

University of Groningen

Socially smart software agents entice people to use higher-order theory of mind in the Mod game

Veltman, Kim; Weerd ,de, Harmen; Verbrugge, Rineke

Published in:

Preproceedings of the 29th Benelux Conference on Artificial Intelligence (BNAIC'2017)

IMPORTANT NOTE: You are advised to consult the publisher's version (publisher's PDF) if you wish to cite from it. Please check the document version below.

Document Version

Publisher's PDF, also known as Version of record

Publication date:

2017

[Link to publication in University of Groningen/UMCG research database](#)

Citation for published version (APA):

Veltman, K., Weerd ,de, H., & Verbrugge, R. (2017). Socially smart software agents entice people to use higher-order theory of mind in the Mod game. In B. Verheij, & M. Wiering (Eds.), *Preproceedings of the 29th Benelux Conference on Artificial Intelligence (BNAIC'2017)* (pp. 253-267). University of Groningen.

Copyright

Other than for strictly personal use, it is not permitted to download or to forward/distribute the text or part of it without the consent of the author(s) and/or copyright holder(s), unless the work is under an open content license (like Creative Commons).

The publication may also be distributed here under the terms of Article 25fa of the Dutch Copyright Act, indicated by the "Taverne" license. More information can be found on the University of Groningen website: <https://www.rug.nl/library/open-access/self-archiving-pure/taverne-amendment>.

Take-down policy

If you believe that this document breaches copyright please contact us providing details, and we will remove access to the work immediately and investigate your claim.

Downloaded from the University of Groningen/UMCG research database (Pure): <http://www.rug.nl/research/portal>. For technical reasons the number of authors shown on this cover page is limited to 10 maximum.

BN AIC

• 2 0 1 7 •

p r e p r o c e e d i n g s

29th Benelux Conference on Artificial Intelligence
November 8-9, 2017, Groningen

Editors: Bart Verheij, Marco Wiering



BN AIC 2017

Benelux Conference on Artificial Intelligence

Preproceedings of the 29th Benelux Conference on Artificial Intelligence
November 8–9, 2017 in Groningen, The Netherlands

Editors
Bart Verheij
Marco Wiering

BNAIC is the annual Benelux Conference on Artificial Intelligence.

This year, the 29th edition of BNAIC is organized by the Institute of Artificial Intelligence and Cognitive Engineering (ALICE), University of Groningen, under the auspices of the Benelux Association for Artificial Intelligence (BN-VKI) and the Dutch Research School for Information and Knowledge Systems (SIKS).

ISBN: 978-94-034-0299-4

Committees

General chairs

Bart Verheij
Marco Wiering

Contact & administration

Elina Sietsema
Carlijne de Vries
Sarah van Wouwe

Local organisation

Luca Bandelli
Abe Brandsma
Tomasz Darmetko
Mingcheng Ding
Ana Dugeniuc
Joel During
Ameer Islam
Siebert Looije
René Mellema
Michaela Mrázková
Annet Onnes
Benjamin Shafrey
Sjaak ten Caat
Albert Thie
Jelmer van der Linde
Luuk van Keeken
Paul Veldhuyzen
Randy Wind
Galiya Yeshmagambetova

Program committee

Stylianos Asteriadis, Maastricht University
Reyhan Aydogan, Delft University of Technology
Floris Bex, Utrecht University
Michael Biehl, University of Groningen
Mauro Birattari, Université Libre de Bruxelles
Hendrik Blockeel, Katholieke Universiteit Leuven
Hans L. Bodlaender, Utrecht University
Sander Bohte, Centrum Wiskunde & Informatica (CWI)
Peter Bosman, Centrum Wiskunde & Informatica (CWI)
Tibor Bosse, Vrije Universiteit Amsterdam
Bruno Bouzy, Paris Descartes University
Frances Brazier, Delft University of Technology
Bert Bredeweg, University of Amsterdam

Tristan Cazenave, Université Paris Dauphine
 Tom Claassen, Radboud University Nijmegen
 Walter Daelemans, University of Antwerp
 Gregoire Danoy, University of Luxembourg
 Mehdi Dastani, Utrecht University
 Patrick De Causmaecker, Katholieke Universiteit Leuven
 Martine De Cock, University of Washington Tacoma
 Mathijs De Weerd, Delft University of Technology
 Benoît Depaire, Hasselt University
 Frank Dignum, Utrecht University
 Virginia Dignum, Delft University of Technology
 Kurt Driessens, Maastricht University
 Madalina Drugan, Technical University of Eindhoven
 Sjur Dyrkolbotn, Western Norway University of Applied Sciences
 Jason Farquhar, Radboud University Nijmegen
 Ad Feelders, Utrecht University
 Bart Goethals, University of Antwerp
 Pascal Gribomont, University of Liège
 Perry Groot, Radboud University Nijmegen
 Marc Gyssens, Universiteit Hasselt
 Frank Harmsen, Ernst & Young Advisory
 Maaike Harbers, Rotterdam University of Applied Sciences
 Tom Heskens, Radboud University Nijmegen
 Koen Hindriks, Delft University of Technology
 Arjen Hommersom, Open University of the Netherlands
 Mark Hoogendoorn, Vrije Universiteit Amsterdam
 Gerda Janssens, Katholieke Universiteit Leuven
 Maurits Kaptein, Eindhoven University of Technology
 Uzey Kaymak, Eindhoven University of Technology
 Walter Kusters, Leiden University
 Johan Kwisthout, Radboud University Nijmegen
 Tom Lenaerts, Université Libre de Bruxelles
 Marco Loog, Delft University of Technology
 Peter Lucas, Radboud University
 Elena Marchiori, Radboud University Nijmegen
 Wannes Meert, Katholieke Universiteit Leuven
 John-Jules Meyer, Utrecht University
 Peter Novák, Delft University of Technology
 Ann Nowé, Vrije Universiteit Brussel
 Aske Plaat, Leiden University
 Eric Postma, Tilburg University
 Henry Prakken, University of Utrecht, University of Groningen
 Jan Ramon, INRIA
 Silja Renooij, Utrecht University
 Nico Roos, Maastricht University
 Stefan Schlobach, Vrije Universiteit Amsterdam
 Pierre-Yves Schobbens, University of Namur
 Johannes Scholtes, Maastricht University
 Lambert Schomaker, University of Groningen
 Martijn Schut, Universiteit van Amsterdam

Evgueni Smirnov, Maastricht University
Matthijs T. J. Spaan, Delft University of Technology
Gerasimos Spanakis, Maastricht University
Jennifer Spenader, University of Groningen
Ida Sprinkhuizen-Kuyper, Radboud University Nijmegen
Thomas Stützle, Université Libre de Bruxelles
Johan Suykens, Katholieke Universiteit Leuven
Niels Taatgen, University of Groningen
Annette Ten Teije, Vrije Universiteit Amsterdam
Dirk Thierens, Utrecht University
Karl Tuyls, University of Liverpool
Antal van den Bosch, Radboud University Nijmegen
Egon L. van den Broek, Utrecht University
Jaap van den Herik, Leiden University
Linda van der Gaag, Utrecht University
Peter van der Putten, Leiden University, Pegasystems
Leon van der Torre, University of Luxembourg
Natalie van der Wal, Vrije Universiteit Amsterdam
Tim van Erven, Leiden University
Marcel van Gerven, Radboud University Nijmegen
Frank van Harmelen, Vrije Universiteit Amsterdam
Martijn van Otterlo, Vrije Universiteit Amsterdam
M. Birna van Riemsdijk, Delft University of Technology
Maarten van Someren, University of Amsterdam
Marieke van Vugt, University of Groningen
Menno van Zaanen, Tilburg University
Joaquin Vanschoren, Eindhoven University of Technology
Marina Velikova, Radboud University Nijmegen
Remco Veltkamp, Utrecht University
Joost Vennekens, Katholieke Universiteit Leuven
Katja Verbeeck, Technologiecampus Gent
Rineke Verbrugge, University of Groningen
Michel Verleysen, Université Catholique de Louvain
Sicco Verwer, Delft University of Technology
Arnoud Visser, Universiteit van Amsterdam
Willem Waegeman, Ghent University
Martijn Warnier, Delft University of Technology
Gerhard Weiss, Maastricht University
Floris Wiesman, Academic Medical Center Amsterdam
Jef Wijsen, University of Mons
Mark H. M. Winands, Maastricht University
Radboud Winkels, Universiteit van Amsterdam
Cees Witteveen, Delft University of Technology
Marcel Worring, University of Amsterdam
Yingqian Zhang, Eindhoven University of Technology

Preface

BNAIC is the annual Benelux Conference on Artificial Intelligence. In 2017, the 29th edition of BNAIC is organized by the Institute of Artificial Intelligence and Cognitive Engineering (ALICE), University of Groningen, under the auspices of the Benelux Association for Artificial Intelligence (BNVKI) and the Dutch Research School for Information and Knowledge Systems (SIKS).

BNAIC 2017 takes place in Het Kasteel, Melkweg 1, Groningen, The Netherlands, on Wednesday November 8 and Thursday November 9, 2017. BNAIC 2017 includes invited speakers, research presentations, posters, demonstrations, a deep learning workshop (organized by our sponsor NVIDIA) and a research and business session.

The four BNAIC 2017 keynote speakers are:

- Marco Dorigo, Université Libre de Bruxelles
Swarm Robotics: Current Research Directions at IRIDIA
- Laurens van der Maaten, Facebook AI Research
From Visual Recognition to Visual Understanding
- Luc Steels, Institute for Advanced Studies (ICREA), Barcelona
Digital Replicants and Mind-Uploading
- Rineke Verbrugge, University of Groningen
Recursive Theory of Mind: Between Logic and Cognition

Three FACt talks (FACulty focusing on the FACts of Artificial Intelligence) are scheduled:

- Bert Bredeweg, Universiteit van Amsterdam
Humanly AI: Creating smart people with AI
- Eric Postma, Tilburg University
Towards Artificial Human-like Intelligence
- Geraint Wiggins, Queen Mary University of London/Vrije Universiteit Brussel
Introducing Computational Creativity

Authors were invited to submit papers on all aspects of Artificial Intelligence. This year we have received 68 submissions in total. Of the 30 submitted Type A regular papers, 11 (37%) were accepted for oral presentation, and 14 (47%) for poster presentation. 5 (17%) were rejected. Of the 19 submitted Type B compressed contributions, 17 (89%) were accepted for oral presentation, and 2 (11%) for poster presentation. None were rejected. All 6 submitted Type C demonstration abstracts were accepted. Of the submitted 13 Type D thesis abstracts, 5 (38%) were accepted for oral presentation, and 8 (62%) for poster presentation. None were rejected. The selection was made using peer review. Each submission was assigned to three members of the program committee, and their expert reviews were the basis for our decisions.

All submissions accepted for oral or poster presentations and all demonstration abstracts appear in the electronic preproceedings, made available on the

conference web site during the conference (<http://bnaic2017.ai.rug.nl/>). All 11 Type A regular papers accepted for oral presentation will appear in the postproceedings, to be published in the Springer CCIS series after the conference.

The BNAIC 2017 conference would not be possible without the support and efforts of many. We thank the members of the program committee for their constructive and scholarly reviews. We are grateful to Elina Sietsema, Carlijne de Vries and Sarah van Wouwe, members of the administrative staff at the Institute of Artificial Intelligence and Cognitive Engineering (ALICE), for their tireless and reliable support. We thank our local organisation team Luca Bandelli, Abe Brandsma, Tomasz Darmetko, Mingcheng Ding, Ana Dugeniuc, Joel During, Ameer Islam, Siebert Looije, René Mellema, Michaela Mrázková, Annet Onnes, Benjamin Shaffrey, Sjaak ten Caat, Albert Thie, Jelmer van der Linde, Luuk van Keeken, Paul Veldhuyzen, Randy Wind, and Galiya Yeshmagambetova, all students in our BSc and MSc Artificial Intelligence programs, for enthusiastically volunteering to help out in many ways. We thank Annet Onnes for preparing the preproceedings, Jelmer van der Linde for developing the web site, Randy Wind for designing the program leaflet, and Albert Thie for coordinating the local organisation.

We are grateful to our sponsors for their generous support of the conference:

- Target Holding
- NVIDIA Deep Learning Institute
- Anchormen
- Quint
- the Netherlands Research School for Information and Knowledge Systems (SIKS)
- SIM-CI
- Textkernel
- LuxAI
- IOS Press
- Stichting Knowledge-Based Systems (SKBS)
- SSN Adaptive Intelligence

We wish you a pleasant conference!

Bart Verheij & Marco Wiering

Contents

Committees	iii
Preface	vi
 I Type A: Regular papers	
Oral presentation	1
Learning-based diagnosis and repair	2
<i>Nico Roos</i>	
Competition between Cooperative Projects	17
<i>Gleb Polevoy and Mathijs de Weerd</i>	
Refining a Heuristic for Constructing Bayesian Networks from Structured Arguments	32
<i>Remi Wieten, Floris Bex, Linda van der Gaag, Henry Prakken and Silja Renooij</i>	
Reciprocation Effort Games	46
<i>Gleb Polevoy and Mathijs de Weerd</i>	
Get Your Virtual Hands Off Me! - Developing Threatening Agents Using Haptic Feedback	61
<i>Linford Goedschalk, Tibor Bosse and Marco Otte</i>	
Tracking Perceptual and Memory Decisions by Decoding Brain Activity	76
<i>Marieke van Vugt, Armin Brandt and Andreas Schulze-Bonhage</i>	
The origin of mimicry: Deception or merely coincidence?	86
<i>Bram Wiggers and Harmen de Weerd</i>	
Feature selection in high-dimensional dataset using MapReduce	101
<i>Claudio Reggiani, Yann-Aël Le Borgne and Gianluca Bontempi</i>	
Simultaneous Ensemble Generation and Hyperparameter Opti- mization for Regression	116
<i>David Roschewitz, Kurt Driessens and Pieter Collins</i>	

Comparison of Machine Learning Techniques for Multi-label Genre Classification	131
<i>Mathijs Pieters and Marco Wiering</i>	
Learning to Play Donkey Kong Using Neural Networks and Reinforcement Learning	145
<i>Paul Ozkohen, Jelle Visser, Martijn van Otterlo and Marco Wiering</i>	
 II Type A: Regular papers	
Poster presentation	160
 A Proposal to Solve Rule Conflicts in the Wang-Mendel Algorithm for Fuzzy Classification Using Evidential Theory	161
<i>Diego Alvarez-Estevéz and Vicente Moret-Bonillo</i>	
 Classification in a Skewed Online Trade Fraud Complaint Corpus	172
<i>William Kos, Marijn Schraagen, Matthieu Brinkhuis and Floris Bex</i>	
 Studying Gender Bias and Social Backlash via Simulated Negotiations with Virtual Agents	184
<i>Laura van der Lubbe and Tibor Bosse</i>	
 Distracted in a Demanding Task: A Classification Study with Artificial Neural Networks	199
<i>Stefan Huijser, Niels Taatgen and Marieke van Vugt</i>	
 Assessing the Spatiotemporal Relation between Twitter Data and Violent Crime	213
<i>Marco Stam, Charlotte Gerritsen, Ward van Breda and Elias Krainski</i>	
 Constructions at Work! Visualising Linguistic Pathways for Computational Construction Grammar	224
<i>Sébastien Hoorens, Katrien Beuls and Paul Van Eecke</i>	
 Generalization of an Upper Bound on the Number of Nodes Needed to Achieve Linear Separability	238
<i>Marjolein Troost, Katja Seeliger and Marcel van Gerven</i>	
 Socially smart software agents entice people to use higher-order theory of mind in the Mod game	253
<i>Kim Veltman, Harmen de Weerd and Rineke Verbrugge</i>	
 Recommending Treatments for Comorbid Patients Using Word-Based and Phrase-Based Alignment Methods	268
<i>Elie Merhej, Steven Schockaert, T. Greg McKelvey and Martine De Cock</i>	
 Reasoning about Norms Revision	281
<i>Fabiano Dalpiaz, Mehdi Dastani and Davide Dell’Anna</i>	

Distribution-driven Regression Ensemble Construction for Time Series Forecasting	291
<i>Florian Wimmenauer, Evgueni Smirnov and Matúš Mihalák</i>	
A Hierarchical Bayesian Network for the Optimization of SRM Assays	306
<i>Jérôme Renaux, Jan Ramon and Andrea Argentini</i>	
Deep Colorization for Facial Gender Recognition	317
<i>Jonathan Hogervorst, Emmanuel Okafor and Marco Wiering</i>	
Predicting Chaotic Time Series using Machine Learning Techniques	326
<i>Henry Maathuis, Luuk Boulogne, Marco Wiering and Alef Sterk</i>	
III Type B: Compressed contributions	
Oral presentation	341
Reactive Versus Anticipative Decision Making in a Novel Gift-Giving Game	342
<i>Elias Fernández Domingos, Juan Carlos Burguillo and Tom Lenaerts</i>	
Evaluating Intelligent Knowledge Systems	344
<i>Neil Yorke-Smith</i>	
The Parameterized Complexity of Approximate Inference in Bayesian Networks	346
<i>Johan Kwisthout</i>	
On the Problem of Making Autonomous Vehicles Conform to Traffic Law	348
<i>Henry Prakken</i>	
The Transitivity and Asymmetry of Actual Causation	350
<i>Sander Beckers and Joost Vennekens</i>	
Multi-View LS-SVM for Temperature Prediction	352
<i>Lynn Houthuys, Zahra Karevan and Johan A.K. Suykens</i>	
Regularized Semi-Paired Kernel CCA for Domain Adaptation .	355
<i>Siamak Mehrkanoon and Johan A.K. Suykens</i>	
Using Values and Norms to Model Realistic Social Agents	357
<i>Rijk Mercur, Virginia Dignum and Catholijn M. Jonker</i>	
Constructing Knowledge Graphs of Depression	360
<i>Zhisheng Huang, Jie Yang, Frank Van Harmelen and Qing Hu</i>	
Learning Multimodal Transition Dynamics for Model-Based Reinforcement Learning	362
<i>Thomas M. Moerland, Joost Broekens and Catholijn M. Jonker</i>	

Simulating Collective Evacuations with Social Elements	364
<i>Daniel Formolo and C. Natalie Van Der Wal</i>	
Melody Retrieval and Classification Using Biologically-Inspired Techniques	367
<i>Dimitrios Bountouridis, Dan Brown, Hendrik Vincent Koops, Frans Wiering and Remco C. Veltkamp</i>	
Omniscient Debugging for Cognitive Agent Programs	370
<i>Vincent Jaco Koeman, Koen V. Hindriks and Catholijn M. Jonker</i>	
Neural Ranking Models with Weak Supervision	372
<i>Mostafa Dehghani, Hamed Zamani, Aliaksei Severyn, Jaap Kamps and W. Bruce Croft</i>	
Participation Behavior and Social Welfare in Repeated Task Allocations	374
<i>Qing Chuan Ye and Yingqian Zhang</i>	
An Empathic Agent that Alleviates Stress by Providing Support via Social Media	376
<i>Lenin Medeiros and Tibor Bosse</i>	
Expectation Management in Child-Robot Interaction	378
<i>Mike Ligthart, Olivier Blanson Henkemans, Koen V. Hindriks and Mark Neerincx</i>	
IV Type B: Compressed contributions	
Poster presentation	380
Chord Label Personalization through Deep Learning of Integrated Harmonic Interval-based Representations	381
<i>Hendrik Vincent Koops, W. Bas de Haas, Jeroen Bransen and Anja Volk</i>	
An Approach for Hospital Planning with Multi-Agent Organizations	383
<i>John Bruntse Larsen and Jørgen Villadsen</i>	
V Type C: Demonstrations	386
Hierarchical Reinforcement Learning for a Robotic Partially Observable Task	387
<i>Denis Steckelmacher, Hélène Plisnier, Diederik M. Roijers and Ann Nowé</i>	

MORL-Glue: A Benchmark Suite for Multi-Objective Reinforcement Learning	389
<i>Peter Vamplew, Dean Webb, Luisa M Zintgraf, Diederik M. Roijers, Richard Dazeley, Rustam Issabekov and Evan Dekker</i>	
RoboCup HQ: A new benchmark focusing on AI, HMI and Autonomous Agents	391
<i>Tijn van der Zant and Lars Zwanepol Klinkmeijer</i>	
The SimuLane Highway Traffic Simulator for Multi-Agent Reinforcement Learning	394
<i>Manon Legrand, Roxana Rădulescu, Diederik M. Roijers and Ann Nowé</i>	
ECrowd: Enterprise Crowdsourcing for Training Cognitive Systems using the Workforce	396
<i>Benjamin Timmermans, Zoltán Szilávik, Manfred Overmeen and Alessandro Bozzon</i>	
Intelligent News Conversation with the Pepper Robot	398
<i>Jonathan Gerbscheid, Thomas Groot and Arnoud Visser</i>	
VI Type D: Thesis abstracts	
Oral presentation	400
Modelling the Generation and Retrieval of Word Associations with Word Embeddings	401
<i>Verna Dankers, Aysenur Bilgin and Raquel Fernández</i>	
The Effect of Tutor Feedback in Language Acquisition Models .	403
<i>Jens Nevens and Katrien Beuls</i>	
Is Mirror Descent a Special Case of Exponential Weights?	405
<i>Dirk van der Hoeven and Tim van Erven</i>	
Modelling Word Associations and Interactiveness for Describer Agents in Word-Guessing Games	408
<i>Verna Dankers, Aysenur Bilgin and Raquel Fernández</i>	
Catch Them If You Can: Malicious Behavior Simulation in Deep Question Answering	410
<i>Nikita Galinkin, Zoltán Szilávik, Lora Aroyo and Benjamin Timmermans</i>	

VII Type D: Thesis abstracts	
Poster presentation	412
‘Well, at least it tried’ The Role of Intentions and Outcomes in Ethically Evaluating Robot Actions	413
<i>Daphne Lenders and Willem F.G. Haselager</i>	
Gamification for Learning by Modelling in Interactive Learning Environments	415
<i>David Stap, Bert Bredeweg and Natasa Brouwer</i>	
Neural Network Reuse in Deep RL for Autonomous Vehicles among Human Drivers	417
<i>Manon Legrand, Roxana Rădulescu, Diederik M. Roijers and Ann Nowé</i>	
An Agent-Based Model for Feasibility and Diffusion of Crowd Shipping	419
<i>Max van de Westelaken and Yingqian Zhang</i>	
Realtime Road User Detection and Classification with Single Pass Deep Learning	421
<i>Robin Manhaeve, Luc De Raedt, Kurt De Grave and Laura Antanas</i>	
Balancing Imbalances	423
<i>Marten Schutten, Marco Wiering and Pamela MacDougall</i>	
Should you link(ed) data?	425
<i>Jesse Bakker, Wouter Beek and Erwin Folmer</i>	
Customer Profiling based on Electronic Payment Transaction Data	427
<i>Michiel Van Lancker, Annemie Vorstermans and Mathias Verbeke</i>	

Part I

Type A: Regular papers
Oral presentation

Learning-based diagnosis and repair

Nico Roos

Data Science and Knowledge Engineering
Maastricht University
roos@maastrichtuniversity.nl

Abstract. This paper proposes a new form of diagnosis and repair based on reinforcement learning. Self-interested agents learn locally which agents may provide a low quality of service for a task. The correctness of learned assessments of other agents is proved under conditions on exploration versus exploitation of the learned assessments. Compared to collaborative multi-agent diagnosis, the proposed learning-based approach is not very efficient. However, it does not depend on collaboration with other agents. The proposed learning based diagnosis approach may therefore provide an incentive to collaborate in the execution of tasks, and in diagnosis if tasks are executed in a suboptimal way.

1 Introduction

Diagnosis is an important aspect of systems consisting of autonomous and possibly self-interested agents that need to collaborate [4–7, 10, 8, 9, 11, 12, 14–18, 20, 19, 29, 30, 21–23, 25, 24, 26–28, 32–34, 37]. Collaboration between agents may fail because of malfunctioning agents, environmental circumstances, or malicious agents. Diagnosis may identify the cause of the problem and the agents responsible [31]. Efficient multi-agent diagnosis of collaboration failures also requires collaboration and requires sharing of information. Agents responsible for collaboration failures may be reluctant in providing the correct information. Therefore it is important to have an incentive to provide the right information. The ability to learn an assessments of other agents without the need to exchange information, may provide such an incentive.

This paper addresses the learning of a diagnosis in a network of distributed services. In such a network, tasks are executed by multiple agents where each agent does a part of the whole task. The execution of a part of a task will be called a service.

The more than 2000 year old silk route is an example of a distributed network of services. Local traders transported silk and other goods over a small part of the route between China and Europe before passing the goods on to other traders. A modern version of the silk route is a multi modal transport, which can consists of planes, trains, trucks and ships. Another example of distributed services is the computational services on a computer network. Here, the processing of data are the distributed services. In smart energy networks, consumers of energy may also

be producers of energy. The energy flows have to be routed dynamically through the network. A last example of a distributed service is Industry 4.0. In Industry 4.0, the traditional sequential production process is replaced by products that know which production steps (services) are required in their production. Each product selects the appropriate machine for the next production step and tells the machine what it should do.

To describe a network of distributed services such that diagnosis can be performed, we propose a directed graph representation. An arc of the graph represents the provision of a service by some agent. The nodes are the points where a task¹ is transferred from one agent to another. Incorrect task executions are modeled as transitions to special nodes.

The assumption that agents are self-interested and no agent has a global view of the network, limits the possibility of diagnosis and repair. We will demonstrate that it is still possible to learn which agents are reliable w.r.t. the quality of service that they provide.

The remainder of the paper is organized as follows. In the next section, we will present our graph-based model of a network of distributed services. Section 3 presents an algorithm for locally learning the reliability of agents providing services. Section 4 presents the experimental results and Section 5 concludes the paper.

2 The model

We wish to model a network of services provided by a set of agents. The services provided by the agents contribute to the executions of tasks. The order of the services needed for a task need not be fixed, nor the agents providing the services. This suggests that we need a model in which services cause state transitions, and in each state there may be a choice between several agent-service combinations that can provide the next service. The service that is provided by an agent may be of different quality levels. We can model this at an abstract level by different state transitions. If we also abstract from the actual service descriptions, then we can use a graph based representation.

We model a network of services provided by a set of agents Ag using a graph $G = (N, A)$, where the N represents a set of nodes and $A = \{(n_i, n'_i, ag_i) \mid \{n_i, n'_i\} \subseteq N, ag_i \in Ag\}_{i=1}^{|A|}$ set of arcs. Each arc $(n, n', ag) \in A$ represents a service (n, n') that is provided by an agent $ag \in Ag$. We allow for multiple services between two nodes provided that the associated agents are different; i.e., several agents may provide the same service.

A set of tasks T is defined by pairs of nodes $(s, d) \in T$. Any path between the source s and the destination d of a task $(s, d) \in T$; i.e., a path (a_1, \dots, a_k) with $a_i = (n_i, n_{i+1}, ag_i)$, $n_1 = s$ and $n_{k+1} = d$, represents a correct execution of the task.

An incorrect execution of a task $(s, d) \in T$ is represented by a path that ends in a node d' not equal to d ; i.e., a path (a_1, \dots, a_k) with $a_i = (n_i, n_{i+1}, ag_i)$,

¹ In smart energy networks the tasks are the directions in which energy must flow.

$n_1 = s$ and $n_{k+1} = d' \neq d$. A special node f is used to denote the complete failure of a service provided by an agent. No recovery from f is possible and no information about this failure is made available.

To describe a sub-optimal execution of a task $(s, d) \in T$, we associate a set of special nodes with each destination node d . These nodes indicate that something went wrong during the realization of the task. For instance, goods may be damaged during the execution of a transport task. The function $D : N \rightarrow 2^N$ will be used for this purpose. Beside the nodes denoting suboptimal executions, we also include the normal execution; i.e., $d \in D(d)$. Moreover, $f \in D(d)$.

To measure the quality of the execution of a task $(s, d) \in T$, we associate a utility with every possible outcome of the task execution: $U(d', d)$ for every $d' \in D(d)$. Here, $U(f, d) \leq U(d', d) < U(d, d)$ for every $d' \in D(d) \setminus d$.

The possible results of a service provided by agent ag in node n for a task $t = (s, d)$ with destination d , will be specified by the function $E(n, d, ag)$. This function $E : N \times N \times Ag \rightarrow 2^N$ specifies all nodes that may be reached by the provided service. The function must satisfy the following requirements:

$$- E(n, d, ag) \subseteq \{n'' \mid (n, n'', ag) \in A\}$$

We also define a probability distribution $e : N \times N \times Ag \times N \rightarrow [0, 1]$ over $E(n, d, ag)$, describing the probability of every possible outcome of the provided service; i.e.,

$$- e(n, d, ag, n') = P(n' \mid n, d, ag) \\ \text{where } n' \in E(n, d, ag) \text{ and } \sum_{n' \in E(n, d, ag)} e(n, d, ag, n') = 1.$$

There may be several agents in a node n that can provide the next service for a task $t = (s, d)$ with destination d . The function $succ : N \times N \rightarrow 2^{Ag}$ will be used to denote the set of agents $succ(n, d) = \{ag_1, \dots, ag_k\}$ that can provide the next service.

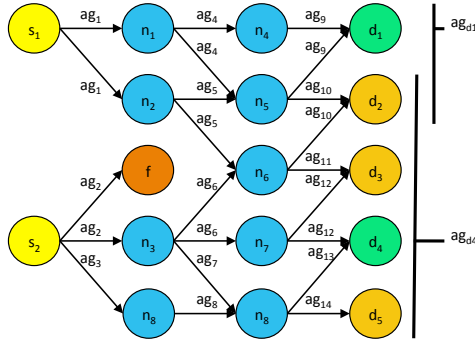


Fig. 1. An example network.

Figure 1 gives an illustration of a network of services represented as a graph. The network shows two starting nodes for tasks, s_1 and s_2 , two successful desti-

nation nodes for tasks, d_1 and d_4 , two unsuccessful destination nodes for tasks, d_2 and d_3 , the failure node f and seven intermediate nodes.

3 Distributed learning of agent reliability

Agents may learn locally diagnostic information using feedback about the result of a task execution. The diagnostic information learned by each agent may enable it to pass on a task in a node to a next agent such that the task is completed in the best possible way. So, an agent must learn the reputation of the agents to which it passes on tasks. This reputation may depend on the node in which the coordination with the next agent takes place as well as on all future agents that will provide services for the task.

We could view our model of a network of services provided by agents as a Markov Decision Process (MDP) [1, 13]. In this markov decision process the nodes in $D(d)$ given the task (s, d) , are absorbing states. Only when reaching a node in $D(d)$ a reward is received. All other rewards are 0. The transition probabilities are given by $e(n, d, ag, n')$. If these probabilities do not depend on the destination; i.e., $e(n, d, ag, n') = P(n' | n, ag)$, then we have a standard markov decision process for which the optimal policy can be learned using Q-learning [35, 36]. However, Q-learning requires that an agent providing a service knows the Q-values of the services the next agent may provide. This implies that we have a Decentralized MDP [2, 3] in which collaboration is needed to learn the optimal Q-values of services. If agents are willing to collaborate, it is, however, more efficient to use the traditional forms of diagnosis [31]. Therefore, in this section, we assume the agents are self-interested and do not collaborate.

To enable local learning of the agents' reputations, we assume that for every task $t = (s, d) \in T$ one and the same agent ag_d is associated with all nodes in $D(d) \setminus f$. Moreover, we assume that each agent that provided a service for the task execution, has added its signature to the task. The incentive for adding a signature is the payment for the provided service. The agent ag_d uses these signatures to make the payments and to inform the agents that provided a service about the success of the whole task execution. The latter information enables each service agent to assess the quality of the agents to which it passes on tasks. If the payments depend on the quality of service of the whole chain, the agents providing services will have an incentive to provide the best possible service and to pass on a task to a next agent such that the quality of the next services is maximized.

An agent ag that provided a service must pass on task $t = (s, d) \in T$ to the next agent if the task is not yet finished. There may be k agents that can provide the next service: ag_1, \dots, ag_k . Assuming that agent ag can identify the current node n and thereby the quality of its own service, ag would like to learn which of the k agents is the most suited to provide the next service for the task.

The agent ag_d associated with the destination d of task $t = (s, d) \in T$ will inform agent ag about the actual quality $d' \in D(d)$ that is realized for the task. This feedback enables agent ag to evaluate the quality of the whole chain of

services starting with a next agent ag_i . So, even if agent ag_i is providing a high quality service, it may not be a good choice if subsequent agents are failing.

An agent ag can learn for each combination of a task destination (the node d) a next agent ag' and the current node n , the probability that the remainder of the task execution will result in the quality $d' \in D(d) \setminus f$. The probability estimate is defined as:

$$pe(d' \mid d, ag', n, i) = \frac{C_{d' \mid i}}{i}$$

where i is the number of times that a task t with destination d is passed on to agent ag' in the node n , and $C_{d' \mid i}$ is the number of times that agent ag_d gives the feedback of d' for task t with destination d .

Agent ag may not receive any feedback if the execution of task t ended in a complete failure, unless agent ag_d knows about the execution of t . In the absence of feedback, agent ag can still learn the probability estimate of a complete failure:

$$pe(f \mid d, ag', n, i) = \frac{C_f \mid i}{i}$$

where $C_f \mid i$ is the number of times that no feedback is received from agent ag_d . An underlying assumption is that agent ag_d always gives feedback when a task is completed, and that the communication channels are failure free.

Estimating the probability is not enough. The behavior of future agents may change over time thereby influencing the probability estimates $pe(d' \mid d, ag', n, i)$. Assuming that the transition probabilities $e(n, n', ag, n'')$ of provided services do not change over time, the coordination between agents when passing on tasks is the only factor influencing the probability estimate $pe(d' \mid d, ag', n, i)$. Since agents have an incentive to select the best possible next agent when passing on a task, we need to address the effect of this incentive on the probability estimates. *First, however, we will investigate the question whether there exist an optimal policy for passing on a task to a next agent and a corresponding probability $P(d' \mid d, ag', n, i)$.*

To answer the above question, utilities of task executions are important. With more than two possible outcomes for a task execution, i.e., $|D(d)| > 2$, the expected utility of a task execution needs to be considered. Therefore, we need to know the utility $U(d', d)$ of all outcomes $d' \in D(d)$. We assume that either this information is global knowledge or that agent ag_d provides this information in its feedback.

Using the utilities of task outcomes, we can prove that there exists an optimal policy for the agents, and corresponding probabilities.

Proposition 1. *Let ag be an agent that has to choose a next agent ag' to provide a service for the task $t = (s, d) \in T$ in node n . Moreover, let $P(d' \mid ag', d, n)$ be the probability of reaching d' given the policies of the succeeding agents.*

The utility $U(d, ag, n)$ an agent ag can realize in node n for a task t with destination d , is maximal if every agent ag chooses a next agent ag^ in every node n in which it can provide a service, such that the term $\sum_{d' \in D(d)} P(d' \mid ag^*, d, n) \cdot U(d', d)$ is maximal.*

Proof. Given a task $t = (s, d) \in T$ we wish to maximize the expected utility agent ag can realize in node n by choosing the proper next agent to provide a service for the task.

$$\begin{aligned}
U(d, ag, n) &= \sum_{d' \in D(d)} P(d' \mid d, n) \cdot U(d', d) \\
&= \sum_{d' \in D(d)} \sum_{ag'} P(d' \mid ag', d, n) \cdot P(ag' \mid d, n) \cdot U(d', d) \\
&= \sum_{ag'} P(ag' \mid d, n) \cdot \sum_{d' \in D(d)} P(d' \mid ag', d, n) \cdot U(d', d)
\end{aligned}$$

Here $P(ag' \mid d, n)$ is the probability that agent ag chooses ag' to be the next agent.

Suppose that the term $\sum_{d' \in D(d)} P(d' \mid ag', d, n) \cdot U(d', d)$ is maximal for $ag' = ag^*$. Then $U(d, ag, n)$ is maximal if agent ag chooses ag^* to be the next agent with probability 1; i.e., $P(ag^* \mid d, n) = 1$. Therefore,

$$U(d, ag, n) = \sum_{d' \in D(d)} P(d' \mid ag^*, d, n) \cdot U(d', d)$$

We can rewrite this equation as:

$$\begin{aligned}
U(d, ag, n) &= \sum_{d' \in D(d)} P(d' \mid ag^*, d, n) \cdot U(d', d) \\
&= \sum_{d' \in D(d)} \sum_{n' \in E(n, d, ag^*)} P(d' \mid d, n') \cdot P(n' \mid ag^*, d, n) \cdot U(d', d) \\
&= \sum_{n' \in E(n, d, ag^*)} P(n' \mid ag^*, d, n) \cdot \sum_{d' \in D(d)} P(d' \mid d, n') \cdot U(d', d) \\
&= \sum_{n' \in E(n, d, ag^*)} e(n, d, ag^*, n') \cdot U(d, ag^*, n')
\end{aligned}$$

Here $P(n' \mid ag^*, d, n)$ is the transition probability of the service provided by agent ag^* , and $U(d, ag^*, n') = \sum_{d' \in D(d)} P(d' \mid d, n') \cdot U(d', d)$ is the expected utility agent ag^* can realize in node n' by choosing the proper next agent to provide a service.

We can now conclude that to maximize $U(d, ag, n)$, agent ag must choose the agent ag^* for which the term $\sum_{d' \in D(d)} P(d' \mid ag', d, n) \cdot U(d', d)$ is maximal, and agent ag^* ensures that $U(d, ag^*, n')$ is maximized. This result enables us to prove by induction to the maximum distance to a node $d' \in D(d)$ that for every agent ag , $U(d, ag, n)$ is *maximal* if every agent ag chooses a next agent ag^* for which the term $\sum_{d' \in D(d)} P(d' \mid ag^*, d, n) \cdot U(d', d)$ is maximal.

- *Initialization step* Let the current node be $d' \in D(d)$. Then the maximum distance is 0 and the current agent is the agent ag_d receiving the result of the task. So, $U(d, ag_d, d') = U(d', d)$.

- *Induction step* Let $U(d, ag_d, n')$ be maximal for all distances less than k . Let n be a node such that the maximum distance to a node in $D(d)$ is k . Then according to the above result, $U(d, ag, n)$ is maximal if agent ag chooses a next agent ag^* for which the term $\sum_{d' \in D(d)} P(d' | ag^*, d, n) \cdot U(d', d)$ is maximal, and for every $n' \in E(n, d, ag^*)$, $U(d, ag^*, n')$ is maximal. The former condition holds according to the prerequisites mentioned in the proposition. The latter condition holds according to the *induction hypothesis*. Therefore, the proposition holds.

□

The proposition shows that there exists an optimal policy for the agents, namely choosing the next agent for which the expected utility is maximized. *The next question is whether the agent can learn the information needed to make this choice.* That is, for every possible next agent, the agent must learn the probabilities of every value in $D(d)$ for a task $t = (s, d) \in T$ with destination d . Since these probabilities depend on the following agents that provide services, the optimal probabilities, denoted by the superscript $*$, can only be learned if these agent have learned to make an optimal choice. So, each agent needs to balance *exploration* (choosing every next agent infinitely many times in order to learn the optimal probabilities) and *exploitation* (choosing the best next agent). We therefore propose the following requirements

- Every agent ag uses a probability $P_i(ag' | d, n)$ to choose a next agent ag' for the task with destination d . The index i denotes that this probability depends on the number of times this choice has been made till now.
- The probability $P_i(ag' | d, n)$ that agent ag will choose agent ag' of which the till now learned expected utility is sub-optimal, approximates 0 if $i \rightarrow \infty$.
- $\sum_{i \rightarrow \infty} P_i(ag' | d, n) = \infty$

The first requirement states that we use a probabilistic exploration. The second requirement ensures that the agent will eventually only exploit what it has learned. The third requirement ensures that the agent will select every possible next agent infinitely many times in order to learn the correct probabilities.

A policy meeting the requirements is the policy in which the agent ag chooses the currently optimal next agent ag' with probability $1 - \frac{1}{(k-1)i}$. Here, k is the number of agents that can perform the next service for a task with destination d , and i is the number of times agent ag has to choose one of these k agents for a task with destination d . The agents that are currently not the optimal choice are chosen with probability $\frac{1}{(k-1)i}$.

We can prove that any approach meeting the above listed requirements will enable agents to learn the optimal policy.

Theorem 1. *Let every agent ag meet the above listed requirements for the probability $P_i(ag' | d, n)$ of choosing the next agent. Moreover, let $P^*(d' | ag, d, n)$ be the optimal probability of reaching the node $d' \in D(d)$ if every agent chooses a next agent ag^* for which the term $\sum_{d' \in D(d)} P(d' | ag^*, d, n) \cdot U(d', d)$ is maximal.*

Then, every agent ag learns $P^*(d' \mid ag', d, n)$ through $pe(d' \mid ag', d, n, i)$ if the number of tasks with destination d for which agent ag has to choose a next agent ag' , denoted by i , goes to infinity.

Proof. We have to prove that: $\lim_{i \rightarrow \infty} pe(d' \mid ag', d, n, i) = P^*(d' \mid ag', d, n)$.

We can rewrite $\lim_{i \rightarrow \infty} pe(d' \mid ag', d, n, i)$ as:

$$\begin{aligned} \lim_{i \rightarrow \infty} pe(d' \mid ag', d, n, i) &= \lim_{i \rightarrow \infty} \frac{C_{d' \mid i}}{i} \\ &= \lim_{i \rightarrow \infty} \sum_{n' \in E(n, d, ag')} \frac{C_{n' \mid i}}{i} \cdot \frac{C_{d' \mid C_{n' \mid i}}}{C_{n' \mid i}} \\ &= \lim_{i \rightarrow \infty} \sum_{n' \in E(n, d, ag')} pe(n' \mid ag', d, n, i) \cdot \frac{C_{d' \mid C_{n' \mid i}}}{C_{n' \mid i}} \\ &= \sum_{n' \in E(n, d, ag')} P(n' \mid ag', d, n) \cdot \lim_{i \rightarrow \infty} \frac{C_{d' \mid C_{n' \mid i}}}{C_{n' \mid i}} \end{aligned}$$

We will prove that $C_{n' \mid i} \rightarrow \infty$ if $i \rightarrow \infty$ and $P(n' \mid ag', d, n) > 0$. That is, for every $x \in \mathbb{N}$, $\lim_{i \rightarrow \infty} P(C_{n' \mid i} > x) = 1$.

$$\begin{aligned} \lim_{i \rightarrow \infty} P(C_{n' \mid i} > x) &= \lim_{i \rightarrow \infty} 1 - P(C_{n' \mid i} \leq x) \\ &= 1 - \lim_{i \rightarrow \infty} \sum_{j=0}^x (P(n' \mid ag', d, n))^j \cdot (1 - P(n' \mid ag', d, n))^{i-j} \\ &= 1 \end{aligned}$$

So, $C_{n' \mid i} \rightarrow \infty$ if $i \rightarrow \infty$. Therefore,

$$\lim_{i \rightarrow \infty} pe(d' \mid ag', d, n, i) = \sum_{n' \in E(n, d, ag')} P(n' \mid ag', d, n) \cdot \lim_{j \rightarrow \infty} pe(d' \mid d, n', j)$$

The estimated probability $pe(d' \mid d, n', j)$ depends on the probability of choosing the next agent. This probability is a function of the j -th time agent ag' must choose a next agent ag'' for a task with destination d in node n' .

$$\lim_{j \rightarrow \infty} pe(d' \mid d, n', j) = \lim_{j \rightarrow \infty} \sum_{ag'' \in succ(n', d)} P_j(ag'' \mid d, n') \cdot \frac{C_{d' \mid ag'', j}}{C_{ag'' \mid j}}$$

where $C_{ag'' \mid j}$ is the number of times that agent ag'' was chosen to be the next agent, and $C_{d' \mid ag'', j}$ is the number of times that subsequently node d' was reached.

We will prove that $C_{ag'' \mid j} \rightarrow \infty$ if $j \rightarrow \infty$ and $P_j(ag'' \mid d, n) > 0$ for every j . That is, for every $x \in \mathbb{N}$, $\lim_{j \rightarrow \infty} P(C_{ag'' \mid j} > x) = 1$. A complicating factor is that $P_j(ag' \mid d, n)$ can be different for every value of j . Let y be the index of

the last time agent ag'' is chosen, and let p_x be the probability of all possible sequences till index y . Then we can formulate:

$$\begin{aligned}\lim_{j \rightarrow \infty} P(C_{ag''} \mid j > x) &= \lim_{j \rightarrow \infty} 1 - P(C_{ag''} \mid j \leq x) \\ &= 1 - p_x \cdot \lim_{j \rightarrow \infty} \prod_{k=y+1}^j (1 - P_k(ag'' \mid d, n)) \\ &= 1 - e^{\ln(p_x) + \sum_{k=y+1}^{\infty} \ln(1 - P_k(ag'' \mid d, n))}\end{aligned}$$

According to the Taylor expansion of $\ln(\cdot)$: $\ln(1 - P_k(ag'' \mid d, n)) < -P_k(ag'' \mid d, n)$. Therefore,

$$\begin{aligned}\lim_{j \rightarrow \infty} P(C_{ag''} \mid j > x) &= 1 - e^{\ln(p_x) - \sum_{k=y+1}^{\infty} P_k(ag'' \mid d, n)} \\ &= 1 - e^{\ln(p_x) - \infty} = 1\end{aligned}$$

The above result implies:

$$\lim_{j \rightarrow \infty} pe(d' \mid d, n', j) = \lim_{j \rightarrow \infty} \sum_{ag'' \in succ(n', d)} P_j(ag'' \mid d, n') \cdot \lim_{k \rightarrow \infty} pe(d' \mid ag'', d, n', k)$$

We can now prove the theorem by induction to the maximum distance to a node $d' \in D(d)$.

- *Initialization step* Let the current node be $d' \in D(d)$. The maximum distance is 0 and the current agent is the agent ag_d receiving the result of the task. So, $\lim_{i \rightarrow \infty} pe(d' \mid ag_d, d, d', i) = P^*(d' \mid ag_d, d, d') = 1$.
- *Induction step* Let $\lim_{j \rightarrow \infty} pe(d' \mid ag', d, n', j) = P^*(d' \mid ag', d, n')$ be maximal for all distances less than k . Moreover, let the maximum distance from n to d' be k .

Then, the expected utility of agent $ag'' \in succ(n', d)$ is:

$$\begin{aligned}\lim_{j \rightarrow \infty} U_j(ag'', d, n') &= \lim_{j \rightarrow \infty} \sum_{d' \in D(d)} pe(d' \mid ag'', d, n', j) \cdot U(d', d) \\ &= \sum_{d' \in D(d)} P^*(d' \mid ag', d, n') \cdot U(d', d) = U^*(ag'', d, n')\end{aligned}$$

According to the requirement,

$$\lim_{j \rightarrow \infty} P_j(ag_j^* \mid d, n') = 1 \text{ for } ag_j^* = \operatorname{argmax}_{ag''} U_j(ag'', d, n')$$

So,

$$\begin{aligned}ag^* &= \lim_{j \rightarrow \infty} ag_j^* \\ &= \lim_{j \rightarrow \infty} \operatorname{argmax}_{ag''} U_j(ag'', d, n') \\ &= \operatorname{argmax}_{ag''} U^*(ag'', d, n')\end{aligned}$$

This implies:

$$\begin{aligned}
\lim_{j \rightarrow \infty} pe(d' \mid d, n', j) &= \lim_{\substack{j \rightarrow \infty \\ ag'' \in succ(n', d)}} \sum P_j(ag'' \mid d, n) \cdot \lim_{k \rightarrow \infty} pe(d' \mid ag'', d, n', k) \\
&= \sum_{ag'' \in succ(n', d)} P^*(d' \mid ag'', d, n') \cdot \lim_{j \rightarrow \infty} P_j(ag'' \mid d, n) \\
&= P^*(d' \mid ag^*, d, n') = P^*(d' \mid d, n')
\end{aligned}$$

Therefore,

$$\begin{aligned}
\lim_{i \rightarrow \infty} pe(d' \mid ag', d, n, i) &= \sum_{n' \in E(n, d, ag')} P(n' \mid ag', d, d) \cdot \lim_{j \rightarrow \infty} pe(d' \mid d, n', j) \\
&= \sum_{n' \in E(n, d, ag')} e(n, d, ag', n') \cdot P^*(d' \mid d, n') \\
&= P^*(d' \mid ag', d, n)
\end{aligned}$$

□

The theorem shows us that each agent can learn which next agent results in an expected high or low quality for the remainder of a task. In order to learn this assessment, the agents must explore all possible choices for a task infinitely many times. At the same time the agents may also exploit what they have learned so far. In the end the agents will only exploit what they have learned. Hence, the learning-based approach combines *diagnosis and repair*.

An advantage of the learning-based approach is that intermitting faults can be addressed and that no collaboration between service agents is required. A disadvantage is that making diagnosis requires information about many executions of the same task. However, as we will see in the next section, a repair is learned quickly at the price that correctly functioning agents may be ignored.

Agents learn an assessment for each possible destination. In special circumstances, they need not consider the destination, and can focus on the next agent that can provide a service for a task. First, the quality of service provided by an agent does not depend on the destination of the task. Second, we do not use utilities for the result of a task and only identify whether a task execution is successful. If these conditions are met, an agent can learn for every next agent the probability that the task execution will be successful.

4 Experiments

To determine the applicability of the theoretical results of the previous section, we ran several experiments. For the experiments, we used a network of n^2 normal nodes organized in n layers of n nodes. Every normal node in a layer, except the last layer, is connected to two normal nodes in the next layer. Moreover, from every normal node in the first layer, every normal node in the last layer can be reached. With every transition a different agent is associated. To model

that these agents may provide a low quality of service, for every transition from normal node n to normal node n' representing the correct execution of a service by an agent, there is also a transition from n to an abnormal node n'' representing the incorrect execution of the service. Here, the abnormal node n'' is a duplicate of the normal node of n' . For every normal node except the nodes in the first layer, there is a duplicate abnormal node denoting the sub-optimal execution of a service. In this model, no recovery is possible. Figure 2 show a 4 by 4 network. The normal nodes that can be used for a normal execution of tasks are shown in yellow, blue and green. The duplicate abnormal nodes representing a sub-optimal execution are shown in orange. The transitions to the latter nodes and the transitions between the latter nodes are not shown in the figure.

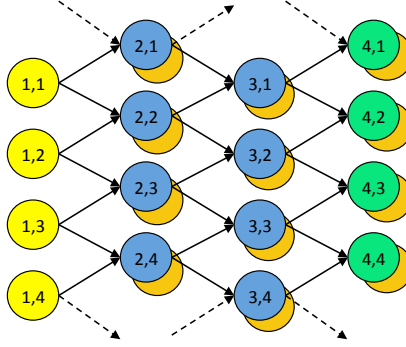


Fig. 2. The network used in the experiments. Note that the dashed arrows denote transitions from nodes (1,4), (2,1) and (3,4) to nodes (2,1), (3,4) and (4,1) respectively.

In our first experiment we determined how often a randomly chosen service is executed in 10000 randomly chosen tasks. We used a network of 10 by 10 nodes in this experiment. Figure 3 shows the cumulative results as a function of the number of processed task. Figure 4 shows in which experiment the service is used.

In the second experiment we used the same network. A fault probability of 0.1 was assigned to the randomly chosen service. Again, we measured how often a service is executed in 10000 randomly chosen tasks. Figure 5 shows the cumulative results as a function of the number of experiments, and Figure 6 shows in which task the service is executed. We clearly see that the agents learn to avoid the agent that provides a low quality of service.

The results show that each agent learns to avoid passing on a task to an agent that may provide a low quality of service. An agent uses the estimated probabilities of a successful completion of a task when passing on the task to the next agent. Nevertheless, as shown in Figure 6, the agents still try the low quality service, but with an increasingly lower probability. This exploration is necessary to learn the correct probabilities.

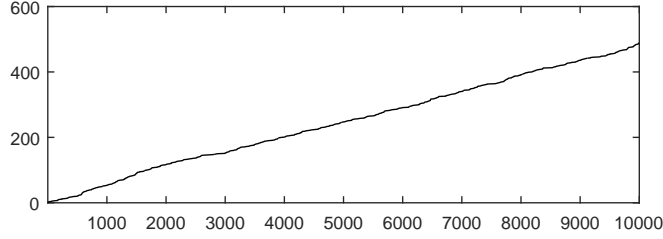


Fig. 3. The number of times a selected service is chosen as a function of the number of processed task.



Fig. 4. The tasks in which a selected service is chosen.

Inspection of the learned probabilities shows that the learning process is slow w.r.t. the total number of executed tasks. Figure 7 shows the learning of the probability that choosing an agent in a node n will result in a good quality of service for a task with a specific destination d . The probability that must be learned is 0.5. The agents only learn when they provided a service for a task with destination d . In Figure 7, the service is executed only 4 times for tasks with destination d of 10000 executions of randomly chosen task. Although the learning process is slow, it is not a problem for the behavior of the network of distributed services. However, it does result in avoiding the services provided by some agents while there is no need for it.

In the third experiment we learned the probability that choosing an agent will result in a good quality of service for a task, independent of the destination of the task. Figure 8 shows the result of the learning process. Again the probability that must be learned is 0.5. The learning process is much faster. However, as

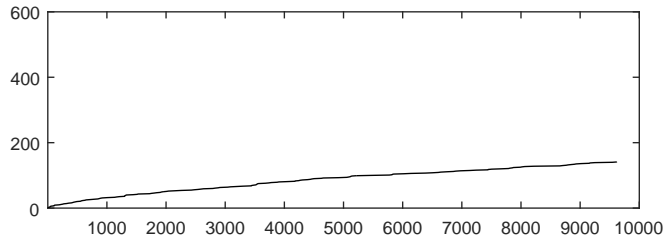


Fig. 5. The number of times a selected service is chosen as a function of the number of processed task.



Fig. 6. The tasks in which a selected service is chosen.

discussed at the end of the previous section, ignoring the destination of a task is only possible if the quality of service does not depend on the destination, and if we only identify whether a task is successful.

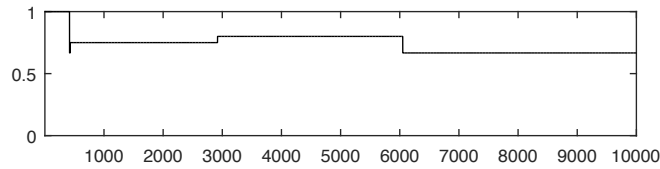


Fig. 7. Learning of the service success probability given a destination.

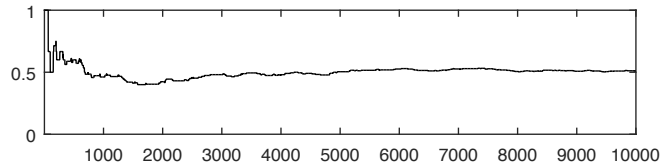


Fig. 8. Learning of the service success probability ignoring the destination.

5 Conclusions

This paper presented a model for describing a network of distributed services for task executions. Each service is provided by an autonomous, possibly self-interested agent. The model also allows for the description of sub-optimal and failed services.

When a task is completed with a low quality, we would like to determine which service was of insufficient quality, which agent was responsible for the provision of this service, and how we can avoid agents that might provide a low quality of service. To answer these questions, the paper investigated an approach for learning in a distributed way an assessment of other agents. The learned information can be exploited to maximize the quality of a task execution. The

correctness of the learned diagnosis an repair approach is proved, and demonstrated through experiments.

An important aspect of the distributed learning approach is that agents do not have to collaborate. Since diagnosis of distributed services is about identifying the agents that are to blame for a low quality of service, this is an important property. It provides an incentive for being honest if agents make a diagnosis in a collaborative setting. Systematic lying will be detected eventually.

This research opens up several lines of further research. First, other policies that balance exploration and exploitation could be investigated. Second, more special cases in which the learning speed can be improved should be investigated. The topology might, for instance, be exploited to improve the learning speed. Third, since agents learn to avoid services of low quality before accurately learning the corresponding probabilities, we may investigate whether we can abstract from the actual probabilities. Fourth, as mentioned in the Introduction and above, the learned assessments provide an incentive for honesty when agents make a collaborative diagnosis. Is this incentive sufficient for agents to collaborate if traditional diagnostic techniques are used?

References

1. R. Bellman. A markovian decision process. *Journal of Mathematics and Mechanics*, 6:679–684, 1957.
2. D. S. Bernstein, R. Givan, N. Immerman, and S. Zilberstein. Sequential optimality and coordination in multiagent systems. *Mathematics of Operations Research*, 6(4):819–840, 2002.
3. C. Boutilier. Sequential optimality and coordination in multiagent systems. In *IJCAI*, pages 478–485, 1999.
4. L. Console and P. Torasso. Hypothetical reasoning in causal models. *International Journal of Intelligence Systems*, 5:83–124, 1990.
5. L. Console and P. Torasso. A spectrum of logical definitions of model-based diagnosis. *Computational Intelligence*, 7:133–141, 1991.
6. R. Davis. Diagnostic reasoning based on structure and behaviour. *Artificial Intelligence*, 24:347–410, 1984.
7. F. de Jonge and N. Roos. Plan-execution health repair in a multi-agent system. In *PlanSIG 2004*, 2004.
8. F. de Jonge, N. Roos, and H. Aldewereld. Multiagent system technologies. In *Multiagent System Technologies*, 2007.
9. F. de Jonge, N. Roos, and H. Aldewereld. Temporal diagnosis of multi-agent plan execution without an explicit representation of time. In *BNAIC-07*, 2007.
10. F. de Jonge, N. Roos, and H.J. van den Herik. Keeping plan execution healthy. In *Multi-Agent Systems and Applications IV: CEEMAS 2005, LNCS 3690*, pages 377–387, 2005.
11. F. de Jonge, N. Roos, and C. Witteveen. Diagnosis of multi-agent plan execution. In *Multiagent System Technologies: MATES 2006, LNCS 4196*, pages 86–97, 2006.
12. F. de Jonge, N. Roos, and C. Witteveen. Primary and secondary plan diagnosis. In *The International Workshop on Principles of Diagnosis, DX-06*, 2006.
13. R. A. Howard. *Dynamic Programming and Markov Processes*. MIT Press, 1960.

14. M. Kalech and G. A. Kaminka. On the design of social diagnosis algorithms for multi-agent teams. In *IJCAI-03*, pages 370–375, 2003.
15. M. Kalech and G. A. Kaminka. Diagnosing a team of agents: Scaling-up. In *AAMAS 2005*, pages 249–255, 2005.
16. M. Kalech and G. A. Kaminka. Towards model-based diagnosis of coordination failures. In *AAAI 2005*, pages 102–107, 2005.
17. M. Kalech and G. A. Kaminka. On the design of coordination diagnosis algorithms for teams of situated agents. *Artificial Intelligence*, 171:491–513, 2007.
18. M. Kalech and G. A. Kaminka. Coordination diagnostic algorithms for teams of situated agents: Scaling up. *Computational Intelligence*, 27(3):393–421, 2011.
19. J. de Kleer, A.K. Mackworth, and R. Reiter. Characterizing diagnoses and systems. *Artificial Intelligence*, 56:197–222, 1992.
20. J. de Kleer and B. C. Williams. Diagnosing with behaviour modes. In *IJCAI 89*, pages 104–109, 1989.
21. R. Micalizio. A distributed control loop for autonomous recovery in a multi-agent plan. In *Proceedings of the Twenty-First International Joint Conference on Artificial Intelligence, (IJCAI-09)*, pages 1760–1765, 2009.
22. R. Micalizio. Action failure recovery via model-based diagnosis and conformant planning. *Computational Intelligence*, 29(2):233–280, 2013.
23. R. Micalizio and P. Torasso. On-line monitoring of plan execution: A distributed approach. *Knowledge-Based Systems*, 20:134–142, 2007.
24. R. Micalizio and P. Torasso. *Plan Diagnosis and Agent Diagnosis in Multi-agent Systems*, pages 434–446. Springer, 2007.
25. R. Micalizio and P. Torasso. Team cooperation for plan recovery in multi-agent systems. In *Multiagent System Technologies, LNCS 4687*, pages 170–181, 2007.
26. R. Micalizio and P. Torasso. Monitoring the execution of a multi-agent plan: Dealing with partial observability. In *Proceedings of the 18th European Conference on Artificial Intelligence (ECAI-08)*, pages 408–412. IOS Press, 2008.
27. R. Micalizio and P. Torasso. Cooperative monitoring to diagnose multiagent plans. *Journal of Artificial Intelligence Research*, 51:1–70, 2014.
28. R. Micalizio and G. Torta. Explaining interdependent action delays in multiagent plans execution. *Autonomous Agents and Multi-Agent Systems*, 30(4):601–639, 2016.
29. O. Raiman, J. de Kleer, V. Saraswat, and M. Shirley. Characterizing non-intermittent faults. In *AAAI 91*, pages 849–854, 1991.
30. R. Reiter. A theory of diagnosis from first principles. *Artificial Intelligence*, 32:57–95, 1987.
31. N. Roos, A. ten Teije, and C. Witteveen. Reaching diagnostic agreement in multi-agent diagnosis. In *AAMAS 2004*, pages 1254–1255, 2004.
32. N. Roos and C. Witteveen. Diagnosis of plan execution and the executing agent. In *Advances in Artificial Intelligence (KI 2005), LNCS 3698*, pages 161–175, 2005.
33. N. Roos and C. Witteveen. Diagnosis of plan structure violations. In *Multiagent System Technologies*, 2007.
34. N. Roos and C. Witteveen. Models and methods for plan diagnosis. *Journal of Autonomous Agents and Multi-Agent Systems*, 19:30–52, 2008.
35. C. J. C. H. Watkins. *Learning from Delayed Rewards*. PhD thesis, Cambridge University, 1989.
36. C. J. C. H. Watkins and P. Dayan. Q-learning. *Machine Learning*, 8(3):279–292, 1992.
37. C. Witteveen, N. Roos, R. van der Krogt, and M. de Weerd. Diagnosis of single and multi-agent plans. In *AAMAS 2005*, pages 805–812, 2005.

Competition between Cooperative Projects

Gleb Polevoy¹ and Mathijs de Weerdt²

¹ University of Amsterdam^{**}, g.polevoy@uva.nl

² Delft University of Technology, m.m.deweerd@tudelft.nl

Abstract. A paper needs to be good enough to be published; a grant proposal needs to be sufficiently convincing compared to the other proposals, in order to get funded. Papers and proposals are examples of cooperative projects that compete with each other and require effort from the involved agents, while often these agents need to divide their efforts across several such projects. We aim to provide advice how an agent can act optimally and how the designer of such a competition (e.g., the program chairs) can create the conditions under which a socially optimal outcome can be obtained. We therefore extend a model for dividing effort across projects with two types of competition: *a quota* or *a success threshold*. In the quota competition type, only a given number of the best projects survive, while in the second competition type, only the projects that are better than a predefined success threshold survive. For these two types of games we prove conditions for equilibrium existence and efficiency. Additionally we find that competitions using a success threshold can more often have an efficient equilibrium than those using a quota. We also show that often a socially optimal Nash equilibrium exists, but there exist inefficient equilibria as well, requiring regulation.

1 Introduction

Cooperative projects often compete with each other. For example, a paper needs to have a certain quality, or to be among a certain number of the best papers to be published, and a grant needs to be one of the best to be awarded. Either the projects that achieve a certain *minimum level*, or those that are among a certain *quota* of the best projects attain their value. Agents endowed with a resource budget (such as time) need to divide this resource across several such projects. We consider so-called public projects where agents contribute resources to create something together. If such a project survives the competition, its rewards are typically divided among the contributors based on their individual investments.

Agents often divide effort across competing projects. In addition to co-authoring articles or books [6, 7, 10] and research proposals, examples include participating in crowdsensing projects [8] and online communities [9]. Examples of quotas for successful projects include investing effort in manufacturing several products, where the market becomes saturated with a certain number of products. Examples of success thresholds are investing in start-ups, where a

^{**} Most of this work was done at Delft University of Technology.

minimum investment is needed to survive, or funding agencies contributing to social projects, where a minimum contribution is required to make the project succeed. Another example is students investing effort in study projects.

The ubiquity and the complexity of such competing projects calls for a decision-support system, helping agents to divide their efforts wisely. Assuming rationality of all the others, an agent needs to know how to behave given the behavior of the others, and the designer of the competition would like to know which rules lead to better results. In the terms of non-cooperative game theory, the objective of this work is to find the equilibria and their efficiency.

Analyzing the NE and their efficiency helps characterizing the influence of a quota or a success threshold on how efficient the stable strategies are for the society and thus increase the efficiency of investing time in the mentioned enterprises. For example, Batchelor [4] suggests increasing the publication standards. However, in addition to maximizing the total value of the published papers, he considers goals such as reducing the noise (number of low quality publications).

To make things clear, we employ this running example:

Example 1. Consider scientists investing time from their time budget in writing papers. A paper attains its value (representing the acknowledgment and all the related rewards) if it stands up to the competition with other papers. The competition can mean either being one of the q best papers, or achieving at least the minimum level of δ , depending on the circumstances. A scientist is rewarded by a paper by becoming its co-author if she has contributed enough to that paper.

Here, the submitters need to know how to split their efforts between the papers, and the conference chairs need to properly organize the selection process, e.g. by defining the quota or threshold on the papers to get accepted.

There were several studies of contributing to projects but the projects did not compete. For example, in the all-pay auction model, only one contributor benefits from the project, but everyone contributes. Its equilibria are analyzed in [5], etc. A famous example is the colonel Blotto game with two players [14], where these players spread their forces among the battlefields, winning a battle if allocating it more forces than the opponent does. The relative number of won battles determines the player's utility. Anshelevich and Hoefer [2] model two-player games by an undirected graph where nodes contribute to the edges. A project, being an edge, obtains contributions from two players. They study minimum-effort projects, proving the existence of an NE and showing that the price of anarchy (PoA)³ is at most 2.

The effort-dividing model [13] used the model of a shared effort game [3], where each player has a budget to divide among a given set of projects. The game possesses a contribution threshold θ , and the project's value is equally shared among the players who invest above this threshold. They analyzed Nash

³ The social welfare is the sum of the utilities of all the players. The price of anarchy [11, 12] is the ratio of the minimum social welfare in an NE to the maximum possible social welfare. The price of stability [15, 1] is the ratio of the maximum social welfare in an NE to the maximum possible social welfare.

equilibria (NE) and their price of anarchy (PoA) and stability (PoS) for such games. However, they ignored that projects may compete for survival. We fill this gap, extending their model by allowing the projects only to obtain their modeled value if they stand up to a competition. To conclude, we study the yet unanswered question of strategic behavior with multiple competing projects.

Compared to the contribution in [10], we model contributing to multiple projects by an agent, and concentrate on the competition, rather than on sharing a project's utility. Unlike devising division rules to make people contribute properly, studied in cooperative game theory (see *Shapley value* [16] for a prominent example), we model given division rules and analyze the obtained game, using non-cooperative game theory.

We formally define the following models:

1. Given a *quota* q , only q projects receive their value. This models the limit on the number of papers to be accepted to a conference, the number of politicians in a city council, the lobbyists being the agents and the politicians being the projects, or the number of projects an organization can fund.
2. There exists a *success threshold* δ , such that only the projects that have a value of at least δ actually receive their value. This models a paper or proposal acceptance process that is purely based on quality.

Our contributions are as follows: We analyze existence and efficiency of NE in these games. In particular, we demonstrate that introducing a quota or a success threshold can sometimes kill existing equilibria, but sometimes allow for new ones. We study how adjusting a quota or a success threshold influences the contribution efficiency, and thereby the social welfare of the participants. We derive that competitions using a success threshold have efficient equilibria more often than those with a quota. We also prove that characterizing the existence of an NE would require more parameters than just the quota or the threshold and the number of the agents and the projects.

We formalize our models in Section 2, analyze the Nash equilibria of the first model and their efficiency in Section 3, and analyze the second model in Section 4. Theorems 2, 3, 5 and 6 are inspired by the existence and efficiency results for the model without competition. Having analyzed both models of competition between projects, Section 5 compares their characteristics, the possibility to influence the authors' behavior through tuning the acceptance criteria, and draws further conclusions. Some proofs are deferred to the appendix (Section A).

2 Model

We build our model on that from [13], since that is a model of investment in common projects with a general threshold. We first present their model for *shared effort games*, which also appears in [3]. From Definition 1 on, we introduce competition among the projects.

There are n players $N = \{1, \dots, n\}$ and a set Ω of m projects. Each player $i \in N$ can contribute to any of the projects in Ω_i , where $\emptyset \subsetneq \Omega_i \subseteq \Omega$;

the contribution of player i to project $\omega \in \Omega_i$ is denoted by $x_\omega^i \in \mathbb{R}_+$. Each player i has a budget $B_i > 0$, so that the strategy space of player i (i.e., the set of her possible actions) is defined as $\{x^i = (x_\omega^i)_{\omega \in \Omega_i} \in \mathbb{R}_+^{|\Omega_i|} \mid \sum_{\omega \in \Omega_i} x_\omega^i \leq B_i\}$. Denote the strategies of all the players except i by x^{-i} .

The next step to define a game is defining the utilities. Let us associate each project $\omega \in \Omega$ with its *project function*, which determines its *value*, based on the total contribution $x_\omega = (x_\omega^i)_{i \in N}$ that it receives; formally, $P_\omega(x_\omega): \mathbb{R}_+^n \rightarrow \mathbb{R}_+$. The assumption is that every P_ω is increasing in every parameter. The increasing part stems from the idea that receiving more effort does not make a project worse off. When we write a project function as a function of a single parameter, like $P_\omega(x) = \alpha x$, we assume that project functions P_ω depend only on the $\sum_{i \in N} (x_\omega^i)$, which is denoted by x_ω as well, when it is clear from the context. The project's value is distributed among the players in $N_\omega \triangleq \{i \in N \mid \omega \in \Omega_i\}$ according to the following rule. From each project $\omega \in \Omega_i$, each player i gets a share $\phi_\omega^i(x_\omega): \mathbb{R}_+^n \rightarrow \mathbb{R}_+$ with free disposal:

$$\forall \omega \in \Omega: \sum_{i \in N_\omega} \phi_\omega^i(x_\omega) \leq P_\omega(x_\omega). \quad (1)$$

We assume the sharing functions are non-decreasing. The non-decreasing assumption fits the intuition that contributing more does not get the players less.

Denote the vector of all the contributions by $x = (x_\omega^i)_{\omega \in \Omega}^{i \in N}$. The utility of a player $i \in N$ is defined to be

$$u^i(x) \triangleq \sum_{\omega \in \Omega_i} \phi_\omega^i(x_\omega).$$

Consider the numerous applications where a minimum contribution is required to share the revenue, such as paper co-authorship and homework. To analyze these applications, define a specific variant of a shared effort game, called a *θ -sharing mechanism*. This variant is relevant to many applications, including co-authoring papers and participating in crowdsensing projects. For any $\theta \in [0, 1]$, the players who get a share are defined to be $N_\omega^\theta \triangleq \{i \in N_\omega \mid x_\omega^i \geq \theta \cdot \max_{j \in N_\omega} x_\omega^j\}$, which are those who bid at least θ fraction of the maximum bid size to ω . Define the θ -equal sharing mechanism as equally dividing the project's value between all the users who contribute to the project at least θ of the maximum bid to the project.

The *θ -equal sharing mechanism*, denoted by M_{eq}^θ , is

$$\phi_\omega^i(x_\omega) \triangleq \begin{cases} \frac{P_\omega(x_\omega)}{|N_\omega^\theta|} & \text{if } i \in N_\omega^\theta, \\ 0 & \text{otherwise.} \end{cases}$$

Let us consider θ -equal sharing, where all the project functions are linear, i.e. $P_\omega(x_\omega) = \alpha_\omega(\sum_{i \in N} x_\omega^i)$. W.l.o.g., $\alpha_m \geq \alpha_{m-1} \geq \dots \geq \alpha_1$. We denote the number of projects with the largest coefficient project functions by $k \in \mathbb{N}$,

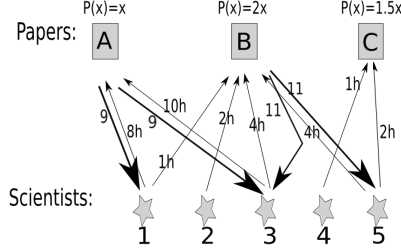


Fig. 1. Scientists contribute time to papers (arrows up), and share the value of the accepted ones (arrows down).

i.e. $\alpha_m = \alpha_{m-1} = \dots = \alpha_{m-k+1} > \alpha_{m-k} \geq \alpha_{m-k-1} \geq \dots \geq \alpha_1$. We call those projects *steep*. Assume w.l.o.g. that $B_n \geq \dots \geq B_2 \geq B_1$.

A project that receives no contribution in a given profile is called a *vacant* project. A player is *dominated* at a project ω , if it belongs to the set $D_\omega \triangleq N_\omega \setminus N_\omega^\theta$. A player is *suppressed* at a project ω , if it belongs to the set $S_\omega \triangleq \{i \in N_\omega : x_\omega^i > 0\} \setminus N_\omega^\theta$. That is, a player who is contributing to a project but is dominated there.

We now depart from [13] and model competition in two different ways.

Definition 1 *In the quota model, given a natural number $q > 0$, only the q highest valued projects actually obtain a value to be divided between their contributors. The rest obtain zero. In the case of ties, all the projects that would have belonged to the highest q under some tie breaking rule receive their value; therefore, more than q projects can receive their value in this case. Formally, project ω is in the quota if $|\{\omega' \in \Omega | P_{\omega'}(x_{\omega'}) > P_\omega(x_\omega)\}| < q$, and ω is out of the quota otherwise, and, effectively, $P_\omega(x_\omega) = 0$.*

The second model is called the success threshold model.

Definition 2 *In the success threshold model, given a threshold δ , only the projects with value at least δ , meaning that $P_\omega(x_\omega) \geq \delta$, obtain a value, while if $P_\omega(x_\omega) < \delta$, then, effectively, $P_\omega(x_\omega) = 0$.*

Example 1 (Continued). Figure 1 depicts a success threshold model, where paper C does not make it to the success threshold, and is, therefore, unpublished. The other two papers are above the success threshold, and get published; such a paper's recognition is equally divided between the contributors who contribute at least θ of the maximum contribution to the paper, and become co-authors.

3 The Quota Model

In this section, we study the equilibria of shared effort games with a quota and their efficiency. We first give an example of an NE, and generalize it to a

sufficiency theorem. Then, we provide equilibrium existence and efficiency theorems for the quota model. Finally, we show that no simple setting of parameters guarantees the existence of an equilibrium or the lack thereof.

Intuitively, introducing a quota can make previously unstable profiles become NE, by making deviations non-profitable. This would increase the price of stability but decrease the price of anarchy. On the other hand, a profile that is an NE without a quota can cease being so in our model, since some projects may obtain no value because of the quota.

Having a quota can lead to counter-intuitive results. In the following example, there can be an NE where no steep project obtains a contribution. The idea is that any deviation from the project where everyone contributes is non-profitable, because it would still leave the other projects out of quota.

Example 2. Given projects 1 and 2, such that $\alpha_2 > \alpha_1$, assume that all the players contribute all their budgets to project 1. If $\alpha_2 B_n < \alpha_1 \sum_{i=1}^{n-1} B_i$ and $q = 1$, then no player can deviate to project 2, as this would still leave that project out of the quota, and therefore, this profile is an NE.

In this NE, the social welfare is equal to $\alpha_1 \sum_{i \in N} B_i$. The optimal social welfare, achieved if and only if all the players contribute all their budgets to project 2, is equal to $\alpha_2 \sum_{i \in N} B_i$. The ratio between the social welfare in this NE and the optimal one is $\frac{\alpha_1}{\alpha_2}$. That ratio is an upper bound on the price of anarchy of this game. In addition, since the optimal profile is also an NE, the price of stability is 1.

The price of anarchy is smaller than $\frac{\alpha_1}{\alpha_2}$ if and only if some agents do not contribute all their budgets. This can only happen in an NE if θ is positive, and if this is the case, then we can have arbitrarily low price of anarchy, down to the case when only agent n contributes, if $\theta B_n > B_{n-1}$, and then, $\text{PoA} = \frac{\alpha_1 B_n}{\alpha_2 \sum_{i \in N} B_i}$.

We now generalize these ideas to the following theorem about possible NE.

Theorem 1. *Consider a θ -equal sharing game with $n \geq 2$ players with budgets $B_n \geq \dots \geq B_2 \geq B_1$ (the order is w.l.o.g.), $0 < \theta < 1$, linear project functions with coefficients $\alpha_m = \alpha_{m-1} = \dots = \alpha_{m-k+1} > \alpha_{m-k} \geq \alpha_{m-k-1} \geq \dots \geq \alpha_1$ (the order is w.l.o.g.), and quota q .*

This game has a pure strategy NE, if $q = 1$ and $B_n < \sum_{i=1}^{n-1} B_i$. Additionally, $\text{PoA} = \frac{\alpha_l \sum_{i: B_i \geq \theta B_n} B_i}{\alpha_m \sum_{i \in N} B_i} \leq \frac{\alpha_l}{\alpha_m}$, for $l \triangleq \min \left\{ j \in \Omega : \alpha_m B_n < \alpha_j \sum_{i=1}^{n-1} B_i \right\}$, $\text{PoS} = 1$.

Proof. If all the players contribute to any single project $j \geq l$, then since $B_n < \sum_{i=1}^{n-1} B_i$, no player can deviate to any project, because this would still leave this project out of the quota. Therefore, this profile is an NE.

In particular, when all the players invest all their budgets in project m , it is an NE, and thus, $\text{PoS} = 1$.

To find the price of anarchy, notice that the worst equilibrium for the social welfare is when everyone contributes to the least profitable possible project, i.e. l , and only those who have a reason to do so contribute. Having an incentive means being not below the threshold amount, θB_n . This equilibrium yields $\alpha_l \sum_{i: B_i \geq \theta B_n} B_i$. ■

This theorem, in accord with the intuition above, shows that reducing the quota can either facilitate an optimal NE, or a very inferior NE. Actually, every efficiency of the form $\frac{\alpha_i}{\alpha_m}$ is possible at equilibrium, which brings us to the question of equilibria appearing and disappearing, which we treat next.

Example 3. A game with NE can cease having equilibria after introducing a quota. For example, consider $\theta \in (0, 1)$, 2 players with budgets $B_1 = \theta B_2$ and 2 projects with the coefficients $\alpha, (1 - \epsilon)\alpha$. This game has an NE if no quota exists, by Theorem 3 from [13]. However, introducing the quota of $q = 1$ implies there is no NE. Indeed, the only candidate profile for an equilibrium is both agents contributing everything to the same project or when both projects obtain the same value. In the former case, agent 2 would like to deviate, to avoid sharing, since the projects are close for small enough an ϵ . In the latter case, agent 2 contributes to both projects, since for small enough an ϵ , agent 1 alone would make the project be out of quota. Since there exists at least one project where agent 1 contributes less than θB_2 , say project i , agent 2 would benefit from contributing to that project all its budget. This is because she would gain at least $(1 - \epsilon)\alpha x_\omega^2$ while losing at most $\alpha(B_1 + x_\omega^2)/2$, which is smaller, for small enough ϵ and θ .

A game can also start having equilibria after introducing a quota. For instance, consider a game with two players, $B_2 = B_1$, $\alpha_m = 1.9\alpha_{m-1}$. Then Theorem 3 from [13] implies that no NE exists, but if we introduce the quota of 1, then both agents contributing to project m is an NE, since a deviator would be out of quota.

We now present an existence theorem. The theorem presents possible equilibria, providing advice on possible stable states. Afterwards, we study efficiency.

Theorem 2. *Consider an equal θ -sharing game with $n \geq 2$ players with budgets $B_n \geq \dots \geq B_2 \geq B_1$ (the order is w.l.o.g.), $0 < \theta < 1$, linear project functions with coefficients $\alpha_m = \alpha_{m-1} = \dots = \alpha_{m-k+1} > \alpha_{m-k} \geq \alpha_{m-k-1} \geq \dots \geq \alpha_1$ (the order is w.l.o.g.), and quota q .*

This game has a pure NE if one of the following holds.⁴

1. $B_1 \geq k\theta B_n$, $k \leq q$ and $\frac{1}{n}\alpha_{m-k+1} \geq \alpha_{m-k}$,
2. $B_1 \geq q\theta B_n$, $k \geq q$ and $B_n < \sum_{j=1}^{n-1} B_j/q$;
3. $B_{n-1} < \frac{\theta}{|\Omega|} B_n$ and all the project functions are equal, i.e. $\alpha_m = \alpha_1$.

The proof provides a profile and shows that no deviation is profitable.

Proof. To prove part 1, distinguish between the case where $k \leq q$ and $k > q$. If $k \leq q$, then the profile where all the players allocate $1/k$ th of their respective budgets to each of the steep projects is an NE for the same reasons that were given for the original model, since here, the quota's existence can only reduce the motivation to deviate.

⁴ If α_{m-k} does not exist, consider the containing condition to be vacuously true.

As for the part 2, consider the profile where all the players allocate $1/q$ th of their respective budgets to each of the q steep projects $m, m-1, \dots, m-q+1$. This is an NE, since the only deviation that is possibly profitable, besides reallocating between the non vacant projects, is a player moving all of her contributions from some projects to one or more of the vacant projects. This cannot bring profit, because these previously vacant projects will be outside of the quota, since $B_n < \sum_{j=1}^{n-1} B_j/q$. As for reallocating between the non-vacant projects, this is not profitable, since $B_1 \geq q\theta B_n$ means that suppressing is impossible. Therefore, this is an NE.

We now prove part 3. Let every player divide her budget equally among all the projects. No player wants to deviate, for the following reasons. All the projects obtain equal value, and therefore are in the quota. Player n suppresses all the rest and obtains her maximum possible profit, $\alpha_m(\sum_{i \in N} B_i)$. The rest obtain no profit, since they are suppressed whatever they do. ■

We now prove an efficiency result, based on Theorem 2.

Theorem 3. *Consider an equal θ -sharing game with $n \geq 2$ players with budgets $B_n \geq \dots \geq B_2 \geq B_1$, $0 < \theta < 1$ (the order is w.l.o.g.), linear project functions with coefficients $\alpha_m = \alpha_{m-1} = \dots = \alpha_{m-k+1} > \alpha_{m-k} \geq \alpha_{m-k-1} \geq \dots \geq \alpha_1$ (the order is w.l.o.g.).⁵, and quota q .*

1. *If at least one of the following holds.*
 - (a) $B_1 \geq k\theta B_n$, $k \leq q$ and $\frac{1}{n}\alpha_{m-k+1} \geq \alpha_{m-k}$,
 - (b) $B_1 \geq q\theta B_n$, $k \geq q$ and $B_n < \sum_{j=1}^{n-1} B_j/q$;*Then, there exists a pure strategy NE and there holds: PoS = 1.*
2. *Assume $B_{n-1} < \frac{\theta}{|\Omega|} B_n$ and all the project functions are equal, i.e. $\alpha_m = \alpha_1$. Then, there exists a pure strategy NE and the following holds: PoS = 1, PoA = $\frac{B_n}{\sum_{i \in \{1,2,\dots,n\}} B_i}$.*

Proof. We first prove part 1a and 1b. According to the proof of parts 1 and 2 of Theorem 2, equally dividing all the budgets among $\min\{k, q\}$ steep projects is an NE. Therefore, PoS = 1.

For part 2, recall that in the proof of part 3 of Theorem 2, we show that everyone equally dividing the budgets between all the projects is an NE. This is optimal for the social welfare, and so PoS = 1. We turn to find the price of anarchy now. If player n acts as just mentioned, while the other players do not contribute anything, then this is an NE, since all the projects are equal and therefore, in the quota, and players $1, \dots, n-1$ will be suppressed at any contribution. An NE cannot have a lower social welfare, since n gets at least $\alpha_m B_n$ in any NE, since this is obtainable alone. Therefore, the fraction between the two social welfare values, namely $\frac{\alpha_m B_n}{\alpha_m \sum_{i \in \{1,2,\dots,n\}} B_i}$, is the PoA. ■

The condition “ $k \geq q$ and $B_n < \sum_{j=1}^{n-1} B_j/q$ ” in Theorem 3 does not hold if the largest budget can be much larger than the rest, implying that we shall

⁵ If α_{m-k} does not exist, consider the containing condition to be vacuously true.

ask whether our optimum NE is guaranteed by part 1a, which requires that the quota has to be at least k . When there are many equally glorious projects to contribute to, meaning that k is large, this constraint becomes non-trivial to implement. The condition “ $k \leq q$ and $\frac{1}{n}\alpha_{m-k+1} \geq \alpha_{m-k}$ ” in Theorem 3 does not hold if the difference between the two largest projects is not big enough, and then the quota has to be at most k if one wants our optimum NE to follow from part 1b. This is non-trivial when we have few most glorious (steep) projects.

We do not know a full characterization of the existence of equilibria; we do know that it would require many parameters. We prove now that the quota with the number of agents and projects do not determine existence.

Proposition 1. *For any quota $q \geq 1$, any number of agents $n \geq 2$ and projects $m \geq 2$, there exists a game which possesses an NE, and a game which does not.*

The proof engineers games with the given parameters with and without NE.

Proof. A game that satisfies the conditions of Theorem 2 provides evidence for the existence.

To find a game without an NE, we first treat the case of $q = 1$. Let all the project coefficient be equal to one another and let

$$B_n > \sum_{i=1}^{n-1} B_i, \quad (2)$$

$$\text{and } B_n > \frac{\sum_{i=1}^n B_i}{|\{i \in N : B_i \geq \theta B_n\}|}. \quad (3)$$

Because of the equality of all the project coefficients and of (2), in an equilibrium, all the agents with budgets at least θB_n will be together with n . Then, (3) implies agent n will deviate, contradictory to having an equilibrium.

For quota $q \geq 2$, let $B_{n-1} < \frac{\theta}{m} B_n$. In any NE, agent n dominates all the rest in the sense that it invests (strictly) more than $B_{n-1}\theta$ in any project that is in the quota, because otherwise, the other agents could get a share at some projects, and assuming $\alpha_l(x + \frac{x}{\theta}) > \alpha_m(\frac{x}{\theta})$ for every project l , agent n would prefer to suppress that. However, if $\alpha_m > \alpha_{m-1}$, n would always prefer to move a bit more contribution to project m , contradictory to the assumption of an NE. ■

4 The Success Threshold Model

In this section, we consider the NE of shared effort games with a success threshold. We allow success thresholds δ be at most the sum of all the budgets times α_m , to let at least one project to obtain its value, in at least one strategy profile. We begin with an example, which inspires a theorem, and then we study existence and efficiency with a given success threshold.

In a profile, we call a project that has a value of at least the threshold an *accepted* project, and we call it *unaccepted* otherwise. In Example 1, the accepted papers are A and B.

Success threshold can cause counter-intuitive results, as follows.

Example 4. Given the projects 1 and 2, such that $\alpha_2 > \alpha_1$, assume that all the players contribute all their budgets to project 1. If $\delta > \alpha_2 B_n$, then no player can deviate to project 2, as this would leave that project unaccepted, and therefore, this profile is an NE.

The conclusions about the prices of anarchy and stability are the same as in Example 2, besides that the price of anarchy can be even zero if $\alpha_1 \sum_{i=1}^n B_i < \delta$.

The exemplified ideas yield the following theorem.

Theorem 4. *Consider an equal θ -sharing game with $n \geq 2$ players with budgets $B_n \geq \dots \geq B_2 \geq B_1$ (the order is w.l.o.g.), $0 < \theta < 1$, linear project functions with coefficients $\alpha_m = \alpha_{m-1} = \dots = \alpha_{m-k+1} > \alpha_{m-k} \geq \alpha_{m-k-1} \geq \dots \geq \alpha_1$ (the order is w.l.o.g.), and success threshold δ .*

This game has a pure NE, if $\alpha_m B_n < \delta$. In addition, $\text{PoA} \leq \frac{\alpha_1}{\alpha_m}$ and $\text{PoS} = 1$. If $\alpha_1 \sum_{i=1}^n B_i < \delta$, then $\text{PoA} = 0$.

Proof. If all the players contribute to any single project, then since $\alpha_m B_n < \delta$, no player can deviate to any project, because this would still leave that project unaccepted. Therefore, this profile is an NE.

In particular, when all the players invest all their budgets in project m , it is an NE, and thus, $\text{PoS} = 1$. When all the players invest in 1, it also is an NE, showing that $\text{PoA} \leq \frac{\alpha_1}{\alpha_m}$, and if $\alpha_1 \sum_{i=1}^n B_i < \delta$, then $\text{PoA} = 0$. ■

This theorem, in accord with the intuition above, shows that increasing the success threshold can either facilitate an optimal NE, or an inferior NE. Actually, every efficiency of the form $\frac{\alpha_j}{\alpha_m}$, for $j \geq \min \{i : \alpha_i \sum_{l=1}^n B_l \geq \delta\}$, is possible at an equilibrium.

Example 5 (Introducing a success threshold can kill or create new NE). The game with $\theta \in (0, 0.5)$, 2 players with budgets $B_1 = 2\theta B_2$ and 2 projects with the coefficients α, α has an NE if no success threshold exists, by Theorem 3 from [13]. If we introduce the success threshold of αB_2 , then in any NE both agents have to contribute to the same project. Then, agent 2 will deviate. For an emerging NE, consider a game with two players, $B_2 = B_1$, $\alpha_m = 1.9\alpha_{m-1}$. Then Theorem 3 from [13] implies that no NE exists, but if we introduce the success threshold of $2B_1\alpha_m$, then both agents contributing to project m constitute an NE, since a deviator would be at a project below the success threshold.

Next, we provide sufficient conditions for the existence of an NE.

Theorem 5. *Consider an equal θ -sharing game with $n \geq 2$ players with budgets $B_n \geq \dots \geq B_2 \geq B_1$ (the order is w.l.o.g.), $0 < \theta < 1$, linear project functions with coefficients $\alpha_m = \alpha_{m-1} = \dots = \alpha_{m-k+1} > \alpha_{m-k} \geq \alpha_{m-k-1} \geq \dots \geq \alpha_1$ (the order is w.l.o.g.), and success threshold δ .*

This game has a pure NE, if one of the following holds.⁶ Define $p \triangleq \left\lfloor \frac{\alpha_m \sum_{i \in N} B_i}{\delta} \right\rfloor$; intuitively, it is the number of the projects that can be accepted.

1. $B_1 \geq k\theta B_n$, $k \leq p$ and $\frac{1}{n}\alpha_{m-k+1} \geq \alpha_{m-k}$,
2. $B_1 \geq p\theta B_n$, $k \geq p \geq 1$ and $\alpha_m B_n < \delta$;
3. $B_{n-1} < \frac{\theta}{|\Omega|} B_n$, all the project functions are equal, i.e. $\alpha_m = \alpha_1$.

Proof. We first prove part 1. The profile where all the players allocate $1/k$ th of their respective budgets to each of the steep projects is an NE for the same reasons that were given for the original model, since here, the requirement to be not less than the success threshold can only reduce the motivation to deviate.

In part 2, consider the profile where all the players allocate $1/p$ th of their respective budgets to each of the p steep projects $m, m-1, \dots, m-p+1$. This is an NE, since the only deviation that is possibly profitable, besides moving budgets between the non vacant projects, is a player moving all of her contributions from some projects to one or more of the vacant projects. This cannot bring profit, because these previously vacant projects will be unaccepted, since $\alpha_m B_n < \delta$. Additionally, any reallocating between the non-vacant projects is not profitable, since $B_1 \geq p\theta B_n$ means that suppressing is impossible. Therefore, the current profile is an NE.

We now prove part 3. We distinguish between the case where the condition $p \geq |\Omega|$ holds or not. If $p \geq |\Omega|$, then the proof continues as in the case of part 3 of Theorem 2, where every player divides her budget equally among all the projects. All the projects are accepted, so no new deviations become profitable.

In the case that $p < |\Omega|$, consider the profile where all the players allocate $1/p$ th of their respective budgets to each of the p projects $m, m-1, \dots, m-p+1$. This is an NE, since the only deviation that is possibly profitable is some player $j < n$ moving all her budget to a vacant project. However, this is not profitable, since the project would be unaccepted, because $B_j \leq B_{n-1} < \frac{\theta}{|\Omega|} B_n < \theta\delta/\alpha_m \leq \delta/\alpha_m$. The penultimate inequality stems from $p < |\Omega| \iff \frac{\alpha_m \sum_{i \in N} B_i}{|\Omega|} < \delta$. Therefore, this is an NE. ■

We now provide an efficiency result, proven in the appendix.

Theorem 6. Consider an equal θ -sharing game with $n \geq 2$ players with budgets $B_n \geq \dots \geq B_2 \geq B_1$, $0 < \theta < 1$ (the order is w.l.o.g.), linear project functions with coefficients $\alpha_m = \dots = \alpha_{m-k+1} > \alpha_{m-k} \geq \alpha_{m-k-1} \geq \dots \geq \alpha_1$ (the order is w.l.o.g.).⁷, and success threshold δ . Define $p \triangleq \left\lfloor \frac{\alpha_m \sum_{i \in N} B_i}{\delta} \right\rfloor$, as in Theorem 5.

1. If at least one of the following holds.

- (a) $B_1 \geq k\theta B_n$, $k \leq p$ and $\frac{1}{n}\alpha_{m-k+1} \geq \alpha_{m-k}$,
- (b) $B_1 \geq p\theta B_n$, $k \geq p \geq 1$ and $\alpha_m B_n < \delta$;

Then, there exists a pure NE and there holds: PoS = 1.

⁶ If α_{m-k} does not exist, consider the containing condition to be vacuously true.

⁷ If α_{m-k} does not exist, consider the containing condition to be vacuously true.

2. Assume $B_{n-1} < \frac{\theta}{|\Omega|} B_n$, all the project functions are equal, i.e. $\alpha_m = \alpha_1$.
Then, there exists a pure NE and $\text{PoS} = 1$. If, in addition, $\alpha_m B_n \geq \delta$, then
 $\text{PoA} = \frac{B_n}{\sum_{i \in \{1, 2, \dots, n\}} B_i}$.

Condition 1a of Theorem 6 implies that if the second best project is close to a best one, then the threshold should be big enough, for condition 1b to guarantee our optimum NE. The contrapositive of the condition 1b implies that if the biggest player is able to make a project succeed on her own, then the threshold should be small enough so that p is at least the number of the most profitable projects, for our optimum NE to be guaranteed by condition 1a.

There exists no simple characterization for the NE existence when $\delta \leq \alpha_m B_n$.

Proposition 2. *For any success threshold $\delta \in [0, \alpha_m B_n]$ and any number of agents $n \geq 2$ and projects $m \geq 2$, there exists a game which possesses an NE, and a game which does not.*

The proof appears in Section A.

5 Conclusions and Further Research

We analyze the stable investments in projects, where a project has to comply to certain requirements to obtain its value. This models paper co-authorship, investment in firms, etc. The goal is to advise which investments are individually and socially preferable. Each agent freely divides her budget of time or effort between the projects. A project that succeeds in the competition obtains a value, which is divided between the contributors who have contributed at least a given fraction of the maximum contribution to the project. We model succeeding in a competition either by a quota of projects that actually obtain their value, or by a success threshold on the value of projects that do.

For purposes like organizing a conference, we ask which quota or success threshold would make the behavior of the players better for the social welfare. Theorem 1 implies that if no player has a budget as large as the total budget of all the other players times the ratio between the least and the largest project coefficient, then the quota of 1 makes many equilibria, including an optimal one, possible. Theorem 4 promises the same by choosing a success threshold that disables any player to make a project successful on her own. The first problem of this approach is that it also allows very inefficient profiles constitute equilibria, asking for some coordination. The second problem is that the discussed equilibria have all the players investing in the same project, which is understandable because of the linear project functions but practically unreasonable in conferences, though possible in other applications, such as sponsorship of large projects like Uber, Lyft, Facebook and VKontakte.

Comparing these models, we see from Theorems 1 and 4 that the success threshold allows ensuring that there exists a socially optimal equilibrium while the quota requires also assuming that the largest effort budget is less than the sum of the other ones times the ratio of the least to the most profitable project

coefficients. In addition, comparing Theorems 3 and 6 shows that provided the smallest budget is at least a certain fraction of the largest one, choosing large enough a threshold or small enough a quota guarantees that an optimal profile will be an equilibrium. Unlike in the described cases, where success threshold seems stronger than quota, we notice that the second part of Theorem 6 actually contains an additional condition, relatively to the second part of Theorem 3, but since the second parts of these theorems refer to the case of a single agent being able to dominate everyone everywhere and all the projects being equally rewarding, this is less practical. To conclude the comparison, sometimes, choosing success threshold has more power, since choosing quota needs to assume an additional relation between the budgets, in order to guarantee that socially optimal equilibria exist. Intuitively, this stems from a quota needing an assumption on what the players are able to do to increase their utility, given the quota, while providing a success threshold can be done already with the budgets in mind.

Both a quota and a success threshold have a concentrating effect: equilibria where the agents contribute to less projects than without any of these conditions.

Many directions to expand the research exist. First, some common projects like papers and books have an upper bound on the maximal number of participants. Also a person has an upper bound on the maximal number of projects she can contribute to. The model should account for these bounds. Second, competition can be of many sorts. For instance, a project may need to have a winning coalition of contributors, in the sense of cooperative games. The fate of the projects that fail the competition can also vary; for example, their value can be distributed between the winning projects. We have extended the sufficiency results for existence from [13], and proven the necessity to be harder for analytical analysis. Simulations or other analytical approaches may be tried to delineate the set of Nash equilibria more clearly. Naturally, project functions do not have to be linear, so there is a clear need to model various non-linear functions. Such a more general model will make the conclusions on scientific investments, paper co-authorship, and the many other application domains more precise, and enable us to further improve the advice to participants as well as organizers. We can look at submitting a paper to a highly-ranked conference and reducing the conference level till the paper gets accepted as on a series of shared effort games with various quotas, success thresholds and participants. If we model the cost of each submission, then the question is to which conference to submit first.

Acknowledgments. This work has been supported by the project SHINE, the flagship project of DIRECT (Delft Institute for Research on ICT at Delft University of Technology). We thank anonymous reviewers for their comments.

References

1. Anshelevich, E., DasGupta, A., Kleinberg, J., Tardos, E., Wexler, T., Roughgarden, T.: The price of stability for network design with fair cost allocation. In: Foundations of Computer Science, 2004. Proceedings. 45th Annual IEEE Symposium on. pp. 295–304 (Oct 2004)

2. Anshelevich, E., Hoefer, M.: Contribution Games in Social Networks, pp. 158–169. Springer Berlin Heidelberg, Berlin, Heidelberg (2010)
3. Bachrach, Y., Syrgkanis, V., Vojnovic, M.: Efficiency and the Redistribution of Welfare. Tech. Rep. May, Microsoft Reserach (2012)
4. Batchelor, G.K.: Preoccupations of a journal editor. *Journal of Fluid Mechanics* 106, 1–25 (5 1981)
5. Baye, M.R., Kovenock, D., de Vries, C.G.: The all-pay auction with complete information. *Economic Theory* 8(2), 291–305 (1996)
6. Bernstein, P.A., DeWitt, D., Heuer, A., Ives, Z., Jensen, C.S., Meyer, H., Özsu, M.T., Snodgrass, R.T., Whang, K.Y., Widom, J.: Database publication practices. In: *Proceedings of the 31st International Conference on Very Large Data Bases*. pp. 1241–1245. VLDB '05, VLDB Endowment (2005)
7. Douceur, J.R.: Paper rating vs. paper ranking. *SIGOPS Oper. Syst. Rev.* 43(2), 117–121 (Apr 2009)
8. Ganti, R., Ye, F., Lei, H.: Mobile crowdsensing: current state and future challenges. *IEEE Communications Magazine* 49(11), 32–39 (Nov 2011)
9. Harper, F.M., Li, S.X., Chen, Y., Konstan, J.A.: Social comparisons to motivate contributions to an online community. In: *Proceedings of the 2nd international conference on Persuasive technology*. pp. 148–159. PERSUASIVE'07, Springer-Verlag, Berlin, Heidelberg (2007)
10. Kleinberg, J., Oren, S.: Mechanisms for (mis)allocating scientific credit. In: *Proceedings of the 43rd annual ACM symposium on Theory of computing*. pp. 529–538. ACM, New York, NY, USA (2011)
11. Koutsoupias, E., Papadimitriou, C.: Worst-case equilibria. In: 16th Ann. Symp. on Theoret. Asp. of Computer Science. pp. 404–413. Trier, Germany (4–6 Mar 1999)
12. Papadimitriou, C.: Algorithms, games, and the internet. In: *Proceedings of the Thirty-third Annual ACM Symposium on Theory of Computing*. pp. 749–753. STOC '01, ACM, New York, NY, USA (2001)
13. Polevoy, G., Trajanovski, S., de Weerd, M.M.: Nash equilibria in shared effort games. In: *Proceedings of the 2014 International Conference on Autonomous Agents and Multi-agent Systems*. pp. 861–868. AAMAS '14, International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC (2014)
14. Roberson, B.: The Colonel Blotto game. *Economic Theory* 29, 1–24 (2006)
15. Schulz, A.S., Moses, N.S.: On the performance of user equilibria in traffic networks. In: *Proceedings of the Fourteenth Annual ACM-SIAM Symposium on Discrete Algorithms*. pp. 86–87. SODA '03, Society for Industrial and Applied Mathematics, Philadelphia, PA, USA (2003)
16. Shapley, L.S.: A value for n-person games. *Contribution to the Theory of Games. Annals of Mathematics Studies* 2, 28 (1953)

A Omitted Proofs

We now prove Theorem 6.

Theorem 6. *Consider an equal θ -sharing game with $n \geq 2$ players with budgets $B_n \geq \dots \geq B_2 \geq B_1$, $0 < \theta < 1$ (the order is w.l.o.g.), linear project functions with coefficients $\alpha_m = \alpha_{m-1} = \dots = \alpha_{m-k+1} > \alpha_{m-k} \geq \alpha_{m-k-1} \geq \dots \geq \alpha_1$ (the order is w.l.o.g.).⁸, and success threshold δ . Define $p \triangleq \left\lfloor \frac{\alpha_m \sum_{i \in N} B_i}{\delta} \right\rfloor$, as in Theorem 5.*

⁸ If α_{m-k} does not exist, consider the containing condition to be vacuously true.

1. If at least one of the following holds.
 - (a) $B_1 \geq k\theta B_n$, $k \leq p$ and $\frac{1}{n}\alpha_{m-k+1} \geq \alpha_{m-k}$,
 - (b) $B_1 \geq p\theta B_n$, $k \geq p \geq 1$ and $\alpha_m B_n < \delta$;
 Then, there exists a pure NE and there holds: $\text{PoS} = 1$.
2. Assume $B_{n-1} < \frac{\theta}{|\Omega|} B_n$, all the project functions are equal, i.e. $\alpha_m = \alpha_1$.
 Then, there exists a pure NE and $\text{PoS} = 1$. If, in addition, $\alpha_m B_n \geq \delta$, then

$$\text{PoA} = \frac{B_n}{\sum_{i \in \{1, 2, \dots, n\}} B_i}.$$

Proof. We first prove parts 1a and 1b. According to proof of parts 1 and 2 in Theorem 5, equally dividing all the budgets among $\min\{k, p\}$ steep projects is an NE. Therefore, $\text{PoS} = 1$.

Part 2 is proven as follows. Since all the players dividing their budgets equally between any $\min\{p, m\}$ projects constitutes an NE, we have $\text{PoS} = 1$.

To treat the PoA, we define the number of projects player n can make accepted on her own, $r \triangleq \lfloor \alpha_m \frac{B_n}{\delta} \rfloor$, and distinguish between the case where $m \leq r$ and $m > r$. If $m \leq r$, consider the profile where player n divides her budget equally between all the projects, while the other players contribute nothing at all. This is an NE, because all the projects are accepted, player n cannot increase her profit and any other player will be suppressed, if she contributes anything anywhere. On the other hand, if $m > r$, consider the profile where player n divides her budget equally between $m, m-1, \dots, m-r+1$, while the other players contribute nothing at all. The only possible deviation is player $j < n$ contributing to a vacant project. However, we have $B_j \leq B_{n-1} < \frac{\theta}{|\Omega|} B_n < \theta\delta/\alpha_m \leq \delta/\alpha_m$. This means that the project would be unaccepted. Therefore, this is an NE.

Therefore, $\text{PoA} \leq \frac{\alpha_m B_n}{\alpha_m (\sum_{i \in N} B_i)}$. Since $\alpha_m B_n \geq \delta$, in any NE, player n receives at least $\alpha_m B_n$, and therefore, $\text{PoA} = \frac{B_n}{\sum_{i \in \{1, 2, \dots, n\}} B_i}$. ■

We finally prove Proposition 2.

Proposition 2. For any success threshold $\delta \in [0, \alpha_m B_n]$ and any number of agents $n \geq 2$ and projects $m \geq 2$, there exists a game which possesses an NE, and a game which does not.

Proof. For $\delta = 0$, which means for no threshold, the theorem follows from Theorem 3 from [13]. Therefore, we assume henceforth a positive success threshold.

A game that satisfies the conditions of Theorem 5 provides an example of the existence. Notice that the p they define is positive, since $\delta \leq \alpha_m B_n$.

To find a game that does not possess an equilibrium, let $\alpha_m = \alpha_1$ and let

$$B_n > \sum_{i=1}^{n-1} B_i, \quad (4)$$

$$B_1 = \dots = B_{n-1} = \theta B_n \text{ and } \delta = \alpha B_n. \quad (5)$$

Because of the equality of all the project coefficients, of (4) and of the choice of the success threshold, in an equilibrium, all the agents with budgets at least θB_n (which are $1, \dots, B_{n-1}$ here) will be together with n on the same single project. Then, agent n will deviate, contradictory to being in an equilibrium. ■

Refining a Heuristic for Constructing Bayesian Networks from Structured Arguments

Remi Wieten¹ ✉, Floris Bex¹, Linda C. van der Gaag¹, Henry Prakken^{1,2}, and Silja Renooij¹

¹ Department of Information and Computing Sciences, Utrecht University, NL

² Faculty of Law, University of Groningen, NL

✉ Corresponding author: g.m.wieten@uu.nl

Abstract. Recently, a heuristic was proposed for constructing Bayesian networks (BNs) from structured arguments. This heuristic helps domain experts who are accustomed to argumentation to transform their reasoning into a BN and subsequently weigh their case evidence in a probabilistic manner. While the underlying undirected graph of the BN is automatically constructed by following the heuristic, the arc directions are to be set manually by a BN engineer in consultation with the domain expert. As the knowledge elicitation involved is known to be time-consuming, it is of value to (partly) automate this step. We propose a refinement of the heuristic to this end, which specifies the directions in which arcs are to be set given specific conditions on structured arguments.

Keywords: Bayesian networks · Structured argumentation

1 Introduction

In recent years, efforts have been made to gain a better understanding of the relation between different normative frameworks for evidential reasoning, such as argumentative and probabilistic approaches [9]. Argumentative approaches are particularly suited for adversarial settings, where arguments for and against a specific conclusion are constructed from evidence. The inferences which are used to draw conclusions from evidence are generally defeasible, in that the conclusion of an argument does not universally hold given the evidence. Arguments can be attacked by other arguments; it can then be established which arguments are accepted and which are rejected. In current argumentative approaches, however, there is no emphasis on incorporating graded uncertainty.

In contrast, probabilistic approaches are well suited for handling graded uncertainty. In particular, Bayesian networks (BNs) [2, 3] are powerful tools to this end. BNs are compact graphical models of joint probability distributions, which allow for evidence evaluation by calculating the probability of the truth of a proposition of interest. However, BNs are generally difficult to construct; in fact, they are often constructed by modelers with the relevant mathematical background, called BN engineers, in consultation with a domain expert.

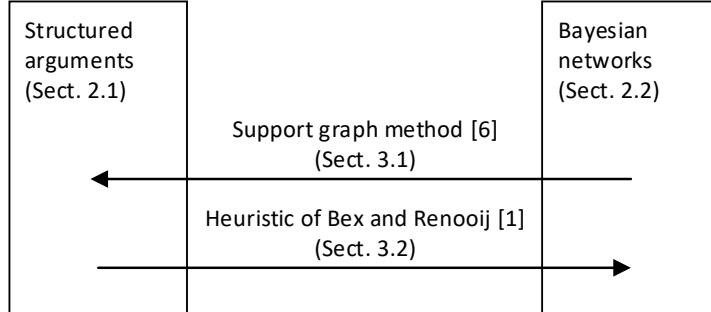


Fig. 1. Outline of Sects. 2 and 3 of this paper.

Recently, a heuristic for constructing BNs from structured arguments was proposed by Bex and Renooij [1]; in this paper, the heuristic will be referred to as the BR heuristic. The heuristic helps domain experts who are more accustomed to argumentation to transform their reasoning into a BN (cf. Fig. 1) and subsequently weigh their case evidence in a probabilistic manner. The focus of the BR heuristic lies on obtaining the graphical structure of the BN, called the BN graph, which captures the independence relations between the domain variables. While the underlying undirected graph, or skeleton, of the BN graph can be automatically constructed by following the BR heuristic, the heuristic prescribes that the arc directions should be set manually by a BN engineer in consultation with a domain expert. Although the heuristic further suggests that the commonly used notion of causality be taken as a guiding principle [3], the resulting graph still has to be verified and refined in terms of the independence relations it represents. This type of knowledge elicitation is known to be time-consuming [7], however, and moreover needs to be repeated for every adjustment to the original arguments. As a consequence, letting arc directions be set by a BN engineer is practically infeasible in investigative contexts such as police investigations, where evidence changes dynamically. It is, therefore, of value to investigate whether the process of setting arc directions can be (partly) automated.

Accordingly, in this paper we propose a refinement of the BR heuristic, which specifies the directions in which the arcs should be set in a BN graph under specific conditions on structured arguments. These conditions are identified by applying a method called the support graph method [6]. This method essentially works in the opposite direction of the BR heuristic, in that structured arguments are constructed from BNs (cf. Fig. 1). By applying the support graph method to BN graphs obtained with the BR heuristic, it is determined whether and under which conditions the original arguments are re-obtained. If the original arguments are not re-obtained from the thus constructed BN graph, it may be concluded that this graph represents the original arguments in a different, possibly incorrect, way. Our refinement of the BR heuristic now ensures that BN graphs from which the original arguments are not returned by the support graph method are not constructed.

The paper is structured as follows. Sections 2 and 3 provide some preliminaries on structured argumentation, BNs, the support graph method and the BR heuristic. In Sect. 4, our refinement to the BR heuristic is proposed, based on observations from applying the support graph method. In Sect. 5, our findings are summarized and possible directions for future research are discussed.

2 Preliminaries

In this section, structured argumentation and BNs are briefly reviewed.

2.1 Structured Argumentation

A simplified version of the ASPIC+ framework for structured argumentation [4] is assumed throughout this paper. Let \mathcal{L} be a non-empty propositional literal language with the unary negation symbol \neg . Informally, \mathcal{L} contains the basic elements which can be argued about. Given a knowledge base $\mathcal{K} \subseteq \mathcal{L}$ of premises, arguments are constructed by chaining inference rules. These rules are defined over \mathcal{L} and are defeasible, in that the conclusion of a defeasible rule does not universally hold given the premises, in contrast with the strict inferences of classical logic. Let \mathcal{R} be a set of defeasible inference rules of the form $d: \phi_1, \dots, \phi_n \Rightarrow \phi$, where ϕ_1, \dots, ϕ_n and ϕ are meta-variables ranging over well-formed formulas in \mathcal{L} . An argument A is then either: (1) ϕ if $\phi \in \mathcal{K}$, where the conclusion of the argument A , denoted by $\text{CONC}(A)$, is equal to ϕ ; or (2) $A_1, \dots, A_n \Rightarrow \phi$ with $\phi \in \mathcal{L} \setminus \mathcal{K}$, where A_1, \dots, A_n are arguments such that there exists a rule $\text{CONC}(A_1), \dots, \text{CONC}(A_n) \Rightarrow \phi$ in \mathcal{R} . In the first case, $\text{CONC}(A)$ is an element from the knowledge base, while in the second case, $\text{CONC}(A)$ follows by applying a defeasible rule to the conclusion(s) of arguments A_1, \dots, A_n , which are called the immediate sub-arguments of A . Generally, a sub-argument of an argument A is either A itself or an argument that is (iteratively) used to construct A . The smallest set of finite arguments which can be constructed from \mathcal{L} , \mathcal{K} and \mathcal{R} is denoted by \mathcal{A} . An argument graph of \mathcal{A} then graphically displays the arguments in \mathcal{A} and their sub-arguments. Fig. 3a shows an example of an argument graph.

The general ASPIC+ framework further includes the notion of attack. Informally, an argument in \mathcal{A} is attacked on one of its non-premise sub-arguments by another argument in \mathcal{A} with the opposite conclusion of that sub-argument. Due to space limitations, the focus of the current paper lies on argument structures without attack relations.

2.2 Bayesian Networks

BNs [3] are graphical probabilistic models which are being applied in many different fields, including medicine and law [2]. A BN is a compact representation of a joint probability distribution $\Pr(\mathbf{V})$ over a finite set of discrete random variables \mathbf{V} . The random variables are represented as nodes in a directed acyclic graph G , where each node¹ can take one of a number of mutually exclusive and exhaustive values; in this paper, we assume all nodes to be Boolean. A node A is a parent of another node B , called the child, in G if G contains an arc from A to B . The BN further includes, for each node, a conditional probability table, or CPT, given its parents; this table specifies the probabilities of the values of the node itself conditioned on the possible joint value combinations of its parents. A node is called instantiated iff it is fixed in a specific value. Given a set of instantiated nodes, conditional probability distributions over the other nodes in the network can be computed using probability calculus [3].

The BN graph captures the independence relations between its variables. Let a chain be defined as a simple path in the underlying undirected graph, or skeleton, of a BN graph. A node V is called a head-to-head node on a chain c if it has two incoming arcs on c . A chain c is blocked iff it includes a node V such that (1) V is an uninstantiated head-to-head node on c without instantiated descendants; (2) V is not a head-to-head node on c and is instantiated. In addition, instantiated end-points of the chain c , that is, instantiated nodes with at most one incoming or outgoing arc on c , serve to block the chain [5]. A chain is inactive if it is blocked; otherwise it is called active. Two nodes $A \neq B$ are called d-separated by a set of nodes \mathbf{Z} if no active chains exist between A and B given instantiations of nodes in \mathbf{Z} . If two nodes are d-separated by \mathbf{Z} , then they are considered conditionally independent given \mathbf{Z} . We note that conditional independence thereby depends on the set of instantiated nodes [8].

An immorality in a BN graph is defined as a triple of nodes (A, B, C) , where A and C are parents of B that are not directly connected by an arc. Two BNs are said to be Markov equivalent iff they share the same skeleton and immoralities. Markov equivalent networks constitute an equivalence class, for which Verma and Pearl [10] proved that any two elements represent the same independence relations over the variables involved. Arcs between nodes that are not involved in an immorality can thus be reversed without changing the represented independence relations as long as no new immoralities arise. Immoralities derive their importance from providing for intercausal reasoning [11]. Specifically, if the head-to-head node involved in an immorality is instantiated, an active chain arises between the parents of the node. These parents can be seen as different causes of the same effect modeled by the head-to-head node. If one of the causes is now observed, then the probability of the other cause being present as well can either increase, decrease or stay the same upon updating, depending on the probabilities in the CPT of the head-to-head node.

¹ The terms ‘node’ and ‘variable’ are used interchangeably.

3 Two Methods for Translating between Structured Arguments and Bayesian Networks

In this section, the support graph method [6] and the BR heuristic [1] are reviewed; the support graph method is used to build structured arguments from BNs, while the BR heuristic is used to construct BN graphs from structured arguments.

3.1 The Support Graph Method

The support graph method, proposed by Timmer and colleagues [6], is a two-phase method for constructing argument structures from BNs. The method allows domain experts who are not familiar with BNs but are accustomed to argumentation to understand the knowledge and reasoning patterns captured by a BN. To this end, the method summarizes all reasoning chains from a set of evidence to a conclusion in a given BN.

In the first phase of the method, a directed graph called the support graph (SG) is constructed from a BN given a variable of interest V^* ; in this SG, all reasoning chains in the BN ending in V^* are captured. The SG does not depend on specific instantiations, and can thus be re-used to build argument structures for different evidence. An SG is iteratively constructed, starting with a graph containing only V^* . New parents are added to existing nodes in the SG as new inference steps are identified in the BN. Three types of inference step are distinguished: (1) an inference step along an arc from a parent to a child; (2) an inference step along an arc from a child to a parent; and (3) an inference step between two parents in an immorality. The last type directly accommodates intercausal reasoning steps which occur between the parents of an immorality, and summarizes the inference from one parent of an immorality to another parent via the common child. In the constructed SG, V^* is the only node without children; every other node in the SG is an ancestor of V^* .

In the second phase of the support graph method, arguments are constructed from the SG for a given set of node instantiations. Given this evidence, the SG is pruned such that only paths remain that start in an instantiated node. From the thus pruned graph, arguments are constructed as follows. The logical language \mathcal{L} is taken to consist of all literals which correspond to the values of the nodes in the BN; two literals $\phi, \psi \in \mathcal{L}$ negate each other iff ϕ and ψ correspond with the different values of the same node. Given the evidence, the knowledge base \mathcal{K} consists of those literals in \mathcal{L} that correspond with the values of the instantiated nodes. The defeasible rules in \mathcal{R} are of the form $(N_1, o_1), \dots, (N_k, o_k) \Rightarrow (N, o)$, where N_1, \dots, N_k are parents of the node N in the pruned SG and o_1, \dots, o_k, o are values of these nodes. From \mathcal{L} , \mathcal{K} , and \mathcal{R} , a set of arguments \mathcal{A} is then constructed.

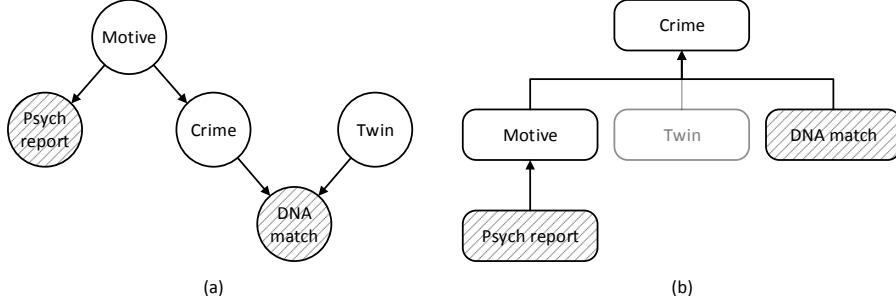


Fig. 2. A BN graph (a) and the corresponding SG for the variable of interest *Crime* (b); *Twin* is pruned from the SG as only *Psych report* and *DNA match* are instantiated.

Example 1. An example by Timmer and colleagues [6] from the legal domain is reviewed to demonstrate the support graph method. In the example, the BN graph from Fig. 2a² is constructed for a criminal case, in which we are interested in whether the suspect committed the crime, that is, whether $Crime = true$. Evidence for this possible conclusion would be the existence of a motive, which may be mentioned in a psychological report. A match between the suspect's DNA and DNA found at the crime scene would further support the proposition that the suspect committed the crime. This finding might also be explained, however, if the suspect had an identical twin. For the variable of interest *Crime*, the SG of Fig. 2b is obtained; the node *Twin* is directly added as a parent of *Crime*, as the triplet $(Crime, DNA\ match, Twin)$ is an immorality in the BN graph. The literals in \mathcal{L} are the possible values of all nodes in the BN graph, that is, \mathcal{L} contains $crime, \neg crime, motive, \neg motive, \dots$. Now, if we assume that *Psych report* and *DNA match* are instantiated with the value *true* conform available evidence, and *Twin* is not instantiated, then the path starting at the node *Twin* is pruned from the SG. The knowledge base \mathcal{K} then consists of *psych report* and *dna match*. Among the defeasible rules extracted from the pruned SG are $d_1: psych\ report \Rightarrow motive$ and $d_2: dna\ match, motive \Rightarrow crime$. The arguments $A_1: psych\ report$, $A_2: dna\ match$, $A_3: A_1 \Rightarrow motive$, and $A_4: A_2, A_3 \Rightarrow crime$ can then be constructed. Also the rules $d_3: psych\ report \Rightarrow \neg motive$ and $d_4: dna\ match, \neg motive \Rightarrow \neg crime$ are extracted from the SG, from which arguments $A_5: A_1 \Rightarrow \neg motive$ and $A_6: A_2, A_5 \Rightarrow \neg crime$ are constructed. These arguments have opposite conclusions of A_3 and A_4 . \square

It should be noted that, when using the support graph method, the reasons pro and con a given conclusion are not distributed over separate arguments, as is usual in argumentation, but are instead encapsulated in a single argument. That is, all literals that are relevant for a specific proposition are taken as the premises

² In figures in this paper, circles are used in BN graphs, rectangles are used in argument graphs and rounded rectangles are used in SGs. Nodes and propositions corresponding to evidence are shaded. Capital letters are used for the nodes in BN graphs and SGs, and lowercase letters are used for propositions.

of an argument for that proposition, which reflects the way in which Bayesian networks internally weigh all evidence.

For every argument that is returned from a BN by the support graph method, the method also returns an argument with the same ‘structure’ but with the opposite conclusion. Timmer and colleagues [6] employ a quantitative step to filter the set of arguments returned. As in the current paper the focus lies on the graphical structures of BNs and not on the modeled probability distribution, this quantitative step is not further discussed here.

3.2 The BR Heuristic for Constructing Bayesian Networks from Structured Arguments

Bex and Renooij [1] have proposed the BR heuristic for constructing BN graphs from structured arguments. This heuristic allows domain experts who are accustomed to argumentation to translate their reasoning expressed as arguments into a BN graph. This graph is then supplemented with CPTs to arrive at a fully specified BN for probabilistic inference over the original arguments. Focusing on argument structures in which no attack relations are present, from a given set of arguments \mathcal{A} constructed from a logical language \mathcal{L} , knowledge base \mathcal{K} , and a set of defeasible rules \mathcal{R} , BN graphs are constructed as follows:

1. For every proposition $\phi \in \mathcal{L}$ used in \mathcal{A} , the BN graph includes a single node V such that $V = \text{true}$ corresponds to ϕ and $V = \text{false}$ corresponds to $\neg\phi$. For every $e \in \mathcal{K}$, the corresponding node is instantiated at the observed value.
2. For every defeasible rule $d: \phi_1, \dots, \phi_n \Rightarrow \phi \in \mathcal{R}$ used in \mathcal{A} , a set of undirected edges between the node associated with ϕ and each of the nodes associated with ϕ_1, \dots, ϕ_n is created for inclusion in the BN graph.
3. The direction of the edges from the previous step is decided upon by a BN engineer in consultation with the domain expert, where a causal direction is chosen if possible, and an arbitrary direction otherwise. The resulting arcs are inserted in the BN graph.
4. The BN engineer verifies that the graph is acyclic and that all chains that should be active in the graph indeed are; if the graph does not yet exhibit these properties, appropriate arcs are removed or reversed, once more in consultation with the domain expert.

Example 2. A simple example is introduced to demonstrate the BR heuristic. The logical language, knowledge base and defeasible rules involved are $\mathcal{L} = \{p, \neg p, q, \neg q, r, \neg r\}$, $\mathcal{K} = \{p\}$ and $\mathcal{R} = \{p \Rightarrow q; q \Rightarrow r\}$. The constructed arguments are $\mathcal{A} = \{A_1: p; A_2: A_1 \Rightarrow q; A_3: A_2 \Rightarrow r\}$; the argument graph of \mathcal{A} is depicted in Fig. 3a. Following steps 1 and 2 of the BR heuristic, the skeleton of the BN graph corresponding to this argument structure consists of nodes P , Q and R , with undirected edges between P and Q and between Q and R . Following step 3, one of the BN graphs of Fig. 3b-e is obtained, depending on how the arc directions are set. \square

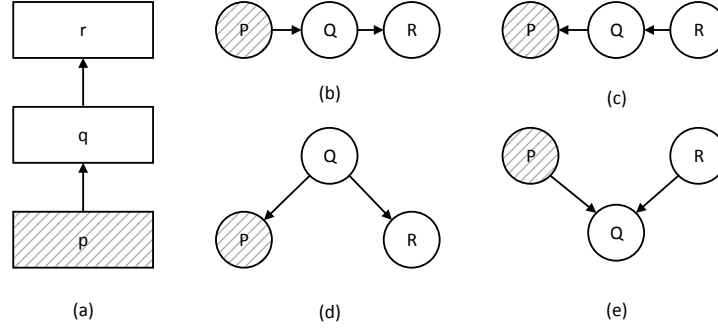


Fig. 3. An argument graph with arguments from p to r via q (a); the four corresponding BN graphs which can be constructed by following the BR heuristic (b-e).

For a given set of arguments \mathcal{A} , the skeleton of the BN graph is automatically constructed by following the first two steps of the BR heuristic. Step 3 then prescribes that the directions of the arcs should be set manually by a BN engineer in consultation with the domain expert, using the notion of causality as a guiding principle (see also [3]). For example, if the domain expert indicates for a defeasible rule $d: p \Rightarrow q$ that p is a typical cause of q , then the arc is set from node P to node Q . Since immoralities can result from following this guiding principle, the independence relations in the constructed BN graph should be verified manually, as prescribed by step 4 of the BR heuristic. This type of knowledge elicitation and verification is known to be a time-consuming and error-prone process in general [7]. Especially for larger or more densely connected BN graphs, it quickly becomes infeasible to verify all independence relations manually, as all possible chains for all possible combinations of instantiated variables need to be investigated. Moreover, the elicitation and verification needs to be repeated for every adjustment to the original argument graph. As this step is practically infeasible in investigative contexts, such as police investigations, in which the evidence for a case changes dynamically, the arc directions are preferably set (semi-)automatically.

4 Refining the BR Heuristic

We propose a refinement of step 3 of the BR heuristic, which specifies the directions in which arcs should be set in a BN graph under specific conditions on structured arguments. These conditions are identified from applying the support graph method. To this end, the arguments to which the BR heuristic is applied are compared to the arguments returned by the support graph method when applied to a BN graph constructed by steps 1-3 of the BR heuristic. In order to apply the support graph method, a variable of interest has to be chosen. In this paper, we assume that there is a single ultimate conclusion in the input argument graph, that is, a single argument that is not an immediate sub-argument of another argument. The node corresponding to this ultimate conclusion is taken

as the node of interest. We further assume that the input arguments for the BR heuristic are linked, in the sense that all premises relevant for a conclusion are encapsulated in a single argument; Fig. 5a shows an example of an argument graph with linked arguments only. Linked argument graphs are similar to the type of argument graphs that are returned by the support graph method.

When applying the support graph method to a BN graph constructed by steps 1-3 of the BR heuristic, a set of arguments is returned. This set may be different from the set of arguments that was used as input for the heuristic. As measures for the differences found, we distinguish between recall and precision, which for a given BN graph respectively measure the proportion of original arguments returned and the proportion of additional arguments returned. Formally, let \mathcal{A} be the set of input arguments for the BR heuristic, let \mathbf{B} be a BN graph constructed from \mathcal{A} by steps 1-3 of the heuristic, and let \mathcal{A}' be the set of arguments returned from \mathbf{B} by the support graph method. We define the recall and precision of \mathbf{B} as follows:

- $Recall(\mathbf{B}) = |\mathcal{A} \cap \mathcal{A}'|/|\mathcal{A}|$
- $Precision(\mathbf{B}) = |\mathcal{A} \cap \mathcal{A}'|/|\mathcal{A}'|$

where \mathbf{B} has maximum recall and precision if these fractions are equal to 1.

In Sect. 4.1, we propose a refinement of the third step of the BR heuristic, which serves to increase the recall of constructed BN graphs. In Sect. 4.2, we address precision. As argued before, Timmer and colleagues [6] propose a quantitative step for filtering the set of arguments returned by the support graph method, which suggests that for improving the precision of constructed BNs, the CPTs need to be taken into account. As in this paper, the focus lies on the graphical structure of a BN, we propose a further refinement of the third step of the BR heuristic based on graphical considerations only.

4.1 Refining the BR Heuristic to Improve Recall

To illustrate how the BR heuristic can be refined such that BN graphs with higher recall are constructed, we revisit Example 2 from Sect. 3.2. By applying steps 1-3 of the heuristic to the argument graph of Fig. 3a, four possible BN graphs over the nodes P , Q and R were constructed, as shown in Figs. 3b-e. These graphs fall into two Markov equivalence classes; the first class consists of the BN graphs of Figs. 3b-d, and the second class consists of the graph of Fig. 3e. Timmer and colleagues [6] proved that for two Markov equivalent BNs and the same node of interest, the same SG is obtained. By applying the support graph method for the node of interest R , we now show that the recall of the original arguments from the BN graph in the second equivalence class is lower than that of the BN graphs in the first class. Since the logical language and knowledge base of the argument structure returned by the support graph method are derived from the BN skeleton, $\mathcal{L}'^3 = \{p, \neg p, q, \neg q, r, \neg r\}$ and $\mathcal{K}' = \{p\}$ are the same

³ The prime symbol is used to denote objects which result from applying the support graph method.

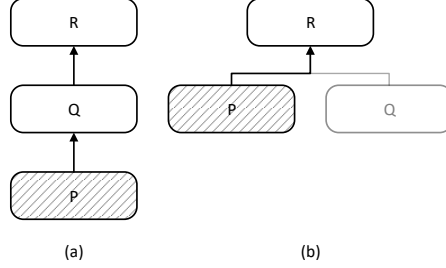


Fig. 4. The (pruned) SG obtained from the BN graphs of Figs. 3b-d (a), and the SG obtained from the BN graph of Fig. 3e (b), where Q is pruned as only P is instantiated.

for all four BN graphs. For the graphs in the first equivalence class, the SG of Fig. 4a is obtained. The defeasible rules of the returned argument structure correspond to the arcs of this SG, that is, $\mathcal{R}' = \{p \Rightarrow q; p \Rightarrow \neg q; \neg p \Rightarrow q; \neg p \Rightarrow \neg q; q \Rightarrow r; q \Rightarrow \neg r; \neg q \Rightarrow r; \neg q \Rightarrow \neg r\}$. As $\mathcal{L} \subseteq \mathcal{L}'$, $\mathcal{K} = \mathcal{K}'$ and $\mathcal{R} \subseteq \mathcal{R}'$, all original arguments $A_1, A_2, A_3 \in \mathcal{A}$ are re-obtained from the SG. Therefore, the BN graphs of Figs. 3b-d have maximal recall.

For the BN graph in the second equivalence class, the SG of Fig. 4b is constructed. In this SG, node P is a direct parent of R and not of Q , as (P, Q, R) is an immorality. We recall that an SG is meant for constructing arguments for different sets of evidence. In the example, where just P is instantiated, node Q is pruned from the SG. The defeasible rules corresponding to this pruned SG are $\mathcal{R}' = \{p \Rightarrow r; p \Rightarrow \neg r; \neg p \Rightarrow r; \neg p \Rightarrow \neg r\}$ and the arguments which can be constructed are $A_1: p$, $A_2': A_1 \Rightarrow r$ and $A_2'': A_1 \Rightarrow \neg r$. Timmer and colleagues [6] employ a quantitative step using the CPTs from the original BN to filter the set of constructed arguments; by this step, arguments A_2' and A_2'' are filtered out, as P and R are independent given that Q is not instantiated. The original arguments A_2 and A_3 are not returned by the support graph method. The recall of the BN graph from Fig. 3e is $\frac{1}{3}$, which is lower than that of the BN graphs in the first equivalence class. It therefore seems desirable to prohibit construction of this BN graph when using the BR heuristic.

Generalizing from the example, let $A_1, \dots, A_n \in \mathcal{A}$, where A_i is an immediate sub-argument of A_{i+1} for all $i \in \{1, \dots, n-1\}$, let $\text{CONC}(A_i) = p_i$, $p_1 \in \mathcal{K}$, and let p_n be the ultimate conclusion of the argument graph of \mathcal{A} . Further assume that no immorality (P_{i-1}, P_i, P_{i+1}) is formed for $i \in \{2, \dots, n-1\}$ by steps 1-3 of the BR heuristic. As no immoralities (P_{i-1}, P_i, P_{i+1}) are present for $i \in \{2, \dots, n-1\}$, upon constructing the SG for the node of interest P_n parents are added iteratively, that is, P_{n-1} is added as a parent of P_n , \dots , P_1 is added as a parent of P_2 . As P_1 corresponds to an instantiated variable, the path starting in P_1 is not pruned from the SG. The support graph method, therefore, returns the arguments A_1, \dots, A_n , and the recall is maximal. On the other hand, if for a given $i \in \{2, \dots, n-1\}$ an immorality (P_{i-1}, P_i, P_{i+1}) would be formed by steps 1-3 of the BR heuristic, then an SG would result in which P_{i+1} is an ancestor of

P_n . As P_{i-1} is directly added as a parent of P_{i+1} , the argument A_i would not be returned, and the recall would not be maximal.

Based on the above observations, the following refinement of step 3 of the BR heuristic is proposed:

- 3'. Let $A_1, \dots, A_n \in \mathcal{A}$, where A_i is an immediate sub-argument of A_{i+1} for any $i \in \{1, \dots, n-1\}$ and where $\text{CONC}(A_i) = p_i$. Then, the directions of the arcs are set such that no immoralities (P_{i-1}, P_i, P_{i+1}) are formed for any $i \in \{2, \dots, n-1\}$. Taking this constraint into account, the directions of the (remaining) arcs are set by a BN engineer in consultation with the domain expert, where a causal direction is chosen if possible.

4.2 A Further Refinement of the BR Heuristic

While in the previous section, simple chains in an argument structure were shown to be best translated in the BN graph by a chain without any immoralities, we now focus on argument structures that do enforce immoralities in the BN graph and propose a further refinement of the refined third step of the heuristic.

Example 3. We consider the linked argument graph of Fig. 5a. The logical language, knowledge base and defeasible rules involved are $\mathcal{L} = \{p, \neg p, q, \neg q, r, \neg r, s, \neg s, t, \neg t\}$, $\mathcal{K} = \{p, q\}$, and $\mathcal{R} = \{p \Rightarrow r; p, q \Rightarrow s; r, s \Rightarrow t\}$; the constructed arguments are $\mathcal{A} = \{A_1: p; A_2: A_1 \Rightarrow r; A_3: q; A_4: A_1, A_3 \Rightarrow s; A_5: A_2, A_4 \Rightarrow t\}$.

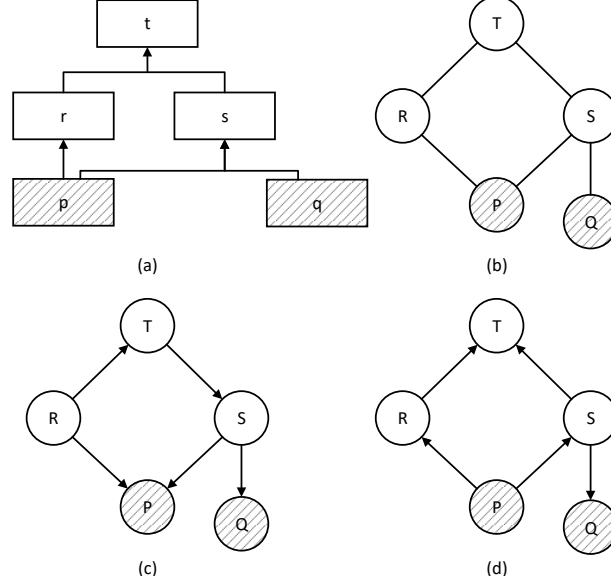


Fig. 5. An argument graph (a) and the corresponding BN skeleton that is constructed by the BR heuristic (b); a corresponding BN graph with the immorality (R, P, S) (c) and a BN graph with the immorality (R, T, S) (d).

Steps 1 and 2 of the BR heuristic result in the BN skeleton of Fig. 5b. In order to obtain an acyclic directed graph from this skeleton, at least one immorality has to be created in the subgraph induced by the nodes P , R , S and T .

According to the refined third step of the BR heuristic, an immorality (T, R, P) should not be formed, as $A_1: p$ is an immediate sub-argument of $A_2: A_1 \Rightarrow r$, which in turn is an immediate sub-argument of $A_5: A_2, A_4 \Rightarrow t$. Similarly, the immorality (T, S, P) should not be formed. Now, the equivalence class of BN graphs is considered which includes just the immorality (R, P, S) ; the BN graph depicted in Fig. 5c is an element of this class. With T as the node of interest, the SG of Fig. 6 is obtained from this graph. The logical language and knowledge base corresponding to this SG are $\mathcal{L}' = \{p, \neg p, q, \neg q, r, \neg r, s, \neg s, t, \neg t\}$ and $\mathcal{K}' = \{p, q\}$, matching those of the original argument graph. The set of defeasible rules \mathcal{R} corresponding to the SG includes the rules $p, q \Rightarrow s$; $p, s \Rightarrow r$; $p \Rightarrow r$; $p, q, r \Rightarrow s$ and $r, s \Rightarrow t$. Among the arguments which can be constructed from the SG are $A_1: p$, $A_2: A_1 \Rightarrow r$, $A_3: q$, $A_4: A_1, A_3 \Rightarrow s$, $A_5: A_2, A_4 \Rightarrow t$, $A'_2: A_1, A_4 \Rightarrow r$, $A'_4: A_1, A_2, A_3 \Rightarrow s$, and $A'_5: A'_2, A'_4 \Rightarrow t$. While the recall of the BN graphs from Fig. 5c is maximal, the precision is not; more specifically, the returned arguments A'_2, A'_4 and A'_5 were not in the original argument set \mathcal{A} .

Now, the equivalence class of BN graphs with just the immorality (R, T, S) is addressed; the BN graph depicted in Fig. 5d is an element of this class. From this BN graph, again the SG of Fig. 6 is constructed for the node of interest T , and thus the same arguments as above are returned. While the precision of the BN graph of Fig. 5d is equal to that of the BN graph from Fig. 5c, we note that the nodes R and S are conditionally independent given the evidence for $\mathbf{Z} = \{P, Q\}$ in the former graph, that is, in the BN graph with just the immorality (R, T, S) . The immediate sub-argument A_4 of A'_2 and the immediate sub-argument A_2 of A'_4 , therefore, appear to be irrelevant, as the associated reasoning is non-existent in this BN graph. As noted before, Timmer and col-

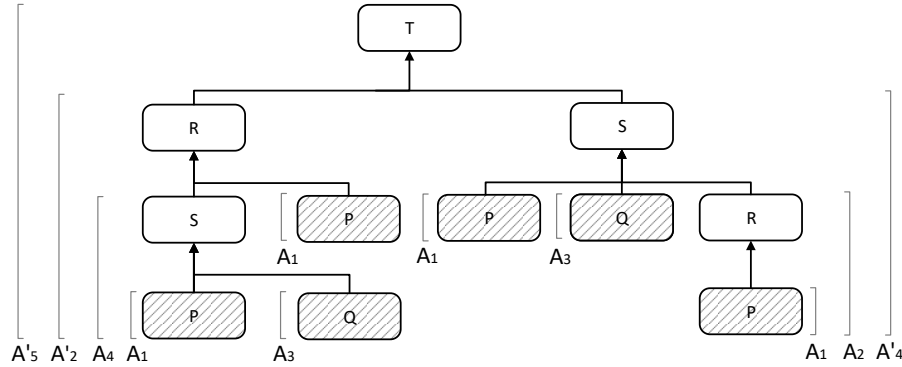


Fig. 6. The SG corresponding to the BN graph of Figs. 5c and 5d, with T as the node of interest; the SG is annotated with some of the possible arguments which can be extracted from it.

leagues [6] employ a quantitative step to filter the set of arguments returned by the support graph method; specifically, as the nodes R and S are conditionally independent given the evidence in the BN graph in Fig. 5d, A_4 and A_2 are filtered out as immediate sub-arguments of A'_2 and A'_4 respectively. Building on the conditional independence relations that can be inferred from the BN graph given the set of instantiated nodes, however, irrelevance of A_4 and A_2 as immediate sub-arguments of A'_2 and A'_4 can be decided upon by graphical considerations only, without involving the CPTs of the nodes. \square

Based on the above example, we propose to set the directions of arcs in a BN skeleton such that no instantiated head-to-head nodes or head-to-head nodes with instantiated descendants are formed, as such head-to-head nodes may introduce unwarranted dependence relations. More specifically, the following refinement of step 3' of the BR heuristic is proposed, which fully specifies the directions of the arcs in a BN graph corresponding to a set of arguments \mathcal{A} :

- 3''. The directions of the arcs in a BN graph are set in the same direction as the arcs in the argument graph, that is, if A is an immediate sub-argument of B , then an arc should be drawn from the node corresponding to $\text{CONC}(A)$ to the node corresponding to $\text{CONC}(B)$.

We note that step 3'' is a further refinement of step 3', as none of the immoralities (P_{i-1}, P_i, P_{i+1}) mentioned in that step are formed if arcs are set in the same direction as in the argument graph. By step 3'', arcs are guaranteed to be set such that head-to-head nodes are not instantiated and do not have instantiated descendants, as the premise arguments in the argument graph, and hence the instantiated nodes in the BN graph, only have outgoing arcs. Finally, we note that step 3'' is not a strict specification of the directions of the arcs in a BN graph; directions can possibly be reversed, given that an element from the same Markov equivalence class as specified by step 3'' is obtained.

5 Conclusion and Future Research

In this paper, we have proposed a refinement of the heuristic of Bex and Renooij [1] for constructing BN graphs from structured arguments. This heuristic is aimed at aiding domain experts who are accustomed to argumentation to transform their reasoning into BNs and subsequently weigh their case evidence in a probabilistic manner. Our refinement consists of fully specifying the directions in which arcs should be set in a BN graph for a given argument structure without attack relations; more specifically, when employing the refined heuristic for a set of arguments \mathcal{A} , the directions of the arcs in the BN graph are set in the same direction as the arcs in the original argument graph of \mathcal{A} . By our refined heuristic, BN graphs with maximal recall are constructed, that is, the original arguments are returned by applying the support graph method to the constructed BN graphs. Furthermore, our refined heuristic prevents the creation of direct intercausal dependence relations between variables in the BN graph that did not

exist between the corresponding propositions in the original argument graph. In the near future, we will evaluate the heuristic in practice by establishing, for example, the extent to which the automatically derived arc directions match the perceived real-world causality or the judgments of domain experts.

In this paper, we focused on improving the recall of BN graphs constructed by the BR heuristic. In our future research, we will address the construction of BN graphs with increased precision. Furthermore, we will extend our research to a more general framework of argumentation [4], not restricting ourselves to linked argument graphs without attack relations.

Bibliography

- [1] F. Bex and S. Renooij. From arguments to constraints on a Bayesian network. In P. Baroni, T.F. Gordon, T. Scheffler, and M. Stede, editors, *Computational Models of Argument: Proceedings of COMMA 2016*, volume 287, pages 95–106. Amsterdam, The Netherlands: IOS Press, 2016.
- [2] N. Fenton and M. Neil. *Risk Assessment and Decision Analysis with Bayesian Networks*. Boca Raton, FL, USA: CRC Press, 2012.
- [3] F.V. Jensen and T.D. Nielsen. *Bayesian Networks and Decision Graphs*. Berlin, Germany: Springer Verlag, 2nd edition, 2007.
- [4] H. Prakken. An abstract framework for argumentation with structured arguments. *Argument & Computation*, 1(2):93–124, 2010.
- [5] R.D. Shachter. A graph-based inference method for conditional independence. In B.D. D’Ambrosio, P. Smets, and P.P. Bonissone, editors, *Proceedings of the Seventh Conference on Uncertainty in Artificial Intelligence*, pages 353–360. San Mateo, CA, USA: Morgan Kaufmann Publishers Inc., 1991.
- [6] S.T. Timmer, J.-J.C. Meyer, H. Prakken, S. Renooij, and B. Verheij. A two-phase method for extracting explanatory arguments from Bayesian networks. *International Journal of Approximate Reasoning*, 80:475–494, 2017.
- [7] L.C. van der Gaag and E.M. Helsper. Experiences with modelling issues in building probabilistic networks. In *International Conference on Knowledge Engineering and Knowledge Management*, pages 21–26. Springer, 2002.
- [8] L.C. van der Gaag and J.-J.C. Meyer. The dynamics of probabilistic structural relevance. In L.C. van der Gaag and J.-J.C. Meyer, editors, *Proceedings of the Eighth Dutch Conference on Artificial Intelligence (NAIC’96)*, pages 145–156. Utrecht, The Netherlands: Utrecht University, 1996.
- [9] B. Verheij, F. Bex, S.T. Timmer, C.S. Vlek, J.-J.C. Meyer, S. Renooij, and H. Prakken. Arguments, scenarios and probabilities: connections between three normative frameworks for evidential reasoning. *Law, Probability & Risk*, 15(1):35–70, 2015.
- [10] T. Verma and J. Pearl. Equivalence and synthesis of causal models. In P.P. Bonissone, M. Henrion, L.N. Kanal, and J.F. Lemmer, editors, *Proceedings of the Sixth Conference on Uncertainty in Artificial Intelligence*, pages 255–270. New York, NY, USA: Elsevier Science Inc., 1991.
- [11] M.P. Wellman and M. Henrion. Explaining “explaining away”. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(3):287–292, 1993.

Reciprocation Effort Games

Gleb Polevoy¹ and Mathijs de Weerd²

¹ University of Amsterdam^{**}, g.polevoy@uva.nl

² Delft University of Technology, m.m.deweerd@tudelft.nl

Abstract. Consider people *dividing their time and effort* between friends, interest clubs, and reading seminars. These are all *reciprocal interactions*, and the reciprocal processes determine the utilities of the agents from these interactions. To advise on efficient effort division, we determine the existence and efficiency of the Nash equilibria of the game of allocating effort to such projects. When no minimum effort is required to receive reciprocation, an equilibrium always exists, and if acting is either easy to everyone, or hard to everyone, then every equilibrium is socially optimal. If a minimal effort is needed to participate, we prove that not contributing at all is an equilibrium, and for two agents, also a socially optimal equilibrium can be found. Next, we extend the model, assuming that the need to react requires more than the agents can contribute to acting, rendering the reciprocation imperfect. We prove that even then, each interaction converges and the corresponding game has an equilibrium.

1 Introduction

In many real-world situations people invest effort in several interactions, such as in discretionary daily activities [16], daily communication between school pupils, sharing files over networks, or in business cooperation. In such an interaction, people tend to reciprocate, i.e., react on the past actions of others (sometimes only if a certain minimum effort is invested) [10, 12]. For example, users of various social networks (Facebook, VKontakte) repeatedly interact in those projects (networks). To recommend how to divide one's limited efforts efficiently, we aim to predict stable strategies for these settings and estimate their efficiency. We study settings with and without a threshold for minimum effort.

Dividing a budget of effort is studied in *shared effort games* [4]. In these games players contribute to various projects, and given their contributions, each project attains a value, which is subsequently divided between the contributors. In order to support decisions regarding individually and publicly good stable strategy profiles in these games, the social welfare (total utility) of strategy profiles is important, and in particular of Nash equilibria (NE). For this, the price of anarchy (PoA) [15], and stability (PoS) [23, 1] are the most famous efficiency measures. The price of anarchy is the ratio of the least social welfare

^{**} Most of this work was done at Delft University of Technology.

in an equilibrium to the optimal social welfare, and the price of stability is the ratio of the social welfare in a best NE to the optimal social welfare.

Bachrach et al. [4] bound the price of anarchy, but only when a player obtains at least a constant share of her marginal contribution to the project’s value; this does not hold for a positive participation threshold. Polevoy et al. [20] have analyzed the Nash equilibria, and price of anarchy and stability also in the case with a threshold. When the threshold is equal to the highest contribution, such shared effort games are equivalent to all-pay auctions. In all-pay auctions, only one contributor benefits from the project. Its equilibria are analyzed by Baye et al. [5] and many others. Anshelevich and Hoefer [2] study graph nodes contributing to edges, which are minimum effort projects. In the literature, the utilities are based on the project values, which are directly defined by the contributions, such as in contributions to online communities, Wikipedia, political campaigns [25], and paper co-authorship [14]. Unlike the existing literature, our paper assumes contributions to the projects determine the interactions, which define utility.

We now review the reciprocation models. Existing models of reciprocation often consider why reciprocation has emerged. The following works consider the emergence of reciprocation. Axelrod [3] studies and motivates direct evolution of reciprocal behavior. Others consider a more elaborate evolution, like Bicchieri’s work on norm emergence [6, Chapter 6] or [27]. Trivers [26] describes how altruism-related emotions like guilt and suspicion have evolved. There exist also other approaches to the nature of reciprocation, such as the *strong reciprocation* [11]. Works like [8, 10, 22] assume the reciprocal behavior and analyze the development of certain interactions, modeling them as appropriate games.

With a model inspired by works on arms races [7, 28] and spouses’ interaction [13], Polevoy et al. [21] formally analyze lengthy repeated reciprocation and show convergence. They define an action on an agent as a convex combination³ between one’s own last action, the considered other agent’s and all the other agents’ last actions. They call this the *floating* reciprocation attitude.

The main contributions of this paper comprise of the analysis of a unifying model of shared effort games with reciprocal projects and creating a basis for further analysis. We define two games: one without a threshold, and another one with a threshold. In the second game, those who are below the threshold in an interaction, are not allowed to participate in the respective interaction. We identify when Nash equilibria exist and find the prices of anarchy and stability. In addition to the main part, where the initial actions are fully reciprocated by the reciprocal agents [26], we model the situation when the budget of an agent to invest in the various projects may fall short of satisfying the requirements of every reciprocal interaction. This forces the agent to curb her investments in some interactions, complicating the process, but we prove it still converges, and therefore, generalizing the definitions to that case is well-defined. We also prove that the corresponding reciprocation effort game and its exclusive thresholded version have an equilibrium. We consider only pure equilibria throughout the paper, even when we do not mention this explicitly. Since the strategies include

³ A combination is convex if it has nonnegative weights that sum up to 1.

all the ways to divide budget among the interactions, the set of pure strategies is already uncountably infinite.

The model of several reciprocal interactions is given in Section 2. Section 3 characterizes the equilibria and their efficiency in a game without a threshold. Then, we analyze the game with a threshold in Section 4. We prove the convergence of an interaction with insufficient budgets and the NE existence in Section 5. Section 6 concludes and outlines new research directions.

2 Model

This section models dividing effort between reciprocal interactions. Adopting the reciprocation model from [21] and the inspired by shared effort games models from [4] and [20], we define a *reciprocation effort game*. First, we define a reciprocal process and the agents' utilities in this process. Next, we define a reciprocation effort game, where agents divide their effort budgets between several such processes. We define a thresholded variation on this game, to model the minimal required investment, in Section 4.

We begin with the reciprocation model, based on models for arms race and arguments. Given agents $N = \{1, \dots, n\}$, at any time $t \in T \triangleq \{0, 1, 2, \dots\}$, every agent acts on any other agent. The action by agent $i \in N$ on another agent $j \in N$ at moment t is characterized by its weight, denoted by $\text{act}_{i,j}(t): T \rightarrow \mathbb{R}$. Since only the weight of an action is relevant, we usually write “action” while referring to its weight. For example, the weights of the actions of helping, nothing, or insulting are in the decreasing order.

In order to define how agents reciprocate, we need the following notation. The kindness of agent i , constant for a given reciprocal process, is denoted by $k_i \in \mathbb{R}$. Agent i 's kindness models i 's inherent inclination to act on any other agent: the larger the kindness, the kinder the agent acts; in particular, it determines the first action of an agent, before the others have acted. We model agent i 's inclination to mimic another agent's action and the actions of all the other participants in the project by reciprocation coefficients $r_i \in [0, 1]$ and $r'_i \in [0, 1]$ respectively, both staying constant for all interactions. r_i is the fraction of $\text{act}_{i,j}(t)$ that is determined by the previous action of j upon i , and r'_i is the fraction that is determined by $\frac{1}{n-1}$ th of the total action on i by all the other agents at the previous time. Fractions sum up to 1, thus $r_i + r'_i \leq 1$. We denote the total received action from all the other agents at time t by $\text{got}_i(t): T \rightarrow \mathbb{R}$; formally, $\text{got}_i(t) \triangleq \sum_{j \in N} \text{act}_{j,i}(t)$.

We now define the actions. At time 0, there is nothing to react to, so the kindness determines the action: $\text{act}_{i,j}(0) \triangleq k_i$.

Definition 1 *At any positive time t , agent i 's action is a weighted average of her own last action (inertia), of that of the other agent j (direct reaction) and of the total action of all the other agents divided over all the others (social reaction):*

$$\text{act}_{i,j}(t) \triangleq (1 - r_i - r'_i) \cdot \text{act}_{i,j}(t-1) + r_i \cdot \text{act}_{j,i}(t-1) + r'_i \cdot \frac{\text{got}_i(t-1)}{n-1}.$$

We have defined how agents reciprocate. An agent's *utility* from a given reciprocation project at a given time is the action one receives minus effort to act, following [19]. This is classical (see, for example, the quasilinear preferences of auction theory [17, Chapter 9.3]). Formally, define the utility of agent i at time t , $u_{i,t}: \mathbb{R}^{n-1} \times \mathbb{R}^{n-1} \rightarrow \mathbb{R}$, as

$$u_{i,t} \left(\left\{ \text{act}_{i,j}(t) \right\}_{i,j \in N}, \left\{ \text{act}_{j,i}(t) \right\}_{i,j \in N} \right) \triangleq \sum_{j \in N} \text{act}_{j,i}(t) - \beta_i \sum_{j \in N} \text{act}_{i,j}(t)$$

where the constant $\beta_i \in \mathbb{R}$ is the importance of performing actions relatively to receiving them for i 's utility. The personal price of acting is higher, equal or lower than of receiving an action if β_i is bigger, equal or smaller than 1, respectively. The minus in front of i 's actions subtracts the effort of acting from one's utility (unless β_i is negative, where that is added). Since the presence of negative actions would mess up this logic (since negative actions would still take effort while increasing the above expression), we assume that actions are always non-negative, which occurs if and only if all kindness values are non-negative. We can have negative influence, but we assume having added large enough a constant to all the actions, to avoid negative actions.

Every such interaction converges, as shown in [21]. To model the utility in the long run, we define the asymptotic utility, or just the *utility*, of agent i , as the limit of her utilities as the time approaches infinity. In formulas, $u_i: (\mathbb{R}^{n-1})^\infty \times (\mathbb{R}^{n-1})^\infty \rightarrow \mathbb{R}$, as $u_i(\bigcup_{t'=0}^\infty \{\text{act}_{i,j}(t), \text{act}_{j,i}(t)\}) \triangleq \lim_{t \rightarrow \infty} u_{i,t}(\text{act}_{i,j}(t), \text{act}_{j,i}(t))$. This is the utility we consider here. This definition of the utility of a process is equivalent to the discounted sum of utilities when the discounting is slow enough. The proof is omitted for the lack of space.

We now define a *reciprocation effort game*. Our agents N participate in m interactions $\Omega = \{1, 2, \dots, m\}$. Each of the m interactions is what we have defined till now, with its own kindness values and actions. The kindness, the actions, the total received action, and the utility in a concrete interaction $\omega \in \Omega$ will be denoted, when the concrete interaction is important, by $(k_i)_\omega$ and $(\text{act}_{i,j}(t))_\omega$, $(\text{got}_i(t))_\omega$, and $(u_{i,t})_\omega$ or $(u_i)_\omega$, respectively. Each player's strategies are the possible contributions to the interactions at time zero (the further contributions are determined by the reciprocation and not by the player). A contribution goes to the whole interaction, not to a particular action on another agent, but it determines the kindness values of the interactions as follows.

Player i 's kindness at reciprocal interaction ω is determined by her contribution to that interaction at time zero, called just "the contribution", divided by the number of other agents who participate in the interaction at ω , accounting for acting on them. This means that i 's kindness at interaction j is $\frac{x_j^i(0)}{n-1}$. Therefore, the sum of all the actions of agent i at time $t = 0$ is equal to her contributions to all the reciprocation projects, which are bounded by her budget b_i . The contribution of player $i \in N$ to interaction project $\omega \in \Omega$ at a general time $t \in T$ is defined as the sum of her actions in that interaction at that time, i.e. $x_\omega^i(t) \triangleq \sum_{j \in N \setminus \{i\}} (\text{act}_{i,j}(t))_\omega$.

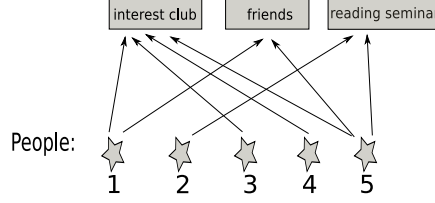


Fig. 1. People divide their own effort between interactions.

An agent contributes something in the beginning of a reciprocation, and from that time on the reciprocation “automatically” uncurls according to Definition 1. We assume that not only the sum of the contributions at $t = 0$, but also the sum of the contributions at any time $t > 0$ is within the acting agent’s budget. Each player i has a normal budget $b_i > 0$ (or just a budget) to contribute from at $t = 0$ and an extended budget $B_i \geq b_i$ that can be used when the actions are required by the reciprocation process at $t > 0$, perhaps resulting in a higher summarized contribution than the voluntarily chosen at $t = 0$. We differentiate between these two budgets, since the need to reciprocate can urge people to act more actively [11], and we assume that B_i s are high enough to allow reciprocation.

Formally, the strategy space of player i consists of her contributions (at time zero), determining her kindness values at the interactions, $\{x^i = (x_\omega^i)_{\omega \in \Omega} \in \mathbb{R}_+^{|\Omega|} \mid \sum_{\omega \in \Omega} x_\omega^i \leq b_i\}$. As mentioned, a “contribution” always means the contribution at $t = 0$. Since the strategy profile $x = (x^i)_{i \in N}$ determines all the interactions, the above defined utilities in a reciprocal interaction, namely $(u_{i,t})_\omega$ and $(u_i)_\omega \triangleq \lim_{t \rightarrow \infty} (u_{i,t})_\omega$, are also functions of x . The utility $u_i(x)$ of a player $i \in N$ in the game is defined to be the sum of the utilities it obtains from the various projects, $u_i(x) \triangleq \sum_{\omega \in \Omega} (u_i(x))_\omega$, completing the definition of a *reciprocation effort game*.

An agent does not have to use up all her budget, so that the inequality $\sum_{\omega \in \Omega} x_\omega^i \leq b_i$ may be strict. The strategies of all the players except i are denoted x^{-i} . We denote the vector of all the contributions by $x = (x_\omega^i)_{\omega \in \Omega}^{i \in N}$.

We now give a concrete example of the model.

Example 1. People choose between going to an interest club, meeting friends, or going to a scientific reading seminar, as illustrated in Figure 1. A player first decides on how much she wants to invest in each interaction, determining her kindness in each one of them. Subsequently, she reciprocates. Each of these projects is an interaction; for instance, in an interest club, a positive action can be supporting another person, while showing contempt would be negative. Interacting, a person continues her previous course of action, represented by $(1 - r_i - r'_i) \cdot \text{act}_{i,j}(t-1)$ in Definition 1, reacts on the other person’s previous action, represented by $r_i \cdot \text{act}_{j,i}(t-1)$, and reacts on the social climate, for which $r'_i \cdot \frac{\text{got}_i(t-1)}{n-1}$ stands.

For the sake of efficiency analysis, we remind that the social welfare is defined as $SW \triangleq \sum_{i \in N} u_i(x)$, and the prices of anarchy [15] and stability [23, 1] are defined as $\frac{\min\{SW(x)|x \text{ is an NE}\}}{\max\{SW(x)|x \text{ is a strategy}\}}$ and $\frac{\max\{SW(x)|x \text{ is an NE}\}}{\max\{SW(x)|x \text{ is a strategy}\}}$, respectively. We define 0/0 to be 1, because 0 from 0 means no loss occurs in the social welfare in the equilibria.

Polevoy et al. [21] prove

Theorem 1. *In an interaction, where for any agent i , $r'_i > 0$ and at least one agent i has $r_i + r'_i < 1$, for all pairs of agents $i \neq j$, the limit $\lim_{t \rightarrow \infty} \text{act}_{i,j}(t)$ exists. The convergence is geometrically fast (exponential). All these limits are equal to each other and it is a convex combination of the kindness values, namely*

$$L = \frac{\sum_{i \in N} \left(\frac{1}{r_i + r'_i} \cdot k_i \right)}{\sum_{i \in N} \left(\frac{1}{r_i + r'_i} \right)}. \quad (1)$$

3 Reciprocation Effort Game without a Threshold

We first completely analyze existence of NE, and then we find all the prices of anarchy and stability. This theorem characterizes the existence of equilibria.

Theorem 2. *Assume that for any agent i , $r'_i > 0$, and in addition, either $n > 2$ or $r_1 + r'_1 + r_2 + r'_2 < 2$. The set of all the NE is exactly all the strategy profiles where every agent with $\beta_i < 1$ somehow divides all her budget among the projects $\{1, \dots, m\}$, and every agent with $\beta_i > 1$ contributes nothing. These strategies are also dominant. In particular, there always exists an NE.*

Proof. Consider an arbitrary player l , and let her strategy (her contributions⁴) be $x^l = (x_1^l, \dots, x_m^l)$. By Formula(s) (1), the limit of the actions at (project) interaction j is

$$\left(\frac{\left(\frac{1}{r_l + r'_l} \cdot (x_j^l) \right)}{\sum_{i \in N} \left(\frac{1}{r_i + r'_i} \right)} + C_j \right),$$

where C_j is independent of l 's strategy. This is both given and received by an agent w.r.t. the $n - 1$ other agents, so we need to multiply the limit by $(n - 1)(1 - \beta_l)$. Summarizing, agent j 's utility from this strategy is

$$(n - 1)(1 - \beta_l) \left(\frac{\left(\frac{1}{r_l + r'_l} \cdot (x_1^l + \dots + x_m^l) \right)}{\sum_{i \in N} \left(\frac{1}{r_i + r'_i} \right)} + C \right),$$

for C that is independent of l 's strategy. Therefore, if $\beta_l < 1$, then l 's strategy is a best response to others' strategies if and only if l arbitrarily divides all her budget among the projects $\{1, \dots, m\}$. On the other hand, if $\beta_l > 1$, then a

⁴ Contributions by default refer to the contributions at time zero.

strategy is a best response if and only if all the contributions are zero. This is true for every agent l , proving that this is an NE. Since each agent is independent of the others, these strategies are also dominant. ■

The possible variations in an NE profile are what the agents with $\beta = 1$ do. This is important for analyzing the efficiency of the NE.⁵ To analyze efficiency, we define: $N^< \triangleq \{i \in N : \beta_i < 1\}$, $N^\leq \triangleq \{i \in N : \beta_i \leq 1\}$, $N^= \triangleq \{i \in N : \beta_i = 1\}$. We now analyze the efficiency of the most and the least efficient equilibria, comparing their social welfare to the maximum possible social welfare.

Proposition 1. *Under the assumptions of Theorem 2, if $(n > \sum_{i \in N} \beta_i)$, we have $\text{PoA} = \frac{\sum_{i \in N^<} \left(\frac{1}{r_i + r'_i} \cdot b_i \right)}{\sum_{i \in N} \left(\frac{1}{r_i + r'_i} \cdot b_i \right)}$, and the PoS is given by the same expression, where we use N^\leq instead of $N^<$. Consequently, if $(n = \sum_{i \in N} \beta_i)$, we have $\text{PoA} = \text{PoS} = 1$. If $(n < \sum_{i \in N} \beta_i)$, then:*

If $N^< \neq \emptyset$, then we have $\text{PoA} = \text{PoS} = -\infty$.

If $N^< = \emptyset$, but $N^\leq \neq \emptyset$, then $\text{PoA} = -\infty$, but $\text{PoS} = 1$.

If $N^\leq = \emptyset$, then $\text{PoA} = \text{PoS} = 1$.

The proof compares the possible social welfare in equilibria with the optimum social welfare.

Proof. The possible social welfare values that an NE can achieve are exactly

$$(n-1)(n - \sum_{i \in N} \beta_i) \frac{\sum_{i \in N^<} \left(\frac{1}{r_i + r'_i} \cdot b_i \right) + \sum_{i \in N^=} \left(\frac{1}{r_i + r'_i} \cdot x^i \right)}{\sum_{i \in N} \left(\frac{1}{r_i + r'_i} \right)},$$

where $0 \leq x^i \leq b_i$. The optimum social welfare is

$$(n-1)(n - \sum_{i \in N} \beta_i) \frac{\sum_{i \in N} \left(\frac{1}{r_i + r'_i} \cdot b_i \right)}{\sum_{i \in N} \left(\frac{1}{r_i + r'_i} \right)}$$

if $(n > \sum_{i \in N} \beta_i)$, and 0 otherwise.

Thus, if $(n > \sum_{i \in N} \beta_i)$, we have

$$\text{PoA} = \frac{\sum_{i \in N^<} \left(\frac{1}{r_i + r'_i} \cdot b_i \right)}{\sum_{i \in N} \left(\frac{1}{r_i + r'_i} \cdot b_i \right)} \text{ and } \text{PoS} = \frac{\sum_{i \in N^\leq} \left(\frac{1}{r_i + r'_i} \cdot b_i \right)}{\sum_{i \in N} \left(\frac{1}{r_i + r'_i} \cdot b_i \right)}.$$

If $(n = \sum_{i \in N} \beta_i)$, we have $\text{PoA} = \text{PoS} = 1$, since the social welfare is always zero, and we define here $0/0 = 1$.

If $(n < \sum_{i \in N} \beta_i)$, then we may get negative social welfare, since zero is optimal, while some NE yield a negative social welfare. Concretely, we have the following subcases:

⁵ $\beta_i > 1$ implies negative utilities that sometimes result in negative PoA and PoS.

If $N^< \neq \emptyset$, then we have $\text{PoA} = \text{PoS} = -\infty$, because any NE has the social

$$\text{welfare of at most } (n-1)(n - \sum_{i \in N} \beta_i) \frac{\sum_{i \in N^{++}} \left(\frac{1}{r_i + r'_i} \cdot b_i \right)}{\sum_{i \in N} \left(\frac{1}{r_i + r'_i} \right)}.$$

If $N^< = \emptyset$, but $N^\leq \neq \emptyset$, then $\text{PoA} = -\infty$ but $\text{PoS} = 1$. The reason is that an NE can have the social welfare from zero and down to $(n-1)(n -$

$$\sum_{i \in N} \beta_i) \frac{\sum_{i \in N^\leq} \left(\frac{1}{r_i + r'_i} \cdot b_i \right)}{\sum_{i \in N} \left(\frac{1}{r_i + r'_i} \right)}.$$

If $N^\leq = \emptyset$, then $\text{PoA} = \text{PoS} = 1$, since an NE has the social welfare of zero.

■

In particular, we have shown that if all the agents find acting easy (i.e., all $\beta_i < 1$), or if all agents really do not like acting (i.e., all $\beta_i > 1$), then $\text{PoA} = \text{PoS} = 1$, so that any NE is optimum for the society. Intuitively, this is because here, all the agents have similar preferences: either everyone wants to act and receive action, or no one does. We have also shown, that if the average agent finds not contributing more important than receiving (i.e., $\sum_{i \in N} \beta_i > n$), but still $\beta_i < 1$ for some agent i , then $\text{PoA} = \text{PoS} = -\infty$, so any NE is catastrophic to the society. Intuitively, this stems from the differences in the agents' preferences. Finally, we see that if $\sum_{i \in N} \beta_i > n$, some agents have $\beta_i = 1$, but none have $\beta_i < 1$, then $\text{PoA} = -\infty$ but $\text{PoS} = 1$, requiring regulation.

Theorem 2 implies that if all the projects have $\beta \leq 1$, then any dividing of all the budget in cooperating is always an NE. This is unintuitive, since usually, some groups are more efficient to interact with than some other groups. The reason for this is that the model assumes that all agents always interact at every project $\omega \in \{1, \dots, m\}$, and only their kindness depends on the strategy. Basically, everyone attends all the interactions, and some people are passive.

4 Exclusive Thresholded Reciprocation Effort Game

We now define a variation on a shared effort game with reciprocation, where only the agents who contribute at least the threshold may interact. First, following [20], we define a *θ -sharing mechanism*. This models, for example, a minimum invested effort to be considered a coauthor, or a minimum effort to master a technology before working with it. Define, for every $\theta \in [0, 1]$, the players who get a share from project ω to be $N_\omega^\theta \triangleq \{i \in N \mid x_\omega^i \geq \theta \cdot \max_{j \in N} x_\omega^j\}$, which are those who bid at least θ fraction of the maximum contribution to ω .

We now define an *exclusive thresholded reciprocation effort game*, as a reciprocation effort game, where exclusively the agents in N_ω^θ interact. Others do not obtain utility and do not even interact. If an agent ends up participating alone at a project, he obtains zero utility from that project, since no interaction occurs. Exclusive thresholded reciprocation effort games model situations when joining an interaction requires contributing enough, like the initial effort it takes

to learn the required technology to contribute to Wikipedia, the effort to become a member of a file sharing community or to start a firm.

In this section, we assume w.l.o.g. that $b_n \geq \dots \geq b_1$. The existence of an equilibrium is easy, since no-one contributing constitutes an NE. Then, we show that also less trivial equilibria exist. Finally, the harder question of equilibrium efficiency is answered for two agents. We first notice a trivial equilibrium.

Observation 1 *The profile where all agents contribute nothing is an NE.*

Proof. In this profile, any agent who deviates by contributing a positive amount to a project will be the only one to interact there, so her utility will still be zero. ■

We call an NE where at any project, at most one agent interacts (reaches the threshold) and positively contributes there, a Zero NE. There may be multiple Zero NE. We have just shown that a Zero NE always exists. A natural question is whether there exist non-Zero NE as well. They do.

Theorem 3. *Assume that all agents have $\beta_i \leq 1$. Assume that for any agent i , $r'_i > 0$ and in addition, for any pair of agents i, j we have $r_i + r'_i + r_j + r'_j < 2$. There exists a non-Zero NE.*

Proof. Consider the profile where all agents $1, \dots, n-1$ contribute their whole respective budgets to project 1, and agent n contributes $\min\left\{b_n, \frac{b_{n-1}}{\theta}\right\}$ to project 1, and nothing to other projects.

This is an NE, for the following reasons. Any agent would be alone at any project other than 1 if it contributed to such a project, and therefore, it will not contribute there. At project 1, the only agent who perhaps can increase her contribution is n , but she will stay alone, if she does, so no deviation is profitable. ■

The next question is the efficiency of the equilibria. Since we always have the Zero NE, and by contributing to the same project the same positive amounts we achieve a positive social welfare, we always have $\text{PoA} = 0$. Regarding the price of stability, we immediately know that it is positive, since there always exists a non-Zero NE. We now show that the price of stability for two agents is 1, meaning that there exists a socially optimal NE.

Proposition 2. *For $n = 2$ and under the assumptions of Theorem 3, $\text{PoS} = 1$.*

Proof. When we have only two players, we can assume w.l.o.g. that in a profile with maximum social welfare, a project that receives a positive contribution, receives it from both agents. Therefore, social welfare is maximized by maximizing the total contribution to the projects where interaction occurs.

Then, the following profile maximizes the social welfare. Agent 1 spreads her budget equally between all the projects. If $b_1 \geq \theta b_2$, then agent 2 divides her budget equally between all the projects, and otherwise, she contributes $\frac{1}{\theta} \frac{b_1}{m}$ to every project. Since this profile constitutes an NE, we conclude that $\text{PoS} = 1$. ■

5 Insufficient Budgets

Till now, we have been assuming that there is enough extended budget to allow the agents make the contributions required by the sum of the reciprocal actions at any time. In this section, we consider dividing effort between reciprocal projects, where the extended budgets B_i may not suffice to reciprocate at some positive time t , and therefore the actions have to be curbed, such that the total action at any time is bounded by the B_i . In Example 1, this can happen if people are unable to keep up with the others because their free time is strictly limited. In order to justify studying the asymptotic behavior here, we prove that for any curbing, the actions in all interactions converge, as time approaches infinity. Then, we study the equilibria of the corresponding game.

The convergence of normal reciprocation is proven in [21], and we now prove the convergence of curbed reciprocation. Consider the undirected interaction graph $G = (N, E)$ of an interaction project, such that agent i can act on j and vice versa if and only if $(i, j) \in E$. Our model assumes that this graph is a clique, meaning that everyone interacts, but this is not necessary for the following theorem. At a given time, let the reciprocation from Definition 1 require actions denoted by the column vector $\mathbf{q} \in \mathbb{R}_+^{|E|}$, in the sense that its (i, j) th coordinate contains $\text{act}_{i,j}$ (for $(i, j) \in E$). Then, the curbing is denoted by $D_{\mathbf{q}} \cdot \mathbf{q}$, where $D_{\mathbf{q}}$ is the diagonal curbing matrix. We omit the subscript \mathbf{q} when the vector on which we act is clear. We denote the curbing matrix at time t by $D(t)$.

Theorem 4. *Consider dividing effort between reciprocal interactions, where every interaction has some connected interaction graph, and for all agents i , $r'_i > 0$. At every interaction, if there exists a cycle of an odd length in the interaction graph, or at least one agent i has $r_i + r'_i < 1$, then, for all pairs of agents $i \neq j$, the limit $L_{i,j} \triangleq \lim_{t \rightarrow \infty} \text{act}_{i,j}(t)$ exists.*

In our model, we assume a completely connected graph, so if at least 3 agents interact, we have an odd cycle, namely a triangle. Therefore, then we only need to assume that for all agents i , $r'_i > 0$.

The proof expresses reciprocation as matrix multiplication. Without curbing, the convergence is proven using the Perron-Frobenius theorem. Keeping convergence when curbing can occur uses the following definition and lemma.

Definition 2 *We remind that a square non-negative matrix A is called primitive, if there exists a positive l , such that $A^l > 0$ (see [24, Definition 1.1]).*

The following lemma, used to prove the theorem, has a value of its own as well. Given a convergent sequence of primitive matrices, the lemma shows that arbitrarily squeezing the matrices keeps the convergence.

Lemma 2. *Given a vector $\mathbf{p}(0) \in \mathbb{R}^d$, a primitive matrix $A \in \mathbb{R}^{d^2}$, such that $\lim_{t \rightarrow \infty} A^t$ exists, and a sequence of diagonal matrices $\{D(t)\}_{t=0}^{\infty}$, $D(t) = \text{diag}(\lambda_1(t), \dots, \lambda_d(t))$, where each $\lambda_i(t) \in (0, 1]$, define the sequence $\{\mathbf{p}(t)\}_{t=0}^{\infty}$ by $\mathbf{p}(t) \triangleq D(t)AD(t-1)A \dots D(1)A\mathbf{p}(0)$. Then, $\lim_{t \rightarrow \infty} \mathbf{p}(t)$ exists.*

Proof. Assume to the contrary, that $\{\mathbf{p}(t)\}$ diverges. Define the sequence $\{\mathbf{p}'(t)\}_{t=0}^{\infty}$ by $\mathbf{p}'(t) \triangleq A^t \mathbf{p}(0)$. Since $\{\mathbf{p}(t)\}$ diverges and $\{\mathbf{p}'(t)\}$ converges, they differ at some point, intuitively speaking. We now formalize this argument. Since $\{\mathbf{p}(t)\}$ diverges and the space is complete, it is not a Cauchy sequence, and so there exists a positive ϵ , such that for each $N > 0$ there exist $n, m > N$, such that $\|\mathbf{p}(n) - \mathbf{p}(m)\| > \epsilon$ ($\|\cdot\|$ is the Euclidean norm). Since $\{\mathbf{p}'(t)\}$ converges, it is a Cauchy sequence, so there exists $N > 0$, such that for all $n, m > N$ we have $\|\mathbf{p}'(n) - \mathbf{p}'(m)\| < \epsilon/2$. If $\|\mathbf{p}(n) - \mathbf{p}(m)\| > \epsilon$ and $\|\mathbf{p}'(n) - \mathbf{p}'(m)\| < \epsilon/2$, we cannot both have $\|\mathbf{p}(n) - \mathbf{p}'(n)\| < \epsilon/4$ and $\|\mathbf{p}(m) - \mathbf{p}'(m)\| < \epsilon/4$. Therefore, for some integer l , $\|\mathbf{p}(l) - \mathbf{p}'(l)\| > \delta$, for some $\delta > 0$, depending solely on ϵ . Since the product defining $\mathbf{p}(l)$ is like that of $\mathbf{p}'(l)$, but with more $D(t)$ matrices, and $D(t) = \text{diag}(\lambda_1(t), \dots, \lambda_d(t))$, where each $\lambda_i(t) \in (0, 1]$, we have $\mathbf{0} \leq \mathbf{p}(l) \leq \mathbf{p}'(l)$. Remembering this, and that matrix A is primitive, thereby propagating a change of an entry to every entry, we can choose l such that every coordinate of $\mathbf{p}(l)$ will be at most α fraction of the corresponding coordinate of $\mathbf{p}'(l)$, for some $\alpha < 1$. The α can be made to depend solely on ϵ , because of the boundedness of all the relevant vectors. So, we have $\mathbf{p}(l) \leq \alpha A^l \mathbf{p}(0)$.

By reiterating the same argument with $\mathbf{p}'_1(t) \triangleq A^t \mathbf{p}(l)$ and $\mathbf{p}_1(t) \triangleq \mathbf{p}(t+l)$, we find $l_1 > 0$, such that $\mathbf{p}_1(l_1) \leq \alpha A^{l_1} \mathbf{p}(l)$. Thus, $\mathbf{p}(l_1 + l) = \mathbf{p}_1(l_1) \leq \alpha A^{l_1} \mathbf{p}(l) \leq \alpha A^{l_1} \alpha A^l \mathbf{p}(0) = \alpha^2 A^{l_1+l} \mathbf{p}(0)$.

Continuing in this manner, and using the boundedness of the converging $\{A^t \mathbf{p}(0)\}$, we prove that $\{\mathbf{p}(t)\}$ converges to zero. A contradiction.⁶ ■

We can now prove Theorem 4.

Proof. We extend the proof of the Theorem 1 from [21], which proves the convergence without the curbing. We show that the curbing still keeps the convergence. We recapitulate the used properties from there, to stay self-contained.

We express the dynamics of interaction in a matrix, and prove the theorem by applying the Perron–Frobenius theorem [24, Theorem 1.1, 1.2], using the above lemma to handle the curbing of actions. Denoting the neighbors of i as $N(i)$, we define the dynamics matrix $A \in \mathbb{R}_+^{|E| \times |E|}$ as

$$A((i, j), (k, l)) \triangleq \begin{cases} (1 - r_i - r'_i) & k = i, l = j; \\ r_i + r'_i \frac{1}{|N(i)|} & k = j, l = i; \\ r'_i \frac{1}{|N(i)|} & k \neq j, l = i; \\ 0 & \text{otherwise.} \end{cases} \quad (2)$$

Assume that for each time $t \in T$, the column vector $\mathbf{p}(t) \in \mathbb{R}_+^{|E|}$ describes the actions at time t . Then, $\mathbf{p}(t+1) = D(t)A\mathbf{p}(t)$, where $D(t)$ is the diagonal matrix, describing the curbing. We call $\mathbf{p}(t)$ an action vector. Initially, $\mathbf{p}(0)_{(i,j)} = k_i$.

We will use the Perron-Frobenius theorem for primitive matrices. We now prepare to use it, and first we show that A is primitive. In the proof of Theorem 1,

⁶ The actual limit does not have to be zero; zero is just the result from the contradictory assumption.

it is shown that A is irreducible and aperiodic, and therefore primitive by [24, Theorem 1.4]. Since the sum of every row is 1, the spectral radius is 1.

According to the Perron-Frobenius theorem for primitive matrices [24, Theorem 1.1], the absolute values of all the eigenvalues except one eigenvalue of 1 are strictly less than 1. The eigenvalue 1 has unique right and left eigenvectors, up to a constant factor. Both these eigenvectors are strictly positive. Therefore, [24, Theorem 1.2] implies that $\lim_{t \rightarrow \infty} A^t = \mathbf{1}\mathbf{v}'$, where \mathbf{v}' is the left eigenvector of the value 1, normalized such that $\mathbf{v}'\mathbf{1} = 1$.

Now, Lemma 2 implies that $L_{i,j}$ exists. \blacksquare

We have proven the reciprocation effort game where curbing can occur is well defined, because all the reciprocation processes converge. We now prove the existence of equilibria in such a game. In the exclusive thresholded model, Observation 1 holds in the curbed case as well, so contributing nothing is an NE. From now on, assume that no threshold exists. Since the curbing renders finding a formula for the actions in the limit unlikely, we take an abstract approach.

Theorem 5. *Consider dividing effort between reciprocal interactions, where for all agents i , $r'_i > 0$. Assume that $n \geq 3$ or at least one agent i has $r_i + r'_i < 1$. Assume that the curbing function $D: \mathbb{R}^n \rightarrow \mathbb{R}^n$ is a weak contraction w.r.t. norm L_∞ , i.e. $\|D_{\mathbf{x}}\mathbf{x} - D_{\mathbf{y}}\mathbf{y}\|_\infty \leq \|\mathbf{x} - \mathbf{y}\|_\infty$.*

Then, there exist small enough β_i s such that an NE exists.

Proof. By Theorem 4, the reciprocation processes converge and so the game is well defined. We prove the existence using Proposition 20.3 from [18]. The strategy set of every player consists of all the possible divisions of the budgets between the projects, which is a nonempty compact convex set.

The continuity of the utility functions follows from the action limits depending continuously on the total contributions of agents to projects. To this end, we can inductively show that at any time t , the change in the action is $\|\mathbf{p}'(t) - \mathbf{p}(t)\|_\infty \leq \|\Delta\mathbf{x}\|_\infty$, where $\mathbf{p}'(t)$ represents the actions at time t if $\mathbf{p}'(0) = \mathbf{p}(0) + \Delta\mathbf{x}$. This boundedness keeps holding in the limit of time approaching infinity as well, implying continuity. For the quasi-concavity of an agent's strategy space, notice that for small enough β_i , agent i would like to increase its contribution exactly till it can increase the limit of the actions of at least one another agent. Finally, Proposition 20.3 implies the theorem. \blacksquare

We also prove that when agents react identically, the game boils down to a single reciprocal interaction.

Proposition 3. *Assume that the curbing is determined by the sum of the actions of an agent and that all the reciprocation coefficients are equal among the agents, i.e. $r_i = r_j$ and $r'_i = r'_j, \forall i \neq j$. Then, the total contribution of agent i at any time $t \in T$, i.e. $x^i(t) \triangleq \sum_{\omega \in \Omega} x_\omega^i(t)$, and the total received action, i.e. $\sum_{\omega \in \Omega} (\text{got}_i(t))_\omega$, are fully determined by the total contributions and the total received actions at time zero of the agents (i and others), regardless how the actions were divided between the projects.*

Proof. We prove this by induction on time. At the *basis*, $t = 0$ and the statement is trivial. At the *induction step*, assume that $\sum_{\omega \in \Omega} x_{\omega}^i(t-1)$ and $\sum_{\omega \in \Omega} (\text{got}_i(t-1))_{\omega}$ are fully determined by the total contributions and the total received actions at time zero and prove this determinacy for $\sum_{\omega \in \Omega} x_{\omega}^i(t)$ and $\sum_{\omega \in \Omega} (\text{got}_i(t))_{\omega}$. Indeed, $x_{\omega}^i(t)$ is equal to

$$\begin{aligned} \sum_{j \neq i} \text{act}_{i,j} &= (1 - r_i - r'_i) \sum_{j \neq i} (\text{act}(t-1))_{\omega} + r_i \sum_{j \neq i} (\text{act}(t-1))_{\omega} + r'_i (\text{got}(t-1))_{\omega} = \\ &= (1 - r_i - r'_i) x_{\omega}^i(t-1) + r_i (\text{got}(t-1))_{\omega} + r'_i (\text{got}(t-1))_{\omega}. \end{aligned}$$

Sum it up over all the projects to obtain

$$\sum_{\omega \in \Omega} x_{\omega}^i(t) = (1 - r_i - r'_i) \sum_{\omega \in \Omega} x_{\omega}^i(t-1) + r_i \sum_{\omega \in \Omega} (\text{got}(t-1))_{\omega} + r'_i \sum_{\omega \in \Omega} (\text{got}(t-1))_{\omega}.$$

Since everything on the right hand side is, by the induction hypothesis, determined by the total contribution and the total received action at time zero, the actions on time t before curbing are determined by them as well. Furthermore, curbing is determined by the total action of the agents, and thus, the curbed actions are also determined by the total contribution and the total received action at time zero.

Regarding the total received action, the derivation of the step is analogous, but it requires moving $1 - r_j - r'_j$, r_j and r'_j out of the parentheses, where we use the equality of these parameters across the agents. ■

6 Conclusions and Further Research

In order to predict investing effort in several reciprocal interactions, we define a game that models dividing efforts between several reciprocal projects. We include an analysis of a model both with and without a contribution threshold.

When no contribution threshold exists, there always exists an equilibrium, and if acting is easy to everyone (for all i , $\beta_i < 1$) or hard to everyone (for all i , $\beta_i > 1$), then every NE is socially optimal. We also show that any dividing of all the budget when acting is easy to everyone is a Nash equilibrium. The result may seem surprising. Intuitively, this happens because everyone participates in each interaction, and the concrete division of the budget does not matter to the social welfare. However, life does not often provide such situations. We also characterize when both efficient and inefficient equilibria exist, calling for regulation.

If a minimum contribution is necessary to participate in interaction, we show that the situation where no-one contributes is an equilibrium. This models the case where people are very passive, and this continues since no-one can start an interaction project on his own. In addition to this trivial equilibrium, we find an equilibrium where all the agents contribute to the same project, like Facebook, instead of participating in the other social networks. This describes the case

when people interact with each other on the same topic. Such a situation is clearly not the only option, since people often have many friendships [9]. For two agents, there exists an equilibrium which is socially optimal.

The choices of strategies by the agents who are indifferent can significantly influence the social welfare. For instance, this happens in the case without threshold to agents for whom acting and receiving action are equally important. Making such agents do what benefits the society can increase the social welfare.

We also model the case when the extended budgets are not big enough and curbing is required. We show that any way of dividing effort between reciprocal interactions results in converging interactions, regardless of how actions are curbed to fit the budgets. We also prove that the resulting reciprocation effort game possesses an equilibrium, with and without threshold.

For future research, we are curious about the efficiency of the equilibria in the game with curbing. Consecutive decisions can be modeled by the agents first contributing to the interactions and then deciding on their reciprocation parameters. Additionally, looking at interactions in large groups where not everyone can act on everyone else would be a natural generalization of our work. Another point is that we assumed that two agents who interact in multiple projects, interact in these projects independently. Modeling the dependency between these interactions is realistic. Analyzing a mixed set of projects, only some of which are interaction projects, would model reality better. Also modeling and analyzing voting to approve who else may participate in an interaction seems promising.

This work models and analyzes a ubiquitous class of interactions and lays the basis for further research, aimed to provide more advice to the agents and to the manager who wants to maximize the social welfare.

Acknowledgments. We thank Prof. Orr M. Shalit from the Technion, Israel for the useful discussions. This work has been supported by SHINE, the flagship project of DIRECT (Delft Institute for Research on ICT at the TU Delft).

References

1. Anshelevich, E., DasGupta, A., Kleinberg, J., Tardos, E., Wexler, T., Roughgarden, T.: The price of stability for network design with fair cost allocation. In: *Found. of Comp. Science*, 2004. *Proceed. 45th Ann. IEEE Symp. on*. pp. 295–304 (Oct 2004)
2. Anshelevich, E., Hoefer, M.: *Contribution Games in Social Networks*, pp. 158–169. Springer Berlin Heidelberg, Berlin, Heidelberg (2010)
3. Axelrod, R.: *The evolution of cooperation*. Basic books, Basic Books (1984)
4. Bachrach, Y., Syrgkanis, V., Vojnović, M.: *Efficiency and the Redistribution of Welfare*. Tech. Rep. May, Microsoft Research (2012)
5. Baye, M.R., Kovenock, D., de Vries, C.G.: The all-pay auction with complete information. *Economic Theory* 8(2), 291–305 (1996)
6. Bicchieri, C.: *The grammar of society: The nature and dynamics of social norms*. Cambridge University Press (2006)
7. Dixon, W.J.: Reciprocity in united states-soviet relations: Multiple symmetry or issue linkage? *American Journal of Political Science* 30(2), pp. 421–445 (1986)

8. Dufwenberg, M., Kirchsteiger, G.: A theory of sequential reciprocity. *Games and Economic Behavior* 47(2), 268 – 298 (2004)
9. Ennett, S.T., Bauman, K.E.: Adolescent social networks: Friendship cliques, social isolates, and drug use risk. Improving prevention effectiveness pp. 47–57 (2000)
10. Falk, A., Fischbacher, U.: A theory of reciprocity. *Games and Economic Behavior* 54(2), 293 – 315 (2006)
11. Fehr, E., Fischbacher, U., Gächter, S.: Strong reciprocity, human cooperation, and the enforcement of social norms. *Human Nature* 13(1), 1–25 (2002)
12. Fehr, E., Gächter, S.: Fairness and retaliation: The economics of reciprocity. *Journal of Economic Perspectives* 14(3), 159–181 (2000)
13. Gottman, J., Swanson, C., Murray, J.: The mathematics of marital conflict: Dynamic mathematical nonlinear modeling of newlywed marital interaction. *Journal of Family Psychology* 13, 3–19 (1999)
14. Kleinberg, J., Oren, S.: Mechanisms for (mis)allocating scientific credit. In: *Proceedings of the 43rd annual ACM symposium on Theory of computing*. pp. 529–538. ACM, New York, NY, USA (2011)
15. Koutsoupias, E., Papadimitriou, C.: Worst-case equilibria. In: *16th Annual Symp. on Theor. Asp. of Computer Science*. pp. 404–413. Trier, Germany (4–6 Mar 1999)
16. Nie, N.H., Hillygus, D.S.: Where does internet time come from?: A reconnaissance. *IT & Society* 1, 1–20 (2002)
17. Nisan, N., Roughgarden, T., Tardos, E., Vazirani, V.: *Algorithmic Game Theory*. Cambridge University Press (2007)
18. Osborne, M.J., Rubinstein, A.: *A Course in Game Theory*, MIT Press Books, vol. 1. The MIT Press (Apr 1994)
19. Polevoy, G., de Weerdt, M., Jonker, C.: The game of reciprocation habits. In: *Proceedings of the 2016 European Conference on Artificial Intelligence. Frontiers in Artificial Intelligence and Applications*, vol. Volume 285: ECAI 2016, pp. 417–425 (2016)
20. Polevoy, G., Trajanovski, S., de Weerdt, M.M.: Nash equilibria in shared effort games. In: *Proceedings of the 2014 International Conference on Autonomous Agents and Multi-agent Systems*. pp. 861–868. AAMAS '14, International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC (2014)
21. Polevoy, G., de Weerdt, M., Jonker, C.: The convergence of reciprocation. In: *Proceedings of the 2016 International Conference on Autonomous Agents and Multiagent Systems*. pp. 1431–1432. AAMAS '16, International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC (2016)
22. Rabin, M.: Incorporating fairness into game theory and economics. *The American Economic Review* 83(5), pp. 1281–1302 (Dec 1993)
23. Schulz, A.S., Moses, N.S.: On the performance of user equilibria in traffic networks. In: *Proceedings of the Fourteenth Annual ACM-SIAM Symposium on Discrete Algorithms*. pp. 86–87. SODA '03, Society for Industrial and Applied Mathematics, Philadelphia, PA, USA (2003)
24. Seneta, E.: *Non-negative Matrices and Markov Chains*. Springer Series in Statistics, Springer (2006)
25. Siegel, R.: All-pay contests. *Econometrica* 77(1), 71–92 (2009)
26. Trivers, R.L.: The evolution of reciprocal altruism. *The Quarterly Review of Biology* 46, 35–57 (3 1971)
27. Van Segbroeck, S., Pacheco, J.M., Lenaerts, T., Santos, F.C.: Emergence of fairness in repeated group interactions. *Phys. Rev. Lett.* 108, 158104 (Apr 2012)
28. Ward, M.D.: Modeling the USA-USSR arms race. *Transactions of The Society for Modeling and Simulation International* 43, 196–203 (1984)

Get Your Virtual Hands Off Me! - Developing Threatening IVAs Using Haptic Feedback

Linford Goedschalk, Tibor Bosse, and Marco Otte

Vrije Universiteit Amsterdam, De Boelelaan 1081,
1081 HV Amsterdam, The Netherlands

Abstract. Intelligent Virtual Agents (IVAs) become widely used for numerous applications, varying from healthcare decision support to communication training. In several of such applications, it is useful if IVAs have the ability to take a negative stance towards the user, for instance for anti-bullying or conflict management training. However, the believability of such 'virtual bad guys' is often limited, since they are non-consequential, i.e., are unable to apply serious sanctions to users. To improve this situation, this research explores the potential of endowing IVAs with the ability to provide haptic feedback. This was realized by conducting an experiment in which users interact with a virtual agent that is able to physically 'touch' the user via a haptic gaming vest. The effect on the loudness of the speech and the subjective experience of the participants was measured. Results of the experiment suggest there might be an effect on the subjective experience of the participants and the loudness of their speech. Statistical analysis, however, shows no significant effect but due to the relatively small sample size it is advisable to further look into these aspects.

1 Introduction

Intelligent Virtual Agents (IVAs) are intelligent digital interactive characters that can communicate with humans and other agents using natural human modalities like facial expressions, speech, gestures and movement [19]. Recently, IVAs have become widely used for numerous applications, varying from healthcare decision support [8] to communication training [20]. In such applications, IVAs play various roles in which they interact with users, for instance as an instructor, therapist or teammate [15].

In the vast majority of these cases, IVAs are friendly and supportive towards the user. Instead, there was less attention for IVAs with a 'negative' or 'aggressive' attitude towards users (i.e., 'virtual bad guys'). This could be considered a missed opportunity, since the concept of virtual bad guys opens up a range of useful applications. Examples include virtual training of aggression de-escalation skills [6], anti-bullying education [23], and Virtual Reality exposure therapy [16].

However, a common difficulty in the design of IVAs is to make them *believable*, i.e., to provide the illusion of being alive [2]. This poses a particular challenge for 'virtual bad guys', since effective applications involving aggressive agents require

that users feel indeed seriously threatened or stressed by the IVA. However, IVAs are typically non-consequential, i.e., they are unable to apply serious sanctions to (or even physically harm) their human interlocutors. As a result, users still perceive IVAs as rather artificial beings, which possibly also influences the way they interact with them.

Triggered by this insight, the question addressed in this paper is how to develop aggressive virtual agents that are taken seriously. This question is tackled by designing and experimentally examining the effects of a threatening IVA that is able to physically 'touch' users by means of haptic feedback, which is realized by means of a haptic gaming vest. More specifically, we investigate whether endowing an IVA with the ability to provide such a physical threat has an impact on the verbal behaviour of users as well as their subjective experience.

The remainder of this article is structured as follows. In Section 2, the recent literature on aggressive virtual agents and on haptic feedback is reviewed. Next, in Section 3 the design of the performed experiment is presented, and the results are provided in Section 4. These results are evaluated in detail in Section 5. A conclusion is provided in Section 6, and Section 7 completes the paper with a discussion.

2 Related Work

The relevant literature for this project covers two main areas, namely aggressive virtual agents and virtual touch. The state-of-the art in these two areas is discussed in the following sub-sections.

2.1 Aggressive Virtual Agents

Research on emotions within IVAs has received much attention in recent years. An important stream of research addresses the development of generic computational models of emotion [12]. Probably the most influential approach is EMA [13], a computational model that formalises the main assumptions behind appraisal theory [11]. Although such models could be used to have agents generate emotional states like 'angry', they do not focus on agents that take a threatening attitude towards humans.

Instead, other research has focused more explicitly on the impact of emotional agents on humans in interpersonal settings. For example, the Sensitive Artificial Listener paradigm enables studying the effect of agents with different personalities on human interlocutors, which provided evidence that IVAs with an angry attitude indeed trigger different (subjective and behavioural) responses than agents with other personalities [17]. Similarly, a study in the domain of negotiation led to the conclusion that IVAs expressing anger (in terms of utterances and facial expressions) lead human negotiation partners to make larger concessions [7]. Another recent study pointed out that a virtual agent that made an 'outburst of aggression' (in terms of shouting to and insulting the user) was able to trigger increased physiological responses [5].

Nevertheless, as also concluded in [5], these responses are still insufficiently strong to be really useful for effective applications where heavy emotional stimuli play a role, such as aggression de-escalation training systems for law enforcement personnel [9] or public transport employees [6]. The assumption underlying the current paper is that this is due to the inability of existing IVAs to apply serious (e.g., physical) sanctions to human interlocutors. This is in line with research in the domain of shooting behaviour training for police officers, which indicates that 'simulated threat' is a necessary criterion to realize an adequate transfer of training from the simulated to the real world [14].

Hence, the current paper aims to bring interactions between humans and aggressive virtual agents to a next level of realism, which is done by introducing two technological innovations, namely immersive Virtual Reality and haptic feedback.

2.2 Virtual Touch

The domain of virtual reality is seeing its technology applied in fields like entertainment, education and even medicine. One of these applications is virtual reality-based training, which can be used for various purposes. In order for the effects of such training applications to be applied in the physical world it is important that the scenarios used resemble this world as closely as possible. This way users are offered an experience as if they were in the physical world itself.

Since a number of years, virtual reality applications are combined with haptic feedback, enabling users to 'touch' objects in the simulated environment. Most of these applications focus on touching static objects rather than conversational agents, for instance for surgical training (e.g., to improve performance in cadaver temporal bone dissection) [22].

Recently, virtual touch is also applied in a more social setting, leading to the area of 'virtual interpersonal touch' (e.g., [1]). For instance, research by Cheok and colleagues explores the use of haptic technology to reproduce multi-sensory sensations related to intimate activities like kissing [21] and hugging [18]. Although the primary use of such technology was to enable intimate touch sensations between humans remotely, it is also claimed to have potential in the area of human-agent interaction. Similarly, other researchers have studied the use of social touch with the aim to make virtual agents more 'warm' or empathic (e.g., [4], [10]). Nevertheless, all of these developments are in the context of 'positive interpersonal touch'. As far as could be determined, research into virtual agents that may touch human conversation partners with the purpose of intimidating or threatening them is still in its infancy.

3 Method

To investigate the effect of touch by a threatening virtual agent on human participants, the following experiment was set up. Participants were asked to interact with a virtual agent in a virtual reality environment through free speech. At

some point during this interaction, the virtual agent would start threatening the user, which was followed by a 'push' that was simulated through haptic feedback. A haptic gaming vest was chosen, because this enabled the users to be provided with a serious physical stimulus, while avoiding physically harming them (which would be the case by using for instance electric surges), which obviously is ethically irresponsible. This section describes the experimental set-up in detail.

3.1 Participants

A convenience sample of 47 people was recruited, most of which were academic students. The age of the participants varied between 18 and 25 years. Participants were randomly assigned to the experimental group that received haptic feedback or the control group that did not receive any feedback. The experimental group consisted of 21 participants (12 male, 9 female), and the control group consisted of 23 participants (13 male, 10 female). Three runs of the experiment resulted in corrupted or incomplete data and so these have been removed.

3.2 Experimental design

Participants were placed in either a condition with haptic feedback during the interaction (condition A) or a condition without haptic feedback (condition B). In both conditions, participants were wearing the haptic feedback equipment (as to eliminate any effects of the equipment itself), but they were not told in advance what was the purpose of the equipment. In the control condition, the haptic vest was turned off, but the participants did not know this.

The experiment used a between-participants design (where each participant is only allocated to one condition) instead of a within-participants design (where each participant would experience both conditions sequentially), because in the latter case, the participants would already expect the virtual push after having experienced the scenario once.

3.3 Tasks

The participants were asked to engage in a virtual reality scenario (displayed on a Head Mounted Display) taking place in the context of a nightclub in which they can freely move around. The participants were tasked with finding the bathroom in the virtual environment, after which they were to return to the bar area they initially started in and have a drink with their friend. However, on their way to the bathroom the participants would encounter a virtual agent named Mason. Mason poses the threat in the virtual environment by acting very aggressively towards the participants. As they walk through the corridor towards Mason, he walks into them and so spills his drink. This sets up a situation in which the agent behaves aggressively towards the users and at some point even physically 'attacks' the user. This attack has the form of a push which is transferred to the user through the haptic feedback vest.

Throughout the nightclub several virtual agents can be seen and heard interacting with each other. The users can only interact with two of the agents in the scenario through free speech responses during a conversation. The first agent they encounter is used to make the participants more comfortable with the free speech interaction paradigm. The second agent, Mason, is used to analyse the responses of the participant. Both agents have been created in such a way that they always give the same responses, no matter what the user says. This was done to minimize the differences across individual trials.

The responses of virtual agent Mason have been inspired by the Sensitive Artificial Listener paradigm [17], which enables users to interact with virtual agents using free speech. The dialogue can be set-up in such a way that the agent always seems to respond to what the user says even if this is not the case and the agent just follows a script.

During the conversation with Mason the participants were free to respond in whatever way they saw fit. Participants were only limited in that they should speak loud and clear, always respond to the agent and use at least one full sentence to respond. The conversation consists of ten user responses which results in a select set of data to analyse. To this end, participants were asked to respond at all times. The participants were alerted to the fact that the microphone might not pick up their voice if they did not speak loud and clear and this would result in difficulties for the analysis of the data.

Both conversations in the scenario are turn-based, meaning that both the agent and the participant take turns while speaking. Only the agents can initiate conversations as to avoid the participants trying to start a conversation with every agent they encounter.

3.4 Variables

Two types of dependent variables were used in this study, namely subjective and objective variables. As subjective variable, the participants' experience was measured through a questionnaire they had to fill in at the end of the experiment. This questionnaire contained the following questions, which had to be answered using a 5-point Likert scale (from 'not at all' to 'very much'):

- Q1 Did you have any experience with the use of head mounted display devices prior to this experiment?
- Q2 Did you have any experience with the use of haptic feedback hardware prior to this experiment?
- Q3 Did you find the virtual scenario to be realistic?
- Q4 Did you find Mason to be aggressive?
- Q5 Did you find Mason to be threatening?

In addition, the following yes-no questions were used:

- Q6 Were you startled when Mason pushed you?
- Q7 Did you look at Mason when he pushed you?
- Q8 Did you first walk through the club before talking to Mason?
- Q9 Did you react differently to Mason after he pushed you?

Some of these questions (Q1, Q2, Q7, Q8) were used as control questions, to avoid that any differences found could be attributed to other factors. For the other questions, the aim was to investigate *whether people have a more intense experience in the condition with haptic feedback than in the condition without* (Research Question 1).

The objective variable that was studied was the verbal behaviour of the participants during the interaction with Mason. More specifically, we used the loudness of their speech as an indicator for the participants' engagement in the scenario, as people who are excited typically speak louder [3]. The relevant data for this were obtained through the use of a web cam that recorded the experiment session. The audio from the recordings was extracted and the amplitudes from the audio files were sampled. This way it could be analysed whether the participants spoke louder or softer after being pushed in the virtual scenario. Hence, the aim was to investigate *if people use louder speech in the condition with haptic feedback than in the condition without* (Research Question 2).

3.5 Material and facilities

The experiment has been conducted in a quiet room in which only the participant and experimenter were present. This room contained a desk with the computer that hosted the virtual environment, a four-legged chair for the participants to sit on during the experiment and a desk with the equipment used during the experiment. The chair on which the participants took place was selected not to be an office chair, as these chairs can turn. When the participant is using the Head Mounted Display to look around in the virtual environment sitting on an office chair would mean they would be able to look behind themselves in the environment while their virtual body would still be facing the other way.

The Virtual Environment was presented to the user using a Head Mounted Display, in this case the Oculus Rift Developer Kit (version 2)¹. Using an advanced high-quality Virtual Environment and a Head Mounted Display requires a high-end gaming computer with a high-end graphics card to ensure smooth performance for an optimally effective Virtual Environment. The computer used an Intel i7-4630 CPU with 16GB DDR4 memory, a 500GB SSD and a Nvidia GTX-780 graphics card with 1GB of memory. To facilitate the haptic feedback a so-called gaming vest was used (the KOR-FX²). These vests incorporate vibration motors that mimic physical impact to the torso. The KOR-FX vest uses two large vibration motors, one on left side of the chest and one on the right side. The vest is wirelessly connected to a control box. This control box accepts low-voltage input (0-5V) and is meant to accept standard sound output of the sound card of a computer. To gain complete control over the haptic feedback, an Arduino One board³ has been used with an analogue line (0-5V) as output to the KOR-FX controller box. The Arduino accepted commands from the Virtual

¹<https://www3.oculus.com/en-us/dk2/>

²<http://korfx.com/>

³<https://www.arduino.cc/en/Main/ArduinoBoardUno/>

Environment, via the USB connection to the computer, to activate the vibration motors in the gaming vest. This way the Virtual Environment had complete control over the haptic feedback to the participants. The Arduino also used a microphone that recorded the volume level of the sound in the environment, i.e. the voice of the participant. The microphone polled from a script inside the Virtual Environment to monitor the speech of the participant. A standard USB game controller was used to control the virtual agent of the user. See Figure 1 for an overview of the system’s architecture.

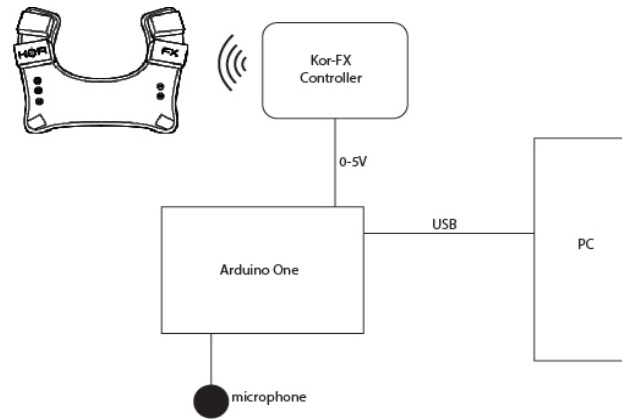


Fig. 1. System architecture.

3.6 Virtual Environment

The Virtual Environment has been developed in Unity Pro (version 5)⁴. A ready-made model from the Unity Asset Store has been purchased for the club environment that has been used in the experiment. This model has been further adapted in order to suit the needs of this research. Atmosphere was added by including special lighting and additional props on the virtual stage. All the humanoid agents in the Virtual Environment have been generated using the iClone Pipeline software (version 6)⁵. The Character Creator⁶ has been used to generate realistic and unique human agents. iClone itself has been used to create the body animations and lip-sync movements.

3DXchange⁷ has been used to convert the agents including their animations into FBX format that could be imported into Unity Pro. Inside Unity

⁴<https://unity3d.com/>

⁵<http://www.reallusion.com/iclone/default.html>

⁶<http://www.reallusion.com/iclone/character-creator/default.html>

⁷<http://www.reallusion.com/iclone/3DXchange.html>

Pro the non-interactive characters were scripted using C#, looping animations and speech to create a livelier atmosphere in the club. The interactive agents, the character in the role of the friend of the participant and Mason, have been separately scripted for more advanced actions. These two agents had a larger set of animations and speech, plus the ability to react to speech of the participants. The agent would monitor if the participant was speaking. If the participants did speak the agent would wait until the participant stopped, allowing for small pauses in speech (of 1 second), or until a maximum amount of time (of 10 seconds) had elapsed. This produced a more realistic reaction of the agent.

Additional scripts made sure that once the participant entered the hallway to the toilets, an encounter with Mason was unavoidable. Both speed and direction of movement of the avatar of the participant were taken over by the script so that the participant and Mason would end up directly in front of each other. A screenshot of the application is shown in Figure 2.



Fig. 2. Screenshot of the application.

3.7 Procedure

After entering the room the participants were asked to sign an informed consent form, allowing for the gathered data to be saved and used for the duration of the research project. Participants also read the health and safety warnings for the Oculus Rift and KOR-FX gaming vest to be able to indicate whether they could safely work with this equipment.

Next, the participants read the experiment instructions and put on the KOR-FX gaming vest. They would take their seat behind the computer and the experimenter would inform them of the instructions once more, highlighting the importance of speaking loud and clear, always responding and using at least one whole sentence to respond with. If the participants had no further questions

they put on the Oculus Rift and the experimenter started a tutorial scenario. In this scenario participants could walk around in order to get accustomed to the controls. The scenario is a grey plain with several blocks placed on it for orientation purposes. The 'ceiling' of the scenario is a sky with a sun. When the participant indicated to understand the controls the experimenter started the recording and the virtual scenario in which all interactions took place.

The participants interacted with the first agent in the environment and then moved on to find the bathroom as instructed. After completing their conversation with virtual agent Mason the screen faded to black and the experimenter stopped the recording. All equipment used was removed and the participant was asked to fill in the questionnaire on the computer. During this the experimenter did not answer any questions the participants had, nor respond to any of their remarks regarding the experiment as not to influence their answers to the questionnaire. For their participation in the experiment, participants were rewarded with a sweet roll after completion of all tasks.

4 Results

This section describes the results of the experiment in detail. First, the subjective measures will be presented, followed by the objective measures.

4.1 Subjective Measures

Figure 3 shows the means of the answers given by the participants to the Likert-scale questions. For example, in condition A (the condition in which haptic feedback was received), the mean of the answers to the question regarding experience with head mounted display devices was 2.33, whereas for condition B it was 2.00.

To analyse whether there was a significant difference between the two conditions regarding the mean answers that were provided to the Likert-scale questions, unpaired t-tests have been performed, under the assumption that the scales reflect continuous data. The results of these tests are displayed in Table 1.

Table 1. T-test results on Likert questions (significance level = 0.05).

Question	P-value	Significant difference
Q1 (HMD experience)	0.37	No
Q2 (haptics experience)	0.62	No
Q3 (scenario realistic)	0.12	No
Q4 (agent aggressive)	0.76	No
Q5 (agent threatening)	0.73	No

Figure 4 displays the results for the yes-no questions for condition A and B, respectively. As an illustration, The figure shows that in condition A, 13

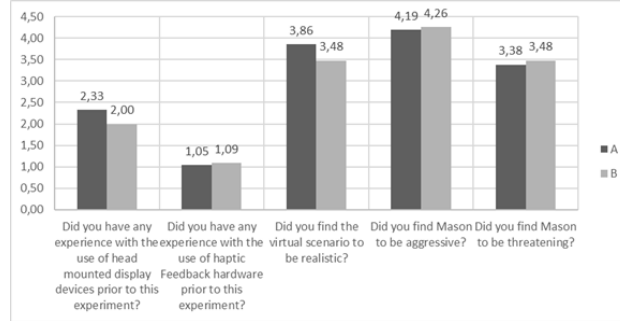


Fig. 3. Answers to the Likert-scale questions.

participants gave a positive answer to Question 6 ('Were you startled when Mason pushed you?'), whereas 8 participants gave a negative answer. Instead, in condition B, 8 participants gave a positive answer to this question, and 15 participants gave a negative answer.

To analyse whether there was a significant difference between the two conditions regarding the answers that were provided, a series of Chi-square tests have been performed. The results of these Chi-square tests are displayed in Table 2.

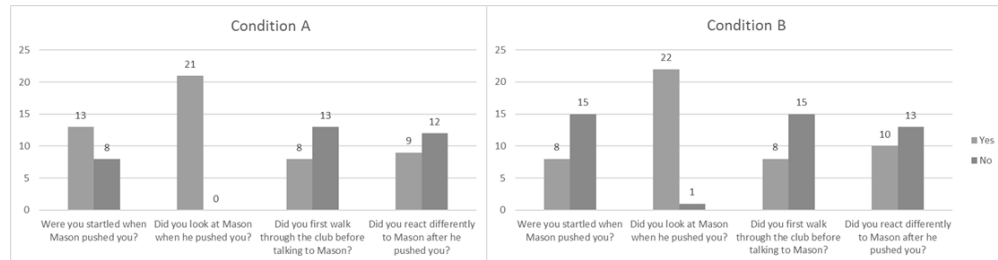


Fig. 4. Answers to yes-no questions for condition A (with haptic feedback) and B (without haptic feedback).

Table 2. Chi-square test results on yes-no questions (significance level = 0.05).

Question	Chi-square value	Variable independence
Q6 (startled by push)	0.07	Yes
Q7 (looked at agent)	0.79	Yes
Q8 (walked around)	0.99	Yes
Q9 (different after push)	0.99	Yes

4.2 Objective Measures

Objective data were obtained by analysis of the audio recording of the experiment sessions. From the audio files, obtained during the experiment sessions using a web cam, the amplitudes of the speech sample concerned with the conversation with Mason have been extracted using Audacity⁸, an audio editing tool. These data were then processed using a script written in Python 2.7 in order to obtain the mean amplitude of the utterances prior to the haptic feedback event (i.e., the virtual push) and of the utterances after the event. This was done to determine whether participants spoke louder or softer after being pushed. A summary of the results is presented in Table 3.

Table 3. Comparison of speech volume before and after the virtual push.

Condition	Louder	Softer
A (haptic feedback)	15	6
B (no haptic feedback)	10	13

Statistical analysis of these data was performed again using Chi-square tests. The Chi-square value of this test was 0.06.

5 Evaluation

In this section the results that have been presented in the previous section will be evaluated in the context of the original research questions. First, the subjective results will be evaluated (Research Question 1) and thereafter the objective results (Research Question 2). Finally, a brief follow-up experiment is described, along with its results.

5.1 Research Question 1

Table 1 shows there is no significant difference between the ratings for prior experience between the experimental and control group (Q1 and Q2). The fact that there is no significant difference between the two groups in the level of experience with any of the devices used, indicates that any effect that is found can not be contributed to this.

There was also no significant difference found between how aggressive (Q4) or threatening (Q5) the participants perceived virtual agent Mason to be. Neither was there any difference between how realistic the experience was for the participants (Q3). This would indicate that the haptic feedback that was provided to participants in condition A did not affect any of these factors. Participants in the control group have indicated to have had almost the same experience as those participants that did receive haptic feedback.

⁸<http://www.audacityteam.org/>

The Chi-square tests applied on the yes-no questions, displayed in Table 2, show that there is variable independence between the experimental group to which the participants were allocated (A or B) and their answers to these questions. This means that the group in which the participants were placed did not affect their answer to these questions. Therefore, on the one hand, any effect of the haptic feedback cannot be attributed to some of the participants looking at Mason and others looking away (Q7) or some participants walking around in the environment first and others heading straight for their goal (Q8). On the other hand, this also means that no effect of haptic feedback on the reaction of the participants (Q9) or them being startled by the virtual push (Q6) is found.

However, the statistical test performed to determine variable independence between participants being startled and receiving haptic feedback returned a P-value of 0.07. In addition, several people in the haptic feedback condition mentioned that they found the feedback experience at least 'surprising'. As this was an experiment with a relatively low number of participants (44) and the significance value used for this test was 0.05, it is advisable to perform a second experiment with a larger sample size in order to determine whether there is actually no correlation between participants receiving haptic feedback and being startled by the virtual agent.

5.2 Research Question 2

The data obtained after processing of the audio files seem to suggest that participants that received haptic feedback on average spoke louder after receiving this feedback compared to participants in the control group. However, the Chi-square test shows variable independence, indicating that the pattern could be obtained through chance. Just as with the subjective data regarding the startling of participants it is important to remark that the Chi-square value is 0.06. Therefore, it would be advisable to perform a second experiment with a larger sample size in order to determine whether there actually is no pattern between loudness of speech and the application of haptic feedback.

5.3 Follow-up Experiment

In order to investigate the effect of haptic feedback on the experience of being startled (Q6) with a (slightly) larger sample, ten additional participants performed the experiment at a later date. Due to problems with the equipment, this follow-up experiment could not be conducted for the objective measures. Since a second analysis of the data would be performed (including the data from the first 44 participants) the significance level was adjusted to 0.025, under the assumption that adding more participants would otherwise always lead to some kind of significant effect.

Analysing the subjective data regarding the startling effect for all 54 participants yielded the results presented in Table 4. After increasing the number of participants, there still seems to be no significant effect of startling. On the contrary: the Chi-square value has increased from 0.07 to 0.1.

Table 4. Chi-square test results on question Q6 (54 participants, significance level = 0.025).

Condition	Yes	No
A (haptic feedback)	15	12
B (no haptic feedback)	9	18
Question	Chi-square value	Variable independence
Q6 (startled by push)	0.1	Yes

6 Conclusion

In this research the effects of negative haptic feedback in the form of a 'push' by a virtual agent in a threatening scenario are explored. To this end an experiment was set-up featuring 44 participants, distributed over two conditions. During this experiment participants interacted with two virtual agents through free speech in a virtual environment. For this an Oculus Rift and the KOR-FX gaming vest have been used. Participants in the experimental group received haptic feedback, through vibrations created by the KOR-FX vest, at a certain point during their conversation with one of the virtual agents. At this moment in the conversation the participants were being attacked by the virtual agent in the form of a push that was synchronised with the haptic feedback that was received. The participants in the control group also used both the Oculus Rift and the KOR-FX vest, but did not receive haptic feedback during their interactions with the virtual agent.

Subjective data were obtained from a questionnaire that was filled in by participants after the experiment had been completed. Objective data were obtained through a recording of the experiment session using a web cam. The audio recording of the experiment was analysed in order to determine the loudness of speech of the participants prior to- and after the haptic feedback event.

Statistical tests indicate that haptic feedback did not have any effect on the experience of the participants in this scenario for the measured variables. The Chi-square test that was performed on the loudness of speech resulted in a value of 0.06, which was close to the significance level of 0.05. In the questionnaire participants were asked whether they were startled when they were pushed in the scenario. The statistical test performed on these answers resulted in a Chi-square value of 0.07 for a significance level of 0.05. As this study featured 44 participants, ten additional participants performed the experiment in order to gain better insight in the near-significant effects. A second statistical analysis of the subjective data for the startling effect resulted in a value of 0.1. As a consequence, no significant effects of the haptic feedback could be demonstrated on the various subjective aspects (Research Question 1) and objective aspects (Research Question 2) measured.

7 Discussion

Despite the fact that no statistically significant effects of haptic feedback were found, this research provides several useful pointers for follow-up research. First of all, the fact that no effect on subjective experience could be demonstrated might be related to the particular set-up of the experiment, which used a relatively low number of participants, and a between-participants design. Since participants played the scenario only once, they had no frame of reference to which they could compare their experience, which made their Likert-scale response difficult to interpret. It might be the case that if participants would experience both conditions (with and without haptic feedback), they would still feel a big difference between them. This is a common problem in user experience research, and it is worthwhile to explore this in more detail.

Secondly, although this research has explored the effect of negative haptic feedback on several aspects of the experience of the user, it has not been exhaustive in that regard. Future research might look into the effect on loudness of speech using more participants in order to ascertain whether haptic feedback might have an effect or not. Other considerations for future research might include alternative ways in which negative haptic feedback influences user experience, the role of the intensity of the feedback (possibly up to the point where the feedback actually hurts), and providing haptic feedback multiple times.

All in all, it is concluded that the lack of significant effects found in the present study should rather be explained by the specific design of this experiment than by the paradigm as a whole, and that follow-up research is required to investigate the full potential of threatening virtual agents based on haptic feedback.

References

1. J. N. Bailenson and N. Yee. Virtual interpersonal touch: Haptic interaction and copresence in collaborative virtual environments. *Multimedia Tools Appl.*, 37(1):5–14, 2008.
2. J. Bates. The role of emotions in believable agents. *Communications of the ACM*, 37(7):122–125, 1994.
3. R. T. Bell. *Sociolinguistics*. BT Batsford, New York, 1976.
4. T. W. Bickmore, R. Fernando, L. Ring, and D. Schulman. Empathic touch by relational agents. *IEEE Transactions on Affective Computing*, 2010.
5. R. Blankendaal, T. Bosse, C. Gerritsen, T. de Jong, and J. de Man. Are aggressive agents as scary as aggressive humans? In *Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems, AAMAS 2015, Istanbul, Turkey, May 4-8, 2015*, pages 553–561, 2015.
6. T. Bosse, C. Gerritsen, and J. de Man. An intelligent system for aggression de-escalation training. In *ECAI 2016 - 22nd European Conference on Artificial Intelligence, 29 August-2 September 2016, The Hague, The Netherlands - Including Prestigious Applications of AI (PAIS 2016)*, pages 1805–1811, 2016.
7. C. M. de Melo, P. J. Carnevale, and J. Gratch. The effect of expression of anger and happiness in computer agents on negotiations with humans. In *10th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2011), Taipei, Taiwan, May 2-6, 2011, Volume 1-3*, pages 937–944, 2011.

8. D. DeVault. Simsensei kiosk: a virtual human interviewer for healthcare decision support. In *Int. conf. on Autonomous Agents and Multi-Agent Systems, AAMAS '14, Paris, France, May 5-9, 2014*, pages 1061–1068, 2014.
9. G. Frank, C. Guinn, R. Hubal, P. Pope, M. Stanford, and D. Lamm-Weisel. Just-talk: An application of responsive virtual human technolog. In *Proceedings of the Interservice/Industry Training, Simulation and Education Conference*, 2011.
10. G. Huisman, J. Kolkmeier, and D. Heylen. With us or against us: Simulated social touch by virtual agents in a cooperative or competitive setting. In *Proceedings of the 14th International Conference on Intelligent Virtual Agents, IVA 2014*, pages 204–213, 2014.
11. R. S. Lazarus. Cognition and motivation in emotion. *American Psychologist*, 46:352–367, 1991.
12. S. Marsella, J. Gratch, and P. Petta. Computational models of emotion. *A blueprint for an affectively competent agent: Cross-fertilization between Emotion Psychology, Affective Neuroscience, and Affective Computing*, 2010.
13. S. C. Marsella and J. Gratch. EMA: A process model of appraisal dynamics. *Cognitive Systems Research*, 10(1):70–90, 2009.
14. A. Nieuwenhuys and R. R. D. Oudejans. Training with anxiety: short- and long-term effects on police officers’ shooting behavior under pressure. *Cognitive Processing*, 12(3):277–288, 2011.
15. J. Rickel. Intelligent virtual agents for education and training: Opportunities and challenges. In *Intelligent Virtual Agents, Third International Workshop, IVA 2001*, volume 2190 of *Lecture Notes in Computer Science*, pages 15–22, 2001.
16. A. Rizzo, G. Reger, G. Gahm, J. Difede, and B. O. Rothbaum. Virtual reality exposure therapy for combat-related PTSD. In *Post-Traumatic Stress Disorder*, pages 375–399, 2009.
17. M. Schroeder, E. Bevacqua, R. Cowie, F. Eyben, H. Gunes, D. Heylen, M. ter Maat, G. McKeown, S. Pammi, M. Pantic, C. Pelachaud, B. Schuller, E. de Sevin, M. Valstar, and M. Woellmer. Building autonomous sensitive artificial listeners. *IEEE Transactions on Affective Computing*, 3(2):165–183, 2012.
18. J. K. S. Teh, A. D. Cheok, R. L. Peiris, Y. Choi, V. Thuong, and S. Lai. Huggy pajama: A mobile parent and child hugging communication system. In *Proc. of the 7th int. conference on Interaction design and children*, pages 250–257, 2008.
19. D. R. Traum, W. R. Swartout, P. Khooshabeh, S. Kopp, S. Scherer, and A. Leuski, editors. *Intelligent Virtual Agents - 16th International Conference, IVA 2016, Los Angeles, CA, USA, September 20-23, 2016, Proceedings*, volume 10011 of *Lecture Notes in Computer Science*, 2016.
20. A. B. Youssef, M. Chollet, H. Jones, N. Sabouret, C. Pelachaud, and M. Ochs. Towards a socially adaptive virtual agent. In *Intelligent Virtual Agents - 15th International Conference, IVA 2015*, volume 9238 of *Lecture Notes in Computer Science*, pages 3–16, 2015.
21. E. Y. Zhang and A. D. Cheok. A networked device for reproducing multisensory kissing. In *Proceedings of the 2016 workshop on Multimodal Virtual and Augmented Reality*, 2016.
22. Y. C. Zhao, G. Kennedy, K. Yukawa, B. Pyman, and S. O’Leary. Can virtual reality simulator be used as a training aid to improve cadaver temporal bone dissection? results of a randomized blinded control trial. *The Laryngoscope*, 121(4):831–837, April 2011.
23. C. Zoll, S. Enz, H. Schaub, R. Aylett, and A. Paiva. Fighting bullying with the help of autonomous agents in a virtual school environment. In *Proceedings of the 7th International Conference on Cognitive Modelling (ICCM)*, 2006.

Tracking Perceptual and Memory Decisions by Decoding Brain Activity

Marieke van Vugt¹, Armin Brandt², Andreas Schulze-Bonhage²

¹ Institute of Artificial Intelligence & Cognitive Engineering, University of Groningen, The Netherlands m.k.van.vugt@rug.nl

² Epilepsy Center, University Medical Center Freiburg, Germany

Abstract. Decision making is thought to involve a process of evidence accumulation, modelled as a drifting diffusion process. This modeling framework suggests that all single-stage decisions involve a similar evidence accumulation process. In this paper we use decoding by machine learning classifiers on intracranially recorded EEG (iEEG) to examine whether different kinds of decisions (perceptual vs. memory) exhibit dynamics consistent with such drift diffusion models. We observed that decisions are indeed decodable from brain activity for both perceptual and memory decisions, and that the time courses for these types of decisions appear to be quite similar. Moreover, the high spatial resolution of iEEG revealed that perceptual and memory decisions rely on slightly different brain areas. While the accuracy of decision decoding can still be improved, these initial studies demonstrate the power of decoding analyses for testing computational models of cognition.

1 Introduction

Decision making is a basic cognitive process that comes to play in many different tasks. Most of the research on decision making focuses on simple tasks such as detecting the direction of randomly moving dots. The theories developed on the basis of those experiments presume that all decisions between two alternatives (at least those consisting of a single stage process) behave with similar dynamics. Specifically, according to drift diffusion models (DDMs; Ratcliff, 1978), decisions follow a drifting diffusion process, where the random walk is driven by the decision information. The DDM starts the process of evidence accumulation at the moment the stimulus comes on the screen, and then slowly drifts towards one of the decision thresholds which each correspond to a particular decision option. As soon as the decision threshold is reached, the response corresponding to the relevant decision option is given. The model has been found to produce excellent fits to performance in a variety of tasks, as well as the detailed shape of the associated response time distributions. The parameters of this model each can be interpreted as specific cognitive processes such as attention allocation in the drift rate parameter and speed-accuracy trade-off in the location of the decision threshold (Ratcliff, 2016). In addition to the drift rate and the decision threshold, the third main parameter of the model is the non-decision time, which

reflects non-decision-related processes such as preparing a motor response and fixed delays in the perceptual system. By varying the values of these parameters, subtle differences in shapes of the response time distributions can be reproduced.

While DDMs were developed exclusively based on behavioral data, more recently it has also been suggested that the brain may implement such diffusion processes. For example, in seminal work, Shadlen and colleagues observed monotonously increasing firing rates of neurons in the lateral intraparietal area while monkeys were deciding about the direction of randomly moving dots (Shadlen and Newsome, 1996, Roitman & Shadlen, 2002). This neural signature was modulated by the strength of the decision evidence (the proportion of dots moving coherently) and the traces seemed to all move up to the same final firing rate around the time of the response. Subsequent studies on monkeys in similar tasks instead placed evidence accumulation in the frontal eye fields (Ferrera et al., 2009, Hanes & Schall, 1996, Purcell et al., 2010), superior colliculus (Ding & Gold, 2012) and caudate (Ding & Gold, 2010). Some of the differences between studies could be traced back to the response modality (e.g., accumulation-like activity is more likely in frontal eye field when monkeys use saccades to indicate their response than when they use reaching).

In humans, accumulation processes have been studied as well, although in that case the challenge is the trade-off between poor temporal resolution of functional magnetic resonance imaging (fMRI) and the poor spatial resolution of electroencephalograph (EEG). fMRI studies have suggested evidence accumulation may take place in the dorsolateral prefrontal cortex (Heekeren et al., 2006), inferior frontal gyrus (Ho et al., 2009, Krueger et al., 2017, Ploran et al., 2007) but as demonstrated in a meta-analysis, in fact almost the whole brain (Mulder et al., 2014). Using EEG, we found neural correlates of evidence accumulation in parietal 4–9Hz theta oscillations when people were making decisions about randomly moving dots (van Vugt et al., 2012). MEG (magnetoencephalography) studies have implicated different brain regions in the accumulation process, such as 14–24Hz beta oscillations over motor cortex; Donner et al., 2009). In addition to these brain oscillations, it has been suggested that two event-related potentials—the centroparieto potential (CPP; O’Connor et al., 2012), and the lateralized readiness potential (van Vugt et al., 2014)—reflect evidence accumulation. While the CPP may arise from parietal cortex, the lateralized readiness potential may come from premotor areas of the brain. However, none of this localization is very specific since it is derived from scalp-recorded EEG, which has poor localization.

An alternative approach to localizing the decision process in the brain has been to use classifiers, which are increasingly popular in neuroscience. The first studies using these methods focused on finding specific moments in time at which decisions can be best classified, rather than tracking the complete decision process over time. For example, Ratcliff and colleagues (2009) found that a logistic regression-based classifier in the period around 400 ms post-stimulus exhibited behavior consistent with evidence accumulation during a face-car discrimination task—the output of this classifier covaried with between-trial differences in the

drift rate. Philiastides and colleagues (2014) followed up on this in a similar face-house discrimination task and showed that a Fisher discriminant analysis could also track the decision process over time and this process appeared to be predominantly localized to parietal regions.

While these results are promising, they do not give very detailed localization of the decision process and restrict themselves to the cortex due to the inherent constraints of EEG data. Moreover, as is clear from the above discussion, different studies have claimed that evidence accumulation occurs in many different brain areas, dependent on the study. One potential reason for contradictory results may be that decisions on the basis of different kinds of evidence may be implemented by different brain regions. For this reason, it is worthwhile to examine whether evidence accumulation looks similar for different kinds of decisions, such as perceptual decisions and decisions about remembered information.

To enhance spatial localization we decode decision information from intracranially-recorded brain oscillations (rather than the scalp EEG used in most previous studies). Intracranial EEG is data with a high degree of spatial and temporal precision that can be obtained from epileptic patients who are implanted with electrodes for clinical purposes (Jacobs & Kahana, 2010). To determine what brain areas are involved in decision making, and where the decision information is available over time, we ran a regularized logistic regression classifier in short (50-ms) time bins, and assessed how classification accuracy develops for memory and perceptual decisions. We focused on classifying 4–9Hz theta oscillations, since we previously demonstrated that those are most informative for decision making (van Vugt et al., 2012). We then looked at the classifier weights to determine what Brodmann areas carry most of the decision information, and whether those differed between perceptual and memory decisions.

2 Methods

2.1 Participants

Participants were recruited from the patients undergoing long-term invasive monitoring for pharmacologically intractable epilepsy at Freiburg University hospital (Germany). Sixteen individuals were recruited and participated in our behavioral experiments.

2.2 Task

To be able to compare perceptual and memory decisions, we created a task with perceptual and memory conditions that used the same set of stimuli. We created synthetic face stimuli by means of the Basel Face model (Paysan et al., 2009), which allowed us to manipulate the stimulus similarity (and hence task difficulty) very precisely. In the perceptual condition (Fig. 1(a)), participants saw two faces facing outward and had to determine whether this face belonged to the same person (i.e., was simply a rotated version of the same face). In the

memory condition (Fig. 1(b)), participants were first shown two faces during a 2000–2075ms (jittered) study period, followed by a 1000–1150ms (jittered) blank delay period, after which a probe face was shown. Participants were then asked to indicate by a button press whether this probe face was identical to one of the two faces presented in the study. The jitter in the task was used to ensure that no spurious oscillatory phase-locking could occur.

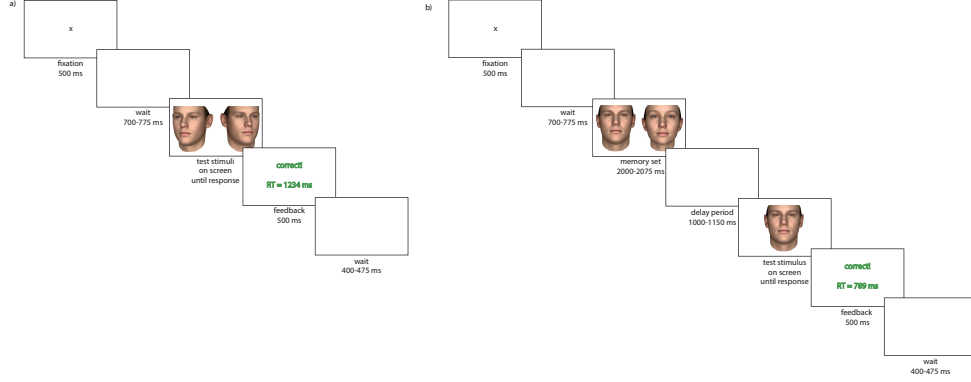


Fig. 1. Example trials of the perceptual (a) and memory (b) condition.

2.3 Recordings

Data were recorded with a 2000Hz sampling rate on the clinical EEG recording system (Compumedics). We then segmented the data into trials of 4000 ms duration, starting 200 ms prior to the onset of the probe stimulus. We checked for the occurrence of epileptiform activity and one of the participants' data had to be discarded due to epileptic spikes in a majority of the trials. Trials whose response time exceeded the trial segment duration (3800 ms) were discarded, and the classifier analysis was done only on correct trials. Similarly, trials with a kurtosis larger than 15 (indicative of epileptic spikes) were removed. The total dataset involves 1178 electrodes.

2.4 Data analysis

Data were analyzed by means of in-house matlab code that was based on toolboxes developed by Jelmer Borst and Per Sederberg. First, we ensured that the two classes to be separated (match/non-match decisions) had an equal number of trials by randomly removing trials from the larger class. We then performed a wavelet transform to obtain EEG time courses in the 4–9Hz theta band, which previously has been shown to be important for decision making (van Vugt et al., 2012). Next, we z-transformed all trials to ensure the data had an average of zero and a standard deviation of one. We then vincentized the data—that is, we turned each trial into an equal number of bins between the stimulus and

the response (with the exception of the first 300 ms, which contains roughly the same peaks irrespective of the response time, so this period was not stretched or compressed and simply divided into 6 bins of 50 ms duration). The number of bins was chosen such that on average, bin duration would be approximately 50 ms.

For each time bin, we trained a regularized logistic regression model to distinguish between the match and non-match responses. Essentially, this model tries to find a matrix \mathbf{W} that maps between the $n \times p$ matrix of examples \mathbf{X} (p is the number of features, n is the number of trials) and the vector of labels (match or non-match) \mathbf{Y} : $\mathbf{W} = (\mathbf{X}^T \mathbf{X} + \lambda \mathbf{I}_p)^{-1} \mathbf{X}^T \mathbf{Y}$. In this equation, \mathbf{I}_p is the $p \times p$ identity matrix and λ is the regularization matrix. The regularization allows the algorithm to deal with many correlated predictors. For each classifier the regularization parameter lambda was determined by means of a search between 0 and 10,000 (Borst et al., 2013). The lambda that minimized the root-mean-square prediction error across all labels was chosen. We then assessed the classifier’s performance using 10-fold cross-validation.

3 Results

3.1 Accuracy across subjects

We first examined how well decisions could be decoded from intracranial EEG data per participant. Fig. 2 illustrates the maximum accuracy across the decision interval (between the moment the probe stimulus came on the screen and the response). As can be seen, there are large differences between individuals in classification accuracy, which ranges from close to chance (55%) to 76%. Most accuracies significantly exceed chance according to a binomial proportion test (see stars in Fig. 2; $p < 0.016$, reflecting a 5% False Discovery Rate).

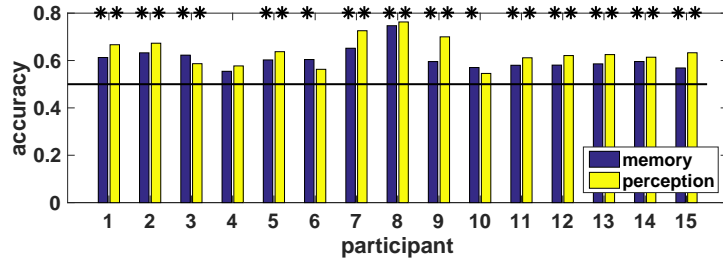


Fig. 2. Maximum decision decoding accuracy for each participant, separately for perceptual and memory trials. Stars indicate classification accuracies that are larger than chance.

Having established that decision classification is possible to some extent, we can now examine our main question: what is the time course of these classifications, and does it differ between perceptual and memory decisions? Fig. 3 demonstrates that while for some participants there is no classification possible, and

Table 1. DDM parameters (mean and standard error of the mean). Perceptual and memory trials were split between low and high-similarity conditions (difficult and easy, respectively). Decision threshold was kept fixed between all conditions. Non-decision time was fixed between the similarity conditions. This model came out as best from a BIC comparison of various model configurations.

condition	drift	decision threshold	non-decision time (s)
perception low similarity	-0.41 (0.13)	0.27 (0.025)	0.92 (0.15)
perception high similarity	0.24 (0.066)	0.27 (0.025)	0.92 (0.15)
memory low similarity	-0.11 (0.014)	0.27 (0.025)	0.50 (0.047)
memory high similarity	0.11 (0.016)	0.27 (0.025)	0.50 (0.047)

the signal hovers around chance level, for others there is meaningful classification, and consistent with the DDM, classification accuracy increases slowly over time (the slope of the classification accuracy is larger than zero for both memory decisions ($t(14)=2.55$, $p = 0.012$) and for perceptual decisions ($t(14)=3.04$, $p = 0.0044$)). Unexpectedly, classification accuracy appears to continue to increase even after the response has been made. One possible interpretation of such a pattern is provided by recent modeling studies that suggest that after the decision has been made, the accumulation process continues with the objective of estimating decision confidence (Pleskac & Busemeyer, 2010).

Comparing the two conditions, classifier accuracy appears to get higher for perception than for memory trials ($M_{memory}=0.61$; $M_{perception}=0.64$; $t(14)=2.82$, $p = 0.014$). In short, while perceptual and memory decisions involve quite different tasks and response times, the decodable decision information forms quite a similar trajectory for both, consistent with the DDM predictions. The only difference is that the classifiers start to increase at different points in time, presumably because the response time in the perception condition is significantly longer than in the memory condition. This difference in response time is presumably caused by the fact that in the perception condition, participants need to first mentally rotate the images before they can make their decision. DDM fits are consistent with this idea (see parameters in Tab. 1): there is a significant effect of task on the non-decision time parameter ($F(1, 60)=14.1$, $p = 0.0004$, similar to previous studies of mental rotation decisions). Another interesting finding is that classification accuracy appears to continue increasing even after the decision has been made. This is consistent with some findings from experiments with monkeys suggesting that after the decision has been made, participants continue to accumulate information to make estimates of their confidence in the decision.

3.2 How do different brain areas contribute to classification?

Next, we asked what brain areas are involved in classification, and how these regions differ between perceptual and memory decisions. Specifically, for every Brodmann area, we reported the proportion of electrodes in that area that were significantly involved in evidence accumulation (i.e., having z-scores larger than 2). Fig. 4 shows the electrodes that were, across participants most involved in

classification, separately for perception and memory decisions. It is clear that most of these electrodes are in lateral parietal and temporal areas. Part of the reason for that is of course that the locations that are relevant for clinical purposes tend to be temporal areas, which are often the sources of epileptic seizures. Table 2 demonstrates that the Brodmann areas that have the largest proportion of electrodes carrying decision information are parietal areas (Brodmann area 7) and perceptual-motor areas (Brodmann areas 1-2-3-5).

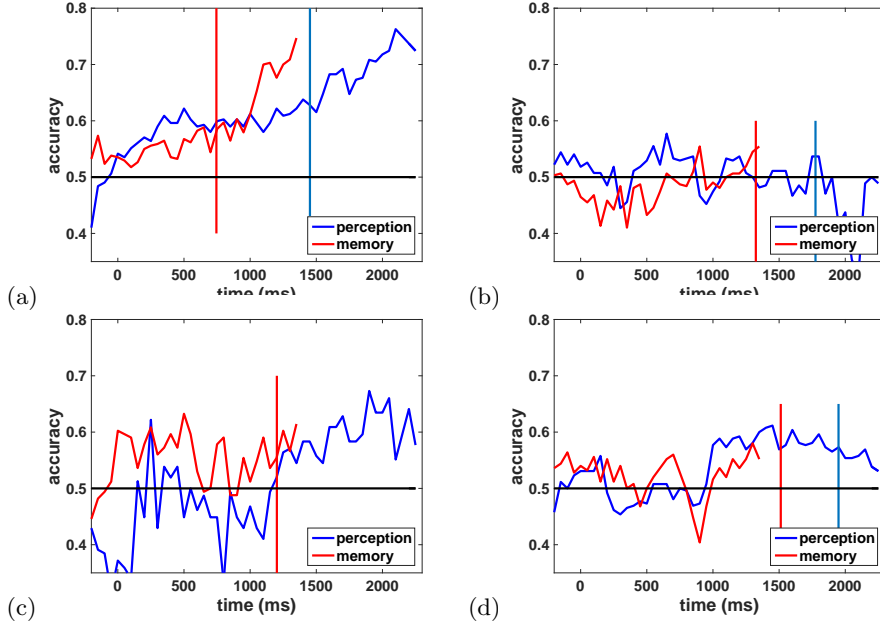


Fig. 3. Time courses of classification accuracy in the 4–9Hz theta band for a participant with good classification (a) and for a participant with poor classification (b). For comparison, two other participants are also shown ((c) and (d)). Vertical lines indicate the average time of the response for the relevant condition (blue for perception, red for memory).

Discussion

We examined the time course of the availability of decision information during a perceptual and a memory decision task. As the DDM predicted, overall the decoded decision evidence shows similar dynamics for perceptual and memory decisions. The difference between the two lies in the time at which they move upwards, which is later for the perception condition than for the memory condition. This is not surprising given that for the perception task the response time is significantly longer than in the memory task. The second question we asked was whether accumulation in the perception and memory conditions relies

on different brain regions. Because we have no full brain coverage we can only make tentative claims, but the data so far suggest that there are differences between those two types of decisions (see also Fig. 4). While memory decisions rely predominantly on Brodmann area 7 (parietal cortex) and sensorimotor cortex, perceptual decisions rely more on Brodmann area 13 (anterior insula) and 30 (visual area).

While we did obtain classification accuracies above chance, classification is far from stellar. Potentially other classifiers such as lasso or artificial neural nets could do better than these. On the other hand, such classifiers run a higher risk of overfitting the data. Another potential approach could be to make use of the known similarity structure of the face stimuli. Previous studies have shown that such decoded similarity structures can help to track memory representations in the brain (Zhang et al., 2015). In addition, we focused here on the theta band, because that frequency was suggested by prior studies. Nevertheless, it could easily be the case that for these data, better decision information could be decoded from other frequencies. Finally, there are large differences between individuals. We examined whether individual differences in task performance (accuracy and response time) could account for these differences in classification accuracy. We found that the data were too unreliable to make any connection between task performance and classifier accuracies (all Bayes Factors between 0.4 and 2.2).

Table 2. Proportion of significant electrodes by Brodmann area for classification of target-lure/match-nonmatch decisions on the basis of 4–9Hz theta activity. The Brodmann areas are ordered by proportion of significant electrodes. Only Brodmann areas with more than 15 electrodes are included. Brodmann areas for which the proportion of significant electrodes is zero: 9, 22, 28,40, 47 and hippocampus.

Brodmann area	memory	perception	$N_{electrodes}$	$N_{participants}$
Brodmann area 7	0.17	0.06	18	1
Brodmann area 13	0.03	0.09	34	10
Brodmann area 20	0.01	0.07	179	13
Brodmann area 36	0.02	0.07	46	9
Brodmann area 37	0.01	0.04	98	12
Brodmann area 41/42	0.03	0.03	33	8
Amygdala	0.00	0.03	31	9
Brodmann area 21	0.01	0.03	246	13
Brodmann area 19	0.02	0.00	57	9
Brodmann area 38	0.02	0.00	101	10

Another weaker part of this study is that it did not make a direct connection to DDM parameters. If the dynamics of the decoded decision process were to covary with model parameter estimates (van Vugt et al., 2012), this would bolster our confidence that we are in fact observing neural correlates of a drift diffusion process. One approach that we can use in the future to examine this is to use the classifier readout to separate the trials into low-evidence and high-evidence,

and then to fit the DDM separately to these classes of trials. Previous studies using such an approach have demonstrated that such within-participant model verification can be used to identify neural correlates of drift diffusion processes (Ratcliff et al., 2009; Ratcliff et al., 2016b).

In short, we have demonstrated how decoding decision information from brain data can help us to better understand the dynamics of decision making. This builds on a larger body of work that uses classifiers to track covert cognitive processes over time. As classifiers become more powerful and can deal with more noisy data, these are very useful tools to help us uncover how the brain “does” cognition. Moreover, we will gain a better and better time-resolved picture of cognitive processes, which can subsequently inform computational models of these processes.

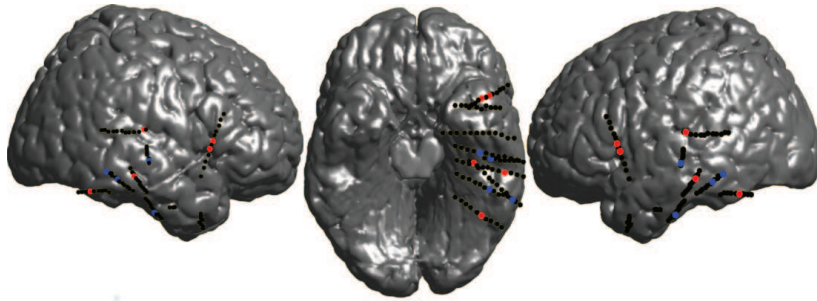


Fig. 4. Electrodes that contribute significantly ($z > 2$) to decision classification. Black dots indicate electrodes that are not showing significant decision-related activity. Red: perceptual decisions. Blue: memory decisions. The electrode locations for two participants were missing, so those are not included in this figure and in Table 1. In addition, Brodmann areas having no electrodes in our dataset are also excluded from the table.

References

- Ratcliff, R.: A theory of memory retrieval. *Psychological Review* **85** (1978) 59–108
- Ratcliff, R., Smith, P.L., Brown, S.D., McKoon, G.: Diffusion decision model: current issues and history. *Trends in Cognitive Sciences* (2016)
- Shadlen, M.N., Newsome, W.T.: Motion perception: seeing and deciding. *Proc Natl Acad Sci U S A* **93** (1996) 628–633
- Roitman, J.D., Shadlen, M.N.: Response of neurons in the lateral intraparietal area during a combined visual discrimination reaction time task. *J Neurosci* **22**(21) (2002) 9475–9489
- Ferrera, V.P., Yanike, M., Cassanello, C.: Frontal eye field neurons signal changes in decision criteria. *Nature Neuroscience* **12** (2009) 1458–1462
- Hanes, D.P., Schall, J.D.: Neural control of voluntary movement initiation. *Science* **274** (1996) 427–430
- Purcell, B.A., Heitz, R.P., Cohen, J.Y., Schall, J.D., Logan, G.D., Palmeri, T.P.: Neurally constrained modeling of perceptual decision making. *Psychological Review* **117**(4) (2010) 1113–

- Ding, L., Gold, J.I.: Neural correlates of perceptual decision making before, during, and after decision commitment in monkey frontal eye field. *Cerebral Cortex* **22** (2012) 1052–1067
- Ding, L., Gold, J.I.: Caudate encodes multiple computations for perceptual decisions. *J Neurosci* **30** (2010) 15747–15759
- Heekeren, H.R., Marrett, S., Ruff, D.A., Bandettini, P.A., Ungerleider, L.G.: Involvement of human left dorsolateral prefrontal cortex in perceptual decision making is independent of response modality. *Proc. Nat. Acad. Sci., USA* **103**(26) (2006) 10023–10028
- Ho, T., Brown, S., Serences, J.T.: Domain general mechanisms of perceptual decision making in human cortex. *J Neurosci* **29**(27) (2009) 8675–8687
- Krueger, P.M., van Vugt, M.K., Simen, P., Nystrom, L., Holmes, P., Cohen, J.D.: Evidence accumulation detected in BOLD signal using slow perceptual decision making. *J Neurosci Meth* **281** (2017) 21–32
- Ploran, E.J., Nelson, S.M., Velanova, K., Donaldson, D.I., Petersen, S.E., Wheeler, M.E.: Evidence accumulation and the moment of recognition: dissociating perceptual recognition processes using fMRI. *J Neurosci* **27**(44) (2007) 11912–11924
- Mulder, M.J., van Maanen, L., Forstmann, B.U.: Perceptual decision neurosciences - a model-based review. *Neuroscience* **277** (2014) 872–884
- van Vugt, M.K., Simen, P., Nystrom, L., Holmes, P., Cohen, J.D.: EEG oscillations reveal neural correlates of evidence accumulation. *Frontiers in Human Neuroscience* **6** (2012) 106
- Donner, T., Siegel, M., Fries, P., Engel, A.K.: Buildup of choice-predictive activity in human motor cortex during perceptual decision making. *Current Biology* **19** (2009) 1581–1585
- O'Connell, R.G., Dockree, P.M., Kelly, S.P.: A supramodal accumulation-to-bound signal that determines perceptual decisions in humans. *Nature Neuroscience* **15**(12) (2012) 1729–1735
- van Vugt, M.K., Simen, P., Nystrom, L., Holmes, P., Cohen, J.D.: Lateralized readiness potentials reveal properties of a neural mechanism for implementing a decision threshold. *PLoS ONE* **9**(3) (2014) e90943
- Ratcliff, R., Philiastides, M., Sajda, P.: Quality of evidence for perceptual decision making is indexed by trial-to-trial variability of the EEG. *Proc. Nat. Acad. Sci., USA* **106**(16) (2009) 6539
- Philiastides, M.G., Heekeren, H.R., Sajda, P.: Human scalp potentials reflect a mixture of decision-related signals during perceptual choices. *J Neurosci* **34**(50) (2014) 16877–16889
- Jacobs, J., Kahana, M.J.: Direct brain recordings fuel advances in cognitive electrophysiology. *Trends in Cognitive Sciences* **14**(4) (2010) 162–171
- IEEE: A 3D Face Model for Pose and Illumination Invariant Face Recognition, Genova, Italy, IEEE (2009)
- Borst, J., Schneider, D.W., Walsh, M.M., Anderson, J.R.: Stages of processing in associative recognition: Evidence from behavior, EEG, and classification. *J Cogn Neurosci* **25**(12) (2013) 2151–2166
- Pleskac, T., Busemeyer, J.R.: Two-stage dynamic signal detection: A theory of choice, decision time, and confidence. *Psychological Review* **117**(3) (2010) 864–901
- Zhang, H., Fell, J., Staresina, B., Weber, B., Elger, C.E., Axmacher, N.: Gamma power reductions accompany stimulus-specific representations of dynamic events. *Current Biology* **25** (2015) 635–640
- Ratcliff, R., Sederberg, P.B., Smith, T., Childers, R.: A single trial analysis of EEG in recognition memory: Tracking the neural correlates of memory strength. *Neuropsychologia* **93** (2016) 128–141
- Provost, A., Heathcote, A.: Titrating decision processes in the mental rotation task. *Psychological Review* **122**(4) (2015) 735–754

The origin of mimicry

Deception or merely coincidence?

Bram Wiggers¹ and Harmen de Weerd^{1,2}

¹ Institute of Artificial Intelligence, University of Groningen

² Professorship User Centered Design, Hanze University of Applied Sciences

Abstract. One of the most remarkable phenomena in nature is mimicry, in which one species (the mimic) evolves to imitate the phenotype of another species (the model). Several reasons for the origin of mimicry have been proposed, but no definitive conclusion has been found yet. In this paper, we test several of these hypotheses through an agent based co-evolutionary model. In particular, we consider two possible alternatives: (1) Deception, in which mimics evolve to imitate the phenotype of models that predators avoid to eat, and (2) Coincidence, in which models evolve a warning color to avoid predation, which coincidentally benefits the mimics. Our agent-based simulation shows that both these hypotheses are plausible origins for mimicry, but also that once a mimicry situation has been established through coincidence, mimics will take advantage of the possibility for deception as well.

1 Introduction

One of the most remarkable phenomena in nature is mimicry, in which one species (*mimic* animals) imitates the phenotype of another species (*model* animals). Typically, the effect is called mimicry when the model species are dangerous to predators. In this case, the mimic species benefits from mimicry when predators mistake the mimic animals for model animals. Depending on characteristics of the mimic species, the model species may benefit (Müllerian mimicry, [11]) or suffer (Batesian mimicry, [1]) from the presence of mimic animals. In this paper, we investigate two possible hypotheses for the origin of mimicry through an agent-based co-evolutionary model.

The pioneer in mimicry research was Bates [1]. Bates found that there are poisonous animals with very bright colors, and camouflaged animals which were not poisonous. Even though the brightly colored animals are more easily detected by predators, they were also identified as dangerous by these predators. This so-called *aposematism* effect became more remarkable when Bates found animals with similar colors and shapes as the toxic animals that were not toxic. This type of mimicry is called Batesian mimicry, in which non-toxic animals imitate the phenotype of toxic animals. This effect has been found in butterflies [11, 1], snakes [12], and various other animals [9].

In Batesian mimicry, mimic animals are not toxic. As a result, whenever a mimic animal is eaten or tasted by a predator and found to be harmless, this

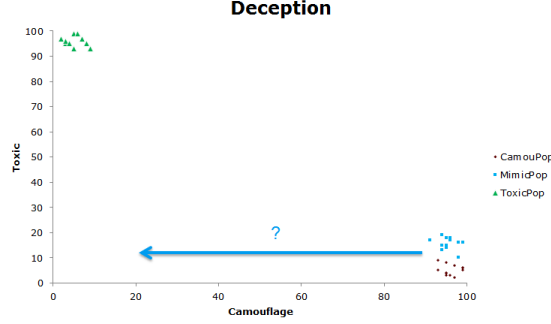


Fig. 1. In deception, the mimic and camouflaged animals start with the same characteristics. The hypothesis states that the mimicry group will move toward the place with low toxicity and camouflage: the mimicry place, since this place has the lowest evolutionary cost.

gives positive feedback to the predator to eat similar animals. This results in model animals being eaten more. Hence, the more mimic animals exist in the habitat of the model, the lower the survival chance of the model. This results in a negative or parasitical effect on the model. Müllerian mimicry [11], on the other hand, involves two species of animals that are both toxic to a certain degree, and therefore both contribute to this anti-predation mechanism.

In this paper, we investigate two possible hypotheses for the origin of Batesian mimicry. A common assumption in the literature is that the mimic animals deliberately deceive their predators by imitating model animals [7, 12]. That is, mimic animals evolve to have the same phenotype as the model animals because this lowers predation. However, mimicry may also come about through coincidence. That is, model animals may evolve a phenotype that allows predators to distinguish them, and which happens to be the phenotype of the mimic animals.

These two hypotheses will be tested through an agent-based co-evolution model. Agent-based modeling has proven its usefulness as a research tool to investigate how behavioral patterns may emerge from the interactions between individuals (cf. [4, 5]). Among others, agent-based models have been used to explain fighting in crowds [8], the evolution of cooperation and punishment [2, 13], and the evolution of language [3]. In this paper, we use agent-based modeling to test two hypotheses on the origins of mimicry. We will elaborate on the hypotheses in the next two sections.

1.1 Deception hypothesis

The deception hypothesis reflects the typical assumption about mimicry. According to the deception hypothesis, mimic animals evolve to have a phenotype that is as similar as possible to the phenotype of model animals. This benefits mimic animals because they are mistaken for animals that are dangerous to

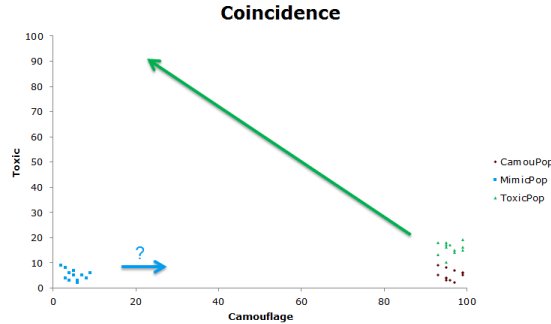


Fig. 2. In coincidence, the toxic model animals and the camouflaged animals start with the same characteristics. The mimicry group is already in place. The toxic population now gets more toxic, and moves toward the lesser camouflaged site.

predators. Therefore, if model animals are less camouflaged, the mimic animals are expected to evolve a lower level of camouflage as well.

This idea is represented graphically in Figure 1. The figure shows three types of prey: model animals (ToxicPop), mimic animals (MimicPop), and a control population of camouflaged animals (CamouPop). Only the model animals are dangerous to the predators, as shown by a high level of toxicity, while both the control and mimic animals start out with high levels of camouflage. The deception hypothesis predicts that, over time, the mimic animals take advantage of the eating behavior of predators and evolve lower levels of camouflage (blue arrow in Figure 1), since being camouflaged has a negative influence on their chance to reproduce. That is, the deception hypothesis describes a process of *speciation*, where one population of prey splits into a population of mimic and a population of control animals.

1.2 Coincidence hypothesis

In contrast with the deception hypothesis, the coincidence hypothesis describes a small role for the mimic animals. The coincidence hypothesis describes the situation in which the mimic animals do not change their phenotype, but that the model animals evolve a distinct phenotype, which happens to be the same as the phenotype of the mimic animals. Note that in this situation, there is an important role for the behavior of the predators. Predators learn to create a discriminatory line between the model animals and the control population, which drives the selective pressure for model animals to evolve a phenotype that is distinct from that of the control animals. The coincidence hypothesis states that the mimic animals may happen to be on the model animal side of this line, and therefore experience a coincidental benefit.

The coincidence hypothesis is described graphically in Figure 2. The figure shows three types of prey: model animals (ToxicPop), mimic animals (Mim-

icPop), and a control population of camouflaged animals (CamouPop). All these populations start out with low levels of toxicity, and are therefore harmless for the predators. In addition, the mimic animals start out with low camouflage, while the control and model animals have high levels of camouflage. The coincidence hypothesis predicts that when the model animals evolve higher levels of toxicity, they will also decrease their level of camouflage (green arrow in Figure 2). In addition, the coincidence hypothesis predicts that the mimic population would not increase its camouflage to more closely resemble the model animals (blue arrow in Figure 2). Note that the coincidence hypothesis also describes a process of *speciation*, but in this case, one population of prey splits into a population of model and a population of control animals.

The idea behind the coincidence hypothesis is that there are relatively few mimic animals who already have a distinctive color because of pre-adaptation [7]. The model animals experience selective pressure towards the phenotype of the mimic animals because of the relative low population sizes of the mimic animals compared to the population size of the control animals.

1.3 Structure of the paper

The remainder of this paper is set up as follows. In Section 2, we will discuss the simulation model, first in general terms and then in more depth. We present our simulation results in Section 3. Section 4 will discuss the results and provides directions for future research.

2 Model

While mimicry is defined in terms of the evolutionary behavior of prey animals, mimicry also depends on the behavior of predator animals. As a result, there are three different ways to study mimicry [10]:

- The evolutionary dynamics way, which studies the evolution of the prey but ignores the behavior of the predators [6];
- The receiver psychology way, which focuses on the behavior of the predators, but tends to ignore the evolution of the prey [7]; and
- The traditional natural historical way, which analyzes the behavior of both predator and prey. In these kinds of research, the co-evolution between predators and prey is studied [1].

In this paper, we follow the traditional natural historical way by explicitly modeling both the evolution of prey animals and the behavior of predator animals. To study this co-evolution, we construct an agent-based model that models individual prey and predator animals. In Section 2.1, we first give a general explanation of our model. A more technical discussion of the model can be found in Section 2.2.

2.1 Model description

Our model of mimicry investigates the co-evolution of predator and prey animals. Prey animals are further subdivided into three separate populations, which we will call the *toxic*, *camouflaged*, and the *mimic* populations.

Predator agents perform two actions: eating prey and reproducing. A predator consists of a neural network that determines whether a predator will eat a prey that it encounters. This network is evolved, which means that predators do not learn over their lifetime, but instead inherit their decision function from their parent. At each time-step of the model, the predator encounters a number of prey. For each encountered prey, the predator decides whether or not to eat the prey, depending on the prey’s phenotype. Eating non-toxic prey increases evolutionary fitness, while eating toxic prey decreases fitness. Reproduction occurs by selecting the agent with the highest fitness from a random sub-set of predators as the parent. The child inherits all characteristics of this parent, subject to a small probability of mutation, which will be elaborated on in Section 2.2.

Prey are defined by three characteristics: camouflage, toxicity, and pattern. A prey’s camouflage determines the probability of being detected, so that a higher camouflage lowers the probability of being encountered by a predator. A prey’s pattern, on the other hand, does not influence the probability of being encountered. A prey’s phenotype consists of its camouflage and pattern. That is, both camouflage and pattern are observable characteristics, while toxicity is a characteristic that cannot be observed by predators.

Prey reproduce by selecting two parents with the highest fitness from a random subset of the population. The fitness of a prey is determined by the number of times it is eaten by a predator. In addition, both toxicity and camouflage decrease a prey’s fitness.

To investigate mimicry, prey animals are subdivided into three separate populations of constant size that reproduce independently. The first population, called the *toxic* prey, has a small genetic drift toward higher toxicity. This population is meant to simulate model animals. Similarly, the *camouflaged* prey experience a small genetic drift towards higher camouflage, and are meant as a control population. The third, *mimic* population does not experience any genetic drift.

2.2 Model details

In this section we look at the model in more detail. In particular, we take a closer look at the eating behavior and knowledge of the predator, the mechanism of reproduction, and the setup of different parameters.

Eating behavior of predators During every time step of the model, each predator encounters a fixed number $Y_{encountered}$ of randomly selected prey animals (see also Algorithm 1). For each of these encounters, the camouflage of the prey animal determines the probability with which the prey is found, so that

Algorithm 1 Eating behavior of predators.

```
let  $C_y$  camouflage of prey  $y$ , scaled  $[-50, 50]$ 
let  $P_y$  pattern of prey  $y$ , scaled  $[-50, 50]$ 
let  $NN(X, Y)$  neural network function of predators
for each encountered prey  $y$  do
    if  $random(100) > C_y$  then ▷ Prey  $y$  is found
        if  $NN(P_y, C_y) > random(1)$  then
            Eat prey  $y$ 
        end if
    end if
end for
```

prey with a high camouflage are more likely to hide from the predators. If the predator finds the prey, it can choose whether or not to eat the prey.

The predator uses a simple feed-forward neural network to propagate the phenotype of the found prey, which results in a decision on whether or not to eat a prey. This neural network consists of two input nodes, which represent the prey's camouflage and pattern; three hidden nodes; and one output node that controls the predator's eating decision. The output node is implemented as a probability between 0 and 1, so that there is a low probability that predators will try to eat prey that they believe to be dangerous.

Each node is connected to all nodes in the next layer. This results in 6 synapses from the input nodes to the hidden nodes and 3 from the hidden nodes to the output node. The total number of synapses (weights) is thus 9 per predator. After each synapse round, an activation function is applied to scale the values between 0 and 1. The activation function used is the sigmoid function.

Reproduction The mechanism with which animals reproduce is different for predators than it is for prey. However, for the selection of the parents, both types use tournament selection.

Predators reproduce asexually, so that every child has a single parent. A child inherits the neural network from its parent, subject to a low probability of mutation. When a weight mutates, a value between -0.25 and 0.25 is added to it. The value is then cut to the domain between -2 and 2. Because of the survival of the fittest principle, the best predators evolve and anticipate on the changes within the prey. This way of learning can be seen as a random search.

Prey, on the other hand, reproduces sexually. The two parents with the highest fitness are chosen with the tournament selection, and the child is a combination of these parents. Each child has camouflage, toxicity, and pattern that is the mean of the corresponding characteristic of its two parents. The values of the prey characteristics have a value between 0 and 100. Each characteristic has a low probability of mutation, in which case a random number between -10 and 10 is added to it. If the new value exceeds the borders of 0 or 100, it is cut off at that value. In the case of genetic drift, the **Genetic Drift** parameter is added to the mutation value, giving more chance for an increasing mutation.

Table 1. Parameter settings used in the simulation runs.

Parameter	Predators	Toxic	Camouflaged	Mimic
Population size	10	300	300	30
Genetic drift	-	3	3	0
Prey encounters ($Y_{encountered}$)	3	-		
Mutation rate	20	2		
Tournament size	3	10		
Lifespan	3	5		
Chance-being-found	101	-		
Camouflage disadvantage	-	3.0		
Toxicity disadvantage	-	0.2		

For both predators and prey, reproduction occurs in generations. After each generation, all animals in the old generation die and are replaced by an identical number of new individuals. This means that there are no animals older than other animals, and that all animals die at the same moment after a predefined number of time-steps. This number of time-steps differs between predators and prey (see the lifespan parameter in Table 1) to reflect differences in learning. Note that prey animals do not die due to being eaten by a predator, but only die when their generation dies. Instead, the fitness of a prey decreases when it is eaten by a predator, reducing the chance for reproduction.

Parameters settings and fitness Within our simulation, the number of predators is fixed, as well as the number of prey within each subpopulation. In every run of the model, there are fewer predators than prey, corresponding to the real world.

The predators have a lower life span than preys, to reflect that they learn faster than the rate at which prey evolves. The prey becomes older, which makes the difference in fitness between preys which are eaten and that are not eaten bigger.

Each time-step, a predator encounters the number of prey divided by the number of predators. This is multiplied by $Y_{encountered}$ to make the selective pressure higher. From the point of view of the prey, it has $Y_{encountered}$ encounters with predators.

In our model, each individual prey and predator represents a group of animals. For this reason, prey does not die when it is eaten. Instead, the fitness of a prey animal y is determined by the number of times it is ‘eaten’ (E_y). In our model, we assume both toxicity and camouflage to be detrimental to fitness. The toxicity (T_y) and camouflage (C_y) of prey y are multiplied by the toxic disadvantage (TD) and camouflage disadvantage (CD) parameters respectively. For example, a prey with a toxicity of 80 and toxicity disadvantage 0.2 will experi-

ence a 16 point penalty to its fitness. The fitness of a prey is updated according to

$$F_y = -(T_y \cdot TD) - (C_y \cdot CD) - (500 \cdot E_y). \quad (1)$$

Note that the most detrimental effect to the fitness of a prey is to be eaten. In addition, a prey's fitness is a non-positive number, with 0 being the highest possible value.

The fitness of a predator is determined by what prey it eats, according to the following formula

$$F_r = \sum_{y \in Eat_r} (T_y - 60). \quad (2)$$

The formula shows that the fitness of the predator r (F_r) depends on the sum of the toxicity of all the prey it has eaten (Eat_r). Toxicity values are reduced by 60, so that predators increase their fitness whenever they eat a prey with toxicity lower than 60, and decrease their fitness otherwise.

3 Results

We used the model outlined in Section 2 to perform simulation runs, the results of which are discussed in this section. The results are divided in four different sections. In Section 3.1, we discuss the deception hypothesis. In Section 3.2, we investigate the coincidence hypothesis. For both these hypotheses, we show results from 100 runs of 14,000 time steps each. After this, in Section 3.3, we will discuss an individual run. Lastly, in Section 3.4 the difference between model animals and mimic animals in the different hypotheses will be discussed.

3.1 Deception hypothesis

For the deception hypothesis (also see Figure 1), the mimic and camouflaged populations start out with high camouflage. The toxic population starts at its final position with high toxicity and little camouflage (that is, brightly colored and toxic animals). The deception hypothesis predicts that the mimic population would evolve to decrease its camouflage while maintaining low toxicity.

Figure 3 shows the average camouflage of the three prey populations across 100 runs. The figure shows that the mimic population indeed decreases its level of camouflage over time. The camouflaged population also initially reduces its level of camouflage, but later returns to high camouflage levels. This can be explained by the genetic drift of the camouflaged population. However, the larger population size of the camouflaged prey also slows down the evolutionary process. In addition, predators find and eat more of the less camouflaged individuals than the more camouflaged individuals, which gives additional selective pressure to increase camouflage.

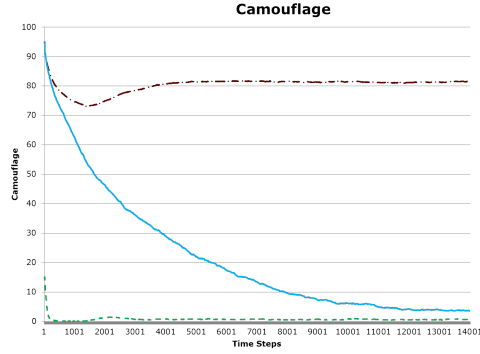


Fig. 3. The average camouflage value for each of the three prey populations across 100 runs for the deception hypothesis. The mimic and camouflaged populations start out with the same high camouflage, while the toxic population starts out with low camouflage.

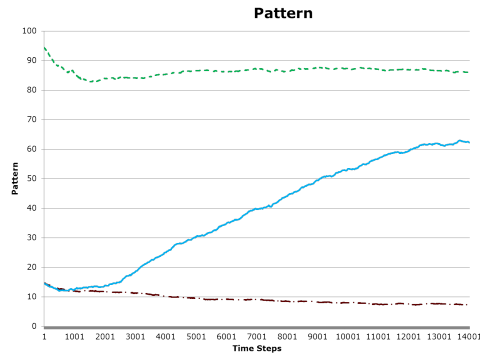


Fig. 4. The average pattern value for each of the three prey populations across 100 runs for the deception hypothesis. The mimic and camouflaged populations start out with the same pattern, which is different from the pattern of the toxic population.

In contrast, Figure 3 shows that the camouflage of the toxic population quickly drops and remains fairly stable. Since the toxic individuals rely on the predators' choice not to eat them, there is an evolutionary pressure to be as distinct as possible from other prey. In this case, the other prey have high camouflage, so the selective pressure encourages the toxic population to lower camouflage.

Figure 4 shows the average pattern of the three prey populations across 100 runs. In this figure, we can see that the pattern of the mimics moves toward that of the toxic population. Both Figure 3 and Figure 4 support the deception hypothesis, since the mimic moves from a position of high camouflage towards a position of no camouflage, thereby mimicking the toxic population. In addition,

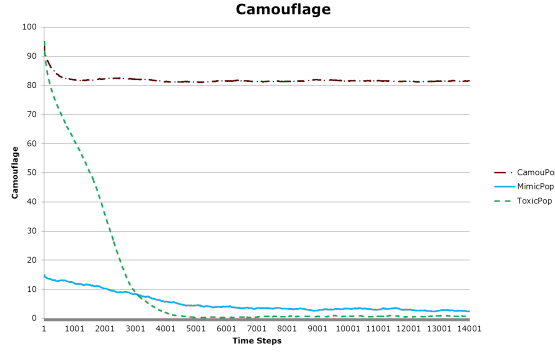


Fig. 5. The average camouflage value for each of the three prey populations across 100 runs for the coincidence hypothesis. The toxic and camouflaged population start out with the same high camouflage, while the mimic populations starts out with low camouflage.

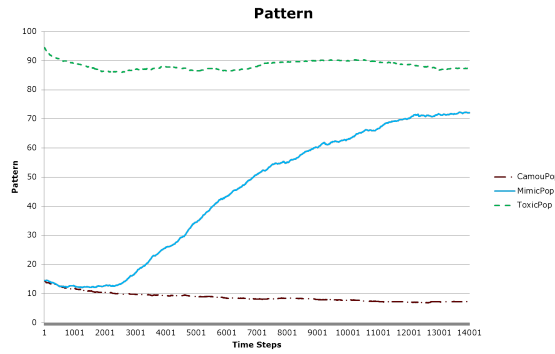


Fig. 6. The average pattern value for each of the three prey populations across 100 runs for the coincidence hypothesis. The mimic and camouflaged populations start out with the same pattern, which is different from the pattern of the toxic population.

the mimics also evolve to have the same pattern as the toxic group. That is, these results suggest that the mimics evolve to trick the predators into not eating them.

3.2 Coincidence

In the coincidence hypothesis, the toxic and camouflaged group start at the same position, with low toxicity and high camouflage. The mimic population, on the other hand, starts out with low toxicity and low camouflage. The coincidence hypothesis predicts that, in order to distinguish itself from the camouflaged population, the toxic population evolves to a position with high toxicity and

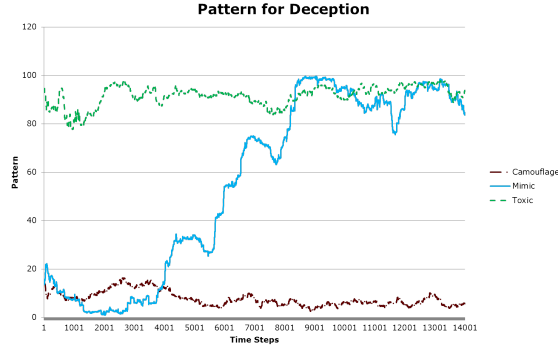


Fig. 7. The pattern of an individual run in the deception hypothesis. The pattern changes with two distinctive bumps, before being similar to the toxic population. After this we see the toxic population moving away from the pattern of the mimics, and the mimics chasing this pattern.

low camouflage, which coincidentally gives the same camouflage as the mimic population (also see Figure 2).

Figure 5 shows the average camouflage of the three prey populations across 100 runs. Note that while the camouflaged population maintains high camouflage, the toxic population evolves to lower its camouflage. In addition, Figure 5 shows that, on average, the mimic population does not increase its camouflage to increase resemblance with the toxic population.

Figure 6 shows the average pattern of the three prey populations across 100 runs. This figure shows that, while the mimic population does not increase its camouflage to resemble the toxic population, the mimic population does evolve to have a pattern that is similar to the toxic population. That is, while Figure 5 supports the coincidence hypothesis, Figure 6 is more suggestive of the deception hypothesis.

3.3 Individual run

While the average results presented in the previous sections give a good impression of the way the prey’s phenotype (i.e., camouflage and pattern) evolves over time, closer inspection shows that the average does not fit any individual run particularly well. For this reason, we take a closer look at a representative individual run in this section.

Figure 7 shows the evolution of pattern for all three prey populations of a representative individual run of 14,000 time-steps. Note that while the average results (Figures 4 and 6) suggest that the pattern of the mimic population gradually evolves over 14,000 time steps, Figure 7 suggests a more rapid evolution. Indeed, individual runs typically show a rapid evolution of the pattern of the mimic population. The average results show a more gradual development because the moment at which this rapid evolution starts is different for each run.

Figure 7 shows that the pattern of the mimic population not only evolves in the direction of the pattern of the toxic population, but also continues to converge on the same value. In addition, due to the differences in population size of the mimic and toxic populations, the pattern of the mimic populations exhibits more volatility than that of the toxic population. This corresponds well with the idea of Holmgren and Enquist [7], who say:

”For mimicry to be established, the movement of the mimic should always be faster than the movement of the model.”

After approximately 9000 steps in the simulation, we notice the pattern of the model animals moves away from that of the mimic animals. When the mimics have a higher pattern, the models get a lower pattern and vice versa. This is consistent with the idea of [7]. In the results we can see that the pattern of the mimics changes faster, but the models try to distinguish themselves from the mimics. The reason for this is that model animals that are more similar to mimic animals are more likely to be eaten by predators, since mimics are harmless for predators. Model animals that look less like the mimics therefore have an evolutionary advantage, a development we can observe in the individual run of the model.

Figure 7 shows that the mimic population changes its pattern in several bumps. These bumps can be explained by the model animals getting less toxic. As a result, the predators start eating more model animals. When we observe the model animals, we can see that they start losing their toxicity when the predators do not eat them, since toxicity is detrimental to individual fitness. However, when the toxicity becomes too low, predators start eating more model animals. In Figure 7, this effect is shown when the pattern of the mimics moves away from the pattern of the model animals.

3.4 Euclidean distance Model and Mimic

Figure 8 shows the distance in phenotype between model and mimic over time. This graph tells us that the distance gets smaller, thus mimicry is created. The euclidean distance is measured as the distance between the mean of the pattern and camouflage between the mimics and the models by the following formula:

$$EUD = \sqrt{(C_{mim} - C_{mod})^2 + (P_{mim} - P_{mod})^2} \quad (3)$$

For the deception hypothesis, the mimic moves towards the model to trick the predators. Over time, we can see that the mimic population becomes increasingly more similar to the model population. This means that a lot of the simulations evolve into mimicry. For the coincidence hypothesis, the toxic group evolves towards the same camouflage as the mimics. After this, the mimics follow the models over the phenotype plane. As a result, the average euclidean distance gets smaller. Since both distances are decreasing, the model supports both hypotheses for mimicry.

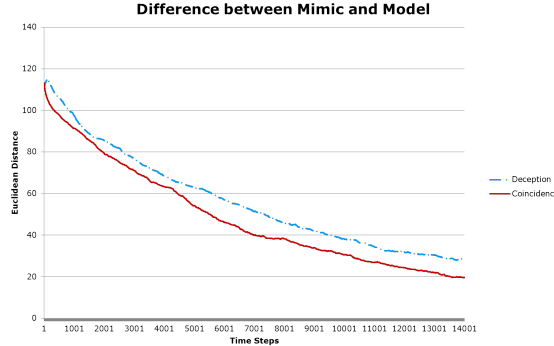


Fig. 8. The Euclidean Distance over time for both the hypotheses. In both hypotheses, the difference between model and mimic decreases over time, arguing for mimicry. The coincidence hypothesis has a little less difference. Keep in mind this is an average over 100 runs, and therefore is not a representation of one run, but a probability of mimic and model being the same.

4 Discussion

In this paper, we constructed an agent-based co-evolutionary model to investigate two possible origins of mimicry. The deception hypothesis predicts that due to selective pressure, mimic animals change their phenotype to resemble the model animals. In contrast, the coincidence hypothesis states that due to selective pressure, model animals change their phenotype to be different from some control population that has high levels of camouflage, and coincidentally get a phenotype that is similar to the mimic animals.

For both these possible origins of mimicry, we can say that they are plausible. The deception has very clear evidence in the deception set-up, where we can see that the mimic animals always change their phenotype to more resemble the phenotype of the model animals. This suggests that mimic animals indeed evolve to deceive predators. For the relatively small population of mimics, it is possible to explore new peaks in the adaptive plane, and successfully deceive predators. For the larger camouflage population, we can see that this population is too big to explore new adaptive peaks.

Our model results also show evidence for the coincidence hypothesis. Given the appropriate starting conditions, the model animals may change their phenotype to more resemble the mimic animals rather than vice versa. However, this can also be explained by the assumed negative fitness contribution of camouflage. This alone may result in both model and mimic animals to experience a selective pressure to reduce camouflage. Indeed, when we consider the pattern alone, the mimics again attempt to deceive the predators by evolving a pattern that resembles that of the model animals.

Note that we considered two different setups to determine the plausibility of the two hypotheses for the origin of mimicry. Both of these setups resulted in

the same ecosystem, with no possibility to determine what was the initial setup. That is, while our results show support for both hypotheses, they do not allow us to draw conclusions about which hypothesis matches best with biological data.

According to Holmgren and Enquist [7], the model animals always attempt to distinguish themselves from mimic animals. By creating distance from the mimics, the predators experience less confusion between model and mimic animals. However, since the mimics follow the phenotype of the models and evolve faster, this is an endless cat and mouse game. In our simulation model, this can be observed in the pattern, where the pattern of the mimic animals closely follows the pattern of the model animals.

Our simulation model can be used to do more research on theories of mimicry. Since the parameters can be easily adjusted, more experiments can be done. Firstly, more experiments can be done with different starting positions of the camouflage, toxicity and pattern values, starting with a control run. In this case, one population without genetic drift would be created to see how preys evolve without other animals. Another example is the toxicity and mimic group starting in the toxic position, and the camouflage on the camouflage position. This would make for another coincidence set-up, which assumes the speciation of the mimicry being a sub-population from the toxic population. Alternatively, the pattern can be altered. The pattern of the camouflage and mimic start the same in this paper, but it can be altered to a situation where the toxic and camouflage population start the same, and see whether the toxic population moves away. Besides this we can see whether the mimics move toward the models, which supports the Coincidence hypothesis.

The dynamics of the model can be altered as well. One possibility is adding more dimensions of recognition. This would mean that instead of 1 pattern, the model would have 50 patterns, which all can be mutated and inherited individually. In these recognition dimensions the scale between 0 and 100 can be removed, so the models and mimics can move through the adaptive space with more freedom. This way, neophobia and the idea of Holmgren and Enquist [7] can be researched in more detail. When the dimensions are implemented we hypothesize that the models will keep evading the phenotype of the mimics and the mimics chasing this phenotype. If the domains are removed, we expect very high and low values in the dimensions, arguing for the very bright colors of the animals.

In our model, we assume that prey consists of three populations of constant size that cannot interbreed. In future work, it would be interesting to see how removing this assumption influences our results. This would create hybrid populations of prey, which may have interesting properties.

To research Müllerian mimicry, more populations can be added which have intermediate values of genetic drift toward toxic. With two toxic populations, a research can be conducted whether the animals imitate each others phenotype or keep their own phenotype. The number dependent theory [11] can be tested in the same way. We hypothesize that when more toxic populations are implemented, there will be one center where all the animals converge to, to make one clear

aposematism. The representation of knowledge can be differentiated. At the moment, the predators have a line in their choice to eat camouflage or not. If a more curved line, or other methods are implemented, the idea of *Novelty and Recognizability* [10] can be researched in more depth. If this is combined with a variation of punishment for toxicity, we expect that neophobia emerge from the simulation.

Lastly, a spatial model can be created, in which agents have a x- and y-coordinate. This way mimicry rings can be researched, which are discussed in great depth by Holmgren and Enquist [10], and found by Bates [1]. When the spatial model is implemented, all the aforementioned can be combined in one simulation, since every place can evolve something else. Especially the borders of different mimicry systems will be interesting to research. Using a bigger adaptation space, better knowledge of the predators and a spatial dimension in the model, we aim to have a better understanding of the origin of mimicry in the future.

References

1. Bates, H.W.: XXXII. Contributions to an insect fauna of the Amazon valley. Lepidoptera: Heliconidae. Transactions of the Linnean Society of London 23(3), 495–566 (1862)
2. Boyd, R., Gintis, H., Bowles, S., Richerson, P.J.: The evolution of altruistic punishment. Proceedings of the National Academy of Sciences 100(6), 3531–3535 (2003)
3. Cangelosi, A., Parisi, D.: Simulating the Evolution of Language. Springer Science & Business Media (2012)
4. Epstein, J.M.: Agent-based computational models and generative social science. Complexity 4(5), 41–60 (1999)
5. Epstein, J.M.: Generative Social Science: Studies in Agent-Based Computational Modeling. Princeton University Press (2006)
6. Gavrillets, S., Hastings, A.: Coevolutionary chase in two-species systems with applications to mimicry. Journal of Theoretical Biology 191(4), 415–427 (1998)
7. Holmgren, N.M., Enquist, M.: Dynamics of mimicry evolution. Biological Journal of the Linnean Society 66(2), 145–158 (1999)
8. Jager, W., Popping, R., van de Sande, H.: Clustering and fighting in two-party crowds: Simulating the approach-avoidance conflict. Journal of Artificial Societies and Social Simulation 4(3), 1–18 (2001)
9. Maan, M.E., Cummings, M.E.: Poison frog colors are honest signals of toxicity, particularly for bird predators. The American Naturalist 179(1), E1–E14 (2011)
10. Mallet, J., Joron, M.: Evolution of diversity in warning color and mimicry: Polymorphisms, shifting balance, and speciation. Annual Review of Ecology and Systematics 30(1), 201–233 (1999)
11. Müller, F.: Ituna and Thyridia: A remarkable case of mimicry in butterflies. Transactions of the Entomological Society of London 1879, 20–29 (1879)
12. Rabosky, A.R.D., Cox, C.L., Rabosky, D.L., Title, P.O., Holmes, I.A., Feldman, A., McGuire, J.A.: Coral snakes predict the evolution of mimicry across new world snakes. Nature Communications 7 (2016)
13. de Weerd, H., Verbrugge, R.: Evolution of altruistic punishment in heterogeneous populations. Journal of theoretical biology 290, 88–103 (2011)

Feature selection in high-dimensional dataset using MapReduce

Claudio Reggiani, Yann-Aël Le Borgne and Gianluca Bontempi

Machine Learning Group, Faculty of Science, Université Libre de Bruxelles,
Boulevard du Triomphe, CP 212, 1050 Brussels, Belgium
`creggiani,yleborgn,gbonte@ulb.ac.be`

Abstract. This paper describes a distributed MapReduce implementation of the minimum Redundancy Maximum Relevance algorithm, a popular feature selection method in bioinformatics and network inference problems. The proposed approach handles both *tall/narrow* and *wide/short* datasets. We further provide an open source implementation based on Hadoop/Spark, and illustrate its scalability on datasets involving millions of observations or features.

1 Introduction

The exponential growth of data generation, measurements and collection in scientific and engineering disciplines leads to the availability of huge and high-dimensional datasets, in domains as varied as text mining, social network, astronomy or bioinformatics to name a few. The only viable path to the analysis of such datasets is to rely on data-intensive distributed computing frameworks [1].

MapReduce has in the last decade established itself as a reference programming model for distributed computing. The model is articulated around two main classes of functions, *mappers* and *reducers*, which greatly decrease the complexity of a distributed program while allowing to express a wide range of computing tasks. MapReduce was popularised by Google research in 2008 [2], and may be executed on parallel computing platforms ranging from specialised hardware units such as parallel field programmable gate arrays (FPGAs) and graphics processing units, to large clusters of commodity machine using for example the Hadoop or Spark frameworks [2–4].

In particular, the expressiveness of the MapReduce programming model has led to the design of advanced distributed data processing libraries for machine learning and data mining, such as Hadoop Mahout and Spark MLlib. Many of the standard supervised and unsupervised learning techniques (linear and logistic regression, naive Bayes, SVM, random forest, PCA) are now available from these libraries [5–7].

Little attention has however yet been given to feature selection algorithms (FSA), which form an essential component of machine learning and data mining workflows. Besides reducing a dataset size, FSA also generally allow to improve

the performance of classification and regression models by selecting the most relevant features and reducing the noise in a dataset [8].

Three main classes of FSA can be distinguished: *filter*, *wrapper* and *embedded* methods [8, 9]. Filter methods are model-agnostic, and rank features according to some metric of information conservation such as mutual information or variance. Wrapper methods use the model as a black-box to select the most relevant features. Finally, in embedded methods, feature evaluation is performed alongside the model training. In this paper, feature metrics are named hereafter *feature score* functions.

Early research on distributing FSA mostly concerned wrapper methods, in which parallel processing was used to simultaneously assess different subsets of variables [10–14]. These approaches effectively speed up the search for relevant subsets of variables, but require the dataset to be sent to each computing unit, and therefore do not scale as the dataset size increases.

MapReduce based approaches, which address this scalability issue by splitting datasets in chunks, have more recently been proposed [15–21]. In [15], an embedded approach is proposed for logistic regression. Scalability in the dataset size is obtained by relying on an approximation of the logistic regression model performance on subsets of the training set. In [16], a wrapper method based on an evolutionary algorithm is used to drive the feature search. The first approaches based on filter methods were proposed in [17, 18], using variance preservation and mutual information as feature selection metrics, respectively. Two other implementations of filter-based methods have lately been proposed, addressing the column subset selection problem (CSSP) [19], and the distribution of data by features in [20]. Recently, a filter-based feature selection framework based on information theory [22] has been implemented using Apache Spark [21].

In this paper we tackle the implementation of *minimal Redundancy Maximal Relevance* (*mRMR*) [23], a forward feature selection algorithm belonging to filter methods. mRMR was shown to be particularly effective in the context of network inference problems, where relevant features have to be selected out of thousands of other noisy features [1, 24].

The main contributions of the paper are the following: *i*) design of minimum Redundancy Maximum Relevance algorithm using MapReduce paradigm; *ii*) open-source implementation for Apache Spark available on a public repository; *iii*) analysis of the scalability properties of the algorithm. In an extended version [25], we also detail how to customize the feature score function during the feature selection process.

The paper is structured as follows. Section 2 provides an overview of the MapReduce paradigm, and Section 3 describes the two main layouts along which data can be stored. Section 4 presents our distributed implementation of mRMR. Section 5 finally provides a thorough experimental evaluation, where we illustrate the scalability of the proposed implementation by varying the number of rows and columns of the datasets, the number of selected features in the feature selection step and the number of nodes in the cluster.

2 MapReduce paradigm

MapReduce [2] is a programming paradigm designed to analyse large volumes of data in a parallel fashion. Its goal is to process data in a scalable way, and to seamlessly adapt to the available computational resources.

A MapReduce job transforms lists of input data elements into lists of output data elements. This process happens twice in a program, once for the *Map* step and once for the *Reduce* step. Those two steps are executed sequentially, and the Reduce step begins once the Map step is completed.

In the Map step, the data elements are provided as a list of key-value objects. Each element of that list is loaded, one at a time, into a function called *mapper*. The mapper transforms the input, and outputs any number of intermediate key-value objects. The original data is not modified, and the mapper output is a list of new objects.

In the Reduce step, intermediate objects that share the same key are grouped together by a *shuffling* process, and form the input to a function called *reducer*. The reducer is invoked as many times as there are keys, and its value is an iterator over the related grouped intermediate values.

Mappers and reducers run on some or all of the nodes in the cluster in an isolated environment, i.e. each function is not aware of the other ones and their task is equivalent in every node. Each mapper loads the set of files local to that machine and processes it. This design choice allows the framework to scale without any constraints about the number of nodes in the cluster. An overview of the MapReduce paradigm is reported in Figure 1a.

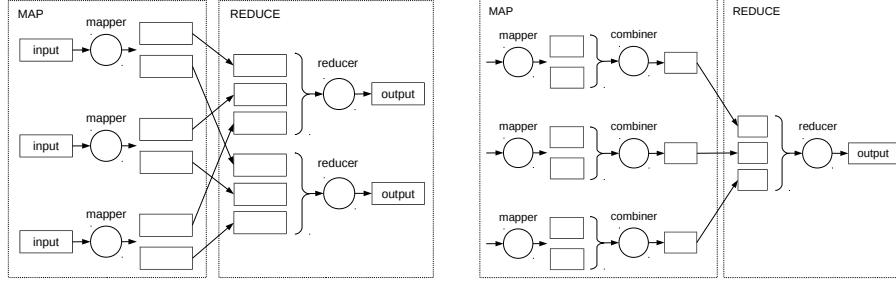
Algorithms written in MapReduce scale with the cluster size, and Execution Time (ET) can be decreased by increasing the number of nodes. The design of the algorithm and the data layout are important factors impacting ET [26].

In ET terms, jobs perform better in MapReduce when transformations are executed locally during the Map step, and when the amount of information transferred during the shuffling step is minimised [27]. In particular, MapReduce is very well-suited for associative and commutative operators, such as *sum* and *multiplication*. These can indeed be partially processed using an intermediate *Combine* step, which can be applied between the Map and Reduce stages.

The combiner is an optional functionality in MapReduce [2]. It locally aggregates mapper output objects before they are sent over the network. It operates by taking as input the intermediate key-value objects produced by the mappers, and output key-value pairs for the Reduce step. This optional process allows to reduce data transfer over the network, therefore reducing the global ET of the job. An illustration of the use and advantages of the combiner is given in Figure 1b.

3 Data Layout

In learning problems, training data from a phenomenon is usually encoded in tables, using rows as observations, and columns as features. Let M be the number of observations, and N be the number of features. Training data can be



(a) MapReduce overview of the data flow. The dataset stored in the distributed storage system is split into chunks across nodes (rectangular *input* boxes). Each chunk is fed as input to the mapper functions, which may output intermediate objects. These objects are shuffled and grouped by keys across the network. Finally, the reducers generate the groups of intermediate objects and output the results. All objects (input, intermediate, output) are identified by a key-value pair.

(b) MapReduce overview of the data flow with the additional combiner function. Intermediate objects produced by mappers are locally aggregated by a commutative and associative function implemented in the combiner logic. In this example three, instead of six, intermediate objects are sent over the network to the reducer function. Combiners provide an efficient way to reduce the amount of shuffled data, and to reduce the overall execution time of the job.

Fig. 1: MapReduce data flow overview with and without combiner.

represented as a collection of vectors, \mathbf{X} ,

$$\mathbf{X} = (\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_M)$$

where

$$\mathbf{x}_j = (x_{j,1}, x_{j,2}, \dots, x_{j,N}) \quad \forall j \in (1, \dots, M).$$

We will refer to this type of structure as the *conventional* encoding, see Table 1.

It is however worth distinguishing two types of tables: *tall and narrow* (T/N) tables, where $M \gg N$, and *short and wide* (S/W) tables, where $M \ll N$.

The distinction is important since MapReduce divides input data in chunks of rows, that are subsequently processed by the mappers. MapReduce is therefore well suited to ingest T/N table, but much less S/W tables, since data cannot be efficiently split along columns. S/W tables are for example encountered in domains such as text mining or bioinformatics [28, 29], where the number of features can be on the order of tens or hundreds of thousands, while observations may be on the order of hundreds or thousands.

In such cases, it can be beneficial to transform S/W into T/N tables, by storing observations as columns and features as rows. We refer to this type of structure as *alternative* encoding, see Table 2.

$x_{1,1}$	$x_{1,2}$	$x_{1,N}$
$x_{2,1}$	$x_{2,2}$	$x_{2,N}$
...
...
$x_{M,1}$	$x_{M,2}$	$x_{M,N}$

Table 1: Conventional encoding: Observations ($x_{i,\cdot}$) are stored along rows, and features ($x_{\cdot,j}$) are stored along columns.

$x_{1,1}$	$x_{2,1}$	$x_{M,1}$
$x_{1,2}$	$x_{2,2}$	$x_{M,2}$
...
...
$x_{1,N}$	$x_{2,N}$	$x_{M,N}$

Table 2: Alternative encoding: Observation ($x_{i,\cdot}$) are stored along columns, and features ($x_{\cdot,j}$) are stored along rows.

4 Iterative feature selection framework

This section first recalls the standard mRMR algorithm [23]. We then detail our MapReduce implementation, for both conventional and alternative encodings. An implementation of a custom feature score function using the Pearson correlation coefficient is reported in the extended article [25].

4.1 minimal Redundancy Maximal Relevance

Let us define the dataset as the table \mathbf{X} with M rows, N columns and discrete values. We define \mathbf{x}_k as the k -th column vector of the dataset and \mathbf{c} as the class vector. Furthermore, let us define L as the number of features to select and i_c^l and i_s^l as the sets at step l ($1 \leq l \leq L$) of candidate and selected features indices, respectively. At $l = 1$, we have $i_c^1 = \{1, \dots, N\}$ and $i_s^1 = \emptyset$. The pseudo-code of the algorithm is reported in Listing 1.1.

Listing 1.1: minimum Redundancy Maximum Relevance Pseudo-code. $I(\cdot)$ is the function that, given two vectors, returns their mutual information. \mathbf{x}_k is the k -th column vector of the dataset and \mathbf{c} is the class vector. L is the number of features to select, i_c^l and i_s^l as the sets at step l ($1 \leq l \leq L$) of candidate and selected features indices.

```

1   $i_c^1 = \{1, \dots, N\}$ 
2   $i_s^1 = \emptyset$ 
3  for  $l = 1 \rightarrow L$ 
4    for  $k \in i_c^l$ 
5       $I_{\mathbf{x}_k, \mathbf{c}} \leftarrow I(\mathbf{x}_k, \mathbf{c})$ 
6    for  $j \in i_s^l$ 
7       $I_{\mathbf{x}_k, \mathbf{x}_j} \leftarrow I(\mathbf{x}_k, \mathbf{x}_j)$ 
8     $g_k \leftarrow I_{\mathbf{x}_k, \mathbf{c}} - \frac{1}{|i_s^l|} \sum_{j \in i_s^l} I_{\mathbf{x}_k, \mathbf{x}_j}$ 
9     $k^* \leftarrow \text{argmax}(g_k)$ 
10    $i_c^{l+1} \leftarrow i_c^l \setminus k^*$ 
11    $i_s^{l+1} \leftarrow i_s^l \cup k^*$ 

```

12 **output** i_s^L

mRMR is an iterative greedy algorithm: at each step the candidate feature is selected based on a combination of its mutual information with the class and the selected features:

$$\begin{aligned} & \operatorname{argmax}_{k \in i_c^l} g_k(\cdot) \\ g_k(\cdot) = & \begin{cases} I(\mathbf{x}_k; \mathbf{c}) & l = 1 \\ I(\mathbf{x}_k; \mathbf{c}) - \frac{1}{|i_s^l|} \sum_{j \in i_s^l} I(\mathbf{x}_k; \mathbf{x}_j) & l > 1 \end{cases} \end{aligned} \quad (1)$$

where $I(\cdot)$ returns the mutual information of two vectors. The *feature score* $g(\cdot)$ is assessed in Lines 5-8 in Listing 1.1.

We redesigned the algorithm using MapReduce paradigm on Apache Spark, distributing the feature evaluation into the cluster. Besides the core features of MapReduce previously described, our design takes advantage of the *broadcast* operator provided in Apache Spark. Broadcasted variables are commonly used in machine learning algorithms to efficiently send additional data to every mapper and reducer as read-only variables [30].

4.2 mRMR in MapReduce with conventional encoding

Let us define the dataset as a Resilient Distributed Dataset (RDD) [31] of M tuples (\mathbf{x}, c) , where \mathbf{x} is the input (observation) vector and c is the target class value.

Considering the dataset with only discrete values, we represent with d_c the set of categorical values of the class, and with d_v the (union) set of unique categorical values of all features. If the dataset has binary values, then $d_c = d_v = \{0, 1\}$. In case of having features with different sets of categorical values, then d_v is the union of unique categorical values of all features.

The input vector is partitioned in candidate and selected features, labeled respectively as \mathbf{x}_c and \mathbf{x}_s ($\mathbf{x}_c \cup \mathbf{x}_s = \mathbf{x}$, $|\mathbf{x}| = N$). Variables L , i_c^l and i_s^l are defined as in the previous section and i_{class} is the class column index. Listings 1.2, 1.3 and 1.4 report the MapReduce job, the mapper and reducer functions, respectively, while an illustrative overview of the data flow is reported in Fig. 2.

Listing 1.2: mRMR MapReduce job with conventional data encoding. L is the number of features to select, i_c^l and i_s^l are the sets at step l ($1 \leq l \leq L$) of candidate and selected features indices. i_{class} is the class column index. d_c is the set of categorical values of the class, and d_v is the (union) set of unique categorical values of all features.

```

1   $i_c^1 = \{1, \dots, N\}$ 
2   $i_s^1 = \emptyset$ 
3  for  $l = 1 \rightarrow L$ 
4    broadcast  $i_{class}, i_c^l, i_s^l, d_v, d_c$ 
```

```

5  scores <- mapreduce(RDD, mapper, reducer)
6  k* <- collectArgmax(scores)
7  i_c^{l+1} <- i_c^l \setminus k*
8  i_s^{l+1} <- i_s^l \cup k*
9  output i_s^L

```

Listing 1.3: mRMR MapReduce mapper function with conventional data encoding. i_c^l and i_s^l as the sets at step l ($1 \leq l \leq L$) of candidate and selected features indices. i_{class} is the class column index. d_c is the set of categorical values of the class, and d_v is the (union) set of unique categorical values of all features. e is a single observation fed as input to the mapper, k and j represent column indices and $contTable$ is the function that creates a contingency table.

```

1  # broadcasted vars: i_class, i_c^l, i_s^l, d_v, d_c
2  mapper(., e)
3  for k in i_c^l
4      output (k, contTable(e_k, e_{i_class}, d_v, d_c))
5  for j in i_s^l
6      output (k, contTable(e_k, e_j, d_v, d_c))

```

Listing 1.4: mRMR MapReduce reducer function with conventional data encoding. k is a column index and t is a collection of contingency tables. The *score* function process all the contingency tables associated with the column with index k and return the feature score.

```

1  reducer(k, t)
2  output(k, score(t))

```

For every $(e_k, e_{i_{class}})$ pair, the mapper task outputs a contingency table, *contTable*, with rows defined as the categorical values in d_c and columns defined as the categorical values in d_v . The element corresponding to row $e_{i_{class}}$ and column e_k is set to 1, while all the others are set to 0. Considering the dataset in Table 3, having one binary class column and four categorical features (with three possible values: -2, 0, 2), an example of emitted contingency table is reported in Table 4. In this example the class vector can only have two possible values: 0 and 1; any feature can only have three possible values: -2, 0 and 2. The input pair $(e_k, e_{i_{class}})$ is (2, 0), therefore the element corresponding to row 0 and column 2 is set to 1, all the others are set to 0.

In case of (e_k, e_j) pair, the contingency table has both rows and columns defined by categorical values in d_v .

At the cost of managing discrete values only, the commutative and associative properties of the contingency table allows the use of the combiner function, thus minimizing the amount of data exchanged across the cluster during shuffling. While the single mapper outputs one or more contingency tables for each

#entry	class	features			
	c	\mathbf{x}_1	\mathbf{x}_2	\mathbf{x}_3	\mathbf{x}_4
1	0	2	0	0	-2
2	0	0	-2	2	0
3	0	0	2	0	-2
4	1	-2	0	0	0
...

Table 3: Example of dataset encoded with conventional layout.

candidate feature, those tables emitted by mappers executed on a given node can be locally reduced via the Combine step. Assuming that the first four entries in Table 3 are processed by four mappers in the same machine, Table 5 is the result of the combiner after the aggregation of four contingency tables of the \mathbf{x}_1 feature produced by the mappers. In this example, the combiner performs an element-wise sum of the contingency tables given as input.

		d_v		
		-2	0	2
d_c	0	0	0	1
	1	0	0	0

Table 4: Contingency table emitted by the mapper function as a result of processing the pair (\mathbf{x}_1, c) of the first entry in Table 3.

		d_v		
		-2	0	2
d_c	0	0	2	1
	1	1	0	0

Table 5: Aggregated contingency table emitted by the combiner function as a result of processing the pair (\mathbf{x}_1, c) of the first four entries in Table 3.

4.3 mRMR in MapReduce with alternative encoding

Data stored in alternative encoding has one column per observation and one row per feature. In this case, let us define the dataset as a RDD of N tuples (k, \mathbf{x}) , where \mathbf{x} is the feature vector and k is the row index ($k \in \{1, \dots, N\}$). Feature and class values could be discrete and continuous as well. With respect to the design of mRMR in MapReduce with conventional encoding, a set of vectors are broadcasted across the cluster: \mathbf{v}_{class} is the class vector, v_s is the collection of selected feature vectors and i_s is the collection of selected feature indices. Variable L is defined as in the previous section and *getEntry* function is a MapReduce task that retrieves the feature vector from the RDD, given a feature index. Listings 1.5 and 1.6 report the MapReduce job and the mapper function, respectively.

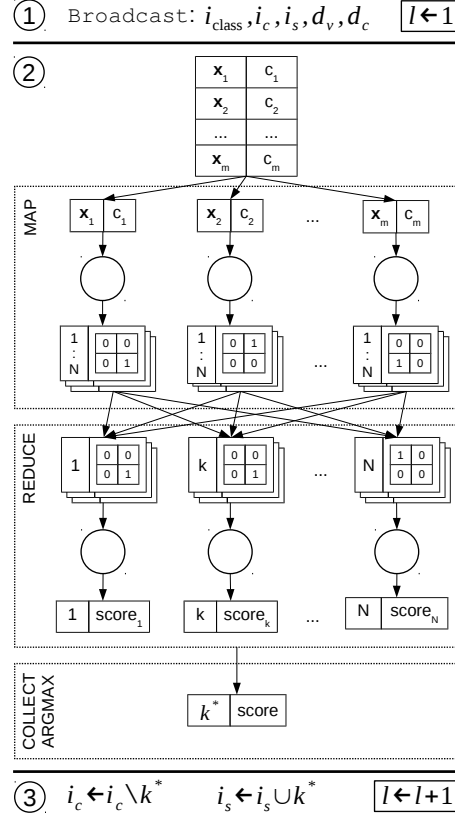


Fig. 2: Illustrative representation of the first iteration of a MapReduce job with discrete values using the conventional encoding. There are as many iterations as the number of features to select. At each iteration, each mapper outputs $N-l+1$ contingency tables for every combination of candidate features and class vector, and, from the second iteration, $(N-l+1) * |i_s^l|$ contingency tables for every combination of candidate and selected features.

Listing 1.5: mRMR MapReduce job with alternative data encoding. *RDD* represents the distributed dataset and L is the number of features to select. $\mathbf{v}_{\text{class}}$ is the class vector, v_s is the collection of selected feature vectors and i_s is the collection of selected feature indices. The *getEntry* function retrieves the feature vector from the RDD, given a feature index.

```

1  $i_s^1 = \emptyset$ 
2  $v_s^1 = \emptyset$ 
3 for  $l = 1 \rightarrow L$ 
4   broadcast  $\mathbf{v}_{\text{class}}, i_s^l, v_s^l$ 
5   scores  $\leftarrow \text{mapreduce}(\text{RDD}, \text{mapper})$ 

```

```

6    $k^* \leftarrow \text{collectArgmax(scores)}$ 
7    $v^* \leftarrow \text{getEntry(RDD, } k^*)$ 
8    $i_s^{l+1} \leftarrow i_s^l \cup k^*$ 
9    $v_s^{l+1} \leftarrow v_s^l \cup v^*$ 
10  output  $i_s^L$ 

```

Listing 1.6: mRMR MapReduce mapper function with alternative data encoding. \mathbf{v}_{class} is the class vector, v_s is the collection of selected feature vectors. The tuple (k, \mathbf{x}) is composed by the feature vector, \mathbf{x} , and the feature index, k . The *score* function processes the vectors and returns the feature score.

```

1  # broadcasted variables:  $\mathbf{v}_{class}, v_s^l$ 
2  mapper( $\cdot, (k, \mathbf{x})$ )
3      score  $\leftarrow \text{score}(\mathbf{x}, \mathbf{v}_{class}, v_s^l)$ 
4      output  $(k, \text{score})$ 

```

While in conventional encoding we used the contingency table as intermediate data structure, the design of mRMR in MapReduce with alternative encoding broadcasts at each iteration all required data for calculation to mappers. This design provides two main advantages: it deals with both discrete and continuous features as well, and the MapReduce job is composed by the Map step only. At the small cost of broadcasting some variables, all operations are executed locally. An illustrative overview of the data flow is reported in Fig. 3.

5 Results

The source code of mRMR implementation in MapReduce with both encodings is available as a Scala library, along with examples, on a public repository (<https://github.com/creggian/spark-ifs>).

We studied the scalability of the implementation of mRMR in MapReduce in both encodings in a cluster with the following specifications: Hadoop cluster of 10 nodes, where each node has Dual Xeon e5 2.4Ghz processor, 24 cores, 128GB RAM and 8TB hard disk; all nodes are connected with a 1Gb ethernet connection. Using Apache Spark v1.5.0, we submit jobs with 4GB of RAM for both the driver and the executors.

For the evaluation of mRMR implementations we used binary artificial datasets. We followed the principles of CorrAL dataset [32], in which four features determine the class value with the following formula: $c = (x_1 \wedge x_2) \vee (x_3 \wedge x_4)$, one is irrelevant and the last one is partially correlated with the class. In all our datasets, the class value (c) depends on the value of 8 features (Formula 2); the remainings are irrelevant.

$$c = ((x_1 \wedge x_2) \vee (x_3 \wedge x_4)) \wedge ((x_5 \wedge x_6) \vee (x_7 \wedge x_8)) \quad (2)$$

We assessed the scalability on the number of rows, the number of columns, the number of selected features, and the number of nodes. We used two kinds

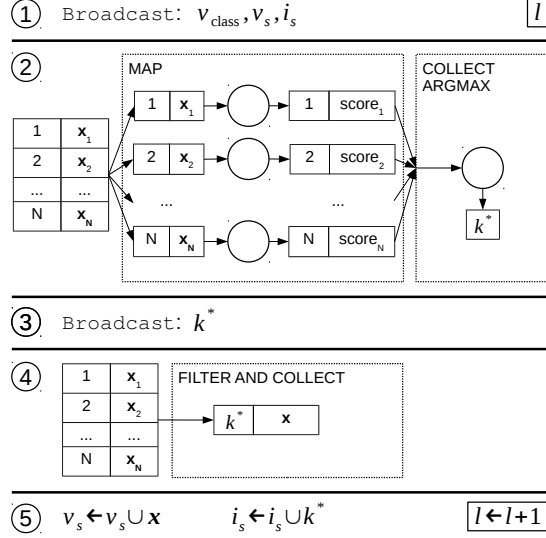


Fig. 3: Illustrative representation of a single iteration of a MapReduce job with alternative encoding. Steps 1-5 represent one iteration of the loop; there are as many iterations as the number of features to select.

of dependent variables: the relative execution time per executor and the computational gain. The former is the ratio between ET divided by ET of $1x$, the latter is the ratio between ET of 1-node and ET. We ran the tests three times to assess the variability of the results; in all figures the maximum, minimum and mean of these three values are connected through a solid vertical line.

5.1 Scalability across the number of rows

We tested the scalability on the number of rows by means of four datasets, each with 1000 columns and an increasing number of rows: 1M, 4M, 7M and 10M (M = millions). We configured the cluster and the algorithm to select 10 features in a distributed environment of 10 nodes (Fig. 4a).

5.2 Scalability across the number of columns

We assessed the scalability on the number of columns using four datasets, each with 1M rows and an increasing number of columns: 100, 400, 700 and 1000. We configured the cluster and the algorithm to select 10 features in a distributed environment of 10 nodes (Fig. 4b).

5.3 Scalability across the number of selected features

We investigated the scalability on the number of selected features using a dataset with 1M rows and 50k (k = thousands) columns. We parametrised the cluster

to distribute the computation over 10 nodes, and the algorithm to select an increasing number of features: 1, 2, 4, 6, 10 (Fig. 4c).

5.4 Scalability across the number of nodes

We tested the scalability across the number of nodes using a dataset with 1M rows and 100 columns. We configure the algorithm to select 10 features, and the cluster to distribute the work over 1, 2, 5 and 10 nodes (Fig. 4d).

By comparing the linear scalability (dotted line) with the actual performances, results show that the scalability of mRMR in MapReduce is linear with respect to the number of rows, as expected by MapReduce design; superlinear with respect to the number of columns; sublinear with respect to the number of selected features and nodes, as expected by our iterative algorithm design and the increasing amount of data exchanged in the network with the increasing of nodes, respectively.

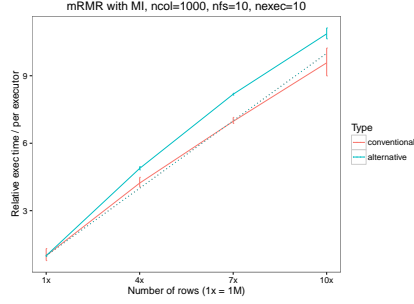
In studying mRMR with conventional and alternative layouts, we chose to use as independent variable the number of rows (columns) instead of the number of observation (features) for the following reason: while in the conventional layout we are able to scale across a very large number of rows, in the alternative layout we are strictly constraint by the amount of memory available in the mapper task to scale across the number of columns. In Figures 4a and 4b, we tested up to 10 million rows and up to one thousand columns, because very high-dimensional S/W tables raises memory errors in the cluster. Hence, even though we show the relative execution time, it would be incorrect to plot performances by increasing the number of observation (features).

The absolute execution time of mRMR MapReduce jobs with alternative encoding is generally 4-6x faster than the respective jobs with conventional encoding.

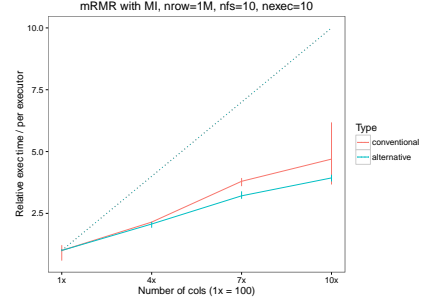
6 Conclusion

In this work we investigated the design and scalability of mRMR algorithm in MapReduce. We proposed two implementations depending on the data layout, which can be easily interfaced in order to customize the feature score function. Despite Hadoop limitations for handling data with a large number of columns, the alternative data layout is a solution to store data from a phenomenon that has a very large number of features. In both conventional and alternative data layouts, we studied the scalability of mRMR in different settings: the number of rows, columns, selected features and nodes. Our experimental results illustrated the scalability of the proposed MapReduce implementations in a large variety of settings.

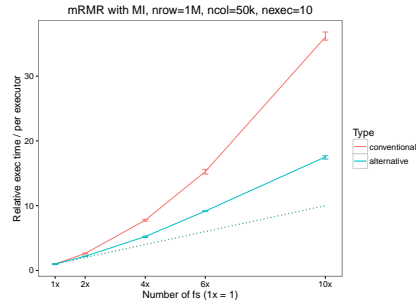
In the future, we intend to extend the approach with continuous features, and to provide an additional portfolio of built-in feature selection algorithms that work with the alternative encoding. While we design and implement known FSA



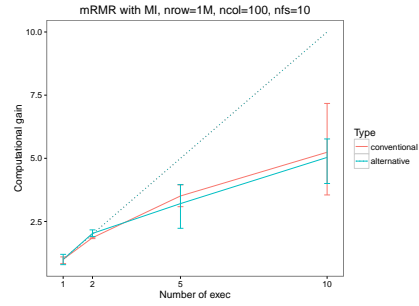
(a) mRMR scalability across the number of rows.



(b) mRMR scalability across the number of columns.



(c) mRMR scalability across the number of selected features.



(d) mRMR scalability across the number of nodes.

Fig. 4: Scalability performance of the mRMR distributed algorithm across number of rows, columns, selected features and nodes.

for MapReduce, novel algorithms that directly take advantage of the distributed nature of the data will be investigated as well. We also plan to extend the scalability study to classification and network inference tasks.

Acknowledgement

The author CR acknowledges the funding of the BridgeIris project (RBC/13-PFS EH-11) supported by INNOVIRIS (Brussels Institute for the encouragement of scientific research and innovation) and The Belgian Kids' Fund. The authors YLB and GB acknowledge the funding of the Brufence project (Scalable machine learning for automating defense system) supported by INNOVIRIS (Brussels Institute for the encouragement of scientific research and innovation).

References

1. Bolón-Canedo, V., Sánchez-Marroño, N., Alonso-Betanzos, A.: Recent advances and emerging challenges of feature selection in the context of big data. *Knowledge-Based Systems* **86** (2015) 33–45
2. Dean, J., Ghemawat, S.: Mapreduce: simplified data processing on large clusters. *Communications of the ACM* **51**(1) (2008) 107–113
3. Yeung, J.H., Tsang, C., Tsoi, K.H., Kwan, B.S., Cheung, C.C., Chan, A.P., Leong, P.H.: Map-reduce as a programming model for custom computing machines. In: *Field-Programmable Custom Computing Machines*, 2008. FCCM’08. 16th International Symposium on, IEEE (2008) 149–159
4. Thusoo, A., Sarma, J.S., Jain, N., Shao, Z., Chakka, P., Anthony, S., Liu, H., Wyckoff, P., Murthy, R.: Hive: a warehousing solution over a map-reduce framework. *Proceedings of the VLDB Endowment* **2**(2) (2009) 1626–1629
5. Chu, C., Kim, S.K., Lin, Y.A., Yu, Y., Bradski, G., Ng, A.Y., Olukotun, K.: Map-reduce for machine learning on multicore. *Advances in neural information processing systems* **19** (2007) 281
6. : Apache mahout: Scalable machine learning and data mining. <https://mahout.apache.org/>
7. Meng, X., Bradley, J., Yuvaz, B., Sparks, E., Venkataraman, S., Liu, D., Freeman, J., Tsai, D., Amde, M., Owen, S., et al.: MLlib: Machine learning in apache spark. *JMLR* **17**(34) (2016) 1–7
8. Guyon, I., Elisseeff, A.: An introduction to variable and feature selection. *J. Mach. Learn. Res.* **3** (March 2003) 1157–1182
9. Kohavi, R., John, G.H.: Wrappers for feature subset selection. *Artif. Intell.* **97**(1-2) (Dec. 1997) 273–324
10. López, F.G., Torres, M.G., Batista, B.M., Pérez, J.A.M., Moreno-Vega, J.M.: Solving feature subset selection problem by a parallel scatter search. *European Journal of Operational Research* **169**(2) (2006) 477–489
11. Melab, N., Cahon, S., Talbi, E.G.: Grid computing for parallel bioinspired algorithms. *Journal of parallel and Distributed Computing* **66**(8) (2006) 1052–1061
12. de Souza, J.T., Matwin, S., Japkowicz, N.: Parallelizing feature selection. *Algorithmica* **45**(3) (2006) 433–456
13. Garcia, D.J., Hall, L.O., Goldgof, D.B., Kramer, K.: A parallel feature selection algorithm from random subsets. In: *Proceedings of the 17th European Conference on Machine Learning and the 10th European Conference on Principles and Practice of Knowledge Discovery in Databases*, Berlin, Germany. (2006)
14. Guillén, A., Sorjamaa, A., Miche, Y., Lendasse, A., Rojas, I.: Efficient parallel feature selection for steganography problems. In: *IWANN* (1). Volume 5517 of *Lecture Notes in Computer Science.*, Springer (2009) 1224–1231
15. Singh, S., Kubica, J., Larsen, S., Sorokina, D.: Parallel large scale feature selection for logistic regression. In: *SDM, SIAM* (2009) 1172–1183
16. Peralta, D., Río, S., Ramírez, S., Triguero, I., Benítez, J.M., Herrera, F.: Evolutionary feature selection for big data classification: A mapreduce approach. *Mathematical Problems in Engineering* **2015** (2015)
17. Zhao, Z., Zhang, R., Cox, J., Duling, D., Sarle, W.: Massively parallel feature selection: an approach based on variance preservation. *Machine Learning* **92**(1) (Jul. 2013) 195–220
18. Sun, Z.: Parallel feature selection based on mapreduce. In: *Computer Engineering and Networking*. Springer (2014) 299–306

19. Ordozgoiti, B., Gómez Canaval, S., Mozo, A.: Massively parallel unsupervised feature selection on spark. In: *New Trends in Databases and Information Systems: ADBIS 2015 Short Papers and Workshops, BigDap, DCSA, GID, MEBIS, OAIS, SW4CH, WISARD, Poitiers, France, September 8-11, 2015. Proceedings.* Springer International Publishing, Cham, Switzerland (2015) 186–196
20. Bolón-Canedo, V., Sánchez-Marono, N., Alonso-Betanzos, A.: Distributed feature selection: An application to microarray data classification. *Applied Soft Computing Journal* **30** (2015) 136–150
21. Ramirez-Gallego, S., Mourio-Taln, H., Martinez-Rego, D., Boln-Canedo, V., Bentez, J.M., Alonso-Betanzos, A., Herrera, F.: An information theory-based feature selection framework for big data under apache spark. *IEEE Transactions on Systems, Man, and Cybernetics: Systems* **PP**(99) (2017) 1–13
22. Brown, G., Pocock, A., Ming-Jie, Z., Lujn, M.: Conditional likelihood maximisation: A unifying framework for information theoretic feature selection. *Journal of Machine Learning Research* **13** (1 2012) 27–66
23. Peng, H., Long, F., Ding, C.: Feature selection based on mutual information: criteria of max-dependency, max-relevance, and min-redundancy. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **27**(8) (2005)
24. Meyer, P.E., Lafitte, F., Bontempi, G.: minet: A r/bioconductor package for inferring large transcriptional networks using mutual information. *BMC Bioinformatics* **9**(461) (2008)
25. Reggiani, C., Le Borgne, Y.A., Bontempi, G.: Feature selection in high-dimensional dataset using MapReduce. *ArXiv e-prints* (September 2017)
26. Blanas, S., Patel, J.M., Ercegovac, V., Rao, J., Shekita, E.J., Tian, Y.: A comparison of join algorithms for log processing in mapreduce. In: *Proceedings of the 2010 ACM SIGMOD International Conference on Management of Data. SIGMOD '10*, New York, ACM (2010) 975–986
27. Sarma, A.D., Afrati, F.N., Salihoglu, S., Ullman, J.D.: Upper and lower bounds on the cost of a map-reduce computation. In: *Proceedings of the VLDB Endowment. Volume 6., VLDB Endowment* (2013) 277–288
28. Ahn, J., Jeon, Y.: Sparse HDLSS discrimination with constrained data piling. *Computational Statistics & Data Analysis* **90** (2015) 74 – 83
29. Jay, N.D., Papillon-Cavanagh, S., Olsen, C., Hachem, N., Bontempi, G., Haibe-Kains, B.: mRMRe: an r package for parallelized mrmr ensemble feature selection. *Bioinformatics* **29**(18) (2013) 2365–2368
30. Karau, H., Konwinski, A., Wendell, P., Zaharia, M.: *Learning Spark: Lightning-Fast Big Data Analytics*. 1st edn. O'Reilly Media, Inc. (2015)
31. Zaharia, M., Chowdhury, M., Das, T., Dave, A., Ma, J., McCauley, M., Franklin, M.J., Shenker, S., Stoica, I.: Resilient distributed datasets: A fault-tolerant abstraction for in-memory cluster computing. In: *Proceedings of the 9th USENIX Conference on Networked Systems Design and Implementation. NSDI'12*, Berkeley, CA, USENIX Association (2012) 2–2
32. Bolón-Canedo, V., Sanchez-Marono, N., Alonso-Betanzos, A.: *Feature Selection for High-Dimensional Data. Artificial Intelligence: Foundations, Theory, and Algorithms.* Springer International Publishing, Cham, Switzerland (2015)

Simultaneous Ensemble Generation and Hyperparameter Optimization for Regression

David Roschewitz, Kurt Driessens, Pieter Collins

Maastricht University, P.O. Box 616 6200 MD Maastricht, The Netherlands,
`d.roschewitz@student.maastrichtuniversity.nl`

Abstract. The development of advanced hyperparameter optimization algorithms, using e.g. Bayesian optimization, has encouraged a departure from hand-tuning. Primarily, this trend is observed for classification tasks while regression has received less attention. In this paper, we devise a method for simultaneously tuning hyperparameters and generating an ensemble, by explicitly optimizing parameters in an ensemble context. Techniques traditionally used for classification are adapted to suit regression problems and we investigate the use of more robust loss functions. Furthermore, we propose methods for dynamically establishing the size of an ensemble and for weighting the individual models. The performance is evaluated using three base-learners and 16 datasets. We show that our algorithms consistently outperform single optimized models and can outperform or match the performance of state of the art ensemble generation techniques.

Keywords: Bayesian optimization, hyperparameter optimization, ensemble generation, regression

1 Introduction

Hyperparameter tuning *for regression* is sparsely covered in research and its combination with ensemble generation appears to be entirely absent. This although both techniques have been successfully applied to classification problems. Our research addresses this omission by examining methods to automatically generate ensembles with tuned hyperparameters for regression problems.

Naïve search methods have commonly been used for hyperparameter optimization of machine learning models. Grid search, for instance, evaluates hyperparameters on a grid with a predefined resolution. This, and its inefficiency in high-dimensional space, limits the usefulness for practical applications. Random search [2] partially alleviates this limitation, but unlike Bayesian optimization, it does not leverage all available information in the tuning process [22]. Researchers have expressed a need to explore both regression problems and deep learning in the context of hyperparameter optimization to encourage the departure from hand-tuning [3, 9].

It is well accepted that ensemble methods, which consist of a combination of multiple base-learners, generally outperform single models for many types of

problems [19]. Therefore, combining ensemble generation with hyperparameter tuning is a natural step for self-optimizing algorithms. Overproduce-and-select (OPAS) methods are a popular choice for ensemble construction, and can achieve best-in-class performance [6]. During an OPAS procedure, the ensemble is constructed selecting learned models from a library of predictors, which could be the models evaluated during a hyperparameter optimization procedure.

Lévesque et al. [15] devised a method that optimizes hyperparameters and simultaneously constructs the ensemble for classification problems. Essentially, it uses all previously trained models in the context of the ensemble for the Bayesian optimization step, which computes the hyperparameters of good base-models to add to the ensemble. They demonstrate that this way of generating ensembles can outperform OPAS methods, with no significant increase in runtime, and the benefit of not requiring a library of trained models.

In their investigation, Lévesque et al. showed that the optimization and ensemble generation step can be effectively combined. Their research, however, does not analyze the suitability of the method for regression problems, which require different treatment. Furthermore, critical elements such as ensemble size and the combination function of the ensemble were fixed. This paper will discuss and develop algorithms based on the research of Lévesque et al. [15] and releases the constraint of a fixed ensemble size. 16 small and medium sized publicly available data sets are used to evaluate the proposed solutions.

The paper is structured as follows: Section 2 formally introduces Bayesian hyperparameter optimization and ensemble generation approaches. Section 3 outlines our contributions, specifically modifications and extensions made to the algorithm. Section 4 presents the experimental set-up, results and analysis.

2 Hyperparameter Optimization

Hyperparameter optimization minimizes a function $f(\gamma)$, where γ is a set of hyperparameters in Γ , the hyperparameter space. In this context, $f(\gamma)$ can be evaluated by training a model with parameters γ , $M(\gamma)$, and computing its performance. This can be measured using a so-called loss function L . Previous observations, \mathcal{D} , of the performance of parameters γ can then be used for various optimization algorithms.

The following sections explain how Bayesian optimization is used to optimize $f(\gamma)$ and how ensembles can be generated from a set of trained models. Lastly the combined optimization and ensemble generation procedure, on which this research is based, is presented.

2.1 Bayesian Optimization

In contrast to naïve search methods such as grid or random search, Bayesian optimization uses *all* previous observations \mathcal{D} to create a probabilistic model, sometimes called a *surrogate function*, $\hat{f}(\gamma)$ of the objective $f(\gamma)$. A so-called

acquisition function then computes the next point in Γ to evaluate. The model is updated with the performance of this point, and the two steps are repeated.

The surrogate function is the *posterior* distribution over the space of functions, induced by the *prior* and our observations \mathcal{D} . The prior captures our beliefs about the space of possible objective functions. For a more complete review of BO see, e.g., Brochu et al. [4] or for a broader overview see, e.g., Shahriari et al. [21].

Gaussian Process Prior Gaussian process (GP) priors are a common choice for Bayesian optimization, due to their flexibility and tractability. A GP is a distribution over functions of type $f : \Gamma \rightarrow \mathbb{R}$, and when combined with observations \mathcal{D} induces a posterior over functions. GPs are defined fully by their mean function $m : \Gamma \rightarrow \mathbb{R}$ and covariance function $k : \Gamma \times \Gamma \rightarrow \mathbb{R}$.

For hyperparameter optimization, the use of an automatic relevance detection (ARD) Matérn 5/2 kernel is suggested in literature [22].

$$k_{\text{M52}}(x, x') = \theta_0 \exp(-\sqrt{5r^2(x, x')}) (1 + \sqrt{5r^2(x, x')} + \frac{5}{3}r^2(x, x')). \quad (1)$$

$$r^2(x, x') = \sum_{d=1}^D (x_d - x'_d)^2 / \theta_d^2. \quad (2)$$

The use of ARD kernels typically results in a total of $D+3$ GP parameters¹, θ : kernel length-scales $\theta_{1:D}$, kernel amplitude θ_0 , observation noise v and constant mean m [22].

Acquisition Function By construction of the GP prior, the surrogate $\hat{f}(\gamma)$ has both a predictive mean function $\mu(\gamma)$ and predictive variance function $\sigma(\gamma)$. The acquisition function $a : \Gamma \rightarrow \mathbb{R}^+$ can then be used to determine the *utility* of any point in Γ . The next set of hyperparameters to evaluate are then simply computed as $\gamma_{n+1} = \text{argmax}_{\gamma} a(\gamma)$. Snoek et al. suggest using the Expected Improvement (EI) acquisition function, which provides a tradeoff between exploitation and exploration [22]:

$$a_{\text{EI}}(\gamma) = \sigma(\gamma)(\alpha(\gamma)\Phi(\alpha(\gamma)) + \phi(\alpha(\gamma))) \quad (3)$$

$$\alpha(\gamma) = \frac{f(\gamma_{\text{best}}) - \mu(\gamma)}{\sigma(\gamma)} \quad (4)$$

where $\Phi(x)$ is the CDF and $\phi(x)$ the PDF of the standard normal distribution.

Since the mean and variance functions are dependent on the GP parameters θ , they can be represented as $\mu(\gamma; \theta)$ and $\sigma(\gamma; \theta)$. For the fully Bayesian treatment

¹We employ the term ‘GP parameters’ to emphasize the difference between these and the hyperparameters subject to optimization in this paper.

of the GP parameters a so-called integrated acquisition function will be used throughout this paper. It is computed through a Monte Carlo estimate, using slice sampling for efficient computation of the required samples. See [22] and their additional material for further details on how the acquisition function is estimated.

2.2 Ensemble Construction from Optimization Output

Post-hoc ensemble generation (PHEG) is a natural way of constructing an ensemble from a library of trained models. PHEG is the *selection* stage of an OPAS method. In the context of optimizing hyperparameters, the history of all trained models serves as input for the ensemble generation.

The PHEG procedure works well in practice using a greedy selection criteria, which can be based on a variety of performance measures such as e.g., mean-squared-error (MSE) [6]. The procedure works as follows:

1. Begin with an empty ensemble.
2. Select the model which maximizes the ensemble performance.
3. Repeat 2 for a given number of iterations, until all models are added or until a point of diminishing returns is reached.

Caruana et al. [6] suggest modifying the procedure to allow selection with replacement. A noteworthy advantage of PHEG procedures, is that it can utilize the same performance function as the final benchmark, irrespective of how the library of models was acquired. When comparing our proposed algorithms to PHEG, we employ selection with replacement.

2.3 Simultaneous Ensemble Generation and Optimization

Hyperparameter optimization and ensemble construction are typically treated as separate procedures, with the possibility of coupling the techniques. Lacoste et al. [14] propose a bootstrapped round-robin technique, where each model of a fixed ensemble is optimized independently. Fundamentally, their ensemble sequential model-based optimization (ESMBO) procedure allows for more computationally efficient model optimization, but uses no information about the ensemble performance in the optimization.

In 2016 Lévesque et al. [15] outlined a simultaneous ensemble generation and optimization approach (SEGO). SEGO considers the performance of the ensemble, E , at every iteration. The loss function, introduced earlier, can be reformulated to evaluate the ensemble, not a single model.

Optimizing the hyperparameters of every model in E would make the objective space excessively high dimensional for large ensemble sizes. An elegant solution used by SEGO is to let the objective function $f(\gamma|E)$ be the loss of the ensemble if a model m trained with parameters γ is placed at index j of E . Hence, the *value* of parameters γ is measured *given* the current ensemble E .

$$f(\gamma|E, j) = L(E[j] \leftarrow M(\gamma)) \quad (5)$$

Throughout the SEGO algorithm, all trained models and their hyperparameter are stored in a history H and P respectively. The round-robin procedure optimizes the model at index $j = i \bmod n$ at iteration i and for ensemble size n . The loss of replacing $E[j]$ with every model in H is computed as l , using Formula 5. The loss of each hyperparameter in P can now be represented by l , and the two lists serve as input to the Bayesian optimization step, which then constructs the surrogate model $\hat{f}(\gamma)$.

The consequence of this type of loss calculation, is that for every iteration the loss must be computed $|H|$ times, resulting in runtime of $O(|H|m)$ for m prediction instances. This overhead is tolerated as the GP computation of the surrogate model contains a matrix inversion, which runs in $O(|H|^3)$. In addition, the assumption that training a learning algorithm is significantly more costly holds in practice.

Algorithm 1 formalizes the SEGO algorithm, but is adjusted to follow the notation of this paper. Note the explicit notion of the cross-validated loss computation, where the loss is averaged. In our research, the loss is estimated through 5-fold cross-validation. We employ *Spearmint* provided by Snoek et al. [22]² as the implementation of Bayesian optimization for hyperparameter tuning. Constant Gaussian noise is assumed.

Algorithm 1: Simultaneous Ensemble Generation and Optimization

Input : $b, n, a(), L(), M()$
Output: Ensemble E ; history of models H

```

1  $E, H, P \leftarrow \emptyset;$ 
2 for  $i \leftarrow 1$  to  $b$  do
3    $j \leftarrow i \bmod n;$ 
4    $l \leftarrow \text{cross-val}([L(E[j] \leftarrow m)]_{m \in H});$ 
5    $\hat{f}(\gamma) \leftarrow \text{BO}(P, l);$ 
6    $\gamma_i \leftarrow \text{argmax}_{a_{EI}(\gamma; \hat{f}(\gamma))};$ 
7    $m_i \leftarrow M(\gamma_i);$ 
8    $l \leftarrow l \cup \{\text{cross-val}(L(E[j] \leftarrow m_i))\};$ 
9    $H \leftarrow H \cup \{m_i\};$ 
10   $P \leftarrow P \cup \{\gamma_i\};$ 
11   $E[j] \leftarrow H[\text{argmin}_k l[k]];$ 
12 end
```

The research performed by Lévesque et al. showed promising results: Their algorithm outperforms a single optimized model. In some cases SEGO also performed better than an ensemble constructed post-hoc from all models trained

²Code available at
<https://github.com/JasperSnoek/spearmint>

throughout the procedure [15]. If a loss function different from the final ensemble evaluation is used for the optimization, they suggest using the PHEG method on the history H from the SEGO procedure.

SEGO was designed and tested for classification tasks only, an omission our research addresses. The authors also note that the requirement of a fixed ensemble size n should be investigated. Furthermore, we find that different loss functions as well as ensemble weighting functions should be explored. This could both improve the algorithm and also demonstrate its robustness. In the following section we introduce SEGO for regression, suitable loss functions, dynamic sizing approaches and a non-constant weighting technique.

3 SEGO for Regression

In order to apply the SEGO procedure to regression problems, the ensemble *integration function* (sometimes referred to as weighting or combination function) and the loss function must be chosen. A default choice for ensembles, is to average the results of all models. The ensemble prediction is therefore a linear combination of all model predictions, with weights $w_{1:n} = \frac{1}{n}$ for ensemble size n .

The residuals of a predictor can be defined as $r_i = \hat{y}_i - y_i$. A typical loss function for regression problems is the mean-squared-error (MSE): $MSE = \frac{1}{n} \sum_{i=1}^n (r_i)^2$. With integration and loss function defined, the baseline SEGO for regression (SEGOR) is established.

3.1 Loss function

The fashion with which the loss of parameters given an ensemble, $f(\gamma|E)$, is computed is instrumental to the iterative optimization of the ensemble. Firstly, the cross-validated loss is used directly to select the next hyperparameters γ_{i+1} , and the loss is also used to choose which model to place at index j .

A known drawback of MSE is that it places disproportionately high weights on large errors. In the context of SEGOR this means a model might be selected if it reduces the error on a difficult-to-predict outlier, but in fact reduces the performance on most unseen instances.

Regularization techniques aim to reduce overfitting, usually by complementing the minimization function with a *regularization term*. Since we are not performing a straight-forward minimization, but using an iterative procedure, adding such a term is non-trivial. However, there exist robust loss functions that aim to reduce overfitting.

One such function is the *Huber loss*, a type of robust M-estimator. In 1964 Huber introduced the idea of a generalized M-estimator minimizing $\sum_{i=1}^n \rho(x_i)$ where ρ is some function on data x [11]. Huber loss is defined as a piecewise

function of residuals whose behaviour is governed by $|r|$. This significantly reduces the impact of large errors, reducing the likelihood to overfit.

$$L_{\text{huber}}(r) = \begin{cases} \frac{1}{2}r^2 & \text{if } |r| < c \\ c(|r| - \frac{1}{2}c) & \text{otherwise} \end{cases}$$

Another, more extreme, type of robust loss function is the *Tukey bisquare*. In contrast to Huber loss, for large $|r|$, the loss is constant. This further reduces the weight given to large errors. Tukey's bisquare has been used successfully as a loss function for other regression problems [1].

$$L_{\text{tukey}}(r) = \begin{cases} \frac{c^2}{6}[1 - (1 - (\frac{r}{c})^2)^3] & \text{if } |r| < c \\ \frac{c^2}{6} & \text{otherwise} \end{cases}$$

The chosen values of c for Huber loss and Tukey's bisquare are $c = 1.345$ and $c = 4.685$, respectively. At these values of c , both loss functions are at least 95% as accurate as least-squares if the data is sampled from a normal distribution. Figure 1 highlights the differences in the loss functions.

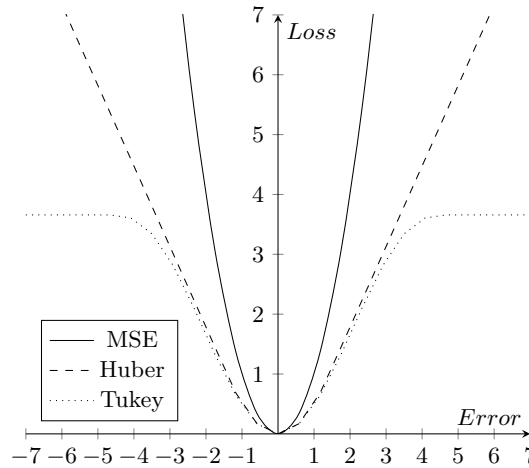


Fig. 1: Comparison of loss functions

3.2 Ensemble size

Much of ensemble research has focused on selecting an ensemble from a library of models [5, 12]. SEGOR currently optimizes parameters given a fixed ensemble size n . Such ensemble generating algorithms are sufficiently novel that there are no established approaches to dynamically adjust the ensemble size throughout the optimization procedure.

Lévesque et al. empirically selected $n = 12$ for their investigation, and our work confirms that this is a reasonable choice, but this might not hold for all types of base-models and datasets. We therefore devised multiple methods of dynamically generating the ensemble *throughout* the optimization process with *no* fixed size.

The fixed ensemble size method (F) will serve as the baseline. Note that the modifications to non-fixed ensemble sizes are performed *after* the BO step, and hence do not affect the input of the hyperparameter tuning. The first proposed method, *best-growing* (BG), initializes the ensemble as $E = \emptyset$, and can add any model $m \in H$, in addition to replacing the model at index j . If a model is added to the ensemble, the remainder of the ensemble is left unchanged. This means that at iteration i , i models are evaluated at index j , and i models are evaluated as additions. An additional set of losses, l_{grow} is computed:

$$l_{\text{grow}} = \text{cross-val}([L(E \cup m)]_{m \in H}) \quad (6)$$

Similarly, the *best-dynamic* (BD) method allows for the deletion of a model in the ensemble in addition to growing:

$$l_{\text{shrink}} = \text{cross-val}(L(E[j] \leftarrow \emptyset)) \quad (7)$$

Each method then decides which action to perform: replace, grow (BG and BD only) or shrink (BD only).

3.3 Integration function

In SEGO, as it was designed for classification, a majority voting ensemble *integration function* was used. As the ensemble prediction in SEGOR is already a linear combination of its component regressors, the weights $w_{1:n}$ for ensemble size n are suitable for optimization [20]. If we express the set of weights w_i for $i = 1 : n$ as w , then by finding $\text{argmin}_w L(\sum_{i=1}^N (r_i(x) * w_i))$ it is possible to find weights minimizing the validation loss, with an increased risk of overfitting the validation set, however. For the hyperparameter tuning step mean-weighting is used as otherwise the individual models' equal impact on the loss-evaluation is no longer guaranteed. We explore weighting using Python's SciPy implementation of BFGS [13].

4 Experiments

The objective of the experimentation is two-fold: Firstly, we want to demonstrate the applicability of SEGO to regression and secondly highlight the performance and robustness of our improvements. Three different types of predictors are used as base-models for hyperparameter tuning tested on 16 small to medium sized datasets. As each experiment requires training a significant number of models in addition to the optimization overhead, and computational resources

were limited for this research, the scope of experimentation had to be restricted. Therefore every dataset was limited to 2999 instances. Each experiment is 5-fold cross validated and the number of Bayesian optimization iterations was set to 100 for every tested approach. The following shorthand will be used: *method - loss-function*, where *s* is a single model and *f* represents SEGOR with a fixed ensemble size of 12; *g* and *d* are the BG and BD ensemble size methods from Section 3.2, respectively. The loss functions *m*, *h* and *t* correspond to MSE, Huber loss and Tukey’s bisquare. Lastly, the suffix *ph* refers to the ensemble generated post-hoc using PHEG.

All regression models were chosen from the scikit-learn library [18]. The base-models and their hyperparameters are:

- DecisionTree (DT):
max_depth, max_features, min_samples_split, min_samples_leaf
- MultiLayerPerceptron (MLP):
learning_rate, activation, hidden_layer_sizes (max. 2 hidden layers)
- ElasticNet (EN):
alpha, l1_ratio, max_iter

The datasets and the number of used features and instances can be viewed in Table 1. Unless cited otherwise, they were retrieved from the UCI Machine Learning Repository [16]

Dataset	Instances	Features
CPU (cpu)	209	6
Boston Housing (bos)	506	13
White Wine (ww)	2999	11
Red Wine (rw)	1599	11
Chicago Speed (csp) [24]	2999	3
MPG (mpg)	398	6
Power Plant (pow)	2999	4
Solar Flare(sf)	1066	23
Facebook Comments (fp)	500	10
Air Quality (aq)	2999	12
Concrete Stength (cs)	1030	8
Cooling Efficiency (ce)	768	8
Heating Efficiency (he)	768	8
Math Grades (mg)	395	56
Yacht Resistance (yr)	308	6
Forest Fire Area (ffa)	517	22

Table 1: Datasets

A single predictor (*s-m*) optimized for 100 iterations and a post-hoc ensemble generated from its history (*s-m-ph*) will serve as a baseline. The performance is measured as the MSE of previously normalized values, irrespective of the loss function used. Furthermore, the diversity of the ensemble will be measured by the mutual information (MI) of the individual models’ predictions. In order to

measure MI of an ensemble, the sum of all pairwise MI is taken, divided by the ensemble size squared [8]. Low values of MI represent ensembles with diverse predictions.

$$MI(Y^m; Y^n) = -\frac{1}{2} \log(1 - \rho^2) \quad (8)$$

where Y^i is the prediction of model i , and ρ is the correlation between two predictors:

$$\rho = \frac{\sum_{i=1 \dots N} (y_i^m - \mu_Y^m)(y_i^n - \mu_Y^n)}{\sqrt{\sum_{i=1 \dots N} (y_i^m - \mu_Y^m)^2 (y_i^n - \mu_Y^n)^2}}. \quad (9)$$

4.1 Performance Results

	s-m	f-m	g-m	d-m	s-m-ph	f-h-ph	g-h-ph	d-h-ph	f-t-ph	g-t-ph	d-t-ph
DT	10.19	4.12	4.56	5.50	6.19	5.69	5.88	7.00	4.81	5.44	6.62
MLP	9.38	7.25	4.81	3.88	5.56	6.31	6.50	5.69	5.88	4.69	6.06
EN	7.00	6.00	6.81	6.00	6.81	6.06	6.00	5.06	5.38	5.25	5.44

Table 2: Mean ranks over 3 hyperparameter spaces

In order to compare the different algorithms, their performance was measured for each of the base-models separately. We compare the combination of different sizing methods and loss functions.

For methods using loss functions other than MSE, a post-hoc ensemble is used to measure performance, as the PHEG procedure can utilize the same performance-function as the final benchmark. Therefore, post-hoc results are used where appropriate to ensure methods are not disadvantaged. An intuitive and insightful way to compare different algorithms is to observe their mean ranks, which are shown in Table 2.

In order to measure performance differences we used two procedures. The first is a two-step process for comparing multiple methods simultaneously. First, a Friedman test is used to test whether there is a significant difference between all methods [7]. In principle, the test considers the mean rank, and is a non-parametric version of the well known ANOVA test. The p -values are 2.8×10^{-5} , 6.0×10^{-4} and 0.72 for DT, MLP and EN respectively. This suggests that for elastic nets, all methods perform similarly. Further investigation revealed that elastic nets performed worse in absolute terms compared to the other base types.

When the Friedman test shows significant differences, a post-hoc test is utilized to determine which methods differ from one another. The Nemenyi test can be used to compare all methods based on mean ranks [7], and includes a compensation for multiple comparisons. Our findings show that the Nemenyi test is more conservative, and is very sensitive to the methods selected for comparison. For instance, *d-h-ph* no longer outperforms *s-m* at $p < 0.05$ for DT space. In the

case of MLPs, $f-m$, $f-h-ph$, $g-h-ph$, $d-h-ph$, $f-t-ph$ and $d-t-ph$ are all above the critical p -value when compared to $s-m$. All performance differences are insignificant even at $p < 0.10$ for elastic nets.

Table 3 shows the results of the Nemenyi post-hoc test, comparing all proposed methods with the single optimized model $s-m$. All other comparisons values are insignificant at $p > 0.10$, and are not shown but were performed simultaneously.

	f-m	g-m	d-m	s-m-ph	f-h-ph	g-h-ph	d-h-ph	f-t-ph	g-t-ph	d-t-ph
DT	0.00	0.00	0.00	0.03	0.00	0.00	0.13	0.00	0.00	0.04
MLP	0.77	0.00	0.00	0.05	0.24	0.33	0.06	0.10	0.00	0.15

Table 3: Nemenyi test p -values for both hyperparameter spaces

The second procedure, the Wilcoxon signed-ranks test is a non-parametric paired test, which can compare the performance of two algorithms given multiple instances [7]. The methodology considers the absolute value of the performance difference, and is therefore more representative than mean ranks. Furthermore, it does not assume sampling from a normal distribution, as the paired t-test would.

In summary, all methods outperform the single optimized model, with $p < 0.05$ for both decision trees and multi-layer perceptrons. For elastic nets however, all methods show insignificant improvement at $p < 0.05$ in the pairwise comparison. Notable is the performance of $f-m$ which outperforms $s-m-ph$ at $p < 0.05$ for decision trees, therefore outperforming an ensemble generated using an OPAS method. Similarly $g-m$ and $d-m$ perform better than $f-m$ for multi-layer perceptrons. This highlights the importance of non-fixed ensemble sizing for certain domains. Tables 4 and 5 show all pairwise p -values of the Wilcoxon tests, the structure of the tables are such that a value in row i and column j represents the p -value that algorithm i outperforms algorithm j .

	s-m	f-m	g-m	d-m	s-m-ph	f-h-ph	g-h-ph	d-h-ph	f-t-ph	g-t-ph	d-t-ph
s-m		1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
f-m	0.00		0.47	0.05	0.00	0.07	0.02	0.01	0.43	0.03	0.00
g-m	0.00	0.55		0.06	0.07	0.19	0.11	0.01	0.70	0.14	0.03
d-m	0.00	0.96	0.94		0.15	0.70	0.43	0.19	0.74	0.63	0.55
s-m-ph	0.00	1.00	0.94	0.86		0.72	0.70	0.23	0.83	0.57	0.74
f-h-ph	0.00	0.94	0.83	0.32	0.30		0.08	0.01	0.75	0.45	0.03
g-h-ph	0.00	0.99	0.90	0.59	0.32	0.93		0.05	0.86	0.74	0.49
d-h-ph	0.00	0.99	0.99	0.83	0.78	0.99	0.95		1.00	0.99	0.96
f-t-ph	0.00	0.59	0.32	0.28	0.19	0.26	0.15	0.00		0.47	0.01
g-t-ph	0.00	0.97	0.87	0.39	0.45	0.57	0.28	0.01	0.55		0.15
d-t-ph	0.00	1.00	0.97	0.47	0.28	0.97	0.53	0.05	0.99	0.86	

Table 4: Pairwise Wilcoxon test p -values for decision trees

	s-m	f-m	g-m	d-m	s-m-ph	f-h-ph	g-h-ph	d-h-ph	f-t-ph	g-t-ph	d-t-ph
s-m		0.99	1.00	1.00	1.00	1.00	0.99	1.00	0.99	1.00	1.00
f-m	0.01		0.97	0.99	0.55	0.65	0.28	0.65	0.47	0.87	0.74
g-m	0.00	0.03		0.70	0.19	0.01	0.06	0.14	0.10	0.33	0.26
d-m	0.00	0.01	0.32		0.07	0.08	0.03	0.02	0.03	0.17	0.10
s-m-ph	0.00	0.47	0.83	0.94		0.53	0.45	0.78	0.61	0.89	0.88
f-h-ph	0.00	0.37	0.99	0.93	0.49		0.35	0.57	0.37	0.85	0.43
g-h-ph	0.01	0.74	0.95	0.98	0.57	0.67		0.28	0.53	0.94	0.55
d-h-ph	0.00	0.37	0.87	0.99	0.23	0.45	0.74		0.55	0.75	0.68
f-t-ph	0.01	0.55	0.91	0.98	0.41	0.65	0.49	0.47		0.65	0.41
g-t-ph	0.00	0.14	0.68	0.84	0.12	0.16	0.07	0.26	0.37		0.13
d-t-ph	0.00	0.28	0.75	0.91	0.13	0.59	0.47	0.33	0.61	0.88	

Table 5: Pairwise Wilcoxon test p -values for multi-layer-perceptrons

We have excluded the weighted methods from the above comparisons, as we can observe weak performance, likely due to overfitting on the validation data or multi-collinearity among the predictors [17].

In conclusion, the proposed methods work well on the DT and MLP hyperparameter space, where the tunable parameters strongly affect the way the regressor learns. The sizing methods performed well, with g - m and d - m trading spots depending on the hyperparameter space. Robust loss functions did not improve generalization accuracy overall, and the Nemenyi test indicates they do not always perform significantly better than a single optimized regressor. We suspect this is related to the relatively small datasets. We believe given more diverse data or if the real loss function cannot be used for learning, robust loss functions would be more applicable.

4.2 Diversity

We can utilize the MI diversity measure to explain the poor performance of the algorithms on the EN hyperparameter space. Table 6 highlights the much greater mean mutual information in the ensemble of elastic nets. This corresponds with the design of elastic nets, where the impact of different parameters has a limited effect on the prediction. Therefore, our hyperparameter tuning focus does not work well for such base-models, and the MI measure can be used for detecting this situation.

	dt	nn	en
mean mut. info.	1.35	1.42	4.28

Table 6: Mean distance measures for the hyperparameter spaces

As all of the aforementioned experiments were conducted with a budget of 100 iterations, the ensemble methods do not converge. We conducted an experiment

using the 'rw' dataset for decision trees, with 300 iterations. The fixed-ensemble-size method $f\text{-}m$ was tested for all sizes in the range $[4, 28]$, with sizes 25, 22 and 19 being the top performers. Our BG method $g\text{-}m$ ranked 5th, with a final mean ensemble size of 27.76. Interestingly, a fixed size of 24 would have resulted in the 20th rank for $f\text{-}m$. This high sensitivity to the a-priori fixed ensemble size highlights the importance of the dynamic ensemble sizing methods. Due to computational limitations, we were unable to investigate true convergence.

5 Conclusion

We presented a method to simultaneously generate ensembles and tune the hyperparameters of its models for regression problems. Furthermore, we introduce robust loss functions and different methods of determining the size of the ensemble on-the-fly. For models with tunable hyperparameter spaces, our proposed techniques significantly outperform single regressors. The proposed sizing methods allow the algorithm to operate without a fixed a-priori ensemble size, a parameter which was shown to impact performance. The proposed robust loss functions have failed to exceed the performance of procedures using MSE, but tend to outperform single models. For models where hyperparameters only slightly affect the diversity in predictions, the suggested methods cannot significantly improve on a single tuned predictor.

Noteworthy is the finding that depending on the chosen base-learner and dataset, a different approach might be the most suitable, highlighted by differences found between decision trees and multi-layer-perceptrons. Most importantly, however, our research demonstrates the suitability of hyperparameter tuning to regression, and showcases the performance of automated meta-learning algorithms, specifically for ensemble generation with no fixed size.

Future Research

With more computational resources available, the proposed methods could be investigated on a greater breadth of larger datasets with more optimization iterations. The utilization of a diversity or complexity term during optimization has had mixed results in other applications [10], but could nonetheless be used to aid in dynamic ensemble sizing. For highly complex hyperparameter spaces, neural networks are suitable replacements for GPs [23]. Our research focused little on ensemble weighting, but other techniques such as stacking or regularized weighting should not be discarded. Furthermore, the suitability of hyperparameter tuning and more advanced techniques such as SEGO(R) have not been extensively explored for deep neural networks, an area which we hope will see increased attention in the future.

Acknowledgements

We want to thank Mediaan for supporting this research and graciously providing compute resources.

Bibliography

- [1] Belagiannis, V., Rupprecht, C., Carneiro, G., Navab, N.: Robust optimization for deep regression. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 2830–2838 (2015)
- [2] Bergstra, J., Bengio, Y.: Random search for hyper-parameter optimization. *Journal of Machine Learning Research* **13**(Feb), 281–305 (2012)
- [3] Bergstra, J.S., Bardenet, R., Bengio, Y., Kégl, B.: Algorithms for hyper-parameter optimization. In: Advances in Neural Information Processing Systems, pp. 2546–2554 (2011)
- [4] Brochu, E., Cora, V.M., De Freitas, N.: A tutorial on bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning. *arXiv preprint arXiv:1012.2599* (2010)
- [5] Caruana, R., Munson, A., Niculescu-Mizil, A.: Getting the most out of ensemble selection. In: Data Mining, 2006. ICDM'06. Sixth International Conference on, pp. 828–833. IEEE (2006)
- [6] Caruana, R., Niculescu-Mizil, A., Crew, G., Ksikes, A.: Ensemble selection from libraries of models. In: Proceedings of the twenty-first international conference on Machine learning, p. 18. ACM (2004)
- [7] Demšar, J.: Statistical comparisons of classifiers over multiple data sets. *Journal of Machine learning research* **7**(Jan), 1–30 (2006)
- [8] Dutta, H.: Measuring diversity in regression ensembles. In: IICAI, vol. 9, p. 17p (2009)
- [9] Feurer, M., Klein, A., Eggenberger, K., Springenberg, J., Blum, M., Hutter, F.: Efficient and robust automated machine learning. In: Advances in Neural Information Processing Systems, pp. 2962–2970 (2015)
- [10] Gu, S., Cheng, R., Jin, Y.: Multi-objective ensemble generation. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery* **5**(5), 234–245 (2015)
- [11] Huber, P.J., et al.: Robust estimation of a location parameter. *The Annals of Mathematical Statistics* **35**(1), 73–101 (1964)
- [12] Johansson, U., Löfström, T., Boström, H.: Overproduce-and-select: The grim reality. In: Computational Intelligence and Ensemble Learning (CIEL), 2013 IEEE Symposium on, pp. 52–59. IEEE (2013)
- [13] Jones, E., Oliphant, T., Peterson, P., et al.: SciPy: Open source scientific tools for Python (2001–). URL <http://www.scipy.org/>
- [14] Lacoste, A., Larochelle, H., Laviolette, F., Marchand, M.: Sequential model-based ensemble optimization. *arXiv preprint arXiv:1402.0796* (2014)
- [15] Lévesque, J.C., Gagné, C., Sabourin, R.: Bayesian hyperparameter optimization for ensemble learning. In: A. Ihler, D. Janzing (eds.) Proceedings of the Thirty-Second Conference (2016) on Uncertainty in Artificial Intelligence, pp. 437–446. AUAI Press (2016)
- [16] Lichman, M.: Uci machine learning repository (2013). URL <http://archive.ics.uci.edu/ml>

- [17] Mendes-Moreira, J., Soares, C., Jorge, A.M., Sousa, J.F.D.: Ensemble approaches for regression: A survey. *ACM Computing Surveys (CSUR)* **45**(1), 10 (2012)
- [18] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E.: Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research* **12**, 2825–2830 (2011)
- [19] Rokach, L.: Ensemble-based classifiers. *Artificial Intelligence Review* **33**(1-2), 1–39 (2010)
- [20] Seni, G., Elder, J.F.: Ensemble methods in data mining: improving accuracy through combining predictions. *Synthesis Lectures on Data Mining and Knowledge Discovery* **2**(1), 1–126 (2010)
- [21] Shahriari, B., Swersky, K., Wang, Z., Adams, R.P., de Freitas, N.: Taking the human out of the loop: A review of bayesian optimization. *Proceedings of the IEEE* **104**(1), 148–175 (2016)
- [22] Snoek, J., Larochelle, H., Adams, R.P.: Practical bayesian optimization of machine learning algorithms. In: *Advances in neural information processing systems*, pp. 2951–2959 (2012)
- [23] Snoek, J., Rippel, O., Swersky, K., Kiros, R., Satish, N., Sundaram, N., Patwary, M., Prabhat, M., Adams, R.: Scalable bayesian optimization using deep neural networks. In: *International Conference on Machine Learning*, pp. 2171–2180 (2015)
- [24] Speed camera violations, chicago data portal. <https://data.cityofchicago.org/Transportation/Speed-Camera-Violations/hhkd-xvj4/data>

Comparison of Machine Learning Techniques for Multi-label Genre Classification

Mathijs Pieters and Marco Wiering

Institute of Artificial Intelligence and Cognitive Engineering
University of Groningen, The Netherlands
`{m.t.pieters,m.a.wiering}@rug.nl`

Abstract. We compare classic text classification techniques with more recent machine learning techniques and introduce a novel architecture that outperforms many state-of-the-art approaches. These techniques are evaluated on a new multi-label classification task, where the task is to predict the genre of a movie based on its subtitle. We show that pre-trained word embeddings contain 'universal' features by using the Semantic-Syntactic Word Relationship test. Furthermore, we explore the effectiveness of a convolutional neural network (CNN) that can extract local features, and a long short term memory network (LSTM) that can find time-dependent relationships. By combining a CNN with an LSTM we observe a strong performance improvement. The technique that performs best is a multi-layer perceptron, with as input the bag-of-words model.

Keywords: natural language processing, multi-label text classification, movie subtitles, CNN model, LSTM network, bag-of-words model

1 Introduction

Text classification is the task of assigning specific categories to documents, examples are spam detection and sentiment analysis. Naive Bayes, a technique based on applying Bayes' Theorem, is frequently used as a baseline method for text classification because it is relatively effective, fast, and easy to implement [11]. Numerous attempts have been made to tackle the poor assumptions of Naive Bayes [8, 17].

Various types of neural networks have been developed throughout the years, many of these techniques are used for natural language processing (NLP) applications. A traditional method is the multilayer perceptron (MLP), trained on the bag-of-words (BoW) model [1]. The BoW model is a sparse representation of texts, ignoring both word order and semantic and syntactic features, treating texts as unordered sets of words. In order to capture the subtleties of language, we seek a dense representation that does capture these features. Many state-of-the-art word embedding techniques [12, 15] are based on the distributional hypothesis [3], stating that *linguistic items with similar distributions have*

similar meanings. These dense representations capture multiple degrees of similarity [14], both semantic and syntactic, such that similar words have similar representations.

Convolutional neural networks (CNN) make use of the internal structure of the dense representation, both in the feature domain, and the temporal (word order) domain. CNN models have achieved remarkable results on various text classification tasks [21, 5]. Whereas CNN models make use of the word order for a specific region size, recurrent neural networks (RNN) have the ability to capture long-term dependencies for texts of any length. More specifically, the Long Short-Term Memory (LSTM) architecture [4] is well suited for longer texts because of its ability to remember information for long periods of time.

In this paper, we introduce a novel dataset which we will use for multi-label text classification. We compare several state-of-the-art techniques, such as the concatenation-CNN and the LSTM network, with more traditional techniques. Furthermore, we introduce a novel architecture that applies a histogram on word embeddings, followed by an MLP. Unlike most research, we trained our own word embeddings, making our setup stand-alone.

In section 2 we introduce our dataset, followed by section 3 where we explain the used methods. The experimental setup is described in section 4, and in section 5 we show and discuss the results. We conclude the paper in section 6 with a conclusion and a proposal for future work.

2 Dataset

The dataset used in the experiment is an intra-lingual movie subtitle corpus, collected by [9], and originates from OpenSubtitles¹. We extracted the English corpus, and removed all tokens apart from the spoken text. Subsequently, we convert all words to lowercase and remove punctuation, see Figure 1. We did not apply stop word removal or stemming. The total dataset consists of 44,171 subtitles, with in total 135,862,112 words and 920,705 unique words. Every subtitle is linked to at least one, and often multiple genres. In total the dataset contains subtitles with 27 different genres, ranging from animation and comedy, to documentary. Because every subtitle can have multiple genres, the classification task is considered a multi-label classification task. This should not be confused with multi-class classification, where every document has exactly one label. Multi-label classification is considered to be significantly more difficult, due to the vast amount of possible label combinations.

Because of the limited availability of computer power we will narrow our focus to the classification of the following genres: "Romance", "Thriller", and "Action". This subset consists of 15,500 subtitles, with in total 48,998,774 words and 448,101 unique words. The distribution of the subtitle lengths is depicted in Figure 2.

¹ <http://www.opensubtitles.org/>

```

<s id="1637">
  <time id="T514S" value="01:38:8,709" />
  <w id="1637.1">May</w>
  <w id="1637.2">the</w>
  <w id="1637.3">Force</w>
  <w id="1637.4">be</w>
  <w id="1637.5">with</w>
  <w id="1637.6">you</w>
  <w id="1637.7"></w>
  <time id="T514E" value="01:38:11,018" />
</s>

```

may the force be with you

Fig. 1. Text preprocessing, single sentence example.

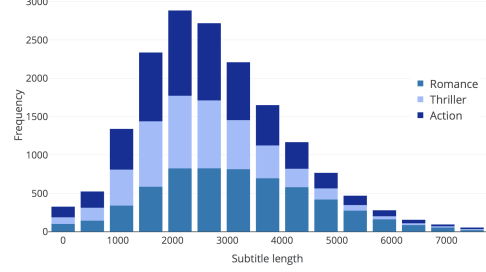


Fig. 2. Distribution of subtitle lengths

3 Methods

In this paper, we will differentiate between models that use the BoW model, and models that use word embeddings. For the first model we have two different methods, and for the latter we will discuss four methods.

3.1 Bag of Words

We will use the BoW model for both the Naive Bayes classifier and the multi-layer perceptron. Let $\mathbf{d} = \{d_1, \dots, d_n\}$ be a collection of documents, where d_{ij} denotes the number of occurrences of word i in document j . Furthermore, let $\mathbf{l} = \{l_1, \dots, l_n\}$ be the according labels, where l_i is itself a set of possibly multiple labels.

Multinomial Naive Bayes transformations We will focus on the Multinomial version of the Naive Bayes model (MNB), where each word position is assumed to be independent of every other word. We will use several improvements proposed by [17], e.g. normalising the weight vectors.

To train the MNB model, we apply the transformations described in equation 1. For a test document t , with word i occurring t_i times, the document is labelled according to equation 2 for some threshold θ .

$$\begin{aligned}
 d_{ij} &\stackrel{(1.1)}{=} d_{ij} \log \frac{\sum_k 1}{\sum_k 1_{i \text{ occurs in } k}} & w_g &\stackrel{(2.1)}{=} \sum_i t_i w_{gi} \\
 d_{ij} &\stackrel{(1.2)}{=} \frac{d_{ij}}{\sqrt{\sum_k (d_{kj})^2}} & w_g &\stackrel{(2.2)}{=} \frac{w_g - \min w_g}{\max w_g - \min w_g} \\
 w_{gi} &\stackrel{(1.3)}{=} \log \sum_{j: g \in l_j} d_{ij} & l(t) &\stackrel{(2.3)}{=} \{g : w_g > \theta\}
 \end{aligned} \tag{1}$$

In equation 1.1, we down-weight common words, a heuristic known as "inverse document frequency". Common words have little influence on the class of a document, but small variations can cause spurious correlations. Note that in most literature a "term frequency" heuristic precedes equation 1.1, we however found that this did not improve the accuracy. Therefore as shown in equation 1.1 we just use the term frequency. In order to prevent that document length affects the classification, we normalize every document according to equation 1.2. Finally, in equation 1.3 we add the weights of all documents belonging to the same genre.

For classification we first multiply each word frequency with the weight, as illustrated in equation 2.1. In standard multi-class classification, we could now assign a label to the class with the highest score. However, since the task is multi-label classification, we have to be able to assign multiple labels to a single document. We do this by first normalizing the weights according to equation 2.2, and then assign each label for which the weight is greater than the predefined threshold θ . By increasing θ we can trade-off recall for precision (defined in section 5.1). We determine this threshold by means of the validation set.

Multi-layer Perceptron The multi-layer perceptron (MLP) has been shown to be effective on a wide variety of tasks, despite its simplicity. We use a fully connected network, with two hidden layers. We use the ReLU activation function, and in every layer we apply L_2 -normalisation before activation. The input of the MLP is again the BoW model, with the n most frequent words. Every word frequency is rescaled according to $d_{ij} = \log(1 + d_{ij})$, reducing the influence of frequently occurring words.

3.2 Skip-gram Model

Many state-of-the-art techniques require dense word vectors as input. It is hypothesised that the techniques developed by e.g. [12] create dense word vectors that contain 'universal' features that can be used for various tasks. We will focus on the Skip-gram model [13]. In this model, each current word is used as an input, and the target is to predict the words that occur within a certain context c before and after the center word, as illustrated in Figure 3. Furthermore, we use Negative sampling (NGE) as objective, where the task is to distinguish the target word from k negative samples drawn from a noise distribution. Since frequent words generally provide less information, we apply subsampling to all words as described in [13]. We train the model using all subtitles in our dataset, in section 4.2 we denote the used hyperparameters. In order to explore the quality of the word vectors we use the Semantic-Syntactic Word Relationship test set, defined in [12]. This test set consists of five types of semantic questions and nine types of syntactic questions. The task is to predict a word, based on the relationship between three given words. An example for the semantic test is: "What word is similar to *Oslo* in the same way as *France* is similar to *Paris*?", the answer would be *Norway*. This test is performed by computing the vector

$\mathbf{x} = \text{vector}(\text{"france"}) - \text{vector}(\text{"paris"}) + \text{vector}(\text{"oslo"})$, and finding the word that has the smallest cosine distance to this vector \mathbf{x} (different from the three question words). An answer is considered correct only if the closest word is identical to the word in the question. Table 1 shows the results on the word analogy task, indicating the effectiveness of the technique as well as generalizability of the used dataset. For the accuracy we denote both the percentage correct, and the number of correct classified pairs combined with the total number of pairs. Note that we used a subsection of the original test set, because some of the test words do not occur in our dataset. We evaluated 6,067 out of the original 8,869 semantic relations, for the syntactic relations we evaluated 10,300 out of 10,675 pairs. The results show that for the semantic relations the categories Common capital city and Man-Woman are learned very accurately, whereas Currency scores poorly. We expect that this is a result of the nature of movie subtitles, relationships (Man-Woman) and famous locations (Common capital city) play an important part in many movies, in contrast to currencies. The syntactic relations show a more balanced result, probably because all nine syntactic categories occur in spoken language.

Table 1. Results of Semantic-Syntactic Word Relationship test set.

Category	Accuracy
<i>Semantic:</i>	43.9 % (2665/6067)
Common capital city	86.6 % (433/506)
All capital cities	43.1 % (996/2310)
Currency	7.40 % (37/502)
City-in-state	35.4 % (824/2328)
Man-Woman	89.3 % (375/420)
<i>Syntactic:</i>	61.8 % (6362/10300)
Adjective to adverb	31.1 % (271/870)
Opposite	25.6 % (180/702)
Comparative	81.6 % (1087/1332)
Superlative	64.4 % (723/1122)
Present participle	62.7 % (622/992)
Nationality adjective	68.6 % (1044/1521)
Past tense	61.3 % (957/1560)
Plural nouns	82.1 % (1093/1332)
Plural verbs	44.3 % (385/869)

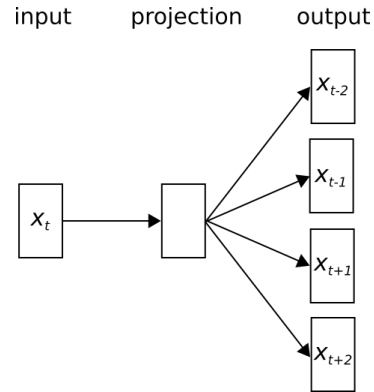


Fig. 3. The Skip-gram architecture, with a context size c of 2.

3.3 MLP on Histogram of Word Embeddings

Previous research has shown that first training a part of the model on an unsupervised task can reduce the training time and increase the accuracy on the supervised task [16]. Because the pre-trained word embeddings contain various features, we expect that a basic model can find relationships between several features in order to learn a supervised task. Preliminary experiments have shown

that taking scalar indicators (such as min, max, or mean) of a single feature over all words in combination with an MLP does not lead to satisfying results. Both the min and max operators can be affected by single, meaningless outliers, whereas the mean operator can potentially reduce significant positive and negative weights to a meaningless average. In order to capture more information we propose to use a histogram, where each word-embedding feature is described by a certain number of bins. Every bin denotes the relative frequency of a range of values for that specific feature. We will now describe the method used to convert a document to a word-embedding histogram, that subsequently can be used as an input for an MLP. Let every subtitle be consisting of n words, such that

$$X = \mathbf{x}_1 \oplus \mathbf{x}_2 \oplus \dots \oplus \mathbf{x}_n \quad (3)$$

where $\mathbf{x}_t \in \mathbb{R}^k$ is the k -dimensional word embedding and \oplus is the concatenation operator. Note that in most literature word embeddings are referred to by "words", we will use "concepts" because the word embeddings are actually the representation of the concept of a word, and not the word itself. The concatenation of the word embeddings results in a matrix $X \in \mathbb{R}^{n \times k}$, where n and k denote the number of words in the subtitle and dimension of the word embedding respectively. In order to use this matrix in combination with a histogram, we first need to scale the values such that we can use bins with a prefixed size and range. We normalize the matrix X according to

$$X_{ij} = \frac{X_{ij} - \min_i X_{ij}}{\max_i X_{ij} - \min_i X_{ij}} \quad (4)$$

We will now make a histogram along every word dimension, using s bins, where every bin has a width of size $1/s$. The range of the bins are denoted by $\{b_1 = [0, \frac{1}{s}), b_2 = [\frac{1}{s}, \frac{2}{s}), \dots, b_s = [\frac{s-1}{s}, 1]\}$. For every bin b_l and every word dimension k we now calculate

$$H_{lk} = \text{card}(\{X_{jk} : X_{jk} \in b_l\}) \quad (5)$$

Subsequently, we calculate the L_1 norm

$$H_{lk} = \frac{H_{lk}}{\sum_{i=1}^s H_{ik}} \quad (6)$$

and calculate the z -score

$$Z_{lk} = \frac{H_{lk} - \mu}{\sigma} \quad (7)$$

where μ , and σ are the mean and standard deviation of all values in H respectively. The resulting matrix $Z \in \mathbb{R}^{s \times k}$ is then used as an input for an MLP.

3.4 Convolutional Neural Network

A Convolutional Neural Network (CNN) is a feedforward neural network, originally used for image classification [7]. CNN models have shown to be effective

on various NLP tasks, by utilising local features of the word embeddings [6]. We will now describe the CNN architecture. Let every document of length n be described by a sequence as defined in equation 3 (padded if necessary). Let $x_{i:i+j}$ denote the concatenation of concept x_i up to x_{i+j} . The convolutional filter $\mathbf{w} \in \mathbb{R}^{h \times k}$ is applied to a window of h concepts, which produces a new feature c_i . Note that k denotes again the word embedding size. We could in theory slide the convolution along the word-features too, there is however no reason to assume that any specific local relationships exist between concepts. The window of concepts $x_{i:i+h-1}$ generates a new feature by

$$c_i = f(\mathbf{w} \circ \mathbf{x}_{i:i+h-1} + b) \quad (8)$$

where \circ is the element-wise multiplication, $b \in \mathbb{R}$ is a bias term, and f is a non-linear function such as the sigmoid, hyperbolic tangent, etc. By applying this filter to all possible windows, we obtain a feature map $\mathbf{c} = [c_1, c_2, \dots, c_{n-h+1}]$. Note that we can use multiple filters, with possibly different filter widths.

In order to capture the more significant events, we subsequently apply a max pooling operation on the feature map \mathbf{c} . Throughout the paper we will differentiate between two types of max pooling, namely max-over-time pooling and 1-D max pooling.

Max-over-time pooling The first technique extracts a single (maximum) scalar from each feature map. By using multiple convolutional filters, with varying filter widths, we obtain several features which are then passed on to a fully connected layer. This architecture was introduced by [6], and is referred to as concatenation-CNN (C-CNN). Whereas the architecture introduced by [6] uses a final softmax layer, we adapt the network for a multi-label problem by using a sigmoid activation output layer.

1-D max pooling Max-over-time pooling reduces a feature map to a single feature, we can also reduce the feature map to several features, for different windows. In order to determine the maximum value for a window of size m we define

$$p_i = \max(c_{i:i+m-1}) \quad (9)$$

with $i = (1, 1+s, 1+2s, \dots)$, where s denotes the size of the stride. In both the convolutional layer and the 1-D max pooling layer we can vary the stride, meaning that instead of moving the filter one step at the time, we move the filter several places per step. We use multiple filters for the same region, making it possible to learn complementary features from the same regions. With l filters, the generated l feature maps are combined to create a matrix $X \in \mathbb{R}^{l \times \lfloor (n-h-m+2)/s \rfloor}$. These feature maps are then used in combination with an LSTM network, as explained in section 3.6.

3.5 Long Short-Term Memory Network

A Recurrent Neural Network (RNN) has the ability to capture time-dependent relations between words. It does this dynamically, without the use of fixed-size context windows. In particular, the Long Short-Term Memory Network (LSTM) [4] excels at tasks where long term dependencies are important. This network has received a lot of attention because of its capability of capturing important events throughout time series, and being relatively unsusceptible of gaps between important events. Given a sequence as described by equation 3, at time step t the LSTM network updates c_t and h_t with input x_t as follows

$$\begin{bmatrix} i_t \\ f_t \\ o_t \\ \hat{c}_t \end{bmatrix} = \begin{bmatrix} \text{sigm} \\ \text{sigm} \\ \text{sigm} \\ \text{tanh} \end{bmatrix} W \cdot [h_{t-1}, x_t] \quad (10)$$

$$c_t = f_t \circ c_{t-1} + i_t \circ \hat{c}_t \quad (11)$$

$$h_t = o_t \circ \tanh(c_t) \quad (12)$$

where c_t and h_t are the memory and hidden state respectively, i_t , f_t , o_t , and \hat{c}_t are the input gate vector, forget gate vector, output gate vector, and current cell state vector respectively. Note that in equations 10 and 12 the functions sigm and \tanh are applied element-wise. In order to map the output of the LSTM network to the output layer, we apply mean-over-time pooling on the output gate vectors o_t , meaning that we calculate the mean of all h_t values over all time steps t . Finally, the mean-over-time pooling is followed by a fully-connected layer with a sigmoid activation function.

The traditional LSTM network may have problems when the change of the parameters of one layer has an effect on the distribution of the input to all subsequent layers, also known as internal covariance shift. A solution proposed by [2], called Batch Normalized LSTM (BN-LSTM), normalizes both the input-to-hidden and hidden-to-hidden transformations by empirically estimating their means and standard deviations.

3.6 CNN-BN-LSTM

We discussed that CNN leverages the local features of words, whereas LSTM dominates in tasks where long term relations play a part. By combining the two techniques, we hope to get the best of both worlds. We start with applying a CNN layer, followed by a 1-D max pooling layer, as discussed in section 3.4. The resulting matrix is then used as input for the LSTM network, such that there are $\lfloor (n - h - m + 2)/s \rfloor$ time steps, each with dimension l . Similar to the procedure discussed in section 3.5, we subsequently apply mean-over-time pooling on the output gate vectors, together with a fully-connected layer with sigmoid activation. An example of this network is shown in figure 4.

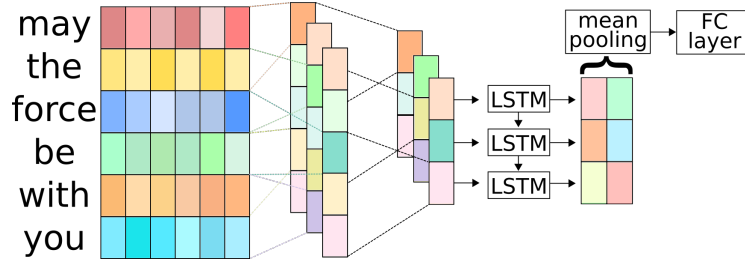


Fig. 4. Graphical representation of the CNN-BN-LSTM network. The hyperparameters of this example are as follows. The word-embedding size is 6. The convolutional layer uses a window size of 2, with a stride of 1, this is followed by 1-D max-pooling with a window size of 3, and a stride of 1.

4 Experimental Setup

4.1 Dataset

The proposed models are tested on the dataset introduced in section 2. We split the dataset into a validation set and a train-test set of respectively 1500 and 14,000 subtitles, so that we tune the hyperparameters on the validation set and use cross validation on the train-test set. We use 7-fold cross validation in order to test the methods, the train set consists each time of 12,000 movies, the test set of 2,000 movies.

4.2 Hyperparameters and Training

The following hyperparameters are all determined by performing a grid search on the validation set. For the MNB model we only take into account words that occur more than 3 times. We use a classification threshold θ of 0.7 for the MNB model. For all other models we use a threshold value of 0.5.

For the BoW-MLP model we use the 50,000 most frequent words. The first hidden layer contains 512 nodes, the second layer 256. In both layers we apply the ReLU activation function, followed by dropout [19] with a dropout rate of 0.5.

Throughout all experiments we use a word embedding size of 300. We use static word embeddings, we thus apply no back propagation on the word embeddings in any of the experiments. We trained the word-embeddings on all subtitles, thus not only on the used subset for the multi-label classification task. The training was performed for 12 epochs, using a learning rate of 0.1, a mini-batch size of 16, a subsample threshold of 10^{-3} , a context size c of 5, and with 15 negative samples.

For the MLP-Histogram model we use 25 bins, followed by 128 hidden nodes in the first layer of the MLP, and 64 nodes in the second layer. Furthermore, in order to prevent overfitting we add Gaussian noise to the input with a mean

of 0, and a standard deviation of 0.02. Additionally, after each layer dropout is applied with a rate of 0.5. Finally, in each layer the ReLU activation function is applied.

The BN-LSTM model uses 300 hidden units, on both the input and output connections we use dropout with a rate of 0.2. We constrain the L_2 -norm of the gradient to not exceed 10, this is known as gradient clipping.

For the CNN-BN-LSTM network we use similar LSTM hyperparameters, proceeded by a CNN. The CNN consists of 200 feature maps, with a window size of 8, a filter stride of 2, followed by a 1-D max pool filter of size 4, with a stride of 2. The activation function used in the CNN is the ReLU. Again we constrain the L_2 -norm of the gradient to a maximum of 10.

In the C-CNN model we use filters of width 3,4 and 5, all with 128 feature maps. We apply dropout with a rate of 0.5, and constrain the L_2 norm again to 10.

For the CNN-BN-LSTM and the C-CNN model we pad the documents to a maximum length of 4000 words. In all models we use a mini-batch size of 20. We train the MLP-BoW and C-CNN for 6 epochs, all other models are trained for 10 epochs. We used the Adadelta update rule [20] for training, while shuffling the mini-batches.

Throughout all experiments (apart from training the word-embeddings) we anneal the learning rate α using exponential decay, defined by $\alpha = \alpha_0 r^{t/k}$, where α_0 is the initial learning rate, r is the decay rate, t is the iteration step, and k indicates the decay step, such that every k steps the learning rate is decayed. In all experiments we use a decay rate r of 0.97. For the MLP-Histogram, BN-LSTM, and CNN-BN-LSTM we used an initial learning rate of 0.1, for the MLP-BoW and C-CNN we use an initial learning rate of 0.005.

5 Results and Discussion

5.1 Metrics

In order to compare our models we will use the F_1 score, which takes into account both the recall and precision. Recall, precision, and the F_1 score for one label are respectively defined as:

$$\text{recall} = \frac{|\{\text{relevant labels}\} \cap \{\text{retrieved labels}\}|}{|\{\text{retrieved labels}\}|} \quad (13)$$

$$\text{precision} = \frac{|\{\text{relevant labels}\} \cap \{\text{retrieved labels}\}|}{|\{\text{relevant labels}\}|} \quad (14)$$

$$F_1 = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}} \quad (15)$$

In order to calculate the final recall, precision, and F_1 -score of the models, we calculate the mean scores over all three genres.

5.2 Results

The results of our models are listed in Table 2. Our baseline method (MNB) does not perform well. The model that performs best is the MLP-BoW model, with an average F_1 -score of 0.77 ± 0.02 . This is a significant higher result ($P < 0.005$) compared to the other models. The novel MLP-Histogram model achieves the second highest F_1 -score. The BN-LSTM does not perform well on it own, however, in combination with a CNN layer (CNN-BN-LSTM) the model obtains the third best results. Finally, the C-CNN model is outperformed by all but two models.

Model	<i>Romance</i>			<i>Thriller</i>		
	Recall	Precision	F_1 -score	Recall	Precision	F_1 -score
MNB	0.93 ± 0.08	0.49 ± 0.14	0.64 ± 0.10	0.02 ± 0.01	0.67 ± 0.14	0.04 ± 0.03
MLP-BoW	0.75 ± 0.13	0.77 ± 0.04	0.76 ± 0.09	0.72 ± 0.01	0.77 ± 0.01	0.74 ± 0.08
MLP-Histogram	0.61 ± 0.03	0.72 ± 0.04	0.66 ± 0.02	0.78 ± 0.04	0.77 ± 0.02	0.77 ± 0.02
BN-LSTM	0.17 ± 0.13	0.72 ± 0.10	0.25 ± 0.14	0.76 ± 0.08	0.74 ± 0.05	0.75 ± 0.02
C-CNN	0.52 ± 0.17	0.69 ± 0.09	0.56 ± 0.10	0.77 ± 0.05	0.76 ± 0.09	0.76 ± 0.05
CNN-BN-LSTM	0.54 ± 0.05	0.74 ± 0.03	0.62 ± 0.04	0.76 ± 0.04	0.79 ± 0.02	0.77 ± 0.03

Model	<i>Action</i>			Mean		
	Recall	Precision	F_1 -score	Recall	Precision	F_1 -score
MNB	0.83 ± 0.11	0.71 ± 0.15	0.73 ± 0.08	0.59 ± 0.02	0.62 ± 0.05	0.47 ± 0.04
MLP-BoW	0.81 ± 0.06	0.82 ± 0.06	0.81 ± 0.05	0.76 ± 0.03	0.79 ± 0.03	0.77 ± 0.02
MLP-Histogram	0.80 ± 0.03	0.79 ± 0.06	0.79 ± 0.02	0.73 ± 0.01	0.76 ± 0.01	0.74 ± 0.01
BN-LSTM	0.75 ± 0.05	0.80 ± 0.06	0.77 ± 0.01	0.56 ± 0.04	0.75 ± 0.04	0.59 ± 0.04
C-CNN	0.75 ± 0.07	0.80 ± 0.08	0.77 ± 0.03	0.68 ± 0.06	0.75 ± 0.02	0.70 ± 0.04
CNN-BN-LSTM	0.78 ± 0.01	0.83 ± 0.05	0.80 ± 0.03	0.69 ± 0.03	0.78 ± 0.03	0.73 ± 0.03

Table 2. The results on the test set, after the specified number of epochs. Both the mean and standard deviation of the cross validation are displayed. We denote the recall, precision, and F_1 -score for the three genres, together with the mean of the recall, precision, and F_1 -score of the three genres.

5.3 Discussion

Our baseline model (MNB) does not perform well, the genre thriller has a very low recall and therefore a low F_1 -score. The other two genres have however a very high recall (higher than all other models). We expect that the poor results on the genre thriller are caused by a combination of how the threshold is determined and the poor assumptions of the MNB model. We also experimented with n -grams, with n ranging from 1 to 3, but the performance decreased for n higher than 1.

The MLP-BoW model outperformed all other (more complex) models. This was in contrast with our expectations, because the model is relatively simple compared to the other machine learning models. Not only is the F_1 -score high, the training time was also relatively short. The fact that this model achieves the highest F_1 -score could suggest that there exist some combinations of important 'indicator words' that are strong predictors for certain genres. We experimented with adding more layers to the network, but this had no significant positive effect on the results. Removing one layer had a negative effect on the accuracy.

Considering that the MLP-Histogram model only takes into account the relative frequency of word embedding feature values the model performs remarkably well. This is another illustration of the 'universal' features of word embeddings. Similar to the MLP-BoW model, the training time is relatively short. Furthermore, this newly proposed model performed best with using the word-embeddings.

The BN-LSTM model performs rather poorly. We expect that this is due to the length of the documents. A careful observation of Figure 2 shows that the genre romance has relatively long subtitles. This could explain the poor results on this genre for models that are susceptible for document length. Although the BN-LSTM network does suffer less from vanishing gradients compared to other RNN networks, the network still has problems with documents of substantial length. Another explanation for the inadequacy of the BN-LSTM model could be that for this task word order is irrelevant and only the occurrence of certain words is important. Preliminary experiments have shown that stacking multiple BN-LSTM layers on top of each other had no effect on the final accuracy. The accuracy increases drastically with the use of batch normalization. Adding batch normalization also causes faster, more stable convergence. Furthermore, the model often diverged without the use of gradient clipping.

Contrary to the MNB model, we saw that for the C-CNN model the use of n -grams (by means of the filter widths) did increase the performance. Although the similar model introduced in [6] achieves state-of-the-art results on various tasks with a similar model, we find only moderate results on our task. One main difference is that the documents in the datasets used in [6] are significantly shorter compared to our dataset, making them less susceptible for outliers that can affect the max-over-time pooling.

By combining a CNN model with a BN-LSTM model (into the CNN-BN-LSTM model) we see a performance improvement compared to a separate C-CNN or BN-LSTM model. By combining the two methods we get the powerful feature extractor of the CNN model, and the capability of detecting long term dependencies of the LSTM model. The downside of this method is that even more hyperparameters have to be tuned. Exploratory research indicated that adding a CNN layer after the BN-LSTM or CNN-BN-LSTM model did not improve the accuracy.

6 Conclusion and Future Work

In this paper we described various techniques that can be used for multi-label classification of movie genres based on subtitles. First, we established a baseline using a multinomial naive Bayes (MNB) classifier combined with several heuristics that "tackle the poor assumptions of MNB" [17]. We trained word embeddings on an unsupervised task, and showed that these embeddings contain indicative features for genre classification. We developed a novel architecture that combines a histogram of the word embeddings with an MLP. Despite the simple nature of this model it outperforms several more complex models. Both the C-CNN network and the BN-LSTM perform poorly on their own. However, by combining both techniques we observe a drastic increase in performance. The model that performs best is the MLP-BoW model, a surprising result given that many papers consider this network to be a baseline method.

We observed that simple models sometimes outperform more complex, state-of-the-art networks. The best network thus completely depends on the problem at hand. Therefore we would like to stress that exploring simpler text-classification methods is of great importance when a new dataset is studied. This directly relates to the principle of Occam's razor, stating that of all possible hypotheses, the one with the fewest assumptions should be used. When we decide to use a specific technique, we make certain assumptions about the data. A simple technique is less prone to overfit the data compared to a more complex technique, because it makes less assumptions about the data. With more assumptions, it is easier to choose parameters such that they only fit the observed data, and do not generalise well.

In follow-up work we would like to consider non-static word embeddings. In [6] it is shown that for certain tasks the performance improves when either non-static word embeddings, or a combination of both static and non-static word embeddings are used. Moreover, we would like to explore the use of random word embeddings and word embeddings trained by others, e.g. [13]. The final F_1 -scores could be improved by using more advanced threshold techniques, and in future research the number of genres should be extended (up to 27). Finally, experiments on more datasets can be conducted, e.g. the Movie Review Sentiment dataset [10] and the Stanford Sentiment Treebank [18].

References

1. James Clark, Irena Koprinska, and Josiah Poon. A neural network based approach to automated e-mail classification. In *Web Intelligence, 2003. Proceedings. IEEE/WIC International Conference on*, pages 702–705, 2003.
2. Tim Cooijmans, Nicolas Ballas, César Laurent, and Aaron C. Courville. Recurrent batch normalization. *CoRR*, abs/1603.09025, 2016.
3. Zellig Harris. Distributional structure. *Word*, 10(23):146–162, 1954.
4. Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

5. Rie Johnson and Tong Zhang. Effective use of word order for text categorization with convolutional neural networks. *arXiv preprint arXiv:1412.1058*, 2014.
6. Yoon Kim. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882*, 2014.
7. Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.
8. David D Lewis. Naive (bayes) at forty: The independence assumption in information retrieval. In *European conference on machine learning*, pages 4–15. Springer, 1998.
9. Pierre Lison and Jörg Tiedemann. Opensubtitles2016: Extracting large parallel corpora from movie and tv subtitles. In *Proceedings of the 10th International Conference on Language Resources and Evaluation*, 2016.
10. Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150, 2011.
11. Andrew McCallum and Kamal Nigam. A comparison of event models for naive bayes text classification. In *AAAI-98 workshop on learning for text categorization*, volume 752, pages 41–48, 1998.
12. Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *CoRR*, abs/1301.3781, 2013.
13. Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013.
14. Tomas Mikolov, Scott Wen-tau Yih, and Geoffrey Zweig. Linguistic regularities in continuous space word representations. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2013.
15. Jeffrey Pennington, Richard Socher, and Christopher D. Manning. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, 2014.
16. Alec Radford, Rafal Józefowicz, and Ilya Sutskever. Learning to generate reviews and discovering sentiment. *CoRR*, abs/1704.01444, 2017.
17. Jason D. M. Rennie, Lawrence Shih, Jaime Teevan, and David R. Karger. Tackling the poor assumptions of naive bayes text classifiers. In *Proceedings of the Twentieth International Conference on Machine Learning*, pages 616–623, 2003.
18. Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642, 2013.
19. Nitish Srivastava, Geoffrey E Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(1):1929–1958, 2014.
20. Matthew D. Zeiler. ADADELTA: an adaptive learning rate method. *CoRR*, abs/1212.5701, 2012.
21. Xiang Zhang, Junbo Zhao, and Yann LeCun. Character-level convolutional networks for text classification. In *Advances in neural information processing systems*, pages 649–657, 2015.

Learning to Play Donkey Kong Using Neural Networks and Reinforcement Learning

Paul Ozkohen¹, Jelle Visser¹, Martijn van Otterlo², and Marco Wiering¹

¹ University of Groningen, Groningen, The Netherlands,
p.m.ozkohen@student.rug.nl,
j.visser.27@student.rug.nl,
m.a.wiering@rug.nl

² Vrije Universiteit Amsterdam^{**}, Amsterdam, The Netherlands,
m.van.otterlo@vu.nl

Abstract. Neural networks and reinforcement learning have successfully been applied to various games, such as Ms. Pacman and Go. We combine multilayer perceptrons and a class of reinforcement learning algorithms known as actor-critic to learn to play the arcade classic Donkey Kong. Two neural networks are used in this study: the actor and the critic. The actor learns to select the best action given the game state; the critic tries to learn the value of being in a certain state. First, a base game-playing performance is obtained by learning from demonstration, where data is obtained from human players. After this off-line training phase we further improve the base performance using feedback from the critic. The critic gives feedback by comparing the value of the state before and after taking the action. Results show that an agent pre-trained on demonstration data is able to achieve a good baseline performance. Applying actor-critic methods, however, does usually not improve performance, in many cases even decreases it. Possible reasons include the game not fully being Markovian and other issues.

Keywords: machine learning, neural networks, reinforcement learning, actor-critic, games, Donkey Kong, platformer

1 Introduction

Games have been a prime subject of interest for machine learning in the last few decades. Playing games is an activity enjoyed exclusively by humans, which is why studying them in the pursuit of *artificial intelligence (AI)* is very enticing. Building software agents that perform well in an area that requires human-level intelligence would thus be one step closer to creating strong, or: general, AI, which can be considered one of the primary goals of the entire field.

Reinforcement learning (RL) techniques have often been used to achieve success in creating game-playing agents [5, 7]. RL requires the use of certain functions, such as a *policy function* that maps states to actions and a *value function*

^{**} The third author acknowledges support from the Amsterdam academic alliance (AAA) on data science.

that maps states to values. The values of these functions could, for example, be stored in tables. However, most non-trivial environments have a large state space, particularly games where states are continuous. Unfortunately, tables would have to become enormous in order to store all the necessary function information. To solve this problem in RL, *function approximation* can be applied, often using *neural networks*. A famous recent example of this is the ancient board game Go, in which DeepMind’s AI *AlphaGo* was able to beat the world’s best players at their own game [7]. Besides traditional games, it was used to learn to play video games. For example, DeepMind used a combination of convolutional neural networks and *Q*-learning to achieve good gameplay performance at 49 different Atari games, and was able to achieve human-level performance on 29 of them [5]. That study shows how an RL algorithm can be trained purely on the raw pixel images. The upside of that research is that a good game-playing performance can be obtained without handcrafting game-specific features. The Deep *Q*-Network was able to play the different games without any alterations to the architecture of the network or the learning algorithms. However, the downside is that deep convolutional networks require exceptional amounts of computing power and time. Furthermore, one could speculate how well performance of each individual game could be improved by incorporating at least some game-relevant features. Still, it is impressive how the network could be generalized to very different games.

An alternative approach is to use hand-crafted game-specific features. One such game where this was successfully applied is Ms. Pac-Man, where an AI was trained to achieve high win rates using higher-order, game-specific features [3]. This approach shows that good performance can be obtained with a small amount of inputs, therefore severely reducing computation time.

In this paper we present an approach to machine learning in games that is more in line with the second example. We apply RL methods to a video game based on *Donkey Kong*, an old arcade game that was released in 1981 by Nintendo [4]. The game features a big ape called Donkey Kong, who captures princess Pauline and keeps her hostage at the end of each stage. It is up to the hero called Jumpman, nowadays better known as Mario, to climb all the way to the end of the level to rescue this damsel in distress. Besides climbing ladders, the player also has to dodge incoming barrels being thrown by Donkey Kong, which sometimes roll down said ladders.

This game provides an interesting setting for studying RL. Unlike other games, Donkey Kong does not require expert strategies in order to get a decent score and/or get to the end of the level. Instead, *timing* is of the utmost importance for surviving. One careless action can immediately lead Mario to certain death. The game also incorporates unpredictability, since barrels often roll down ladders in a random way. The intriguing part of studying this game is to see whether RL can deal with such an unpredictable and timing-based continuous environment. We specifically focus on the very first level of the Donkey Kong game, as this incorporates all the important elements mentioned above while also making the learning task simpler. Other levels contain significantly different mechanics, such as springs that can launch Mario upwards if he jumps

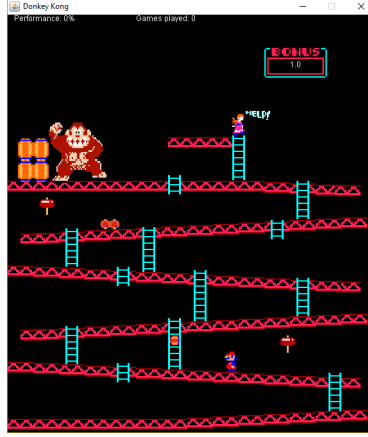


Fig. 1: Recreation of Donkey Kong.

on it, or vertically moving platforms. We do not consider these mechanics in this research.

For this study we used a specific RL technique called *actor-critic* [9]. In each in-game step, the *actor* (player) tries to select the optimal action to take given a game state, while the *critic* tries to estimate the given state’s value. Using these state-value estimates, the critic gives feedback to the actor, which should improve the agent’s performance while playing the game. More specifically, we employ a variant of actor-critic: the *actor-critic learning automaton* (ACLA) [14].

Both the actor and the critic are implemented in the form of a *multilayer perceptron* (MLP). Initializing the online learning with an untrained MLP would be near-impossible: the game environment is too complex and chaotic for random actions to lead to good behavior (and positive rewards). In order to avoid this, both the actor and the critic are trained offline on demonstration data, which is collected from a set of games being played by human players.

The main question this paper seeks to answer is: is a combination of neural networks and actor-critic methods able to achieve good gameplay performance in the game Donkey Kong? In the next sections we will first define the domain and its features, after which we discuss our machine learning setup and methodology and we conclude with results and discussion.

2 The Domain: a Donkey Kong Implementation

A framework was developed that allows the user to test several RL techniques on a game similar to the original Donkey Kong. The game itself can be seen in Fig. 1 and was recreated from scratch as a Java application.

The goal of the game is to let the player reach the princess at the top of the level. The agent starts in the lower-left corner and has to climb ladders in order to ascend to higher platforms. In the top-left corner, we find the game’s

antagonist Donkey Kong, who throws barrels at set intervals. The barrels roll down the platforms and fall down when reaching the end, until they disappear in the lower-left of the screen. When passing a ladder, each barrel has a 50% chance of rolling down the ladder, which adds a degree of unpredictability to the barrel’s course. The player touching a barrel results in an instant loss (and “game over”), while jumping over them nets a small score. Additionally, two power-ups (hammers) can be picked up by the player when he collides with them by either a walking or jumping action, which results in the barrels being destroyed upon contact with the agent, netting a small score gain as well. This powerup is temporary. The agent can execute one out of seven actions: *walking* (left or right), *climbing* (up or down), *jumping* (left or right) or *doing nothing* (standing still). The game takes place in a 680×580 window. Mario moves to the left and right at a speed of 1.5 pixels, while Mario climbs at a speed of 1.8 pixels. A jump carries Mario forward around 10 pixels, which implies it requires many actions to reach the princess from the initial position.

While this implementation of the game is quite close to the original game, there are several differences between the two versions of the game:

- The game speed of the original is slower than in the recreation.
- The barrels are larger in the original. To reduce the difficulty of our game, we made the barrels smaller.
- The original game contains an oil drum in the lower-left corner which can be ignited by a certain type of barrel. Upon ignition, the barrel produces a flame that chases the player. This has been entirely left out in the recreation.
- The original game consists of several different levels. The recreation only consist of one level, which is a copy of the first level from the original.
- The original game uses some algorithm for determining whether a barrel will go down a ladder or not, which appears to be based on the player’s position relative to the barrel and the player’s direction. The code of the original is not available, so instead we opted for a simple algorithm where the barrels’ odds of rolling down a ladder is set to be simply 50% at any given time.

The built environment supports *manual* mode, in which a human player can interact with the game, and two *automated* modes in which an MLP is used to control Mario (either only using an actor network, or learning with a critic). While there are a few notable differences between the original game and our recreation both versions are still quite similar. It is therefore reasonable to assume that any AI behavior in the recreation would translate to the original.

3 Generalization: Multilayer Perceptrons

The actor and critic are implemented in the form of an MLP, a simple feed-forward network consisting of an input layer, one or more hidden layers and an output layer. Like the game itself, the MLP was built from scratch in *Java*, meaning no external packages were used.

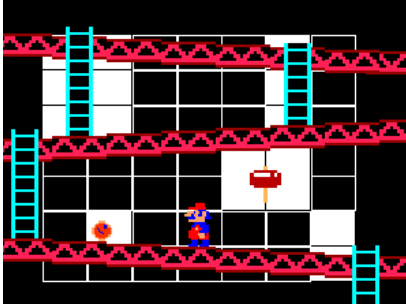


Fig. 2: Visualization of the vision grid that tracks objects directly around the agent, granting Mario local vision of his immediate surroundings. Note that while only one grid can be distinguished, there are actually three vision grids stacked on top of each other, one for each object type.

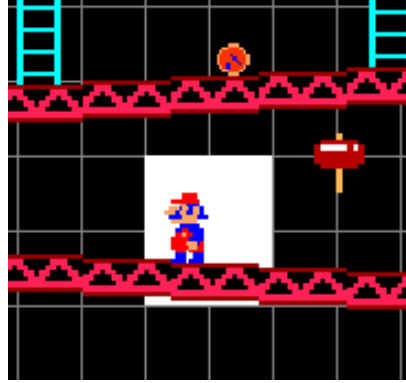


Fig. 3: Visualization of the level-wide grid that tracks the current location of the agent. While not visible in this image, the grid spans the entire game environment.

3.1 Feature Construction for MLP Input

This section provides an overview of how inputs for the MLPs are derived from the game state. Two algorithms employ several varieties of grids that are used to track the location of objects in the game. Each cell in each grid corresponds to one input for the MLP. Besides these grids, several additional inputs provide information about the current state of the game.

There are three types of objects in the game that the agent can interact with: *barrels*, *powerups* and *ladders*. We use three different vision grids that keep track of the presence of these objects in the immediate surroundings of Mario. A similar method was used by Shantia et al. [6] for the game *Starcraft*.

First of all, the MLP needs to know how to avoid getting killed by barrels, meaning it needs to know where these barrels are in relation to Mario. Barrels that are far away pose no immediate threat. This changes when a barrel is on the same platform level as Mario: at this point, Mario needs to find a way to avoid a collision with this barrel. Generally, this means trying to jump over it. Barrels on the platform level above Mario need to be considered as well, as they could either roll down a ladder or fall down the end of the platform level, after which they become an immediate threat to the agent. The second type of objects, ladders, are the only way the agent can climb to a higher platform level, which is required in order to reach the goal. The MLP therefore needs to know if there are any ladders nearby and where they are. Finally, the powerups provide the agent the ability to destroy the barrels, making Mario invincible for a short amount of time. The powerups greatly increase the odds of survival, meaning it's important that the MLP knows where they are relative to Mario.

In order to track these objects, we use a set of three grids of 7×7 cells, where each grid is responsible for tracking one object type. The grids are fixed on Mario, meaning they move in unison. During every time step, each cell detects whether it's colliding with the relevant object. Cells that contain an object are set to 1.0, while those that do not are set to 0.0. This results in a set of $3 \times 49 = 147$ Boolean inputs. The princess is always above the player, while barrels that are below the player pose no threat whatsoever. We are therefore not interested in what happens in the platform levels *below* the agent, since there rarely is a reason to move downwards. Because of this, these *vision grids* are not centered around the agent. Instead, five of the seven rows are above the agent while there is only one row below. An example of the vision grid is shown in Fig. 2.

The MLP requires knowledge of the location of the agent in the environment. This way it can relate outputs (i.e. player actions) to certain locations in the map. Additionally, this knowledge is essential for estimating future rewards by the critic, which will be explained further in section 5. The agent's location in the game is tracked using a 20×20 grid that spans the entire game environment. Like the vision grid, each cell in the agent tracking grid provides one boolean input. The value of a cell is 1.0 if the agent overlaps with it, 0.0 if it does not. This *agent tracking grid* provides $20 \times 20 = 400$ Boolean inputs. An example tracking grid can be seen in Fig. 3.

There are some additional features, such as Booleans that track whether Mario is currently jumping or climbing. The total amount of features is the sum of 147 vision grid cells, 400 agent tracking grid cells and 4 additional inputs, resulting in 551 in total. The four additional inputs are extracted from the game state as follows:

- A boolean that tracks whether the agent can climb (i.e. is standing close enough to a ladder). This prevents the agent from trying to climb while this is not possible.
- A boolean that tracks whether the agent is currently climbing. This prevents the agent from trying to do any other action besides climbing while on a ladder.
- A boolean that tracks whether the agent currently has an activated powerup. This is used to teach the MLP that it can destroy barrels while under the influence of a powerup, as opposed to having to jump over them.
- A real decimal number in the range $[0,1]$ that tracks how much time a powerup has been active. We compute it as the ratio $\frac{t}{d}$ between the time passed since the powerup was obtained (t) and the total time a powerup remains active (d).

3.2 MLP output

For the **actor** the output layer consists of seven neurons, each neuron representing one of the seven possible player actions: moving left or right, jumping left or right, climbing up or down, or standing still. During training using demonstration data, the target pattern is encoded as a *one-hot vector*: the target for

the output neuron corresponding to the action taken has a value of 1.0, while all other targets are set to 0.0. During gameplay, the MLP picks an action based on *softmax* action selection [9]. Here, each action is given a probability based on its activation. Using a *Boltzmann distribution*, we can transform a vector a of length n , consisting of real output activation values, into a vector $\sigma(a)$ consisting of n real values in the range $[0, 1]$. The probability for a single output neuron (action) i is calculated as follows:

$$\sigma(a_i) = \frac{e^{a_i/\tau}}{\sum_{j=1}^n e^{a_j/\tau}} \quad \text{for } i = 1, \dots, n \quad (1)$$

where τ is a positive real temperature value which can be used to induce exploration into action selection. For $\tau \rightarrow \infty$, all actions will be assigned an equal probability, while for $\tau \rightarrow 0$ the action selection becomes purely *greedy*. During each in-game timestep, each output neuron in the actor-MLP is assigned a value using Eq. 1. This value stands for the probability that the actor will choose a certain action during this timestep. The output layer of the **critic** consists of one numerical output, which is a *value estimation* of a given game state. This will be explained further in section 5.2.

3.3 Activation Functions

Two different activation functions were used for the hidden layers: the sigmoid function and the Rectified Linear Unit (ReLU) function. Given an activation a , the sigmoid output value $\sigma(a)$ of a neuron is:

$$\sigma(a) = 1/(1 + e^{-a}) \quad (2)$$

The ReLU output value is calculated using:

$$\sigma(a) = \max(0, a) \quad (3)$$

Both activation functions are compared in order to achieve the best performance for the MLP. This will be elaborated upon in section 6.

4 Learning From Demonstration

RL alone is sometimes not enough to learn to play a complex game. Hypothetically, we could leave out offline learning and initialize both the actor and the critic with an untrained MLP, which the critic would have to improve. In a game like Donkey Kong however, this would lead to initial behavior to consist of randomly moving around without getting even remotely close to the goal. In other words: it would be hard to near-impossible for the actor to reach the goal state, which is necessary for the critic to improve gameplay behavior. This means that we need to *pre-train* both the actor and the critic in order to obtain a reasonable starting performance. For this, we utilized *learning from demonstration*

(LFD) [1]. A dataset of input and output patterns for the MLP was created by letting the first two authors play 50 games each. For each timestep, an input pattern is extracted from the game state as explained before. Additionally, the action chosen by the player at that exact timestep (and the observed reward) is stored. The critic uses the reward to compute a target value as is explained later. All these corresponding input-output patterns make up the data on which the MLPs are pre-trained.

5 Reinforcement Learning Framework

Our game is modeled as a *Markov Decision Process* (MDP), which is the framework that is used in most RL problems [9, 13]. An MDP is a tuple $\langle S, A, P, R, \gamma \rangle$, where S is the set of all states, A is the set of all actions, $P(s_{t+1}|s_t, a)$ represents the transition probabilities of moving from state s_t to state s_{t+1} after executing action a and $R(s_t, a, s_{t+1})$ represents the reward for this transition. The discount factor γ indicates the importance of future rewards. Since in Donkey Kong there is only one main way of winning the game, which is saving the princess, the future reward of reaching her should be a very significant contributor to the value of a state. Furthermore, as explained in Section 2, the agent does not move very far after each action selection. When contrasted with the size of the game screen, this means that around 2000 steps are needed to reach the goal, where 7 actions are possible at each step, leading to a very challenging environment. For these reasons, the discount factor γ is set to 0.999, in order to cope with this long horizon, such that values of states that are, for example, a 1000 steps away from the goal still get a portion of the future reward of saving the princess. A *value function* $V(s_t)$ is defined, which maps a state to the expected value of this state, indicating the usefulness of being in that state. Besides the value function, we also define a *policy function* $\pi(s_t)$ that maps a state to an action. The goal of the RL is to find an *optimal* policy $\pi^*(s_t)$ such that an action is chosen in each state in order to maximize the obtained rewards in the future. The environment is assumed to satisfy the *Markov property*, which assumes that the history of states is not important to determine the probabilities of state transitions. Therefore, the transition to a state s_{t+1} depends only on the current state s_t and action a_t and not on any of the previous states encountered.

In our Donkey Kong framework, the decision-making agent is represented by Mario, who can choose in each state one of the seven actions to move to a new state, where the state is uniquely defined by the combination of features explained earlier. Like in the work done by Bom et al. [3], we use a fixed reward function based on specific in-game events. Choosing actions in certain states can trigger these events, leading to positive or negative rewards. We want the agent to improve its game-playing performance by altering its policy. Rewards give an indication of whether a specific action in a specific state led to a good or a bad outcome. In Donkey Kong, the ultimate goal is to rescue the princess at the top of the level. Therefore, the highest positive reward of 200 is given in this situation. One of the challenging aspects of the game is the set of moving

Event	Reward
Save princess	+200
Jumping over a barrel	+3
Destroy barrel with powerup	+2
Pick up powerup	+1
Getting hit by barrel	-10
Needless jump	-20

Table 1: Game events and their corresponding rewards. A 'needless' jump penalty is only given if the agent jumped, but did not jump over a barrel nor did the agent pick up a powerup.

barrels that roll around the level. Touching one of these barrels will immediately kill Mario and reset the level, so this behavior should be avoided at all costs. Therefore, a negative reward of -10 is given, regardless of the action chosen by Mario. Jumping around needlessly should be punished as well, since this can lead Mario into a dangerous state more easily. For example, jumping in the direction of an incoming barrel can cause Mario to land right in front of it, with no means of escape left. The entire set of events and the corresponding rewards are summarized in Table 1.

5.1 Temporal Difference Learning

Our RL algorithms are a form of *temporal difference* (TD) learning [8, 9]. The advantage of TD methods is that they can immediately learn from the raw experiences of the environment as they come in and no model of the environment needs to be learned. This means that we can neglect the P -part of the MDP tuple explained earlier. TD methods allow learning updates to be made at every time step, unlike other methods that require the end of an episode to be reached before any updates can be made (such as *Monte Carlo* algorithms). Central to TD methods is the value function, which estimates the value of each state based on future rewards that can be obtained, starting at this state. Therefore, the value of a state s_t is the expected total sum of discounted future rewards starting from state s_t :

$$V(s_t) = E \left[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \right] \quad (4)$$

Here, s_t is the state at time t , γ is the discount factor and R_{t+k+1} is the reward at time $t+k+1$. We can take the immediate reward observed in the next state out of the sum, together with its discount factor:

$$V(s_t) = E \left[R_{t+1} + \gamma \sum_{k=0}^{\infty} \gamma^k R_{t+k+2} \right] \quad (5)$$

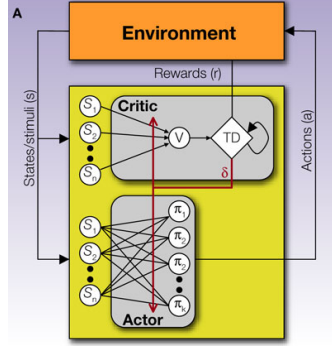


Fig. 4: The architecture of Actor-Critic methods [10].

We observe that the discounted sum in Eq. 5 is equal to the definition of the value function $V(s_t)$ in Eq. 4, except one time step later into the future. Substituting Eq. 4 into Eq. 5 gives us the final value function prediction target:

$$V(s_t) = E[R_{t+1} + \gamma V(s_{t+1})] \quad (6)$$

Therefore, the predicted value of a state is the reward observed in the next state plus the discounted next state value.

5.2 Actor-Critic methods

Actor-critic methods are based on the TD learning idea. However, these algorithms represent both the policy and the value function separately, both with their own weights in a neural network or probabilities/values in a table. The policy structure is called the actor, which takes actions in states. The value structure is called the critic, which criticizes the current policy being followed by the actor. The structure of the actor-critic model is illustrated in Fig. 4.

The environment presents the representation of the current state s_t to both the actor and the critic. The actor uses this input to compute the action to execute, according to its current policy. The actor then selects the action, causing the agent to transition to a new state s_{t+1} . The environment now gives a reward based on this transition to the critic. The critic observes this new state and computes its estimate for this new state. Based on the reward and the current value function estimation, both R_{t+1} and $\gamma V(s_{t+1})$ are now available to be incorporated into both making an update to the critic itself, as well as computing a form of feedback for the actor. The critic looks at the difference of the values of both state s_t and s_{t+1} . Together with the reward, we can define the feedback δ_t at time t , called the TD error, as follows:

$$\delta_t = R_{t+1} + \gamma V(s_{t+1}) - V(s_t) \quad (7)$$

When a terminal state is encountered (hit by a barrel or saving the princess) the value of the next state, $\gamma V(s_{t+1})$, is set to 0. The tendency to select an action has to change, based on the following update rule [9]:

$$h(a_t|s_t) = h(a_t|s_t) + \beta\delta_t, \quad (8)$$

where $h(a_t|s_t)$ represents the tendency or probability of selecting action a_t at state s_t and β is a positive step-size parameter between 0 and 1.

In the case of neural networks, both the actor and the critic are represented by their own multilayer perceptron. The feedback computed by the critic is given to the actor network, where the weights of the output node of the actor corresponding to the chosen action are directly acted upon. The critic is also updated by δ_t . Since the critic approximates the value function $V(s_t)$ itself, the following equation (where the updated V' is computed from V):

$$\begin{aligned} V'(s_t) &= V(s_t) + \delta_t, \\ &= V(s_t) + R_{t+1} + \gamma V(s_{t+1}) - V(s_t) \end{aligned}$$

is reduced to:

$$V'(s_t) = R_{t+1} + \gamma V(s_{t+1}), \quad (10)$$

which is, once again, the value function target for the critic. We can see that as the critic keeps updating and improves its approximation of the value function, $\delta_t = V'(s_t) - V(s_t)$ should converge to 0, which decreases the impact on the actor likewise, which can converge to a (hopefully optimal) policy.

We employ the actor-critic algorithm called *actor-critic learning automaton* [14]. This algorithm functions in the same basic way as standard actor-critic methods, except in the way the TD error is used for feedback. As explained before, standard actor-critic methods calculate the feedback δ_t and use this to alter the tendency to select certain actions by changing the parameters of the actor. ACLA does not use the exact value of δ_t , but only looks at whether or not an action selected in the previous state was good or bad. Therefore, instead of the value, the sign of δ_t is used, and a one-hot vector is used as the target.

6 Experiments and Results

In our experiments we define the performance as the percentage of games where the agent was able to reach the princess: $\frac{gamesWon}{gamesPlayed}$. In the first experiment, the parameters for the MLP trained using learning from demonstration were optimized in order to achieve a good baseline performance. We then perform 10 runs of 100 games to see how the optimized actor performs without any RL. For the second experiment, we compare the performance of only the actor versus an actor trained with ACLA for 5 different models. Between each model, the performance of the Actor-MLP is varied: we do not only want to see if ACLA is able to improve our best actor, but we want to know whether it can increase the performance of lower-performing actors as well.

Model	N hidden layers	N hidden nodes	learning rate	Activation function	Performance
1	2	200	0.01	sigmoid	56.6% (SE: 1.08)
2	1	50	0.001	sigmoid	29.9% (SE: 1.08)
3	1	100	0.005	ReLU	48.6% (SE: 2.02)
4	2	50	0.001	ReLU	50.6% (SE: 1.46)
5	1	80	0.01	sigmoid	12.6% (SE: 0.90)

Table 2: Details of the 5 models that were used in the RL trials. The performance means the % of trials in which Mario gets to the Princess in 100 games. The results are averaged over 10 simulations with MLPs trained from scratch.

6.1 Model Selection for RL

During the RL experiments, the ACLA algorithm was applied to a few different actor networks. The networks were selected based on their performance on the 10×100 games. For example, the first model we considered is an Actor trained with the combination of the best parameters for the sigmoid activation function, found as a result of a separate parameter optimization phase. We consider two networks using the sigmoid activation functions and two networks using the ReLU activation function. The fifth model differs from the other 4: this model is only pre-trained for 2 epochs. This small amount of pre-training means that the model is quite bad, leaving much room for possible improvement by the critic. Besides model 5, the two sigmoid models were trained until a minimum change in error between epochs of 0.00005 was reached, while the two ReLU models had a minimum change threshold of 0.0007. The reason that the ReLU models' threshold is higher than the Sigmoid models', is that preliminary results have shown that the error did not decrease further after extended amounts of training for MLPs using ReLU. Table 2 displays and details all 5 models that we considered and tried to improve during the RL trials together with their performance and standard error (SE).

6.2 Online Learning Experiments

This section explains how the RL trials were set up. Each of the 5 models is trained during one ACLA session. This learning session lasts 1000 games, where the temperature of the Boltzmann distribution starts at a value of 8. This temperature is reduced every 200 games, such that the last 200 games are run at the lowest temperature of 4. Preliminary results showed that most networks performed best at this temperature. The ACLA algorithm is applied at every step, reinforcing positive actions. The learning rate of the actor is set to 0.0001, so that ACLA can subtly push the actor into the right direction. The critic also uses a learning rate of 0.0001. Such low learning rates are required to update the approximations (that were already trained well on the demonstration data) *cautiously*. Setting the learning rate too high causes the networks to become unstable. In this event, state values can become very negative, especially when the actor encounters a lot of negative rewards.

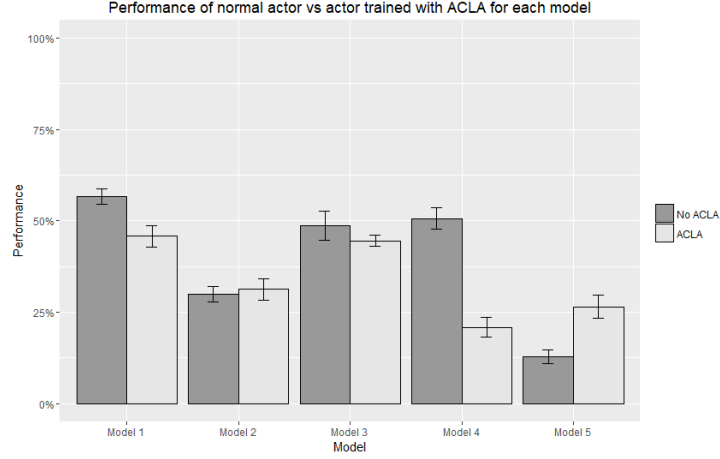


Fig. 5: Performances of the actor trained with vs. without ACLA for each model. The error bars show two standard errors (SE) above and below the mean

After the 1000-games training sessions, the performance of the actors trained with ACLA were compared to the performance of the actors before training with ACLA. For each of the 5 models, both actors were tested in 10×100 games, both with a fixed temperature of 4. The results of the trained actor performances are shown in Table 3. The final results are shown in Fig. 5, where the performance of each model’s actor versus the model’s actor trained with ACLA are shown.

Statistic	Model 1	Model 2	Model 3	Model 4	Model 5
MEAN	45.8%	31.2%	44.5%	20.8%	26.4%
SE	1.45	1.46	0.76	1.34	1.59

Table 3: Results of the models trained with ACLA on 10 runs

6.3 Analysis of Results

Looking at Fig. 5, the differences in performance can be seen for each model, together with standard error bars which have a length of $4 \times SE$. From this figure, we see that the error bars of models 2 and 3 overlap. This might indicate that these differences in performances are not significant. The other 3 models do not have overlapping error bars, suggesting significance. In order to test for significance, we use a nonparametric Wilcoxon rank sum test, since the performance scores are not normally distributed. The Wilcoxon rank sum test confirms a significant effect of ACLA on models 1 ($W = 41.5$, $p < 0.05$), 4 ($W = 100$, $p < 0.05$) and 5 ($W = 0$, $p < 0.05$), but not on models 2 ($W = 41.5$, $p > 0.05$)

and 3 ($W = 27$, $p > 0.05$). These results seem to confirm the observations made earlier with respect to the error bars in Fig. 5.

6.4 Discussion

Using parameter optimization, we were able to find an MLP that is able to obtain a reasonable baseline performance by using learning from demonstration. The best model, model 1, was able to achieve an average performance of winning the game-level of 56.6%. In addition to this, several MLPs were trained with different parameter settings, resulting in a total of 5 neural net models. The performance of these 5 models varies based on how robustly the actor-critic method is able to improve these models.

While the performance achieved by an actor that is only trained offline is reasonable, ACLA does not usually seem to be able to improve this any further. Even worse, the actor’s performance can start to decline over time. Only a model that is barely pre-trained on demonstration data can obtain a significant improvement. We therefore conclude that a combination of neural nets and actor-critic is in most cases not able to improve on a reasonable policy that was obtained through learning from demonstration.

7 Conclusions

We have presented our Donkey Kong simulation and our machine learning experiments to learn good gameplay. We have employed LfD to obtain reasonable policies from human demonstrations, and experimented with RL (actor-critic) to learn the game in an online fashion. Our results indicate that the first setting is more stable, but that the second setting has possibly still potential to improve automated gameplay. Overall, for games such as ours, it seems that LfD can go a long way if the game can be learned from relevant state-action examples. It may well be that for Donkey Kong, in our specific level, the right actions are clear from the current game state and additional delayed reward aspects play a less influencing role, explaining the lesser effect of RL in our experiments. More research is needed to find out the relative benefits of LfD and RL. Furthermore, in our experiments we have focused on the global performance measure of percentage of games won. Further research could focus on more finegrained performance measures using the (average) reward, and experiment with balancing the various (shaping) rewards obtained for game events (see Table 1).

Future research could result in better playing performance than those obtained in this research. Actor-critic methods turned out to not be able to improve the performance of the agent. Therefore, other reinforcement learning algorithms and techniques could be explored, such as Q-learning [12], advantage learning [2] or Monte Carlo methods. A recent method has been introduced called the Hybrid Reward Architecture, which has been applied to Ms. Pac-Man to achieve a very good performance [11]. Applying this method to Donkey Kong could yield better results. Additionally, it would be interesting to see whether having more

demonstration data positively affects performance. Since we only focused on the very first level of the game, further research is needed to make the playing agent apply its learned behavior to different levels and different mechanics.

Bibliography

- [1] Atkeson, C.G., Schaal, S.: Robot learning from demonstration. In: Proceedings of the International Conference on Machine Learning. pp. 12–20 (1997)
- [2] Baird, L.: Residual algorithms: Reinforcement learning with function approximation. In: Proceedings of the Twelfth International Conference on Machine Learning. pp. 30–37 (1995)
- [3] Bom, L., Henken, R., Wiering, M.: Reinforcement learning to train Ms. Pac-Man using higher-order action-relative inputs. In: IEEE Symposium on Adaptive Dynamic Programming and Reinforcement Learning (2013)
- [4] [http://donkeykong.wikia.com/wiki/Nintendo: Donkey Kong fansite wiki](http://donkeykong.wikia.com/wiki/Nintendo:Donkey_Kong_fansite_wiki), accessed Sept. 2017
- [5] Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., Riedmiller, M.: Human-level control through deep reinforcement learning. *Nature* 518, 529–533 (2015)
- [6] Shantia, A., Begue, E., Wiering, M.: Connectionist reinforcement learning for intelligent unit micro management in Starcraft. In: The 2011 International Joint Conference on Neural Networks. pp. 1794–1801 (2011)
- [7] Silver, D., Huang, A., Maddison, C.J., Guez, A., Sifre, L., van den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., Dieleman, S., Grewe, D., Nham, J., Kalchbrenner, N., Sutskever, I., Lillicrap, T., Leach, M., Kavukcuoglu, K., Graepel, T., Hassabis, D.: Mastering the game of Go with deep neural networks and tree search. *Nature* 529(7587), 484–489 (Jan 2016)
- [8] Sutton, R.S.: Learning to predict by the methods of temporal differences. *Machine Learning* 3(1), 9–44 (1988)
- [9] Sutton, R.S., Barto, A.G.: Reinforcement Learning: An Introduction. Cambridge: The MIT Press (1998)
- [10] Takahashi, Y., Schoenbaum, G., Niv, Y.: Silencing the critics: understanding the effects of cocaine sensitization on dorsolateral and ventral striatum in the context of an actor/critic model. *Frontiers in Neuroscience* 2(1), 86–99 (2008)
- [11] van Seijen, H., Fatemi, M., Romoff, J., Laroche, R., Barnes, T., Tsang, J.: Hybrid reward architecture for reinforcement learning (2017), retrieved from <https://arxiv.org/abs/1706.04208>
- [12] Watkins, C.J.: Learning from delayed rewards (1989), PhD Thesis, University of Cambridge, England
- [13] Wiering, M., van Otterlo, M.: Reinforcement Learning: State of the Art. Springer (2012)
- [14] Wiering, M.A., Van Hasselt, H.: Two novel on-policy reinforcement learning algorithms based on TD(λ)-methods. In: 2007 IEEE International Symposium on Approximate Dynamic Programming and Reinforcement Learning. pp. 280–287 (2007)

Part II

Type A: Regular papers
Poster presentation

A Proposal to Solve Rule Conflicts in the Wang-Mendel Algorithm for Fuzzy Classification Using Evidential Theory

Diego Alvarez-Estevez¹[0000-0001-5790-0577] and Vicente Moret-Bonillo²[0000-0002-9435-3151]

¹ Haaglanden Medisch Centrum, The Hague, The Netherlands
diego.alvarez@udc.es

² University of A Coruña, A Coruña, Spain
vicente.moret@udc.es

Abstract. This paper addresses the problem of solving rule conflicts in a modified version of the Wang-Mendel algorithm for the induction of fuzzy classification rules. At this respect we propose a solution based on the reinterpretation of the conflict resolution mechanism as an evidence reassignment problem. The Evidential-theory framework developed by Dempster and Shafer is used for this purpose, and different alternative conflict handling strategies are explored. Experiments are carried out using a benchmark of well-known classification problems. Our preliminary results are encouraging toward supporting the usefulness of the proposed approach.

Keywords: Learning from examples, Wang-Mendel algorithm, Fuzzy classification, Evidential Theory.

1 Introduction

Several methods are available in the literature for the induction of fuzzy rules in the context of supervised learning, also often referred as learning from examples. Examples can be found in fields such as function approximation [1-3], clustering [4-5], and of course, classification [6-9]. In this paper we focus on the domain of solving classification problems. As such, given a set of N input patterns of the form $\vec{x}_i = (x_{i1}, x_{i2}, \dots, x_{im})$, $i = 1..N$ and a set of output classes $C = \{C_1, C_2, \dots, C_K\}$, then the classification problem consists in finding the appropriate mapping f such that $f(\vec{x}_i)$ returns the class $C_i \in C$ to which the pattern \vec{x}_i belongs, $\forall i = 1..N$. The goal is therefore to find the best possible approximation $\widehat{f}(\vec{x})$ of $f(\vec{x})$. Hence, ideally one would like that $f(\vec{x}) = \widehat{f}(\vec{x})$. Notice that we are implicitly assuming that each input pattern has one dominant class membership in C , that is, it should be always possible to find j such that $\mu_j(\vec{x}) > \mu_i(\vec{x}), \forall i \neq j, i = 1..K$. We are, on purpose, not considering the case in which the same pattern could be assigned to two or more classes with exactly the same degree of membership. Further, in the context of this work, we will interpret this situation as a *conflict*.

In particular in this paper we analyze the case of the Wang-Mendel (WM) algorithm for the induction of fuzzy IF-THEN rules [10]. Specifically we focus on the version adapted by Chi et al. [11] for pattern classification (hereafter referred as the WM/Chi method). As it will be described later on, the WM/Chi method for the generation of fuzzy classification rules includes, as part of the algorithm, a rule conflict resolution step by which, among the generated rules that share the same antecedent while pointing to different classes, one winner rule has to be chosen to solve the conflict, discarding the rest. The original algorithm (both on its regression and classification forms) proposes a method to choose the winner rule based on the calculation of ad-hoc scorings over the involved conflicting rules. Different methods to obtain these scorings have been proposed, for example, in Chi et al. [11] and in Cordón et al. [12]. In this work, we contribute by exploring an alternative approach by reinterpreting the conflict resolution mechanism as an evidence reassignment problem, and for that purpose we explore the use of the Evidential Theory (ET) framework developed in the past by Dempster and Shafer [13]. The idea is explained throughout the subsequent sections, and experiments are carried out to test this approach by using several well-known machine learning classification benchmarks.

2 Methods

2.1 Overview of the WM Algorithm

The WM algorithm for the induction of fuzzy rules was one of the first approaches to design fuzzy rule based systems by learning from examples [10]. The interest in this method resides in its simplicity and its straightforward approach, which at the same time has proven capabilities to provide reasonably good performance. These properties have made of this algorithm one of reference benchmark methods in the field [9, 14-17]. Originally the WM algorithm was developed to be used in Mamdani-type fuzzy reasoning systems on the context of regression and function approximation [10]. A first extension of this algorithm for its use in classification problems was proposed by Chi et al. [11]. Later on, Cordón et al. used the Chi's et al. approach to investigate the consequences of using different rules types, including the possibility to assign weights to the generated rules. They also explored the use of different aggregation strategies [12]. In Wang [14] the WM method is reviewed presenting it as a flexible fuzzy system approach to data mining. In this review, the original method for function approximation is extended, but also classification problems are considered using a piecewise constant fuzzy predictive model. In this work we will focus on the Chi's et al. [11] formulation by considering the mapping function of the system as a direct application from \mathbb{R}^m to C . This approximation has the advantage of avoiding the necessity to define fuzzy partitions over the output space. In our view, this qualifies for a simpler and more natural classification approach.

2.2 The WM/Chi Method for Classification

Let us assume that a set of N input-output patterns of the form

$$(\vec{x}_i, C_i), \vec{x}_i \in R^m, C_i \in C, i = 1, 2, \dots, N \quad (1)$$

are available, which are split into two disjoint subsets. We refer to them as the training (TR) and the testing (TS) sets, respectively, containing N_{TR} and N_{TS} input-output patterns each, therefore $N = N_{TR} + N_{TS}$. Our learning algorithm is allowed to use only the information contained in TR in order to construct $\hat{f}(\vec{x})$, i.e. to infer a set of R rules that would integrate the so-called Knowledge Base (KB) of the system.

In the following we describe the WM/Chi method for classification problems (for detailed information see [11]). The algorithm is composed of three steps:

-Step 1: Divide the Input Space into Fuzzy Regions

The original method does not impose any specific partition for the input variables, but it proposes an equally spaced partition based on triangular fuzzy sets [10]. In this kind of partition each domain interval $k, k=1 \dots m$, is divided into $Q_k = 2n + 1$ regions, n being a natural number, where the center of each corresponding membership function (MF) lies in the center of the region, and the extrema lie at the center of the neighboring regions. Several other methods for fuzzy partition design can be used as well which can be found in the literature [4, 14].

-Step 2: Generate Fuzzy Rules from the Given Data Pairs

For each input-output pair $(\vec{x}_p, C_p) \in TR, \vec{x}_p \in R^m, p = 1 \dots N_{TR}$, a rule R_p is generated by computing the membership values $\mu_{jk}(x_{pk})$ for $k=1 \dots m, j=1 \dots Q_k$. For each k then select the fuzzy set $A_{j_{max}k}$ in which the maximum membership takes place, $j_{max} = \{j \mid \mu_{j_{max}k}(x_{pk}) \geq \mu_{jk}(x_{pk}) \text{ for all } j=1 \dots Q_k\}$.

The following rule is then generated:

$$\text{Rule } R_p: \text{IF } x_1 \text{ is } A_{j_{max}1} \wedge \dots \wedge x_m \text{ is } A_{j_{max}m}, \text{ THEN } C_p \quad (2)$$

The process is repeated for all the training patterns and an initial set of N_{TR} “and” rules is therefore obtained as the output.

-Step 3: Remove conflicting rules

Each training pattern generates a rule, and therefore there might be rules that “conflict” with each other as result. The definition of “conflict” is of fundamental importance here. In the original paper for function approximation a conflict is considered when two or more rules share the same antecedent part while having different consequents [10]. The natural extension for classification followed by Chi et al. on their adaptation for classification [11] is to consider a conflict when two or more rules have the same antecedent and output to different classes. Indeed, one should notice that allowing rules to share the same antecedent while pointing to different classes is likely (depending on the aggregation strategy) to cause the conflicting situation of the same

input being assigned to different classes with the same activation strength. This is a problem when one has to unambiguously assign a unique final class label. Indeed, this is especially the case, for example, when the fuzzy inference engine does only use the rule with the highest activation to perform the classification (the so-called one-winner aggregation approach, as it was the case in the original WM method [10]).

To avoid this problem a conflict resolution strategy was originally proposed in Wang and Mendel [10] and in Chi et al. [11], based on the scoring of the conflicting rules. Using this mechanism, only the rule with the maximum score is selected to be included in the final KB, discarding the rest. To compute the score, in Chi et al. [11] the following custom product strategy based on the respective associated input activation was proposed:

$$s_p = \prod_{k=1}^m \mu_{j_{max}k}(x_{pk}) \quad (3)$$

This is an adaptation for classification of the original scoring rule proposed in Wang and Mendel [10]. Alternative approaches have been proposed too, for example, a scoring approach based on the relative proportion of patterns supporting each of the conflicting rules was also suggested in Chi et al. [11]. On the other hand, in the unlikely situation of a tie in the scorings, then the conflict resolution mechanism should provide of a complementary selection criterion. This situation however, is not explicitly mentioned in the original method. One simple possibility is just to pick one of the conflicting rules at random, or else use any other alternative (tie breaking) criterion.

As a result of this step, we obtain a subset of R rules, $R \leq N_{TR}$, where conflicting rules have been removed. This set of rules defines the final KB of the fuzzy inference system.

2.3 Conflict Resolution Using Evidential Theory

While the original proposal of choosing the rule with the maximum score seems in principle straightforward, this criterion explicitly discards all the information carried out by the non-selected rules. However, it seems logical to consider that this discarded information might relevantly contribute to the final output. After all, the existence of these (preliminary) rules is a consequence of the information carried out by the (training) data, as described in step 1) of the WM/Chi algorithm. Depending on the amount data available, the number of output classes and the schema of the fuzzy rules (i.e. for example we could allow a simple rule to point to more than one class at the same time, see for example [12]), the complexity of the relationships can rapidly increase, and the amount of discarded information might indeed be relevant and have potential contribution to the final optimal decision. In this scenario the application of a more formal and powerful evidence combinational model seems adequate.

The Evidential Theory (ET) framework proposed by Dempster and Shafer is, at this respect, attractive, among others for which: (i) it easily allows modelling of uncertainty associated to evidences and hypotheses, (ii) it allows considering sets of hypotheses without the need that the confidence placed on a particular hypothesis spreads in any particular mode on each of the remaining ones, (iii) it elegantly reflects the lack of knowledge, so commonly associated to reasoning processes, (iv) it is a formal model

which contains probability theory as a particular case, and also some of the evidence combination functions of other effective (but ad-hoc) models such as the model of certainty factors, developed by Shortliffe and Buchanan, and used in the well-known MYCIN system [18].

Our proposal for the reinterpretation of the conflict resolution problem in the context of the ET framework is rather simple. As such, when two or more rules conflict (according to the above mentioned notion of conflict) then we might just consider that each rule's output class accounts for a possible hypothesis, and that each individual rule's associated score (regardless of the specific method for computing this score) is an independent item of evidence pointing to the consequent hypothesis.

Let us consider the following simple example in the context of a two-class problem, in which we have three conflicting rules R_1 , R_2 , and R_3 , pointing respectively to classes C_1 , C_1 , and C_2 . Let us also assume the rule scoring approach provides the following respective rule scores $s = \{0.7, 0.1, 0.6\}$. According to the above described reinterpretation, and put in terms of the ET framework, then the universe set representing all possible hypotheses of the domain U , results $U = \{H_1, H_2\}$, or equivalently $U = \{C_1, C_2\}$. Consequently the power set $\Omega = 2^U = \{\emptyset, \{C_1\}, \{C_2\}, \{C_1, C_2\}=U\}$, contains all the possible subsets of U , including the empty set \emptyset . The resulting Basic Belief Assignment (BBA) function, $m: \Omega \rightarrow [0,1]$, assigns the following belief masses due to the conflicting rules:

$$R_1: m_1(C_1)=0.70 \Rightarrow m_1(\Omega_1)=1-0.70=0.30$$

$$R_2: m_2(C_1)=0.10 \Rightarrow m_2(\Omega_2)=1-0.10=0.90$$

$$R_3: m_3(C_2)=0.60 \Rightarrow m_3(\Omega_3)=1-0.60=0.40$$

Taking into account that

$$\sum_{A \subseteq U} m(A) = 1 \quad (4)$$

$$m(\emptyset) = 0$$

and applying the Dempster's rule of combination, such that

$$m_{1,2}(Z) = \frac{\sum_{Z=X_i \cap Y_j} m_1(X_i)m_2(Y_j)}{\sum_{X_i \cap Y_j \neq \emptyset} m_1(X_i)m_2(Y_j)} \quad (5)$$

we obtain the following final mass assignments:

$$m(C_1)=0.52$$

$$m(C_2)=0.29$$

$$m(\Omega)=0.19$$

At this point the conflict resolution criterion might be modified to choose one of the conflicting rules, as the one whose class associated mass assignment is the maximum (*criterion A*). Using this criterion notice that the result matches the one of the classical approach in the sense that $\max\{0.7, 0.1, 0.6\} = 0.7 \Rightarrow C_1$. In this case the clas-

sical conflict resolution approach and the proposal using ET agree. However, we may just consider exactly the same example, but adding one additional rule to the conflicting set: $R_4 \rightarrow C_2$ with associated score $s_4 = 0.5$. The interested reader might do the corresponding calculations to obtain in this case that:

$$m(C_1) = 0.35$$

$$m(C_2) = 0.52$$

$$m(\Omega) = 0.13$$

and therefore, in this case the selected class would be C_2 , which differs from the analogous result applying the original WM/Chi criterion: $\max\{0.7, 0.1, 0.6, 0.5\} = 0.7 \Rightarrow C_1$.

Another possible rule conflict resolution criterion (*criterion B*) might be to allow the implementation of all the possible rules in the KB, by assigning a rule's weight in each case, equal to the resulting final mass assignments. Actually, using this alternative conflict resolution criterion, we are not selecting a unique output class for the antecedent to solve the conflict, but allowing several rules to have the same antecedent while pointing to different classes with different strength a priori. That is, our KB will be composed in fact of rules of the form:

$$\text{Rule } R_p: \text{IF } x_1 \text{ is } A_{j_{\max 1}} \wedge \dots \wedge x_m \text{ is } A_{j_{\max m}}, \text{ THEN } C_p \text{ with } w_p = m(C_p) \quad (6)$$

Experiments will be carried out in the next sections to compare both approaches.

3 Experimental Setup

The main objective here is to evaluate the proposed approach as described in Section 2. For this purpose we have chosen to use a set of well-known classification benchmarks selected from the University of Carolina Irvine (UCI) machine learning repository [19]. For each dataset we schedule 10x10-fold cross-validation in which, for each fold, we apply the WM/Chi method to build the KB and evaluate the performance over the corresponding TS set. For comparison purposes, for each iteration, the resolution of conflicting rules in step 3 of the algorithm will be performed applying the classical approach as stated originally by Wang and Mendel [10] and later by Chi et al. [11] (baseline method), and then using the two alternative criteria (*A* and *B*) using the ET framework, as described before in Section 2.3.

Table 1 characterizes the selected datasets. In Table 1 the first column refers to the name of the dataset (for detailed information please check [19]), the second column shows the number of instances that form the dataset, column three indicates the input dimensionality, and the last column shows the number of output classes and their corresponding distribution.

Note that during the cross-validation process all datasets are partitioned using the original class distribution according to column four of Table 1, thus no class distribution normalization is attempted. In our experiments, for all the datasets, the input space is partitioned using three equally spaced triangular fuzzy sets.

Table 1. Summary characterization of the classification benchmarks selected for the tests. In the fourth column the notation “number of classes: distribution” is used

Dataset	Patterns	Attributes	Classes
IRIS	150	4	3: [50, 50, 50]
WINE	178	13	3: [59, 71, 48]
BREAST CANCER	569	32	2: [257, 212]
GLASS	214	9	6: [70, 76, 17, 13, 9, 29]
PIMA	768	8	2: [500, 268]

Two specific realizations of a general fuzzy inference engine are tested: (a) using the classical one-winner reasoning method (*max*) in which only the rule with the highest activation is used to classify the current input pattern; and (b) using a general reasoning model that combines the information provided by all the activated rules, in our case using a normalized addition aggregation operator (*sum*). The objective is to test the validity of the previous analyses of Cordon et al. using WM/Chi approach, who hypothesized that the one-winner approach would tend to misuse the information provided by the places of overlapping fuzzy subsets, usually offering worst results than combinative approaches [12]. In both approaches the product is used to implement the *T-norm* operator that calculates the matching degree between each rule R_j and the current input pattern \vec{x} . The following fuzzy inference engines therefore result on each case:

In the case of *max*:

$$y_k(\vec{x}) = \max_{j=1}^{R_k} \left\{ w_j \prod_{i=1}^m \mu_{ji}(x_i) \right\} \quad (7)$$

In the case of *sum*:

$$y_k(\vec{x}) = \frac{\sum_{j=1}^{R_k} \{w_j \prod_{i=1}^m \mu_{ji}(x_i)\}}{\sum_{j=1}^R \{w_j \prod_{i=1}^m \mu_{ji}(x_i)\}} \quad (8)$$

where the index k denotes the corresponding output class, and m is the input dimensionality. Notice as well the rule weighting term w_j is introduced which accounts for the weight associated to the rule R_j in the case of (6), and otherwise $w_j = 1$ which is equivalent to (2) (i.e. when applying the original WM/Chi’s formulation).

Finally, the score associated to each of the rules i involved in a conflict would be here calculated using (3) according to the original proposal in Wang and Mendel [10] and in Chi et al. [11], i.e.:

$$s_i = \bigwedge_{k=1}^m \mu_{A_k} = \prod_{k=1}^m \mu_{j_{maxk}}(x_{pk}) \quad (9)$$

Notice this has been chosen on the matter of allowing an easier comparison with the original method, however, some other approaches are available, such as the ones proposed in Cordón et al. [12] or in Chi et al. [11] (strategy 2).

4 Results

Experimental results are shown in Tables 2-3. For all tables the first column determines the test dataset, and column two describes the KB generation method. Notation is as follows: *Chi* denotes the original WM/Chi's approach, *Chi EvA* denotes the variation in which ET is used to solve rules conflicts under *criterion A* (see Section 2.3), and *Chi EvB* denotes the variation in which ET is used to solve rules conflicts under *criterion B* (see Section 2.3). Columns three and four show respectively the average classification accuracy and the kappa index achieved for TS. Best results for each dataset according to the criterion of the "maximum Kappa score achieved in TS" are shown in bold. When a method achieves worst performance than the baseline reference (Chi), then it is highlighted in italics.

It can be seen from the results in Tables 2-3 that by using the proposed approach to solve conflicts under the ET framework, and for all the experiments carried out, equal or better performance is achieved under *criterion A*. Results under *criterion B* show a less clear trend, sometimes achieving the best absolute performance, and sometimes underperforming the reference baseline. By comparing the two alternative aggregation approaches (*max*, Table 2, and *sum*, Table 3) the results support here the hypothesis of the use of combination aggregation methods (see [12]), for which best performances for all datasets are achieved when using *sum*.

Table 2. Experiment results over TS using the one-winner aggregation approach. Results are shown as mean \pm std. Best performance on each dataset is highlighted in bold. In italics means the method performs worse than the baseline method (Chi)

Dataset	Method	Accuracy TS	Kappa TS
PIMA	Chi	0.5868 \pm 0.0471	0.1410 \pm 0.0995
	Chi EvA	0.7114 \pm 0.0563	0.3231 \pm 0.0904
	Chi EvB	0.7229 \pm 0.0627	0.3255\pm0.1102
IRIS	Chi	0.9133 \pm 0.0850	0.8588 \pm 0.1346
	Chi EvA	0.9133 \pm 0.0850	0.8588 \pm 0.1346
	<i>Chi EvB</i>	<i>0.8800\pm0.0752</i>	<i>0.8015\pm0.1127</i>
WINE	Chi	0.9234 \pm 0.0409	0.8778 \pm 0.0638
	Chi EvA	0.9251 \pm 0.0406	0.8803 \pm 0.0636
	Chi EvB	0.9423 \pm 0.0403	0.9077\pm0.0643
CANCER	Chi	0.9428 \pm 0.0209	0.8750 \pm 0.0433
	Chi EvA	0.9428 \pm 0.0209	0.8750 \pm 0.0433
	<i>Chi EvB</i>	<i>0.9410\pm0.0381</i>	<i>0.8730\pm0.0810</i>
GLASS	Chi	0.4745 \pm 0.1127	0.2785 \pm 0.1608
	Chi EvA	0.6209 \pm 0.0949	0.4858\pm0.1212
	Chi EvB	0.5887 \pm 0.1211	0.4319 \pm 0.1443

Table 3. Experiment results over TS using the normalized addition (*sum*) aggregation approach. Results are shown as mean \pm std. Best performance on each dataset is highlighted in bold. In italics means the method performs worse than the baseline method (Chi)

Dataset	Method	Accuracy TS	Kappa TS
PIMA	Chi	0.7137 \pm 0.0485	0.3548 \pm 0.1066
	Chi EvA	0.7375 \pm 0.0583	0.3901\pm0.1034
	<i>Chi EvB</i>	0.7274 \pm 0.0690	<i>0.3249\pm0.1192</i>
IRIS	Chi	0.9600 \pm 0.0328	0.9323 \pm 0.0559
	Chi EvA	0.9600 \pm 0.0328	0.9323 \pm 0.0559
	<i>Chi EvB</i>	<i>0.9333\pm0.0424</i>	<i>0.8888\pm0.0706</i>
WINE	Chi	0.9483 \pm 0.0532	0.9205 \pm 0.0803
	Chi EvA	0.9489 \pm 0.0535	0.9213 \pm 0.0807
	Chi EvB	0.9503 \pm 0.0459	0.9222\pm0.0696
CANCER	Chi	0.9535 \pm 0.0357	0.8979 \pm 0.0766
	Chi EvA	0.9535 \pm 0.0357	0.8979 \pm 0.0766
	<i>Chi EvB</i>	<i>0.9451\pm0.0398</i>	<i>0.8810\pm0.0854</i>
GLASS	Chi	0.5696 \pm 0.0994	0.3829 \pm 0.1428
	Chi EvA	0.6146 \pm 0.0781	0.4499 \pm 0.0925
	Chi EvB	0.6348 \pm 0.1314	0.4961\pm0.1562

5 Discussion

In this work we present a method for handling rule conflicts that appear during the application of the WM algorithm for the induction of fuzzy IF-THEN rules. Specifically we have focused on the context of classification problems, following the extension of the WM method described by Chi et al. [11], and proposed an approach by reinterpreting the conflict resolution mechanism as an evidence combination problem. For this purpose the Evidential Theory framework proposed by Dempster and Shafer has been used [13].

Although the results here exposed should still be regarded as preliminary, they are encouraging toward the usefulness of the proposed method. Two possible conflict resolution criteria (here referred as *A* and *B*) have been proposed. The results among our classification benchmark have shown equal or better performance in all the datasets under *criterion A*. Under *criterion B* we were able to achieve the best absolute performance in some datasets, however, for other datasets the method underperformed the reference baseline.

Further testing is indeed needed. Whether it is possible to determine beforehand the exact circumstances over which the approach presented would effectively contribute to achieve better classification performance than the classical WM/Chi approach still needs to be investigated. In general results might not generalize for all situations, and possibly the contribution of the conflict resolution mechanism to the overall performance might become blurred by mightier configuration choices, such as the parameterization of the inference engine, the implementation of the fuzzy logic operators, or the particularities of the input dataset.

We expect as well the dimensionality of the feature space to be a factor of consideration. At this respect, in our analyses –as well as for all the referenced approaches in the literature– equally spaced triangular fuzzy sets were used for the partition of the

input space. Even though this kind of partition is not optimal, literature has shown that it provides reasonably good performance when the number of input fuzzy sets has been chosen correctly. Our choice to use here three equally spaced input fuzzy sets obeys, in one hand, to the motivation of keeping the experimentation procedures rather simple. On the other hand the number of rule conflicts is expected to increase when lowering the dimensionality of the feature input space (more examples will fall under the same fuzzy input region). As we were interested in evaluating the effects of varying the conflict resolution mechanism over the performance, keeping of a low dimensional feature input space seemed as of being a proper choice. Future work to objectively quantify the amount of rule conflict occurring on each case will be valuable to assess the actual impact of each of the conflict resolution approaches.

Further understanding of the full possibilities that Evidential Theory might bring into this context is also needed. We might be, in fact, not yet taking advantage of the full capabilities of this model. Specifically we are currently ignoring the amount of belief that is shared among all the outcomes, i.e. $m(\Omega)$, which cannot be assigned to specific focal elements. It might be useful to consider this as a relevant piece of information to some ending, such as carrying out some sort of output weighting or normalization. In addition, notice that according to the formulation presented in Section 2.3, we are in fact dealing with a simplified version of ET in which, in practice, each rule provides evidence supporting only one of the possible classes/hypothesis of U . In contrast, Evidential Theory could be applied to allow rules to point to several outputs at the same time. In other words, we are currently limiting to a Multiple-Input Single-Output scenario, however benefits might derive when considering Multiple-Input Multiple-Output rules of the form *IF x_1 is $A_1 \wedge \dots \wedge x_m$ is A_m THEN $C_1 \wedge \dots \wedge C_k$* . As Dempster-Shafer theory has as a special case the probability theory, it would be interesting as well to explore the links to probabilistic conflict resolution. In particular a parallelism could be established between rules (6) and the rules of probabilistic fuzzy classifiers [20].

Last but not least, even though in this work we have focused on classification problems, the methodology here proposed can be directly applied as well to regression problems, and therefore to the original WM formulation [10]. Possible benefits of the approach into this context need still to be assessed.

Future work will be carried out in order to provide answers to the aforementioned questions.

References

1. J.-S. Jang, "ANFIS: adaptive-network-based fuzzy inference system," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 23, no. 2, pp. 665-685, 1993.
2. W. Yu and X. Li, "Fuzzy identification using fuzzy neural networks with stable learning algorithms," *IEEE Transactions on Fuzzy Systems*, vol. 12, no. 3, pp. 411-420, 2004.
3. D. Alvarez-Estevéz, J. Principe and V. Moret-Bonillo, "Information theoretic fuzzy modeling for regression," in *IEEE World Congress on Computational Intelligence*, Barcelona, Spain, 2010.
4. S. Chiu, "Fuzzy model identification based on cluster estimation," *Journal of Intelligent &*

- Fuzzy Systems: Applications in Engineering and Technology*, vol. 2, no. 3, pp. 267-278, 1994.
5. J. Bezdek, R. Ehrlich and W. Full, "FCM: The fuzzy c-means clustering algorithm," *Computers & Geosciences*, vol. 10, no. 2-3, pp. 191-203, 1984.
 6. D. Nauck and R. Kruse, "Obtaining interpretable fuzzy classification rules from medical data," *Artificial Intelligence in Medicine*, vol. 16, pp. 149-169, 1999.
 7. S. Pal and S. Mitra, "Multilayer perceptron, fuzzy sets and classification," *IEEE Transactions on Neural Networks*, vol. 3, no. 5, pp. 683-697, 1992.
 8. D. Alvarez-Estevéz, J. Principe and V. Moret-Bonillo, "Neuro-fuzzy classification using the Correntropy criterion: application to sleep depth estimation," in *2010 International Conference on Artificial Intelligence*, Las Vegas, NV, 2010.
 9. J. Alonso and L. Magdalena, "HILK++: an interpretability-guided fuzzy modeling methodology for learning readable and comprehensible fuzzy rule-based classifiers," *Soft Computing*, no. 15, pp. 1959-1980, 2011.
 10. L.-X. Wang and J. Mendel, "Generating fuzzy rules by learning from examples," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 22, no. 6, pp. 1414-1427, 1992.
 11. Z. Chi, H. Yan and T. Pham, *Fuzzy algorithms: with applications to image processing and pattern recognition*, World Scientific Publishing Co. Pte. Ltd., 1996.
 12. O. Cordon, M. del Jesus and F. Herrera, "A proposal on reasoning methods in fuzzy rule-based classification systems," *International Journal of Approximate Reasoning*, vol. 20, no. 1, pp. 21-45, 1999.
 13. G. Shafer, *A Mathematical Theory of Evidence*, Princeton University Press, 1976.
 14. L.-X. Wang, "The WM method completed: a flexible fuzzy system approach to data mining," *IEEE Transactions on Fuzzy Systems*, vol. 11, no. 6, pp. 768-782, 2003.
 15. H. Ishibuchi, Y. Kaisho and Y. Nojima, "Design of linguistically interpretable fuzzy rule-based classifiers: a short review and open questions," *Journal of Multi-valued Logic and Soft Computing*, vol. 17, pp. 101-134, 2011.
 16. J. Sanz, A. Fernández, H. Bustince and F. Herrera, "Improving the performance of fuzzy rule-based classification systems with interval-valued fuzzy sets and genetic amplitude tuning," *Information Sciences*, vol. 180, no. 19, pp. 3674-3685, 2010.
 17. M. Elkano, M. Galar, J. Sanz and H. Bustince, "CHI-BD: A fuzzy rule-based classification system for Big Data classification problems," *Fuzzy Sets and Systems*, 2017.
 18. E. Shortliffe and B. Buchanan, "A model of inexact reasoning in medicine," *Mathematical Biosciences*, vol. 23, 1975.
 19. M. Lichman, 2013. [Online]. Available: <http://archive.ics.uci.edu/ml>.
 20. J. van den Berg, U. Kaymak and W. van de Bergh, "Fuzzy classification using probability-based rule weighting," in *Proceedings of the 11th IEEE International Conference on Fuzzy Systems*, Hawaii, USA, 2002.

Classification in a Skewed Online Trade Fraud Complaint Corpus

William Kos, Marijn Schraagen, Matthieu Brinkhuis, and Floris Bex

Department of Information and Computing Sciences, Utrecht University

{w.h.kos, m.p.schraagen¹, m.j.s.brinkhuis, f.j.bex}@uu.nl

Abstract. This paper explores how machine learning techniques can be used to support handling of skewed online trade fraud complaints, by predicting whether a complaint will be withdrawn or not. To optimize the performance of each classifier, the influence of resampling, word weighting, and word normalization on the classification performance is assessed. It is found that machine learning can indeed be used for this purpose, by improving the baseline performance in comparison to the skewness ratio up to 13 pp using Logistic Regression. Furthermore, the results show that data alteration techniques can improve classifier performance on a skewed dataset up to 13.5 pp.

Keywords: Classification, Law Enforcement, Skewed Data.

1 Introduction

The Dutch National Police maintains an online interface that allows civilians to report their complaints regarding trade fraud over an online medium (e.g. eBay). Since an increasing amount of complaints are being filed [9], it is desirable to make an automatic distinction between complaints worth investigating and those not worth investigating. One valuable distinction which can be made early in the process is that between a complaint which will be withdrawn by either the complainant or the police and a complaint that will not be withdrawn. The current research examines whether either one of nine machine learning classifiers trained on free text complaint data can be used for this purpose. Complicating this task is the class distribution in the data, where a majority of 83.3% is labelled as "not withdrawn". To prevent this skewness from affecting classifier performance, data alteration techniques are applied, of which the influence on the classification performance is assessed. Specifically, resampling using multiple training sample distributions [12, 16], word normalization using either stemming [17] or lemmatization [4], and word weighting using either binary [19] or TF-IDF weighting [24] were applied to examine their influence on a classifier trained using a textual skewed dataset.

¹ Corresponding author

2 Related Work

Data mining techniques, including classification, are increasingly being applied to the field of crime analysis. Chen et al. explored data mining techniques used for crime analysis [7], and Sharma and Panigrahi have categorized over 40 approaches using machine learning techniques for fraud detection [21].

In [1], associative classification has been used, which is a technique where both association rules and classification are combined, to accurately discriminate phishing websites from legitimate websites. Making use of Bayesian networks, [2] have been able to predict the characteristics of a homicide offender based on crime scene variables (e.g. police report or autopsy report) more accurate than a team of police experts. These characteristics could be used by police officers to identify a possible suspect. In [9], naive Bayes, Bayesian network, decision tree, and association rule techniques were compared for their speed and accuracy in classifying crimes and accidents in Denver City, and it was found that association rules result in the highest accuracy. In their collaboration with the West Midlands Police, [15] used a Bayesian network to predict whether a certain property in the UK's Midlands will be re-victimized or not and within what timespan. Next to this, a neural network was used to classify possible offenders for their likelihood of conducting unsolved crimes.

Some of the research performed in this field focuses mainly on how to deal with a skewed dataset (i.e. one class is overly present). In their research, [5] combined a rule-based association system with a neural network to detect credit card fraud. Using their combined classifier, they are able to achieve an accuracy of 99.955%. In [16], research has been performed on which classification method is best to be used for fraud detection with a skewed dataset. It was proposed to use a stacking-bagging method, in which a naive Bayes, neural network and decision tree classifier are combined.

Previous work on the classification of online police complaints has to the knowledge of the authors not yet been presented.

3 Experimental Setup

3.1 Dataset

The online trade fraud complaints dataset used in this research has been provided by the Dutch National Police. This dataset consists of 51.386 entries, manually labelled by police employees on whether a complaint has been withdrawn and if so, for what reason. In total, 8.609 (16.7%) entries have been labelled *withdrawn*, resulting in a skewed dataset. The dataset contains a total of 60 attributes, including the binary class labels, and contains a free-text field in which the complainant's story that led to the complaint is included. Note that only the textual complaints description is used in the current research (see Section 5 for further discussion). An anonymized and translated example of both complaint types is included in Table 1.

Table 1: Data example

Withdrawn	John Doe advertised a rental home. In hindsight it all appears to be fake.
Not withdrawn	I have bought a bottle of Dom Perignon and a bottle of Crystal 1999 from John Doe via Marktplaats and transferred 100 euro to NL01ABCD0123456789. Up to now, I have not received anything and John Doe does not respond to my e-mails.

3.2 Classification technique selection

As no previous work on classifying online trade fraud complaints has yet been presented, it was unknown which classification techniques would result in the best performance. Therefore, the set of techniques used in this research was based upon a combination of previous work on classification for textual, skewed, and criminal data. An overview of the techniques used in this research is described by Sebastiani [20], who compared the classification results of ensembling based (e.g. AdaBoost), SVM, logistic regression, association rule (e.g. RIPPER), KNN, decision tree, neural network, and probabilistic classifiers used in individual papers on the highly skewed Reuters dataset, which contains a collection of news documents.

3.3 Research framework

In order to ensure the analyses in this research were performed under equal conditions, a research framework was created. Analysis is conducted using Weka [22] combined with R [18] using the package RWeka [10]. Linguistic analysis is performed by Frog [20], incorporated in the framework using the package Frogr². The research framework consists of two sections: the preprocessing section and the training/testing section.

Preprocessing

Since this research is based upon the prediction of the *withdrawn* label using a complaint's free-text field, these variables were first selected from the dataset, after which a corpus was created where each document represented a single complaint. The words in each document were transformed to their base form using either a Dutch adapted stemmer [17] or lemmatizer [4]. In a subsequent cleaning phase, all punctuation was removed and Dutch stop words were removed according to a predefined list³. Next, the corpus was split using stratified 10-fold cross validation [14], so that the ratio of minority to majority classes in each fold was maintained and each complaint was used once for testing and nine times for training.

² W. Van Atteveldt. *Frogr: R client for the frog tagger and parser for Dutch*. R package version 0.100, 2014.

³ <http://snowball.tartarus.org/algorithms/dutch/stop.txt>

Training and testing

Each of the 10 folds resulting from the preprocessing section was used for training and testing a classifier following the same procedure. First, 90% of the fold assigned for training the classifier was transformed into a term-document matrix containing 1:3-grams. All terms present in the term-document matrix were evaluated for their presence in the documents and the term-document matrix was reduced to the 100 most occurring n-grams, which are used as features for machine learning. Here, it has been opted to use the 100 most occurring terms over a standard feature selection algorithm to reduce the dimensionality of the problem and thereby clearly discriminate the influence of the mentioned data alteration techniques on the classification performance, which is the focus of this research. In preliminary experiments it was found that the classification performance improved when increasing the amount of features used for training up to 100, after which it stabilized. Furthermore, it was found that using 100 features greatly improves the framework runtime, while barely influencing the classification performance in comparison to using more features.

Next, the term-document matrix was resampled using either random undersampling or SMOTE [6] following the resampling techniques described by Japkowicz and Stephen [12] to reduce dataset skewness from influencing a classifier's performance. With random undersampling, the minority class entries were kept constant and the majority class entries were randomly downsampled according to the training sample distribution. For SMOTE, the minority class entries were scaled up to match the amount of majority class entries and the majority class entries were randomly up- or downsampled according to the training sample distribution. The classifier was then built on the resampled term-document matrix using Weka's default classifier implementations [22]. To avoid interference with the aspects under investigation in this research, we have opted to use the basic parameter settings provided by Weka.

After the classifier had been built it was tested using the assigned 10% of the fold. If a term used for training the classifier did not occur in the term-document matrix of the test set, a value of 0 was assigned to it for each document indicating its absence.

Finally, the evaluation results of each individual fold were combined and averaged resulting in an overall evaluation for the classifier under the predefined settings.

3.4 Evaluation

The performance on classifying online trade fraud complaints was evaluated using the macro-averaged F_1 -measure, following the approach of Yang and Liu [23]. When using the accuracy as a metric, the overall performance would be highly influenced by the data skewness. The macro-averaged F_1 -measure, however, equally weighs both recall and precision of the minority and majority class, thereby clearly showing the overall performance evenly influenced by both classes. The computation of the F_1 -measure defines the *not withdrawn* and *withdrawn* classes as positive and negative, respectively:

Actual class	Predicted class		
		not withdrawn	withdrawn
	not withdrawn (nwd)	TP	FN
	withdrawn (wd)	FP	TN

The F_1 -measure is defined in terms of precision and recall, as follows:

$$F_1 = \frac{F_{wd} + F_{nwd}}{2} = \frac{\frac{2 \cdot P_{wd} \cdot R_{wd}}{P_{wd} + R_{wd}} + \frac{2 \cdot P_{nwd} \cdot R_{nwd}}{P_{nwd} + R_{nwd}}}{2} =$$

$$\frac{2 \cdot \frac{TP}{TP + FP} \cdot \frac{TP}{TP + FN}}{\frac{TP}{TP + FP} + \frac{TP}{TP + FN}} + \frac{2 \cdot \frac{TN}{TN + FN} \cdot \frac{TN}{TN + FP}}{\frac{TN}{TN + FN} + \frac{TN}{TN + FP}}$$

$$\frac{2}{2}$$

To determine which classifier holds the most additional value, classification results should also be compared to a baseline [23]. As this research focused on the influence of data alteration techniques to the performance of a classifier, it was opted to compare all results to their unimproved counterpart. Regarding word normalization, however, it was determined to omit test cases where no word normalization was applied for runtime optimization. Based on three randomly generated feature lists using both stemming and lemmatization, it was decided to use stemming as the baseline for word normalization, as lemmatization generally diverges further from the non-normalized word forms. Overall, when evaluating the influence of data alteration techniques on classifier performance, it was thus compared to the results obtained using a training set which was not sampled, unweighted, and normalized using stemming.

4 Results and Discussion

4.1 Evaluating the overall results

During this research a total of 9 classifiers have been evaluated under different resampling, word normalization and word weighting conditions. An overview of the individual results has been combined in Table 2, which shows for each classifier the baseline compared to the optimal training setup, shown as resampling type, percentage of minority cases, weighting type.

When evaluating the observations of each individual classification algorithm, it becomes apparent that the difference in optimized performance between the best classifier (i.e. Logistic regression) and the worst classifier (i.e. K-nearest neighbor) is only minor with 4.2 percentage points (pp). Five classifiers hold the best optimized classification performance within a range of 1 pp, namely logistic regression, multinomial naive Bayes, support vector machine, multivariate naive Bayes, and association rule classifi-

Table 2: Overview individual classifier results (F1 score)

U/O: Undersampling/Oversampling; N/B/T: No/Binary/TF-IDF weighting				
Classifier	Baseline	Optimal	Setup	Difference
Multinomial naive Bayes	0.572	0.590	U30B	0.018
Logistic regression	0.466	0.594	U40N	0.128
Decision tree	0.527	0.560	U30T	0.033
Multivariate naive Bayes	0.560	0.586	N16B	0.026
Association rule	0.456	0.585	U40N	0.129
Neural network	0.454	0.564	U30B	0.110
K-nearest-neighbor	0.503	0.552	U40B	0.049
Support vector machine	0.454	0.589	O40T	0.135
AdaBoost	0.454	0.556	U40B	0.102

ers. Even though the differences between the individual baseline and optimal classification results vary in size, this minor difference in optimal performance suggests that a classifier can only be improved up to a certain extent. This would thus imply that, after optimizing, the selection of an appropriate classification algorithm should depend less upon performance, but more on other metrics (e.g. runtime). In our results, the macro-averaged F-measures do not exceed 0.594, which results from a low performance on the minority class. Underlying this relatively low overall performance could be two reasons:

- The features used in the free-text field for both withdrawn and not withdrawn complaints show a high resemblance. When, e.g., warning the police for a possible fraudster, which is not a valid complaint, words like *marktplaats* (online trading website), *oplichter* (fraudster), and *product* (idem) are often used. Such explanative features are also used in a complaint which has not been withdrawn. Comparing a subset of individual complaints revealed that it is possible that the free-text fields do not contain enough information to be distinct, thereby reducing the overall performance of a classifier trained on this dataset using features resulting from a bag-of-words approach.
- Since in this research it has been opted to use the 100 most occurring terms, which are used in both withdrawn and not withdrawn complaints, the overall performance of the classifiers could be reduced compared to when the more distinctive features would be selected using a feature selection algorithm.

4.2 Evaluating data alteration results

Resampling

With respect to resampling, each individual classification technique follows a similar trend as illustrated for an undersampled multinomial naive Bayes classifier in Figure 1. When increasing the training sample distribution, the precision of the minority class slightly decreases, while the recall of the minority class strongly increases. The majority class follows an opposite pattern where the precision of the majority class slightly increases, while the recall of the majority class strongly decreases. Even though

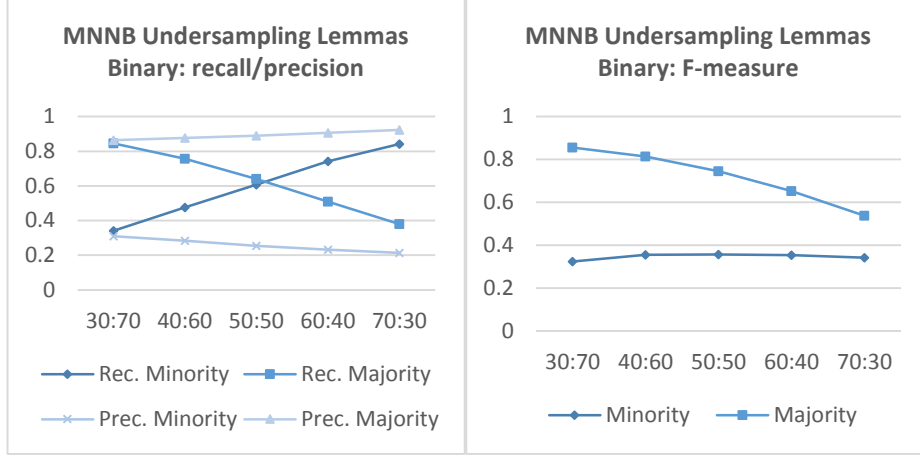


Figure 1: Recall, precision (left) and F-measure (right) using multinomial naive Bayes. X-axis: training sample distribution. Y-axis: recall/precision/F-measure.

resampling is intended to increase the minority performance without detriment of the majority performance, the observed pattern can be explained. During the training phase a classifier will, at first, become less inclined to automatically label a test case as the majority class as the skewness is lifted, after which it will be inclined towards the minority class as it becomes overly present in the training set. The rate at which the recall and precision change differs per classification technique, however, the precision of the minority class does, independent of the training sample distribution, remain centered around 30% for each classification technique. In a similar way, the precision of the majority class remains centered around 90%. These patterns do, however, imply that, in the current setup, the F-measure of the minority class can only be improved up to a certain extent, due to its harmonic nature as illustrated in 1. For the majority class, the F-measure only decreases due to the decline in recall, thus nullifying the influence of the high precision. Combining both patterns results in the conclusion that an optimal resampling percentage should thus be at a level where the initial increase in minority performance neutralizes the decrease in majority performance. This initial increase is the largest when there are less minority cases, which implies that the resampling percentage should be in favor of the majority class, which is confirmed by the optimal training sample distributions in Table 2.

Word weighting

Table 2 shows that for most classifiers, the right choice of weighting technique can outperform an unweighted baseline. Which technique to use depends upon the classification technique, however, binary weighting often outperforms TF-IDF weighting. This finding could imply that merely the presence of a term is enough for a classifier to be based upon, which is consistent with earlier findings with respect to SVM and naive Bayes classifiers [13, 19].

Word normalization

For word normalization Table 3 lists the difference in performance using either stemming or lemmatization for all classifiers under the optimal setup. The table shows that all classifiers perform better using lemmatization, however, the differences are only minor (0.5 to 1.8pp). Overall, with a few exceptions, lemmatization has little but consistently improved recall and precision of both the minority and majority class.

Table 3: Overview of word normalization influence

Classifier	Stemming	Lemmatization	Difference
Multinomial naive Bayes	0.584	0.590	0.006
Logistic regression	0.583	0.594	0.011
Decision tree	0.549	0.560	0.011
Multivariate naive Bayes	0.581	0.586	0.005
Association rule	0.574	0.585	0.011
Neural network	0.551	0.564	0.013
K-nearest-neighbor	0.543	0.552	0.009
Support vector machine	0.578	0.589	0.011
AdaBoost	0.538	0.556	0.018

4.3 Evaluating additional results

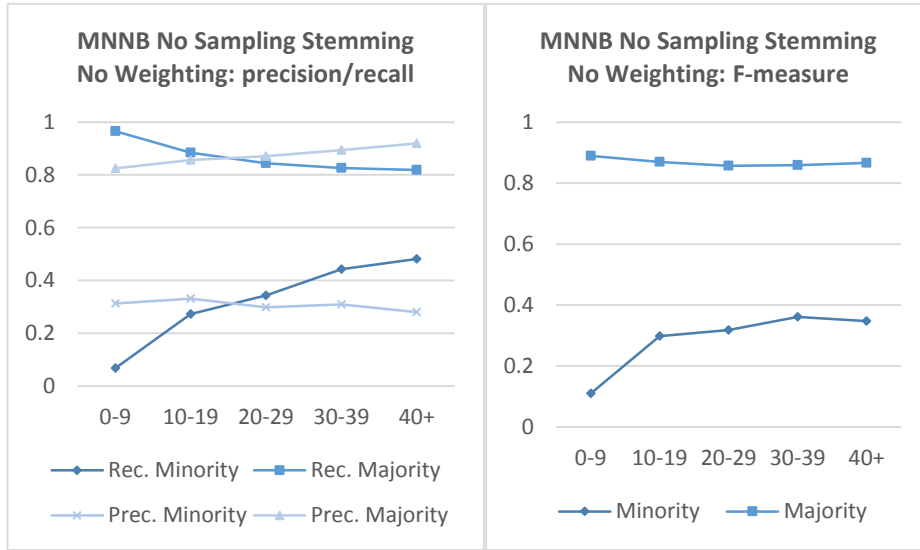
In section 4.1, two possible reasons have been mentioned with respect to the relatively low classification performance on the minority class. To initially examine these possible causes, follow-up experiments have been performed in which the influence of each cause is independently verified. Furthermore, an experiment has been executed regarding the probabilistic classifiers as the findings in this research did not match the prescribed literature.

The performance of probabilistic classifiers

As outlined in Section 4.1, probabilistic classifiers (i.e. multinomial naive Bayes and multivariate naive Bayes) have a good performance on the current data. In contrast, Yang and Liu [23] found logistic regression, SVM, and KNN to significantly outperform neural networks and naive Bayes with respect to the macro-averaged F-measure on the Reuters-21578 corpus. Even though it is based upon the micro-averaged F-measure, the result comparison of Sebastiani [20] also showed the lower performance of probabilistic classifiers on the skewed Reuters corpus in comparison to the other classifiers in his research. Here it should be noted that the Reuters corpus is a multi-labeled dataset instead of binary, and that the classifiers in the overview of Sebastiani [20] have not been improved using resampling. However, when comparing the unsampled results, the probabilistic classifiers still outperform all other classifiers in this research including logistic regression and association rules. Looking at the probabilistic classifiers individually, it was observed that the difference in performance between an unsampled and a resampled classifier was negligible. Combining this with the skewed characteristic of the dataset led to the assumption that on a highly skewed dataset, determining the

Table 4: Results per feature set using multinomial naive Bayes

#Features in test set	Set size	Recall minority	Recall majority	Precision minority	Precision majority	F-measure minority	F-measure majority
0-9	1529.2	0.068	0.965	0.313	0.825	0.110	0.889
10-19	1921.8	0.273	0.884	0.331	0.856	0.298	0.870
20-29	1016.4	0.343	0.844	0.298	0.871	0.318	0.857
30-39	459.9	0.443	0.827	0.309	0.894	0.361	0.859
40+	211.3	0.482	0.819	0.281	0.920	0.348	0.866

**Figure 2:** Recall, precision (left) and F-measure (right) using multinomial naive Bayes. X-axis: #features in test set, Y-axis: recall/precision/F-measure.

posterior probability in a naive Bayes classifier is mainly influenced by the class likelihood and less by the class priori. To test this hypothesis, an experiment has been set up in which the class predictions were evaluated with respect to the total amount of features in the test set which have been used for building the classifier. For this experiment, the multinomial naive Bayes classifier has been trained according to the baseline and framework as described in section 3.4. The results of this experiment are contained in Table 4 and depicted in Figure 2.

Evaluating the above results reveals a clear relationship between the predictive accuracy of a classifier and the amount of features used in the test set. Using less features causes a multinomial naive Bayes classifier to predict more test cases as the majority class compared to using more features. This observation supports the assumption that the influence of the priori decreases as more features are present in the test set. Furthermore, this experiment shows that the predictive power of a probabilistic classifier on a skewed dataset depends upon the amount of features used in the test set. Since in this research the 100 most common features have been used, the amount of features used in the test set are likely to be correlated with the length of the test set, thereby resulting in

the assumption that the predictive power of a classifier on textual data depends on the length of the document.

Table 5: Results using multinomial naive Bayes with bi-normal separation

Selection method	Recall minority	Recall majority	Precision minority	Precision majority	F-measure minority	F-measure majority
BNS	0.033	0.989	0.384	0.836	0.060	0.906
Occurrence	0.241	0.892	0.314	0.854	0.272	0.872

The influence of features

In section 4.2, it was mentioned that independent of the training sample distribution, the precision of the minority class remains fixed around 30%. This implies that, even though the 100 most common features contain more information than simply classifying all test cases as either the minority or majority class, the information richness is restricted. To examine whether feature selection could be used to overcome this limitation in information richness, an experiment following the same training procedure as above has been set up in which bi-normal separation [8] is used to select the most informative features. The results of this experiment are compared to the classifier trained using the 100 most occurring features in Table 5.

Given that the selected features are highly distinctive, the low minority recall using bi-normal separation suggests a reduced influence of the priori, which, as was discussed earlier, implies that few features are used for testing when training the classifier with 100 selected features. Combining this with the above example leads to the conclusion that the distinctive features selected using bi-normal separation do not cover a wide spectrum of the test cases. Since the most distinctive features occur in only a limited amount of complaints, the assumption is supported that the features in the dataset do not contain enough information to be used for making a distinction on whether a complaint will be withdrawn or not.

Comparing the results of the two feature selection methods, it can be observed that for 100 features using bi-normal separation as the feature selection metric only reduces the performance of multinomial naive Bayes classifier (-8.9pp). Combining this with above conclusion that the distinctive features in this dataset cover only a small spectrum of the test cases, it can be concluded that when training a multinomial naive Bayes classifier on a dataset with little distinctive features the best feature selection metric is to use the most common features. Further research is required to conclude whether this finding also applies to other classifiers.

5 Conclusions and Future Research

In this research, it has been examined whether a complaint's free-text field can be used to predict whether a complaint will be withdrawn or not. Literature studies on the use of machine learning techniques for characteristics of the dataset (i.e. criminal, textual,

and skewed) revealed the combination of machine learning with data alteration techniques to result in the best classification performance, which was confirmed during the data analysis. The data analysis furthermore revealed that the best optimized machine learning technique to be used for making the prediction is Logistic regression. Using a Logistic regression classifier, a recall of 33.5% and a precision of 29.7% can be attained for the minority class, while for the majority class a recall of 84.0% and a precision of 86.3% can be attained, which exceeds the unsampled ratio of minority to majority complaints (16.7%/83.3%).

Regarding probabilistic classifiers, it was found that using more features in the test set can improve the predictive power. Furthermore, it was found that when training a multinomial naive Bayes classifier on a dataset with little distinctive features the best feature selection metric is to use the most common features.

Even though it can be concluded that a complaint's free-text field can indeed be used to some extent to predict whether a complaint will be withdrawn or not, the performance was restricted. Since a complaint does not solely exist of a free-text field, but contains 59 other attributes, such as location fields, payment method, social media usage. Exploratory analysis showed the potential of this approach. Textual meta-attributes such as word or document length could be included as features as well. In future research a selection of these attributes may lead to a more accurate classifier. In addition, the decision to reject a complaint may be influenced by developments during the investigation following the initial filing of the complaint, which is outside of the scope of the current dataset.

Acknowledgements

This research has been funded by the Dutch National Police in the project 'Intelligence Amplification for Cybercrime' (IAC) [3].

References

1. Abdelhamid, N., Ayesh, A., Thabtah, F.: Phishing detection based Associative Classification data mining. *Expert Systems with Applications*, 41(13): 5948–5959, 2014.
2. Baumgartner, K., Ferrari, S., Palermo, G.: Constructing Bayesian networks for criminal profiling from limited data. *Knowledge-Based Systems*, 21(7):563–572, 2008.
3. Bex, F.J., Testerink, B. and Peters, J. (2016) A.I. for Online Criminal Complaints: From Natural Dialogues to Structured Scenarios. *ECAI 2016 workshop on Artificial Intelligence for Justice*.
4. van den Bosch, A., Busser, G.J., Daelemans, W., Canisius, S.: An efficient memory-based morphosyntactic tagger and parser for Dutch. In: van Eynde, F., Dirix, P., Schuurman, I., Vandeghinste, V. (eds.) *Selected Papers of the 17th Computational Linguistics in the Netherlands Meeting*, pages 99–114, 2007.
5. Brause, R., Langsdorf, T., Hepp, M.: Neural Data Mining for Credit Card Fraud Detection. In: *Proceedings of the 11th IEEE International Conference on Tools with Artificial Intelligence*, pages 103–106, 1999.

6. Chawla, N., Bowyer, K., Hall, L., Kegelmeyer, W.: SMOTE: Synthetic Minority Over-Sampling Technique. *Journal of Artificial Intelligence Research*, 16:321-357, 2002.
7. Chen, H., Chung, W., Xu, J., Wang, G., Qin, Y., Chau, M.: Crime Data Mining: A General Framework and Some Examples. *Computer*, 37(4):50-56, 2004.
8. Forman, G.: An Extensive Empirical Study of Feature Selection Metrics for Text Classification. *Journal of Machine Learning Research*, 3:1289-1305, 2003.
9. Gupta, A., Syed, A., Mohammad, A., Halgaguge, M.: A Comparative Study of Classification Algorithms using Data Mining: Crime and Accidents in Denver City the USA. *International Journal of Advanced Computer Science and Applications*, 7(7):374-381, 2016.
10. Hornik, K., Buchta, C., Zeileis, A.: Open-Source Machine Learning: R Meets Weka. *Computational Statistics*, 24(2):225-232, 2009.
11. Inspectie Veiligheid en Justitie. Aanpak van internetoplichting door de politie: inspectieonderzoek naar een vorm van cybercrime, 2015.
12. Japkowicz, N., Stephen, S.: The Class Imbalance Problem: A Systematic Study. *Intelligent data analysis*, 6(5):429-449, 2002.
13. Jurafsky, D., Martin, J.: *Speech and Language Processing*. Prentice Hall, 2014.
14. Kohavi, R.: A Study of Cross-Validation and Bootstrap for Accuracy Estimation and Model Selection. In: Mellish, C (ed.) *Proceedings of the 14th International Joint Conference on Artificial Intelligence*, pages 1137-1143, 1995.
15. Oatley, G., Ewart, B: Crimes analysis software: 'pins in maps', clustering and Bayes net prediction. *Expert Systems with Applications*, 25(4):569-588, 2003.
16. Phua, C, Alahakoon, D, Lee, V: Minority Report in Fraud Detection: Classification of Skewed Data. *ACM SIGKDD Explorations Newsletter*, 6(1):50-59, 2004.
17. Porter, M.: An algorithm for suffix stripping. *Program*, 14(3):130-137, 1980.
18. R Core Team. R: A language and environment for statistical computing. *R Foundation for Statistical Computing*, Vienna, Austria, 2015.
19. Schneider, K-M: On Word Frequency Information and Negative Evidence in Naive Bayes Text Classification. In: Vicedo, J., Martínez-Barco, P, Muñoz, R., Saiz Noeda, M. (eds.) *Advances in Natural Language Processing. Lecture Notes in Computer Science*, 3230, pages 474-485, 2004.
20. Sebastiani, F.: Machine Learning in Automated Text Categorization. *ACM Computing Surveys (CSUR)*, 34(1):1-47, 2002.
21. Sharma, A., Panigrahi, P.: A Review of Financial Accounting Fraud Detection based on Data Mining Techniques. *International Journal of Computer Applications*, 39(1):37-47, 2012.
22. Witten, I, Frank, E.: *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann, 2005.
23. Yang, Y., Liu, X.: A re-examination of text categorization methods. In: *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 42-49, 1999.
24. Zhang, W., Yoshida, T., Tang, X.: A comparative study of TF*IDF, LSI and multi-words for text classification. *Expert Systems with Applications*, 38(3):2758-2765, 2011.

Studying Gender Bias and Social Backlash via Simulated Negotiations with Virtual Agents

L.M. van der Lubbe and T. Bosse

VU Amsterdam, De Boelelaan 1105, 1081 HV Amsterdam, the Netherlands,
lauravanderlubbe@hotmail.com, t.bosse@vu.nl

Abstract. This research investigates whether (female and male) virtual negotiators experience a social backlash during negotiations with an economical outcome when they are using a negotiation style that is congruent with the opposite gender. To this end, first some background research has been done on gender differences in negotiations and the social backlash that is experienced by men and women. Based on this literature study, an application has been implemented (using the tools Poser Pro, Renpy and IVONA) that enables users to engage in a salary negotiation with a virtual agent that plays the role of employee. Next, an experiment has been conducted in which 93 participants had an interactive negotiation with the virtual employee. Results show that the effect of gender on negotiation outcome and social backlash was less pronounced in this experiment than expected based on the literature. However, several factors, such as the experience of the participants and the provided context, could explain these findings.

Keywords: virtual agents, salary negotiations, social backlash, gender bias, gender pay gap.

1 Introduction

The gender pay gap refers to the phenomenon that women on average have lower salaries than men for the same type of job, even when adjusting for external factors like working time. One of the possible explanations of this phenomenon is the fact that men often achieve better negotiation outcomes than women [10], which may be caused by the fact that behaviors congruent with the female gender role are typically not seen as efficient for negotiations [2]. Unfortunately, adjusting the negotiation style in such a way that it is no longer congruent with the gender role can lead to a social backlash (e.g., being perceived as less friendly) [2]. Social backlash is the effect that violation of stereotypes results in social reprisals [14]. Female negotiators suffer more from social backlash than male negotiators [6], both regarding the economic outcome of the negotiation and the evaluation of the negotiator [15].

The goal of this research is to investigate whether this effect of gender on negotiation outcome and social backlash is also present when people negotiate with virtual agents. By using a virtual agent in the role of either a male or female employee, and a human participant in the role of manager, it is possible

to manipulate only the variables of interest and keep other variables constant among different conditions (which is hard to achieve in an experiment based on human-human interaction). For this research it is expected that the negotiation outcomes in the experiment are comparable to human-human negotiation outcomes. This expectation is motivated based on research that has shown that the effects of emotions of the negotiation partner on the outcomes of human-machine negotiations are comparable to the effects found within human-human negotiations [11]. Furthermore, this is in line with studies showing that people see computers as social actors, leading to interactions that are comparable to human-human interactions [13]. Research has also shown that negotiating agents can be used as training tool for humans [5]. If this research finds that the effect of gender on negotiations in virtual negotiations is comparable to the effect in human-human negotiations, virtual agents may be used in the future to develop interventions to make people aware of their gender bias during negotiations.

Two negotiation styles, one congruent with the male gender role and one congruent with the female gender role, have been implemented in a female and male virtual agent in the role of employee. It is investigated whether females and males negotiating with a negotiation style opposite to the style of their gender role are punished for this, meaning their outcome benefits less from the other negotiation style compared to the opposite gender. Both genders need to be compared to measure whether male and female negotiators are treated differently, potentially causing a social backlash that is bigger for women compared to men.

Section 2 will give an overview of the literature on negotiation and gender differences. Section 3 explains the research method used, including the hypotheses and experimental design. Section 4 describes the implementation of the virtual agent application. Section 5 presents an overview of the found results and Section 6 concludes the paper with a discussion.

2 Background literature

This section provides an overview of the relevant literature in order to formulate hypotheses suiting our main research question and support the negotiation scenario that has been written for the experiment. First of all, a general overview of the gender differences in negotiations and the experienced backlash is given. Next, the aspects that influence a negotiation and the negotiation styles fitting both gender roles are explored.

2.1 Gender differences in negotiations and social backlash

Gender is one of the most frequently studied factors that play a role during negotiations (for overview articles see [9,10,16]). Gender differences can influence many different aspects of negotiations and their outcomes; the meta-analysis of [10] presents a summary of these aspects. One of the most important aspects is the difference in economical outcome, since this might explain the gender pay gap [10].

The more flexible and individual nature of employment terms nowadays can be disadvantageous for women, since they face many obstacles during negotiations [9]. One such obstacle that is of particular interest for the research of this paper is the social backlash that women may encounter during negotiations. Behaviors that are typically considered to be beneficial during negotiations (e.g., being assertive) are often not considered to be in accordance with the female gender role [2]. When women adjust their behavior to fit the task of negotiating, going against the female gender role, they might experience a social backlash [2]. This means that women who go against their female gender role be ‘punished’ for this behavior during negotiations. This backlash is not only visible in the economical outcomes of the negotiation but also in how the employee is evaluated by the negotiation partner [15]. It has been found that female negotiators experience more social backlash than male negotiators [6].

Not only the gender of the negotiating employee is influencing negotiations, the gender of the interlocutor might also influence the outcome of a negotiation [10]. A backlash is more likely to occur with female evaluators [9].

2.2 Aspects influencing negotiations

In addition to gender alone, many related factors can influence the outcome of a negotiation (for an overview see [10]). This subsection gives an overview of such influencing aspects that might need to be taken into account when writing a scenario for the negotiation that will be implemented for this research.

- Negotiation style: Women overall demonstrate a slightly less competitive negotiation style, compared to men [16]. The female negotiation style can be seen as a more cooperative or altruistic style [4], it is also described as an accommodating style, which leads to lower outcomes than the competitive negotiation style of men [16]. Female negotiators ask for less and make more generous offers compared to male negotiators [4].
- Advocacy: When people negotiate on behalf of someone else different social roles are involved compared to when they are negotiating for themselves [10]. This alters the behaviors that are considered gender typical. It also changes the strategy that negotiators use [1].
- Initiation of the negotiation: Women are less likely to initiate a negotiation [3, 10] and women might be punished more for initiating negotiations [3].
- Type of negotiation: The meta analysis of [10] distinguishes two types of negotiations, called distributive and integrative negotiations. Salary negotiations have a distributive form: it is not about a mutual gain, but about a gain for the employee. An assertive negotiation style, which is congruent with the male gender role, is often an advantage in such a negotiation [10].
- Domain of the job: Women might experience fewer difficulties when negotiating about a job that fits the female stereotype [15].
- Nonverbal aspects: Cooperative negotiators express behaviors that can be described as “warm”, whereas competitive negotiators express “cold” behaviors [7, 8]. This could be linked to the negotiation styles described earlier.

Although there is proof that nonverbal and verbal communication should be congruent with each other, a lack of congruence between those does not lead to the expected confusion [8]. Moreover, nonverbal messages are more important than verbal messages [8].

The nonverbal behavior of the respondent, combined with the behavior of the proposer and the history of the two parties, can be used to automatically predict the outcome of a negotiation [12].

3 Research Method

In Section 3.1 the main research question and hypotheses are introduced and motivated based on the literature presented in the previous section. Next, the design of the experiment is described in Section 3.2.

3.1 Research question and hypotheses

Our main research question can be formulated as follows: *How does the gender of a virtual agent influence its negotiation outcome and social backlash during salary negotiations with a human negotiation partner?*

In order to answer this research question, the following hypotheses are formulated, based on the background literature from Section 2.1:

1. The outcomes of negotiations are lower for female virtual agents than for male virtual agents, independent of the negotiation style used.
2. Both female and male virtual agents experience a social backlash when negotiating using a negotiation style that is not congruent with their gender role.
3. The gender of the human participant influences the outcome of the negotiation and the experienced backlash. Male-male, female-male, male-female and female-female negotiations will have different outcomes and different experienced backlashes.

It is important to note that the ‘outcome’ of a negotiation refers to a monetary measurable outcome. It is measured in an absolute way (i.e., the value of the agreement deal) as well as a relative way (i.e., the relative value of the agreement deal compared to the opening offer). Furthermore, the social backlash is measured in terms of the participants’ subjective evaluation of the negotiation. To this end, participants had to evaluate a number of statements on a 7-point Likert scale (see Section 4.4 for more details).

3.2 Experimental design

A total of 93 people (55 male, 38 female) participated in this study. All participants were at least 18 years old and had an adequate level of English to understand the negotiation. Only 15% of the participants had professional negotiation experience. At the end of the experiment participants were asked to

guess what the research was about. Based on the results, it is assumed that none of the participants knew the exact goal of the research. Although a few participants (both male and female) did mention gender, most participants mentioned something like testing realism of virtual agents.

Within the experiment, four conditions were used:

- Female virtual employee - Assertive negotiation style (21 participants; 62% male, 38% female)
- Female virtual employee - Non-assertive negotiation style (25 participants; 56% male, 44% female)
- Male virtual employee - Assertive negotiation style (24 participants; 58% male, 42% female)
- Male virtual employee - Non-assertive negotiation style (23 participants; 61% male, 39% female)

For the experiment a between-subjects design was chosen, to prevent results from being affected by negotiations conducted earlier. Each participant was randomly assigned to one of the conditions. The type of virtual employee and the gender of the participant were independent variables, whereas negotiation outcome and social backlash (as defined in the previous sub-section) were dependent variables.

To conduct the virtual negotiation a compressed folder, containing an application that can be run without application, was downloaded by the participants via a website which randomly assigned the participants to a condition.

This application contains the interactive scenario that is further explained in Section 4. At the end of the interactive scenario the participant received a hyperlink to the questionnaire described in Section 4.4.

4 Implementation

A negotiation scenario was created based on the literature of Section 2 and was used for the experiment. In Section 4.1 the choices made in the design of this scenario are explained. Section 4.2 explains the scenario that resulted from those choices. Section 4.3 describes the technical implementation of the application, and Section 4.4 describes the questionnaire used for the experiment.

4.1 Design choices

To enable a valid comparison, the number of factors that could influence the outcome of a negotiation (see Section 2.2) in favor of one of the two genders should be minimized. In order to ensure this, the following factors are taken into account:

- Advocacy: The virtual negotiator will negotiate on behalf of their own instead of on behalf of someone else to make sure that the gender typical behavior and thereby most efficient strategy is not influenced by this factor.

- Initiation of the negotiation: The negotiations in the scenario should not be initiated explicitly by the virtual negotiator. Therefore, the created scenario states that the negotiation is part of a yearly evaluation moment and every employee is asked to negotiate about their salary. This way the negotiation does not need to be initiated by one of the parties, since women are less likely to do this according to their gender role.
- Type of negotiation: Using a distributive negotiation type is useful to investigate the social backlash since the effective negotiation style for this type is typical for one of the genders. It can be assumed that the highest outcome is realized when the virtual employee uses an assertive negotiation style.
- Domain of the job: To make sure that no stereotypes are activated, it is important to have a scenario about a job that is as gender neutral as possible.
- Nonverbal aspects: For the implementation of the scenario the nonverbal behaviors will be kept neutral, meaning they cannot be classified as specifically “warm” or “cold”. This is done in order to make sure that the type of nonverbal communication cannot influence the negotiation results. The nonverbal behaviors of the virtual employees are kept constant in each condition to ensure an equal influence on all conditions.

4.2 Resulting scenario

Before starting the actual negotiation, participants have to read a short description about the background of the scenario, in which they are told they play the role of manager in a large home store¹. Figure 1 shows an overview of the possible developments of the negotiation, which has the form of a turn-based dialogue (the text can be found in appendix A). The rectangles represent the possible behaviors of the virtual employee, whereas the arrows indicate the options the participant has. As shown, every turn the participant has three options, namely to accept the last offer (which leads to an agreement), reject the last offer (leading to an unsuccessful end of the negotiation), and to make a (predefined) counteroffer.

As mentioned earlier, the negotiation style of the virtual employee is either assertive or non-assertive. These styles are not only reflected in the absolute values of the offers and the text said by the virtual employee, but also by the likelihood to accept or further negotiate when using a non-assertive negotiation style. Since it is found in the literature that non-assertive negotiators in particular have a tendency to accept an offer earlier, each turn of the non-assertive employee (but not the assertive employee) involves a probability of accepting the last offer, hence reaching an agreement. This is indicated with a percentage for the likelihood of each reaction type (accepting or counteroffer). Although this makes analyzing the results more complicated, since there is more variation across participants within the non-assertive condition, this approach makes the non-assertive conditions more realistic. In all cases where no percentages are mentioned, the response of the virtual employee is deterministic.

¹ The used background sketch can be found at <https://goo.gl/rMIq8y>

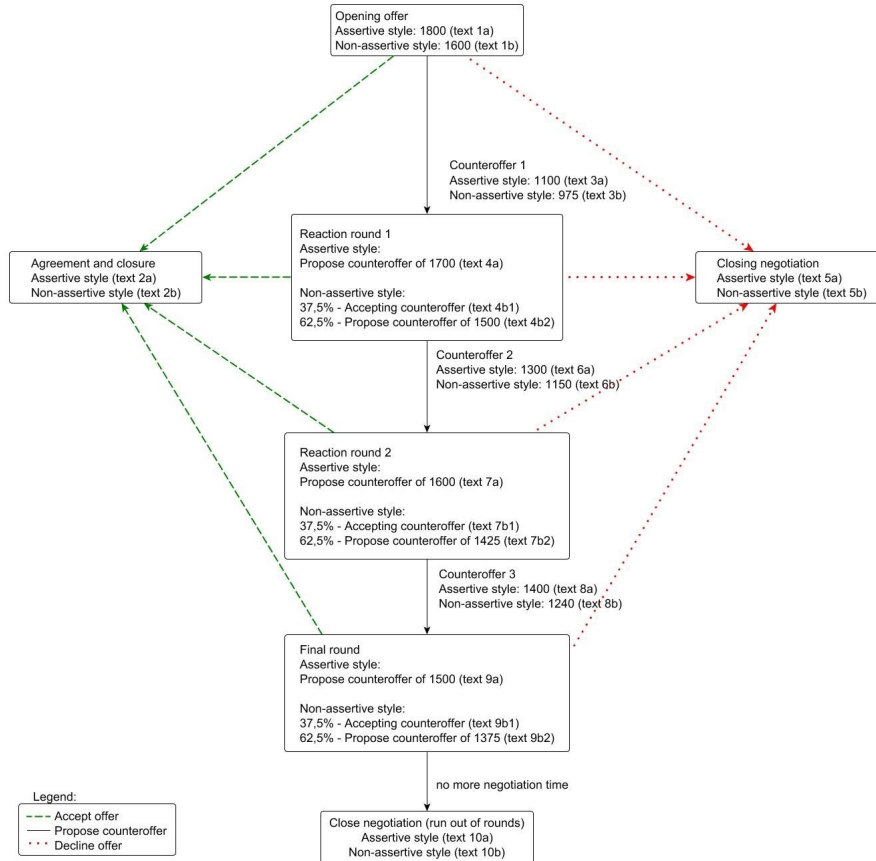


Fig. 1. Graphical overview of scenario flow

In order to make the results comparable, the percentages of all offers relative to the opening offer are kept equal among the two negotiation styles. For the non-assertive negotiation style, in which the counteroffers of the participant can be accepted by the virtual employee, a few extra offers (and also percentages) have been created.

4.3 Modeling the virtual negotiation

To model the appearance of the virtual employee, Poser Pro 2014² has been used. A standard office scene is used as scene for the virtual negotiation. The standard figures Rex and Roxie are used as virtual employees Mark and Marie. Their clothing is taken as similar as possible (a gray t-shirt with short sleeves

² <http://my.smithmicro.com/poser-3d-animation-software.html>

for both characters), as well as their hair color (black) and the manually created pose they are sitting in. In Figure 2 the resulting scene setting for the characters Mark and Marie are shown.

To translate the written scenario into a spoken text that can be used as speech of the virtual employee, IVONA³ voices have been used. For the implementation a male and female voice with the same accent (American English) have been chosen, as well as the same sound quality (22 kHz). This ensures that the results are not influenced by a difference in accent or sound quality between the conditions. With Poser Pro 2014's Talk Designer the characters' lips can move according to the sound and text that is provided. Another feature of the Talk Designer is the possibility to automatically add some head movements and eye blinks to the talking. The rate for this was kept equal among all the different texts and genders. The head movements are constrained in order to make sure that the employees keep looking at the camera; these constraints are also kept equal for both employees. The Talk Designer also allows to add emotion to the facial expressions; however, all emotions have been set to 0 to make sure they do not influence the results, and because the relation between negotiation and affect is outside the scope of this paper.

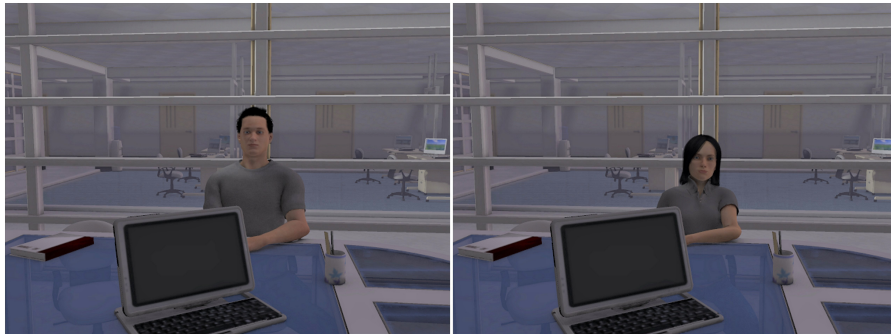


Fig. 2. Scene setting of Mark (left) and Marie (right)

For each utterance of the virtual employee, a separate video has been rendered. To create a interactive negotiation with these videos a Python⁴ script in Renpy⁵ was used. At the beginning of the negotiation, the description of the context mentioned earlier is presented to the user for 60 seconds, or until (s)he skips it. For each decision point, a multiple choice menu is presented to the user.

³ <https://www.ivona.com/us/>

⁴ <https://www.python.org/>

⁵ <https://www.renpy.org/>

4.4 Questionnaire

At the end of the negotiation the participant is provided with a hyperlink to a questionnaire, which has been created in Google Forms.

Within this questionnaire (see Appendix B), the first two questions are used to acquire information about the outcome of the negotiation, expressed in terms of the round in which the negotiation ended and the monetary outcome reached in the negotiation. The round number and outcome are already filled in, the participant only needs to check them.

In the remaining questions, the following aspects are addressed:

1. Background information about the participant, among which his or her negotiation experience.
2. Feelings of the participant towards the negotiation and its outcome.
3. How the participants feels about the virtual employee, using both open and closed questions on a seven point Likert scale. Among others, the participants are asked to rate typically female and male characteristics, based on the questionnaire of [2].
4. The participant's opinion on the implementation of the virtual employee.
5. Whether the participant knows the topic of the research.

The questions are created to serve two purposes; on the one hand there are questions that are used to test the hypotheses of this research (in particular, the questions highlighted in bold in Appendix B), on the other hand some questions have been added to distract the participants from the actual purpose of the research (e.g., asking them how realistic they found the appearance of the agent).

5 Results

5.1 Hypothesis 1 - Negotiation outcomes

Figure 3 shows the salary raises that were achieved by the four types of virtual employees for all negotiations that ended in a deal. Importantly, however, the negotiations that did not end in a deal (either because the human participant stopped the negotiation, or because no deal was reached after 4 rounds of interaction⁶) are not included in this figure. The number of negotiations in which no deal was reached were as follows: Female Assertive 6, Female Non-Assertive 5, Male Assertive 7, Male Non-Assertive 4.

As can be seen in Figure 3, especially for the non-assertive employees, the salary raises that were reached by women and men are very similar. Our first hypothesis stated that the outcomes of the negotiations would be lower for female virtual employees than for male virtual employees, independent of the negotiation style used. To test this hypothesis, two one-sided independent t-tests have been performed: one comparing the absolute outcomes of all female employees

⁶ Participants knew that they had only one more choice moment when they reached round 4

(both assertive and non-assertive) with those of all male employees, and one comparing the relative outcomes (in percentage of the opening offer). For negotiations that ended without a deal being reached, the outcome was considered to be zero. Moreover, a Bonferroni correction for the two tests was applied to reduce the chance of a Type-I error. These tests pointed out that there was no significant difference between the genders, both regarding the absolute outcome ($p=0.43$) and the relative outcome ($p=0.45$). Hence, using a significance level of 0.05, Hypothesis 1 was not confirmed.

On the contrary, a closer look at Figure 3 reveals that, when considering the assertive employees only, the female employees on average achieved slightly higher raises than the male employees. Yet, this difference turned out to be not statistically significant. However, when leaving out the negotiations in which no deal was reached (no deals were first considered as an outcome of 0), the difference actually is significant ($p<0.05$). This is a surprising insight because most of the existing literature claims that it does not pay off for women to negotiate using an assertive style.

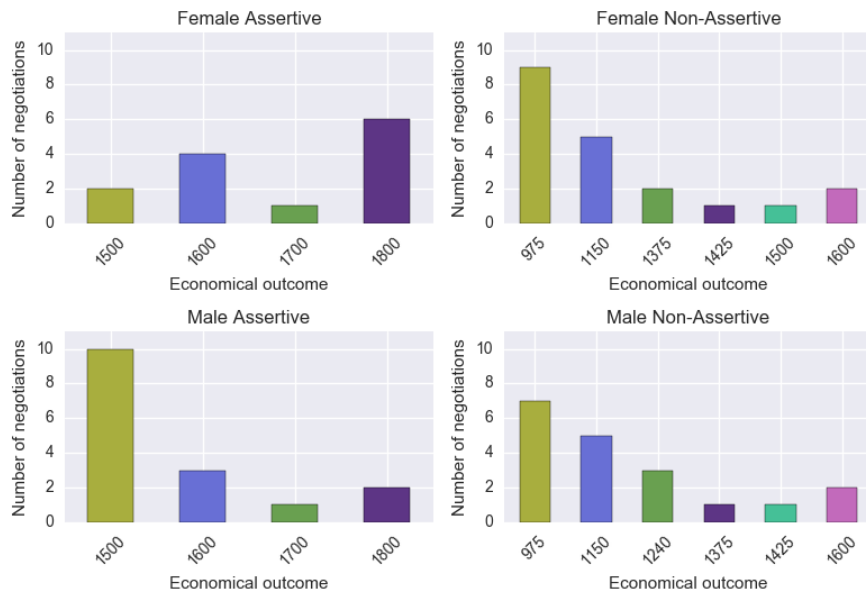


Fig. 3. Salary raises reached in negotiations ending with a deal

5.2 Hypothesis 2 - Social backlash

The second hypothesis focuses at the social backlash experienced by both genders, measured in terms of subjective measures. The main results of the subject-

tive evaluations are shown in Figure 4. This figure contains five characteristics: two of these (friendly and sensitive) are assumed to be typically female properties, whereas two others (arrogant and independent) are typically male properties [2]. The fifth characteristic concerns whether the employee used appropriate language.

As can be seen in the figure, most differences between the male and female negotiators are rather small. For each characteristic, we tested, both for the female and for the male employees, whether there was a significant difference between the rating of the assertive and the non-assertive employee. Again, this was done using independent t-tests with Bonferroni correction.

For the female employees, significant differences were found for the characteristics Friendly ($p < 0.001$), Arrogant ($p < 0.001$), and Language ($p < 0.005$). For the male employees, significant differences were found for the characteristics Friendly ($p < 0.001$), Arrogant ($p < 0.001$), Language ($p < 0.001$), and, interestingly, Sensitive ($p < 0.005$).

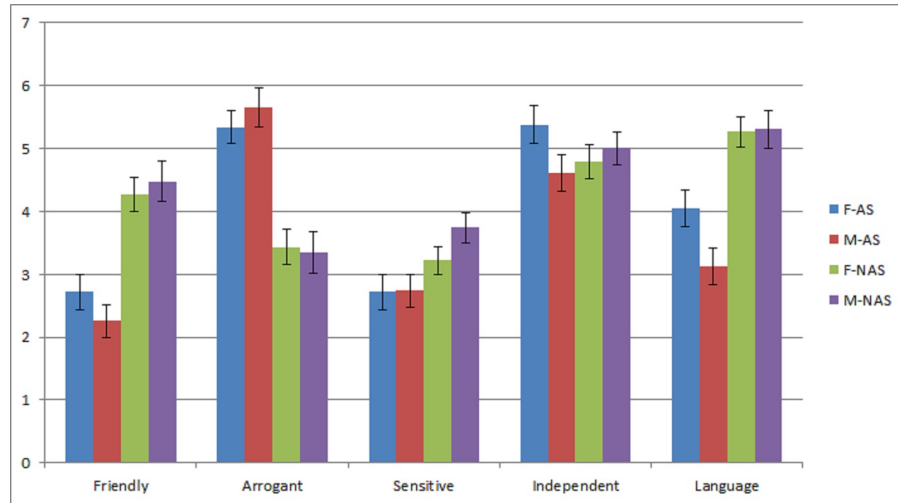


Fig. 4. Subjective evaluations (F:female, M:male, AS:assertive, NAS:non-assertive)

When looking at Hypothesis 2, it can partly be confirmed. The original formulation of the hypothesis was that both genders would experience a social backlash when negotiating using a negotiation style that is not congruent with their gender role. Since the stereotypical female style is non-assertive, we can conclude that our female agents suffered from a backlash on some points: when using an assertive style, they were considered less friendly, more arrogant, and using less appropriate language. Instead, such negative effects cannot be observed when male agents deviate from their stereotypical style (i.e., when they take a non-assertive instead of an assertive style). On the contrary, in this case

they are considered more friendly and sensitive, less arrogant and using more appropriate language. However, it should be noted that the evaluation did not include any negative features that apply to non-assertive people (such as ‘weak’ or ‘insecure’). If such features were included, we might have found backlashes for male agents as well (cf. [6]).

All in all, on the basis of Figure 4 it can be concluded that the male and female agents received more or less the same evaluations. Nevertheless, some interesting differences between the genders were found. In particular, there are significant differences for sensitivity (non-assertive male agents were considered more sensitive than non-assertive female agents, $p < 0.05$) and for appropriate language (assertive female agents were considered to use more appropriate language than assertive male agents, $p < 0.05$).

5.3 Hypothesis 3 - Gender of participants

Hypothesis 3 stated that the gender of the participant influences the outcome. To test this, the tests performed for Hypothesis 1 and 2 were repeated, but now in a two way ANOVA to check whether there is an interaction effect with the gender of the participant. However, for none of the tests a significant interaction effect was found. Hence, Hypothesis 3 could not be confirmed.

6 Discussion

This research investigated how the gender of a virtual agent influences its negotiation outcome and social backlash during salary negotiations with a human negotiation partner. The results pointed out a number of things. First, there were no big gender differences in terms of the monetary outcome of the negotiations, and surprisingly the assertive females even obtained slightly higher outcomes than the assertive male negotiators. Second, as expected, female employees suffered from several social backlashes when using an assertive style instead of a non-assertive style. Contrary to our hypothesis, no backlashes were found for male employees using a non-assertive style (instead, they were evaluated more positively), although this may be explained by the lack of features such as ‘insecure’ in our study. Finally, no effect of the gender of the evaluators was found.

It is interesting to speculate on why the results differ from the literature at some points. There are several factors that might explain this. Most of these factors have to do with the specific setup of this experiment, involving a human-agent negotiation with relatively limited interaction possibilities. For instance, participants made remarks about the virtual employee being a bit robotic or fake, both because of its voice and the behaviors during the negotiation. As mentioned, the participants also rated the virtual employee on realism and human-likeness of its sound (even though these questions were not used to test the hypotheses). The mean scores for this were 3.73 (of 7) for the realism and 2.91 (of 7) for the human-likeness of the voice. This implies that the virtual employee was not really experienced as a human interlocutor, which might make the results

not generalizable for human negotiations. An improvement for future research might be to incorporate different emotions matching the negotiation styles to further improve the credibility of the virtual agents. In addition, researching the possibilities of incorporating human speech as a way to interact and negotiate with a virtual agent, giving the participant a more natural experience with more interaction freedom, could be an interesting artificial intelligence challenge.

Other factors that might explain the results involve the background of the human participants. A lot of participants mentioned their lack of negotiation experience in the questionnaire. The fact that 85% of the participants had no professional negotiation experience might therefore have influenced the results.

Although the participants were asked to negotiate as if it was a real negotiation, a limiting factor could be the fact that it was a simulation instead of a real negotiation. There was no reward for the participants; this could have affected their motivation to strive for a specific negotiation outcome.

Other remarks made by the participants were that they had limited information and context about the employee and its work, which hampered their ability to make appropriate choices. Also, the short negotiation length and bidding options were sometimes experienced as a restriction. These factors will need to be further addressed in the future.

References

1. Amanatullah, E.T., Morris, M.W.: Negotiating gender roles: Gender differences in assertive negotiating are mediated by women's fear of backlash and attenuated when negotiating on behalf of others. *Journal of personality and social psychology* 98(2), 256 (2010)
2. Amanatullah, E.T., Tinsley, C.H.: Punishing female negotiators for asserting too much or not enough: Exploring why advocacy moderates backlash against assertive female negotiators. *Organizational Behavior and Human Decision Processes* 120(1), 110–122 (2013)
3. Bowles, H.R., Babcock, L., Lai, L.: Social incentives for gender differences in the propensity to initiate negotiations: Sometimes it does hurt to ask. *Organizational Behavior and Human Decision Processes* 103(1), 84–103 (2007)
4. Eckel, C., De Oliveira, A., Grossman, P.J.: Gender and negotiation in the small: are women (perceived to be) more cooperative than men? *Negotiation Journal* 24(4), 429–445 (2008)
5. Gratch, J., DeVault, D., Lucas, G.M., Marsella, S.: Negotiation as a challenge problem for virtual humans. In: *International Conference on Intelligent Virtual Agents*. pp. 201–215. Springer (2015)
6. Heilman, M.E., Wallen, A.S.: Wimpy and undeserving of respect: Penalties for men's gender-inconsistent success. *Journal of Experimental Social Psychology* 46(4), 664–667 (2010)
7. Johnson, D.W.: Effects of warmth of interaction, accuracy of understanding, and the proposal of compromises on listener's behavior. *Journal of Counseling Psychology* 18(3), 207 (1971)
8. Johnson, D.W., McCarty, K., Allen, T.: Congruent and contradictory verbal and nonverbal communications of cooperativeness and competitiveness in negotiations. *Communication Research* 3(3), 275–292 (1976)

9. Kulik, C.T., Olekalns, M.: Negotiating the gender divide lessons from the negotiation and organizational behavior literatures. *Journal of Management* 38(4), 1387–1415 (2012)
10. Mazei, J., Hüffmeier, J., Freund, P.A., Stuhlmacher, A.F., Bilke, L., Hertel, G.: A meta-analysis on gender differences in negotiation outcomes and their moderators. *Psychological Bulletin* 141(1), 85 (2015)
11. de Melo, C.M., Carnevale, P., Gratch, J.: The effect of expression of anger and happiness in computer agents on negotiations with humans. In: *The 10th International Conference on Autonomous Agents and Multiagent Systems-Volume 3*. pp. 937–944. International Foundation for Autonomous Agents and Multiagent Systems (2011)
12. Park, S., Gratch, J., Morency, L.P.: I already know your answer: Using nonverbal behaviors to predict immediate outcomes in a dyadic negotiation. In: *Proceedings of the 14th ACM international conference on Multimodal interaction*. pp. 19–22. ACM (2012)
13. Reeves, B., Nass, C.: *How people treat computers, television, and new media like real people and places*. CSLI Publications and Cambridge (1996)
14. Rudman, L.A.: Self-promotion as a risk factor for women: the costs and benefits of counterstereotypical impression management. *Journal of personality and social psychology* 74(3), 629 (1998)
15. Tinsley, C.H., Cheldelin, S.I., Schneider, A.K., Amanatullah, E.T.: Women at the bargaining table: Pitfalls and prospects. *Negotiation Journal* 25(2), 233–248 (2009)
16. Walters, A.E., Stuhlmacher, A.F., Meyer, L.L.: Gender and negotiator competitiveness: A meta-analysis. *Organizational behavior and human decision processes* 76(1), 1–29 (1998)

A Negotiation dialogue

The texts of this dialogue are partly based on the work of [2].

1. Opening offer
 - (a) It is nice that we could have this discussion today. I would propose a raise of €1.800,- per year (€150,- per month) since my results have been excellent lately.
 - (b) Thank you for the opportunity to renegotiate my salary. I am very happy with my position and glad that I can stay for another year. Would a salary raise of €1.600,- per year (€133,- per month) be possible? I think that that's a reasonable raise considering my latest results.
2. Agreement and closure
 - (a) Excellent. I was already expecting you to agree on this proposal. I look forward to work for you for another year.
 - (b) Great, thank you! I am very happy that we could agree on this. I am looking forward to work for you for another year!
3. Counteroffer 1
 - (a) Thanks for coming in today to discuss your salary and hopefully make an agreement on this today as well. The offer you are making is too high, what about €1.100,- per year?
 - (b) Thanks for coming in today to discuss your salary and hopefully make an agreement on this today as well. The offer you are making is too high, what about €975,- per year?
4. Reaction round 1
 - (a) Assertive style
 - i. That offer is insulting, it is way too low for me. I propose €1.700,- per year. You would be foolish not to seriously consider this counteroffer.
 - (b) Non-assertive style
 - i. Thank you for that offer, I am very pleased. I accept this offer and I look forward to work another year for your store.
 - ii. Thank you for that offer but I was hoping to earn more. I propose €1.500,- per year.
5. Closing negotiation
 - (a) I think you are unbelievable to quit the negotiations at this point. I hope that you will reconsider my offers because I am not sure whether I will stay under this conditions.

- (b) I am sorry that you consider this offer a deal breaker. I hope that in the future you are willing to renegotiate about my salary since I really wish to continue working for you.
- 6. Counteroffer 2
 - (a) No, that is not a good offer. How about €1.300,- per year? That seems fair.
 - (b) No, that is not a good offer. How about €1.150,- per year? That seems fair.
- 7. Reaction round 2
 - (a) Assertive style
 - i. I don't think that's fair at all. Frankly, I am shocked that you would offer me so little. I have other offers that are looking much more desirable right now. How about €1.600,-?
 - (b) Non-assertive style
 - i. Thank you for that offer, I am very pleased. I accept this offer and I look forward to work another year for your store.
 - ii. No. I appreciate your offer but unfortunately I just cannot accept it. How about €1.425,-? This seems like a reasonable compromise.
- 8. Counteroffer 3
 - (a) Well, that's not acceptable for us. We could do €1.400,-.
 - (b) Well, that's not acceptable for us. We could do €1.240,-.
- 9. Final round
 - (a) Assertive style
 - i. I can't agree to that. Your offer is unreasonable; I'd refuse to work for so little. There is no way you can possibly expect me to work for less than €1.500,-.
 - (b) Non-assertive style
 - i. Thank you for that offer, I am very pleased. I accept this offer and I look forward to work another year for your store.
 - ii. I think we are making some progress, but unfortunately I still can't accept it. I was hoping you might find a salary of €1.375,- reasonable.
- 10. Close negotiation (run out of rounds)
 - (a) I am sorry that we ran out of time for this negotiation. It's a shame that you did not accept any of my offers. Can we schedule another meeting?
 - (b) I am sorry that we ran out of time for this negotiation. I hope that we can have another meeting at another moment. Thank you for your time today.

B Questionnaire

Questions indicated in bold are used for the subjective measures as shown in Figure 4.

1. In what 'round' did the negotiation end? [Open question - Mandatory]
2. What was the 'outcome' of the negotiation? [Open question - Mandatory]
3. What is your gender? [Male/Female - Mandatory]
4. What is your age? [Open question - Mandatory]
5. Do you have any professional experience with salary negotiations in the role of manager? [Yes/No - Mandatory]
6. How do you feel about the negotiation in general? [Likert: 1=Very negative - 7=Very positive - Mandatory]
7. Can you specify your feelings about the negotiation in more detail? [Open question]
8. Are you satisfied with the reached outcome? [Likert: 1=Completely unsatisfied - 7=Completely satisfied - Mandatory]
9. Can you specify your feelings about the reached outcome in more detail? [Open question]
10. Do you think that in a real negotiation you would have given the same responses to the employee? [Yes/No/Other .. - Mandatory]
11. Describe the virtual employee in one word: [Open question - Mandatory]
12. How realistic would you rate the appearance of the virtual employee? [Likert: 1=Not realistic at all - 7=Very realistic - Mandatory]
13. How human did the virtual employee sound? [Likert: 1=Not human at all - 7=Very human - Mandatory]
14. **How friendly do you think the virtual employee was?** [Likert: 1=Not friendly at all - 7=Very friendly - Mandatory]
15. **How arrogant do you think the virtual employee was?** [Likert: 1=Not arrogant at all - 7=Very arrogant - Mandatory]
16. **How sensitive to others do you think the virtual employee was?** [Likert: 1=Not sensitive at all - 7=Very sensitive - Mandatory]
17. **How independent do you think the virtual employee was?** [Likert: 1=Not independent at all - 7=Very independent - Mandatory]
18. **How appropriate was the language used by the virtual employee?** [Likert: 1=Not appropriate at all - 7=Very appropriate - Mandatory]
19. Can you specify your feelings about the virtual employee in more detail? [Open question]
20. What do you think this research is about? [Open question - Mandatory]

Distracted in a Demanding Task: A Classification Study with Artificial Neural Networks

Stefan Huijser¹, Niels Taatgen, Marieke van Vugt

Institute of Artificial Intelligence, University of Groningen
Nijenborgh 9, 9747 AG Groningen, Netherlands

¹s.huijser@rug.nl

Abstract. An important issue in cognitive science research is to know what your subjects are thinking about. In this paper, we trained multiple Artificial Neural Network (ANN) classifiers to predict whether subjects' thoughts were focused on the task (i.e., *on-task*) or if they were distracted (i.e., *distracted thought*), based on recorded eye-tracking features and task performance. Novel in this study is that we used data from a demanding spatial complex working memory task. The results of this study showed that we could classify on-task vs. distracted thought with an average of 60% accuracy. Task performance was found to be the strongest predictor of distracted thought. Eye-tracking features (e.g., pupil size, blink duration, fixation duration) were found to be much less predictive. Recent literature showed potential for eye-tracking features, but this study suggests that the nature of the task can greatly affect this potential. Rehearsal effort based on eye-movement behavior was found to be the most promising eye-tracking feature. Although speculative, we argue that eye-movement features are independent of the content of distracted thought and may therefore provide a more generic feature for classifying distracted thought.

Keywords: Distracted Thought, Mind-wandering, Demanding Task, Artificial Neural Networks.

1 Introduction

One important challenge cognitive scientists face in their research is to determine what someone is thinking about: Are my subjects doing the task, how are they solving it, have they wandered off? In particular the latter question is challenging. Knowing whether someone is distracted is difficult, as it is usually not accompanied with (outwardly) observable behavior. Researchers that study mind-wandering, a self-generated and task-unrelated thought process [1], have commonly solved this problem by using self-report. This method can give you a peak into the richness of thought. Although shown to produce valid results [2], it is hard to implement them correctly [3]. Furthermore, the discrete nature of the method leaves the researcher oblivious to the dynamics of the thought process. For these reasons, mind-wandering researchers have explored alternative methods to measure mind-wandering.

One candidate method is to infer the characteristics of distracted vs. non-distracted

thought from eye-tracking measurements. Distracted thought refers here to all thinking that is irrelevant to performing the task at its time of occurrence. Eye-tracking recordings are often conducted in conjunction with self-report, allowing researchers to track thought throughout the experiment. Examples of such research have identified that mind-wandering and inattentiveness (i.e., my mind is ‘blank’) are related to lower phasic and tonic pupil activity [4–6]. Studies with reading tasks have found that mind-wandering results in longer and more frequent fixations [7]. Also, it has been shown that increased and more prolonged blinking is correlated with mind-wandering [8]. Taken together, these studies show that distracted thought such as mind-wandering results in measurable patterns in eye-tracking features.

The successes in associating features from eye-tracking recordings to mind-wandering raised the question whether such features are also predictive of mind-wandering, or distracting thought in general, on a single-trial level. Recently, a study by Grandchamp and colleagues [8] explored this question by training a support vector machine classifier to predict distracted vs. non-distracted thinking with a collection of eye-tracking features. The researchers reached a classification accuracy of 80%. Noticeably, pupil size was found to be the most reliable predictor, with low average pupil size being indicative of distracted thought. In conclusion, this study showed that eye-tracking features, with pupil size in particular, could be used to classify distracted thinking in our subjects.

In this paper, we will also examine how eye-tracking related features are related to distracted thought, by training an artificial neural network classifier to predict if subjects were performing the task (i.e., *on-task thought*, OTT) or experienced *distracted thought* (DT). Novel in this study is that we will use data from a relatively demanding task, whereas previous classification studies only used simple task paradigms. In addition, we will consider both eye-tracking related features and trial-to-trial task performance in classifying on-task and distracted thought examples. Including both will allow us to compare the contribution of eye-tracking features, which can be measured continuously, with task performance measured only on a trial-to-trial basis.

2 Methods

The data set we use in this study was collected in a previous eye-tracking experiment in which subjects performed a working memory task and occasionally reported if their recent thought was on-task or distracted. We refer to Huijser, van Vugt, and Taatgen [9] for full details on the task, materials, and data acquisition.

2.1 Subjects

In total 38 individuals agreed to participate in the experiment. All subjects were native Dutch speakers and had normal or corrected-to-normal vision. We excluded the data of 5 subjects due to excessive data loss in the eye-tracking recordings. In addition, we removed the data of one extra subject, as this subject did not perform the task as required. This left us with 32 subjects for analysis and classification. All subjects

signed an informed consent prior to the start of the experiment and received a small monetary compensation when finished.

2.2 Task

The task performed by the subjects was a spatial complex working memory (SCWM) task. This task required subjects to memorize a sequence of targeted locations in a 4x4 grid, while also performing a processing subtask in between each presented location. On every trial, we first presented a storage target for 1 second. This allowed the subjects to encode the location of the target. Following this storage phase, a 4 seconds self-paced processing phase started, in which subjects made binary decisions (i.e., yes/no) on word stimuli presented in the same grid. The words moved every second to a random but different position to prevent visual rehearsal of the storage targets and to keep subjects engaged in the task¹. Following the processing phase, the grid was emptied for 2 seconds, allowing subjects to rehearse or to potentially get distracted. The eye-tracking recordings during this ‘blank’ phase were used to classify examples as on-task thought or distracted thought (see section *Thought-probe* below).

Storage, processing, and blank phases were repeated a number of times equal to a span of three or four. The experiment included 96 trials, with half of the trials being of span 3 and the other half span 4.

2.3 Thought-probe

To determine whether subjects experienced on-task or distracted thought on a particular trial, we sampled recent thought content with thought-probes. Thought-probes are self-report questions aimed at assessing recent or current conscious experience, and were conducted on random but equally distributed moments in the experiment (i.e., half of the trials, resulting in 48 thought-probe trials).

In this experiment, we used an adapted version of a thought-probe question introduced by Stawarczyk, Majerus, Maj, Van der Linden, and D’Argembeau [10] and Unsworth & Robison [4]. The question was (translated from Dutch), ‘What were you thinking about before you were prompted to answer?’, with the following response options: (1) I tried to remember the location of the X’s; (2) I was still thinking about the words from the decision task (= processing task); (3) I was evaluating aspects of the task (e.g., my performance, how long it takes, difficulty of the task); (4) I was distracted by my environment (sound/ temperature etc.) or by my physical state (hungry/thirsty); (5) I was mind-wandering/ I thought about task unrelated things, (6) I was not paying attention, but I did not think about anything specific. Response option 1 was labeled as *on-task thought* (OTT). The remaining options together were labeled as *distracted thought* (DT). Response option 3,4, and 6, were labeled as *task-related*

¹ The experiment of Huijser and colleagues [9] involved two conditions in the processing phase. These conditions used different word stimuli. As we were not interested in the conditions, we collapsed trials of both conditions in this study. Therefore, we do not discuss the two conditions here.

interference, *external distraction*, and *inattentiveness* respectively [see also 10]. We labeled option 2 as *mental elaboration* and option 5 as *mind-wandering* [see also 9].

2.4 Eye-tracking measurement

Equipment. An Eyelink 1000 eye-tracker system from SR Research was used to sample pupil size (in arbitrary units) and gaze position (X-, Y-coordinates in pixels) at a rate of 250 Hz. Blink events were automatically detected by the Eyelink software.

Preprocessing. Before training the classifier, we first performed a selection of pre-processing steps on the eye-tracking recordings. First of all, we discarded the pupil size and gaze position data for all automatically detected blinks including 100 ms before and after the blink events. In addition, we removed remaining artifacts by discarding sudden downward jumps (>200 units in pupil size, approx. $0.5 SD$). Thereafter, we linearly interpolated the resulting missing data and subsequently downsampled the full data set to 100 Hz.

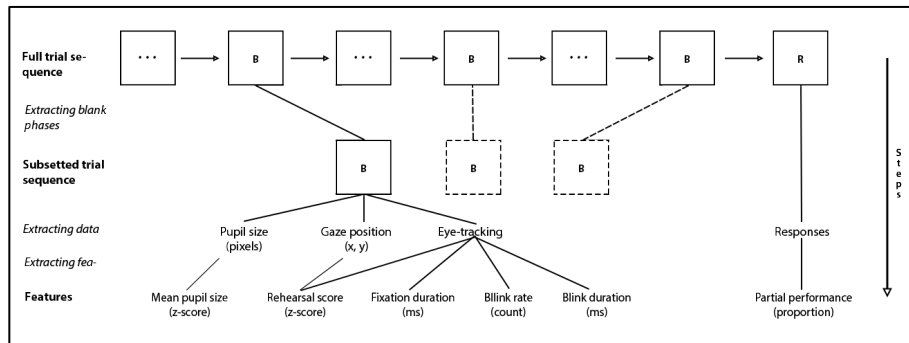


Fig. 1. Overview of the feature extraction procedure in a span 3 trial. To extract the features from the data set, we collected the responses and selected all the blank phases (i.e., thereby discarding the storage and processing phases). The recorded data in each blank phase and the collected response were subsequently used to compute the features. Square boxes with dots refer to the storage and processing phases. The square boxes with B refer to blank phases.

2.5 Classifier

We trained an artificial neural network (ANN) classifier [see e.g., 11] on the recorded eye-tracking and task performance data to track whether subjects were on-task or distracted. From the data we only selected the blank phases (see Figure 1) as input for the ANN, because blank phases were assumed to involve most distracted thinking. As only half of the trials were followed by a thought-probe we solely considered these trials for analysis and classification.

Table 1. Correlations between the features. Bold values represent significant correlations ($p < 0.05$).

	PS	RE	FD	BR	BD	PP
PS	1	-0.08	0.03	-0.09	-0.13	-0.03
RE	-0.08	1	0.06	0.05	0.03	0.10
FD	0.03	-0.24	1	-0.12	-0.08	0.01
BR	-0.09	0.05	-0.12	1	0.60	-0.05
BD	-0.13	0.03	-0.08	0.60	1	-0.04
PP	-0.03	0.10	0.01	-0.05	-0.04	1

Extracting features. As input for the ANN, we extracted six features from the collected data: *mean pupil size* (PS), *rehearsal effort* (RE), *fixation duration* (FD), *blink duration* (BD), *blink rate* (BR), and *partial performance* (PP) on a single trial (i.e., percent correct). All features were z-score standardized ($\mu = 0$, $\sigma = 1$) by-subject prior to training the network. We decided to exclude blink rate from the classification analysis because it was correlated with blink duration ($r = 0.60$; see Table 1). We favored blink duration over blink rate because its values were more normally distributed. For details on how we computed the features we refer to the sections below. An overview of the feature extraction procedure can be found in Figure 1.

Mean pupil size. We first made sure that each blank phase was equally long by removing all pupil size values after 2 seconds from the start of the blank phase. The mean pupil size was subsequently computed for each blank phase by averaging the pupil size time series in each blank phase. The pupil size time series used for this calculation were corrected for gaze position [see 11].

Rehearsal effort. We determined rehearsal effort for each blank phase by counting the number of correct fixations in each blank. Correct fixations were defined as fixations on locations where previously in the trial a storage target was shown. To account for within-trial differences, we divided the number of correct fixations by the total amount of fixations in each blank phase. The resulting score was z-score standardized by-subject.

Fixation duration & Blink duration. The duration of fixations and blinks were determined by calculating the difference between the offset and onset of fixations and blinks in a blank phase. The software of the eye-tracker provided the time stamps for the onset and offset. When the end of a fixation or blink fell outside of the blank phase period, we did not cut off the duration but still regarded the full fixation or blink duration.

Blink rate. We computed the blink rate by counting the number of blink onsets in each blank phase.

Partial performance. The partial performance was calculated by counting the number of correct responses in each trial and dividing that by the span of the trial.

Algorithm. In this study, we created two ‘types’ of ANNs: full and lesioned. The full network received input from all features and therefore included five input neurons $\{x_i \mid x_1, \dots, x_5\}$. Lesioned networks excluded one of the features and were designed to determine the contribution of a feature to classification by comparing a lesioned network to the full network. We created in total five lesioned networks, each excluding one of the features, and each network contained four input neurons. All networks had a single hidden layer with four neurons and one output neuron. We did try networks with two hidden layers and with different amounts of neurons. However, performance on the more complex ANNs did not result in significant improvement of classification performance. Therefore, we chose to the simpler network with one hidden layer over the two-layered alternatives.

All hidden and output neurons in full and lesioned networks used a *logistic (sigmoidal) activation* function. The logistic function takes input ‘x’ and transforms it into values between 0 and 1. We chose to use logistic function as it has been shown to perform well in the context of binary classification [see 12].

$$f(x) = \frac{1}{1 - e^{-x}} \quad (1)$$

As error function we applied *binary cross-entropy*,

$$E(X, Y) = -\frac{1}{n} \sum_{i=1}^n y_i \log o(x_i) + (1 - y_i) \log (1 - o(x_i)) \quad (2)$$

where ‘n’ is the total amount of training examples, $Y = \{y_1, \dots, y_n\}$ the true target labels (0 or 1), $X = \{x_1, \dots, x_n\}$ the input, and $o(x)$ the output of the network (between 0 and 1). When the output of the network is close to the real target label, the resulting error from the function is close to zero. The rationale of choosing cross-entropy as error function is that it take the binary characteristics of the data into account and therefore provides a more optimal solution compared to other error functions such as mean squared error [see e.g., 13].

To optimize the weights in our networks we used *mini-batch stochastic gradient descent* with *Nesterov accelerated gradient* [15] as backpropagation algorithm,

$$v_t = \gamma v_{t-1} - \eta \frac{1}{B} \sum_{b=0}^{B-1} \frac{\partial E(W_{t-1}, \mathbf{m}_b)}{\partial W_{t-1}} \quad (3)$$

$$W_t = W_{t-1} - \gamma v_{t-1} + (1 + \gamma) v_t \quad (4)$$

where weights ‘W’ are updated for every mini-batch \mathbf{m}_b of ‘B’ training examples with a learning rate η . To improve the stability and convergence of the gradient descent, we used NAG to update the weights with respect to the direction of the previous weight updates. This was done by adding a momentum factor γv_{t-1} to the previous weight W_{t-1} , where momentum γ is a constant and v_{t-1} is the velocity of the gradient in the previous iteration t . To understand NAG, one can interpret the gradient as a ball that runs down a hill (i.e., the error/loss curve). When the ball moves in the right direction, it gains momentum. If there is a ‘bump’ in the curve (i.e., local minimum), the momentum will make sure that the ball can continue to progress. Because the movement of the gradient is calculated by predicting its path, the ball can also slow down when the slope of the curve goes up (i.e., when error is expected to increase). The result is that the ball will roll to the bottom of the curve, where the error is at its minimum (i.e., global minimum), without getting stuck in local minima and without overshooting the global minimum by rolling up the curve again.

Training and testing. Training and test sets were derived by performing *stratified 10-fold cross validation* on the full data set containing a total of 5374 examples. The resulting folds each had a training set with 4835 examples and a test set with 539.

Every network (i.e., full and lesioned) for each fold was trained independently for 500 epochs. The weights (W) were initialized at a random value between -0.1 and 0.1 and the bias was set to 1. We trained every network with a mini-batch size (B) of five and used a fixed learning rate (η) of 0.0001 and a momentum (γ) of 0.9.

Target labels for on-task thought examples were coded as 1, distracted thought examples were coded as 0. Because the network outputs probability values, we interpreted output values smaller than 0.51 as a prediction for class 0, and values equal or greater than 0.51 as a prediction for class 1. We chose 0.51 as threshold, because 51% of the examples in our data set were on-task.

3 Results

3.1 Logistic regression – how predictive are our features?

Before turning to the ANNs, we first wanted to check how predictive the individual features are of on-task vs. distracted thought. We fitted our features (excluding blink rate) as fixed effects on a logistic regression model with our subjects as random intercepts. Test results of this model are shown in Table 2.

We found that trial-to-trial partial performance (PP), mean pupil size (PS), rehearsal effort (RE), and blink duration (BD) were significant predictors of being on-task or distracted in the blank phase (all $p < 0.05$). The length of fixations (FD) was not significant, suggesting that fixation duration is indiscriminant of on-task and distracted states. While an increase in performance, pupil size, and rehearsal effort was found to be predictive of on-task thought, longer blinks were associated with being distracted. Partial performance was the strongest predictor, with one standard deviation (SD) increase in performance making it 1.77 times more likely to be on-task. One SD in-

crease in mean pupil size made it 1.18 times more likely to be on-task, and 1.09 times for rehearsal effort. To put these numbers in perspective, if the chance of being on-task is 50 percent, performing one *SD* better makes this chance $50 * 1.77 = 88.5$ percent. For pupil size, rehearsal effort, and blink duration, this is 59, 54.5, and 46.5 percent respectively. Therefore we argue that partial performance is the most practically significant predictor. Although much less, eye-tracking features are also predictive for on-task vs. distracted thought.

3.2 ANN results – how well can we predict on-task and distracted states?

Now we know how predictive our features are, we can look at how the ANNs performed. All the important statistics are described in Table 3.

We found that all ANN classifiers, except for the lesioned network excluding partial performance, were able to predict on-task vs. distracted thought above chance at around 60%. Noticeably, the differences in classification performance between the full network and lesion networks excluding eye-tracking features were only very small. As the logistic regression model already showed that the influence of the eye-tracking features was small, this suggests that the contribution of eye-tracking features to the ANN classifier was minimal.

We observed a drop in classification accuracy when partial performance was excluded (i.e., L. PP classifier). The accuracy of the L. PP classifier was just above chance at 52.87%, and had an area under the receiver operating characteristic curve (AUC) of 0.539. The AUC is a measure of diagnostic ability and takes the ratio between true and false positives into account. An AUC of 0.539 means that amount of true and false positives was very similar, suggesting that the classifier did not learn the problem and potentially performed random guessing. More detail on the behavior of the classifier can be found in Figure 2 (right). Here, we see that the classifier made similar predictions on both on-task and distraction examples. Most of the output values were centered at 0.48, which happens to be the probability of encountering an distracted thought example (i.e., 0.487). Therefore, the classifier did not learn to classify on-task and distraction based on the data. Instead, it learned the underlying probability of distracted thought examples.

Table 2. Results from the logistic regression model with subjects as random intercept. Features are interpreted as significant when $p < 0.05$. Values in the e^β column are interpreted as “ x times more likely with one unit increase in *feature*.”

Features	β	<i>SE</i>	e^β	<i>z</i>	<i>p</i>
PS	0.169	0.035	1.18	4.840	<0.001
RE	0.086	0.031	1.09	2.759	0.01
FD	0.028	0.031	1.03	0.887	0.37
BD	-0.069	0.031	0.93	-2.198	0.03
PP	0.573	0.035	1.77	16.441	<0.001

Table 3. Classification performance, collapsed over folds, of full and lesioned ANNs after training for 500 epochs. We reported the overall accuracy (acc, in %), area under the ROC curve (AUC), sensitivity to predicting distracted thought (DT), and specificity (i.e., accuracy of predicting on-task thought (OTT) on where DT is the true label). Full = all features. L. = lesioned.

	<i>Overall acc (%)</i>	<i>AUC</i>	<i>OTT acc (%)</i>	<i>DT acc (%)</i>
Full	60.31	0.623	53.26	67.76
L. PS	60.87	0.621	51.81	70.44
L. RE	60.61	0.622	52.35	69.33
L. BD	60.70	0.623	53.22	68.61
L. FD	60.48	0.623	52.79	68.61
L. PP	52.87	0.539	77.30	27.03

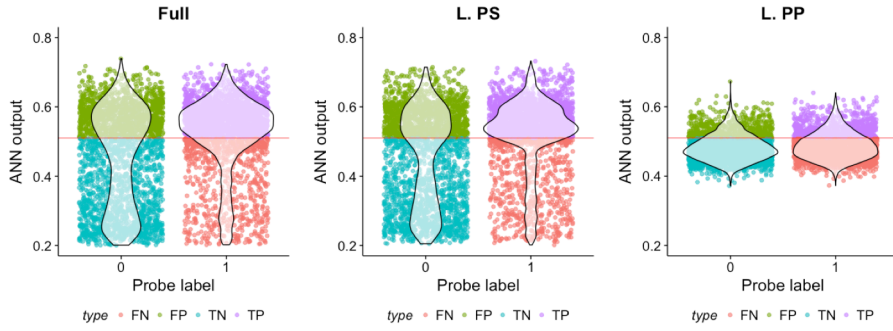


Fig. 2. The figures above show all output values and its density (i.e., shaded area) of the full ANN (left), the pupil size lesioned ANN (L. PS), and the trial performance lesioned ANN (L. PP). The value of the prediction is displayed on the y-axis, and the true probe labels are displayed on the x-axis. The red horizontal line shows the threshold used to classify a predicted value as on-task (value ≥ 0.51) or distracted (value < 0.51). The colors of the predictions show if the prediction is a true positive (TP; purple), true negative (TN; blue), false positive (FP, green), or a false negative (FN; red).

When we look in more detail at the classification performance of the full and eye-tracking lesioned classifiers, we found that the accuracy on on-task thought examples was much better compared to distracted thought examples. Accuracy on on-task examples was on average 69.10%, while for distracted thought examples it was only just above chance at 52.85% (see Table 3).

Now knowing that classification performance on distraction was poor, we wanted to examine why this was the case. We turned to the raw thought-probe data, and assessed the accuracy of the classifiers on each distraction response category (see Table 4). We observed that performance of the full and eye-tracking lesioned classifiers was below chance on examples where subjects engaged in mental elaboration (47.95% on

average) and in task-related, but interfering thought (44.31% on average). On the other hand, performance on the remaining distraction categories (i.e., external distraction, mind-wandering, and inattentiveness) was relatively good with accuracies up to 70%. It is likely that the classifier confused mental elaboration and task-related interference examples for on-task examples. This makes sense, because the thought content in these distraction categories is more closely related to the task in comparison to the other categories.

Table 4. Overview of the classification accuracy on different thought-probe categories (in columns) for all ANN classifiers (in rows). The bottom row ‘n’ gives the total amount of examples for each category in the full data set.

	On-task	Mental elaboration	Task-related interference	External distraction	Mind-wandering	Inattentiveness
Full	67.76	48.43	45.04	62.72	65.15	69.63
L. PS	70.44	47.19	42.83	61.82	63.00	69.63
L. RE	69.33	47.52	44.10	62.42	63.81	68.69
L. BD	68.61	48.51	44.88	64.42	63.81	69.16
L. FD	68.61	48.10	44.72	61.81	64.61	68.69
L. PP	27.03	74.63	74.02	79.10	86.33	83.64
n	2612	1210	635	330	373	214

3.3 ANN results – individual differences

The last thing we examined was the performance of the full ANN classifier on the different subjects in this study. We found that accuracy was above chance (accuracy > 0.51) for most of the subjects ($n = 26$, total = 32, 81.3%), and relatively good (accuracy > 0.7) for 21.8% of the subjects ($n = 7$). Notwithstanding, we found accuracies below chance of a handful, but substantial number of subjects ($n = 6$, 18.8%).

As performance of the classifier differed between subjects, we explored these differences in more detail. First, we examined whether the ratio of on-task thought/distracted thought reports of each subject correlated with accuracy. Because the classifier performed better on on-task examples, it might be that differences in the amount of on-task reports impacted the performance of the classifier. We found a small correlation ($r = 0.21$), but this was not significant ($p = 0.25$). Therefore, there was likely no relationship between the ratio of on-task/distracted reports and the performance of the classifier.

Second and last, we investigated if individual differences in the features were related to changes in classification accuracy. We calculated the mean value of each feature (excluding blink rate) for on-task and distracted thought examples for each subject. Subsequently, we calculated a difference score by subtracting the mean feature value of distracted thought examples from the on-task examples. Correlating these difference scores with classification accuracy revealed that differences in partial performance were strongly correlated with classification accuracy ($r = 0.79$, $p < 0.001$). In addition, we found a moderate positive correlation for rehearsal effort ($r =$

0.35, $p = 0.045$). Therefore, it is likely that the individual differences in classification performance were caused by variation in partial performance between on-task and distracted thought examples, and to a smaller extent by differences in rehearsal effort. Surprisingly, we found no evidence of a relationship between classification accuracy and across subject differences in pupil size ($r = 0.03$, $p = 0.86$). This was unexpected, because previous research [8] and our logistic regression model showed that pupil size was a reliable predictor. The ANN classifier was therefore inconsistent with this work.

4 Discussion

The aim of this study was to classify on-task and distracted thought examples with eye-tracking related and performance related features from a demanding task. The results showed that we were able to classify on-task and distracted thought above chance (60% accuracy) on the basis of input from both eye-tracking (i.e., mean pupil size, rehearsal, blink duration, and fixation duration) and performance features (i.e., trial-to-trial partial performance). Noticeably, classification performance dropped to chance level when trial-to-trial partial performance was excluded from training and testing. This suggests that performance of our full network, receiving input from all features, was largely carried by partial performance. We conclude that the eye-tracking features only contributed little to the performance of the classifiers.

On-task thought was easier to classify than distracted thought. Detailed investigation of the classifiers' predictions on individual distracted thought categories showed that performance had stark differences. Whereas performance on mind-wandering, external distraction, and inattention examples was relatively good (i.e., accuracy > 0.6), performance on mental elaboration and task-related interference was poor (i.e., accuracy < 0.5). It is likely that the latter distracted thought categories were confused for on-task examples. Taking into account that they were also the most prevalent distracted thought categories, it is implied that they contributed to the low performance on distracted thought examples.

Exploring the performance of the full classifier on the individual subjects showed that accuracy was above chance for the majority of subjects (i.e., 81%), and above 70% accuracy for a substantial group of subjects (i.e., 25%). We found that individual differences in the decrement (i.e., on-task – distracted thought) in partial performance and rehearsal effort were correlated with the differences in classification accuracy. Small decrements in both features were associated with chance level or below chance level performance. Importantly, we found no such relationship between the ratio of on-task and distracted thought examples in individual subjects and the classification performance. This means that the accuracy of the classifier on individual subjects can only be explained by differences in the feature data. From these results, we conclude that partial performance, and rehearsal effort to a lesser extent, contributed to successful classification.

Comparing the results of this study to other work is difficult, since there are only few mind-wandering classification studies, and these studies have used different feature sets and methodologies. Nevertheless, comparisons can be made on parts of the

results. We found that our classifier underperformed in comparison to other work. Other classification studies in the literature have reported average or median classification accuracies of 72 to 81% [see 6,8,15]. It should be noted that these studies did not include behavioral features such as trial-to-trial performance, but instead used only eye-tracking features [8] or fMRI-based features in combination with eye-tracking [6]. Both studies found that eye-tracking features contributed significantly to classification performance. Although it is difficult to determine why the present study did not identify such a contribution, it is possible that the challenging and dynamic nature of our task caused more noise and variance in the eye-tracking measures, and therefore made it more difficult to learn a good decision boundary.

Another possible explanation for the relatively weak contribution of eye-tracking features is that the content of distracted thought in our study was different from the other mentioned classification studies. Most subjects reported experiencing distracting thoughts as a result of the stimuli from the processing task (i.e., mental elaboration) in our study. In previous work using the same data set [9], we showed that this category of thought was not associated with a different mean pupil size compared to on-task. Similar results were found for task-related interference, while typical mind-wandering categories (i.e., mind-wandering and inattentiveness) were associated with a significantly lower mean pupil size [4]. Mind-wandering categories, however, only constituted 11% of all reports. Tasks with lower cognitive demand have found to involve more mind-wandering [e.g., 16], and given that the other mentioned classification studies used simpler tasks, this may at least explain why mean pupil size was a better predictor in their studies.

While different distracted thought categories affected mean pupil size differently, it may highlight the strength of eye-movement features. The advantage of eye-movement, such as fixations, saccades, and blinks, is that they are ubiquitous in most tasks [15]. For instance, the fixations on and saccades towards locations on the display where targets were presented (i.e., rehearsal effort) are cues that subjects rehearsed the targets in our paradigm. The absence of such eye-movement behavior means that the subject is not performing the task, and is therefore off-task/distracted. Although rehearsal effort was not significantly different from on-task for all distracted thought categories, there was a similar trend towards lower rehearsal effort visible for each distracted thought category. This highlights the potential for eye-movement features as predictors for distracted thought in general, and may also explain why rehearsal effort showed up in the individual differences analysis.

5 Conclusion and Future Directions

At the beginning of this paper we raised the research question: How are eye-tracking features and task performance related to distracted thought? From this study we conclude that trial-to-trial performance is the strongest predictor of distracted thought. Although eye-tracking related features have shown great potential, this study seems to indicate that the nature of the task and the content of distracted thought can greatly affect this potential. Rehearsal effort based on eye-movement behavior was found to

be the most promising eye-tracking feature. Although speculative, we suggest that eye-movement features are independent of the content of distracted thought and may therefore provide a more generic feature for classifying distracted thought. An interesting avenue for future research is to explore the eye-movement data in more detail, to determine which eye-movement features are relevant for classification. For example, recent research has shown that beta-process hidden markov models are able to extract dynamic patterns from time series examples [18] Applying such methods to eye-movement data may provide more insight in which patterns in the time series determine on- or distracted thought behavior.

Acknowledgements

This research was supported by a grant from the European Research Council (MULTITASK – 283597) awarded to Niels Taatgen. We would like to thank Marco Wiering for his comments and suggestions in the early stages of this research.

References

- [1] J. Smallwood and J. W. Schooler, “The Science of Mind Wandering: Empirically Navigating the Stream of Consciousness,” *Annu. Rev. Psychol.*, vol. 66, no. 1, pp. 487–518, 2015.
- [2] J. C. McVay and M. J. Kane, “Drifting from slow to ‘D’oh!’: working memory capacity and mind wandering predict extreme reaction times and executive control errors,” *J. Exp. Psychol. Learn. Mem. Cogn.*, vol. 38, no. 3, pp. 525–549, 2012.
- [3] Y. Weinstein, “Mind-wandering, how do I measure thee with probes? Let me count the ways,” *Behav. Res. Methods*, no. December, 2017.
- [4] N. Unsworth and M. K. Robison, “Pupillary correlates of lapses of sustained attention,” *Cogn. Affect. Behav. Neurosci.*, vol. 16, no. 4, pp. 601–615, 2016.
- [5] J. Smallwood, K. S. Brown, C. Tipper, B. Giesbrecht, M. S. Franklin, M. D. Mrazek, J. M. Carlson, and J. W. Schooler, “Pupillometric evidence for the decoupling of attention from perceptual input during offline thought,” *PLoS One*, vol. 6, no. 3, 2011.
- [6] M. Mittner, W. Boekel, a. M. Tucker, B. M. Turner, A. Heathcote, and B. U. Forstmann, “When the Brain Takes a Break: A Model-Based Analysis of Mind Wandering,” *J. Neurosci.*, vol. 34, no. July, pp. 16286–16295, 2014.
- [7] T. Foulsham, J. Farley, and A. Kingstone, “Mind wandering in sentence reading: decoupling the link between mind and eye,” *Can. J. Exp. Psychol.*, vol. 67, no. 1, pp. 51–9, 2013.
- [8] R. Grandchamp, C. Braboszcz, and A. Delorme, “Oculometric variations during mind wandering,” *Front. Psychol.*, vol. 5, no. FEB, 2014.
- [9] S. Huijser, M. K. van Vugt, and N. A. Taatgen, *The Wandering Self: Tracking the Mind Wandering State in a Complex Working Memory Task*. Manuscript submitted for publication, 2017.

- [10] D. Stawarczyk, S. Majerus, M. Maj, M. Van der Linden, and A. D'Argembeau, "Mind-wandering: Phenomenology and function as assessed with a novel experience sampling method," *Acta Psychol. (Amst.)*, vol. 136, no. 3, pp. 370–381, 2011.
- [11] J. Brisson, M. Mainville, D. Mailloux, C. Beaulieu, J. Serres, and S. Sirois, "Pupil diameter measurement errors as a function of gaze direction in corneal reflection eyetrackers.," *Behav. Res. Methods*, vol. 45, no. 4, pp. 1322–1331, 2013.
- [12] M. Minsky and S. Papert, *Perceptrons*. Oxford, England: M.I.T. Press, 1969.
- [13] S. Dreiseitl and L. Ohno-Machado, "Logistic regression and artificial neural network classification models: A methodology review," *J. Biomed. Inform.*, vol. 35, no. 5–6, pp. 352–359, 2002.
- [14] L. M. Silva, J. Marques de Sá, and L. A. Alexandre, "Data classification with multilayer perceptrons using a generalized error function," *Neural Networks*, vol. 21, no. 9, pp. 1302–1310, 2008.
- [15] Y. Nesterov, "A method for solving a convex programming problem with convergence rate $O(1/k^2)$," *Sov. Math. Dokl.*, vol. 27, no. 2, pp. 372–376, 1983.
- [16] R. Bixler and S. D'Mello, "Toward Fully Automated Person-Independent Detection of Mind-Wandering," in *International Conference on User Modeling, Adaptation, and Personalization*, 2014, pp. 37–48.
- [17] J. Rummel and C. D. Boywitt, "Controlling the stream of thought: Working memory capacity predicts adjustment of mind-wandering to situational demands," *Psychon. Bull. Rev.*, vol. 21, no. 5, pp. 1309–1315, 2014.
- [18] E. B. Fox, M. C. Hughes, E. B. Sudderth, and M. I. Jordan, "Joint modeling of multiple time series via the beta process with application to motion capture segmentation," *Ann. Appl. Stat.*, vol. 8, no. 3, pp. 1281–1313, 2014.

Assessing the Spatiotemporal Relation between Twitter Data and Violent Crime

Marco Stam¹, Charlotte Gerritsen², Ward van Breda³, and Elias Krainski⁴

¹ Department of Economics, Leiden University, The Netherlands

² Netherlands Institute for the Study of Crime and Law Enforcement, The Netherlands

³ Sentimentics bv, The Netherlands

⁴ Departamento de Estatística, Universidade Federal do Paraná

Abstract. Social media have grown over the past decade to become an important factor in the way people share information. This paper explores the feasibility of using Twitter data for predictive policing models within the city of Amsterdam, the Netherlands. A novel, powerful approach to Bayesian inference is used to assess the correlations between the spatiotemporal distributions of tweets containing predefined keywords and that of violent crime incidents. Spatiotemporal log-Gaussian Cox processes are considered using the stochastic partial differential equations (SPDE) approach for space correlated over time with autoregressive dynamics and fitted using the integrated nested Laplace approximations (INLA) method. The findings show that the occurrence of such tweets raises the probability of incidents occurring nearby in space-time. This novel insight has unveiled the promising potential of Twitter-based predictive policing models within the city of Amsterdam.

1 Introduction

Social media usage has grown at an astonishing pace over the past decade, with users sharing information ranging from news stories to their personal mood states. Having been launched in 2006, Twitter is one of the most popular social media platforms. Its approximately 320 million monthly active users generate more than 500 million messages daily, providing a powerful insight into the public's sentiment and activities (<https://about.twitter.com/company>). As such, its usefulness as a data source for various stakeholders in fields that rely on human factors is drawing great attention. An advantage of Twitter is its (partially) open nature, with freely available application programming interfaces (APIs) and high amounts of publicly available messages, which makes it a very accessible source for data mining.

The main focus of this paper is the potential of Twitter for issues related to the field of criminology. The human factor in crime leads to the assumption that insights into people's activities, preoccupations and mood states can be useful for monitoring, predicting and preventing various types of crimes. The potential of social media such as Twitter has therefore not escaped the attention of researchers within the field of criminology, especially of those focused on 'predictive policing', which encompasses the usage of statistical analysis of data to anticipate and prevent future crime or to

respond more adequately to it [16]. In general this enables a more effective and efficient allocation of the often limited resources of law enforcement agencies compared to traditional policing approaches. This is mainly done through analysis of the spatial or spatiotemporal distributions of crime. By assessing the risks of various types of crimes in certain geographical areas, law enforcement can allocate its resources accordingly. A limitation however of traditional predictive policing models is their dependency on historical crime data [29]. These models are therefore locally descriptive and lack the possibility of implementation in other geographical areas. Models based on data derived from social media sources can overcome this lack of portability, as they are not based on specific geographical attributes.

This paper aims to explore the potential of Twitter-based predictive policing models within the city of Amsterdam, the Netherlands. To this end, a novel approach to Bayesian inference is applied to approximate parameter values that describe the distributions of certain tweets and violent crime incidents in geographical space and time. This is done by fitting spatiotemporal log-Gaussian Cox processes onto the individual point processes, which is considered a good method to model the collected data and its spatial and temporal dependence [15]. The applied approach enables us to subsequently correlate the spatiotemporal distributions of the point processes with each other.

2 Related Work

The potential of Twitter as a data source has not gone unnoticed by the academic community. A sizeable body of research surrounding Twitter has developed in areas such as computer science, economics, sociology and criminology. Many of these studies focus on its predictive power, covering a broad range of future events. By monitoring the public's general mood state via Twitter it has been shown possible to predict the up- or downward trend in the closing values of the Dow Jones Industrial Average of the following day [5, 31].

Other research has shown the ability of Twitter-based models to accurately predict phenomena such as the development of an influenza epidemic, civil unrest and box office results [1, 2, 3, 27]. The accuracy of these models trumped that of the traditionally employed methods of prediction and monitoring, respectively those of the Centers for Disease Control and Prevention and the Hollywood Stock Exchange [1, 3]. A prime example of the important role that social media fulfills can be found in the "Arabic Spring" uprisings in the Middle East and North Africa, which were largely facilitated by the communicational means offered by social media platforms [9]. Analyzing the activity on Twitter showed that changes therein preceded certain incidents such as mass protests. The general implication of these researches is that Twitter provides a useful insight into phenomena that contain human factors.

The body of research surrounding Twitter within the Netherlands is considerably smaller. However, there is some work confirming the predictive potential of Dutch Twitter corpora as well. By analyzing the content of Dutch tweets it has been proven possible to predict the activities its senders were likely to engage in later on that day [30]. Analysis of the prevalence of party names and accompanying sentiment in Twitter messages enabled the prediction of the Dutch senate and parliament election results [18, 25]. This relatively simple approach resulted in a less accurate prediction than that of traditional polling companies.

These researches show that Twitter offers insight into phenomena related to varying academic fields. The most relevant related work for the current project however is that of Gerber [8]. This comprehensive research has assessed the feasibility of predicting 25 types of crime using Twitter. Coherent topics within the collected tweets were identified via automatic semantic analysis and subsequently used to train a predictive model. This approach was applied over the complete corpus of geotagged tweets sent from within the city of Chicago, Illinois (USA), which resulted in improved accuracy of the prediction of 19 types of crime compared to the traditional kernel density estimation approach. This includes crimes of the legal definitions of aggravated assault and battery within the state of Illinois, which correspond in part to the definition of violent crime incidents used in this research.

Although Gerber's research shows promising results for this study, they differ from each other in several ways. An important difference is that the proportion of tweets written in the Dutch language which are geotagged is lower than those written in English, with 2.9 versus 3.4 percent respectively [24]. Although this can be considered a fairly small difference, it is amplified by Chicago having a much larger population size than Amsterdam. There is also the potential that differences between the Dutch and English Twitter corpora may affect the results. Apart from the clear language difference there is the possibility that Dutch Twitter users deviate from their American counterparts in additional unforeseen ways as to the way they share information via Twitter. These differences emphasize the relevance of this research and the need to assess the potential of Twitter for predictive policing in the Netherlands.

3 Data Collection

Data have been collected over two separate time periods of 68 and 108 days, concerning the spatiotemporal distribution of two main variables within the city of Amsterdam. The first variable are tweets sent within those time periods, originating from within the city boundaries. The second main variable is violent crime incidents that have occurred over those periods within the city. Information concerning these incidents has been gathered by aggregating RSS-feeds, i.e., Rich Site Summary feeds, a collection of formats primarily designed for web syndication. Two publicly accessible web feeds were aggregated of the p2000 communications network of the emergency services within the region of Amsterdam-Amstelland. 'Incidents' are defined as "violent crime incidents in the public domain to which the police has responded". These violent crime incidents include physical assaults and assaults with the use of a deadly weapon, i.e., firearms or sharp objects. Street names are used over the p2000 communications network to indicate the locations of these incidents. The point location coordinates for the incidents were subsequently approximated as the center points of those streets by using the Google Maps API. As the region Amsterdam-Amstelland not only covers the city of Amsterdam but also adjacent towns, a subsection was made of incidents originating within Amsterdam itself. Of the 714 captured incidents, 682 were located within the city boundaries.

The tweets were gathered using the Sentimentics software, which is primarily designed for text-based sentiment analysis (<http://sentimentics.com/>). Its application in this current project was however limited to the usage of predefined keywords to filter tweets from the entire Dutch twitter corpus. To facilitate the analyses the keywords have been grouped into three 'tweet keyword clusters' (TKCs), which contain keywords respectively related to: "physical violence", "agitation/aggression" and

“substance abuse”. As the tweets are extracted from the Dutch Twitter corpus, Dutch keywords were used. The three TKCs are composed to correspond to their identically numbered sub-hypotheses. As such, TKC_1 mainly contains keywords directly related to physical violence (I), such as “fight” (vechtpartij) and “fighting” (vechten). Keywords more indirectly related to physical violence are however also incorporated in this cluster. These include keywords not directly indicating physical assault, but rather various acts indicating public disorder that are often accompanied by acts of violence, e.g., “rioting” (rellen). Keywords referring to agitated or aggressive mood states (II) were incorporated in TKC_2 , such as, “angry” (boos) and “enraged” (woedend). The third cluster (TKC_3) includes keywords related to substance abuse (III), e.g., “drunken” (dronken) and “intoxicated” (beschonken).

Only 2.9 percent of the collected tweets include geotags, resulting in a number of 5.960 geotagged tweets in dataset 1 and 6.744 geotagged tweets in dataset 2. Less than half of these tweets originated from within the municipality of Amsterdam, specifically 2.209 and 3.012 tweets for dataset 1 and 2 respectively. A large proportion of these tweets however did not originate from private individuals, but rather entities engaged in reporting about crime or police activity. Such tweets were deemed unsuitable and therefore excluded from the analysis. Data cleaning lead to a final count of 193 usable tweets over the first time period (dataset 1). The second time period over which data was collected resulted in a total count of 221 usable tweets (dataset 2).

Both the tweets as well as the incidents are spatiotemporal, random point processes that are considered to be aggregated due to a stochastic environmental heterogeneity. The most frequently applied models for such data are Cox processes, which are ‘doubly stochastic’ as they are inhomogeneous Poisson processes with a random intensity surface ($\lambda(s)$) [15]. More specifically, the collected data is to be modeled as log-Gaussian Cox processes (LGCPs), i.e. Cox processes of which the logarithm of the intensity surface ($\log(\lambda(s))$) is a Gaussian process. One of the useful properties of LGCPs is that the intensity surface and the underlying Gaussian process can be predicted using Bayesian methods.

4 Analysis

The data preparation and analyses were performed in the R software environment for statistical computing, version 3.2.2 (R Development Core Team, 2015). To adequately analyze the spatiotemporal distributions of our data a model is needed which incorporates Tobler’s ‘first law of geography’ [26]: “Everything is related to everything else, but near things are more related than distant things”. This quote is mostly aimed at spatial autocorrelation, but can be extended to correlation between multiple variables. For spatiotemporal analyses, the concept can be extended into a third, temporal dimension, i.e., that events near to each other in time are more related than events more distant in time. Accounting for these terms is possible using the R-INLA package for Bayesian computing with R (<http://r-inla.org>). This package contains a fairly new alternative for Bayesian inference to the computationally intensive Markov chain Monte Carlo (MCMC) method. This so-called integrated nested Laplace approximation (INLA) approach enables fast approximate inference for latent Gaussian models [17]. This is especially useful as MCMC methods become increasingly hard to run with the advances in data collection and modeling leading to increases in model size and complexity. Shortly summarized, the INLA approach is fast by taking advantage of the precision matrix sparsity when considering Gaussian approximations in the nested

Laplace approximations. The first Laplace approximation is used to find the parameter modal configuration. The second is considered for improving the latent field marginal distributions. In addition, efficient exploration around the mode is done when doing the numerical integration over the parameter space in order to account for uncertainty. Whereas MCMC has to explore the posterior distributions in a stochastic way, INLA does it in an analytical way [12, 14].

The R-INLA package was created to make the approach more accessible to applied researchers from various fields and contains a myriad of functions for various methods of Bayesian modeling. Particularly useful for spatial analyses are the ‘stochastic partial difference equation’ (SPDE) models. These SPDE-models provide an additional gain in computational efficiency when modeling spatial data with Gaussian fields in the Matérn class [13, 21, 22]. This is achieved by using SPDEs to link the continuously indexed Gaussian fields to the discretely indexed Gaussian Markov random fields. In doing so the continuous representation of space remains, while the computations are being performed on a discrete scale. Gaussian Markov random fields (GMRFs) have sparse precision matrices, which enable a drastic reduction in computational burden compared to the Gaussian fields dense matrices. As such, it also largely eliminates the often encountered ‘big n problem’ of Gaussian fields, which can quickly become computationally unfeasible. Removing this limitation has great potential for spatial statistics, wherein Gaussian fields play an essential role. The supported class of latent Gaussian models, which is a very broad range of models in itself, ranges from linear mixed models to spatiotemporal models [14]. The additional integration of the SPDE approach in R-INLA expands upon this further, enabling us to use R-INLA for point process models, including the spatiotemporal LGCPs applied in this research. Inference for log-Gaussian Cox processes can be performed via INLA due to the model assumption that there is a latent Gaussian Random Field [10, 20].

Bayesian inference is widely used in spatial statistics. Main factors to its popularity in this field are its ability to incorporate uncertainty and to deal with missing data. Despite its popularity, using this approach for statistical analysis with unknown parameter values is quite controversial [4]. This is mainly due to the significant effect the often arbitrarily specified prior distributions can have on the resulting posterior distributions. Accordingly, the INLA approach also approximates posterior marginal distributions for all of the model parameters. R-INLA however offers the ability to use Penalized Complexity priors, which largely negate the effect of weak prior information and subjectivity [23].

5 Model Properties

To answer our research questions, the incidents were modeled as a response variable and the tweets as covariates. At the time of this research there was no single comprehensive method for this in R as the data is misaligned, i.e., both point processes occur at different spatiotemporal locations. However, the R-INLA package offers a multistage solution due to its many general functions. One SPDE model can be considered for each of the tweets and one for the incidents, which is particularly useful for misaligned data [4, 11]. The joint model was built as a simplified coregionalization model where the tweets fields are treated as covariates for the incidents field. This approach for multivariate spatial point processes is based on Wackernagel’s linear model of coregionalization [7, 19, 28]. We have considered the spatiotemporal model in [6] implemented in a way to allow complete misalignment of the tweets and

incidents, following Section 3.2 in [12], and [11]. It only requires the data to be bounded by identically space-time observation windows [11]. The final model specification will be discussed in the next section.

6 Specification and Results

The first steps in using the SPDE approach to fit a spatiotemporal LGCP, are to specify a temporal and spatial mesh upon which the model is fitted. To accommodate the comparison of multiple point processes they must be fitted on the same meshes, bounded by one space-time observation window [11]. The applied model was fitted over both datasets separately using the same specifications. These datasets are however bounded by different temporal observation windows and therefore require individual temporal meshes. The aforementioned time knots were manually specified on the temporal meshes. Although generally limited by computational capacity, this computationally feasible maximum could not be reached due to the limited amount of tweets. To maintain a high enough number of tweets per time knot, the highest achievable amount of time knots was 12. These were applied equally spaced over the time window, resulting in relative counts of 1 time knot per 5.7 days for dataset 1 and 1 knot per 9 days for dataset 2.

The creation of the spatial mesh is done using constrained refined Delaunay triangulation by default. This method partitions the spatial area into triangles around mesh nodes in such a way that any location within it is closer to its own mesh node than to any other mesh node [11]. For geostatistical analysis it is recommended to define the mesh nodes on the event locations to increase efficiency and precision. This is however not necessary for LGCPs and would also defeat its purpose if it is defined on only one of multiple point processes that are to be fitted upon it. As all of the LGCPs need to be fitted on one spatial mesh, it was defined separately from the event locations. This is quite straightforward with R-INLA, although it requires some tuning to get the ideal specification. A finer mesh can be specified where it is preferable and conversely empty areas can be covered in a cruder mesh to save unnecessary computational costs. A shapefile of the municipality of Amsterdam was used to subset the tweets and incidents originating from within the city of Amsterdam. The spatial mesh was initially placed over the entire city, using the city borders as boundaries. The limited amount of tweets however required a reduction in spatial mesh nodes to accommodate raising the number of time knots to a level that adequately models the temporal variance. To accommodate this, a subsection of the area was made, which encompasses the vast majority of the tweets and incidents while greatly reducing the number of spatial mesh nodes. Combined with a cruder spatial mesh specification and the usage of convex hulls, the number of nodes was lowered from 948 to 62 nodes for dataset 1 and 65 nodes for dataset 2. This led to lower absolute counts of 182 tweets and 207 incidents in dataset 1 and 208 tweets and 379 incidents in dataset 2. The relative count however is higher with this specification, with 2.9 tweets and 3.3 incidents per mesh node for dataset 1 and 3.2 tweets and 5.8 incidents per node for dataset 2.

The initial model specification was to run four LGCPs separately for each of the four point processes and subsequently assess the correlations using a spatiotemporal coregionalization approach. However, to reduce the risk of oversmoothing, the temporal resolution needs to be high enough to catch enough variation. Raising the number of time knots increases the model complexity, which needs to be supported by a proportionate amount of data.

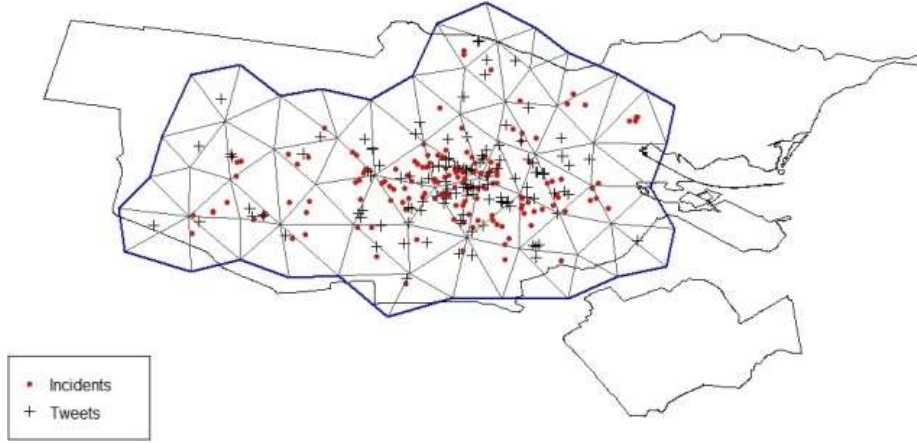


Figure 1. The final specification of the spatial mesh with 62 mesh nodes projected over the city of Amsterdam overlaid by the spatial distributions of the incidents and tweets of dataset 1.

The limited number of tweets resulted in the necessity to compensate this added complexity by reducing the number of model parameters. The coregionalization approach consists of assessing the correlation between each variate and the spatial and temporal correlation of each variate. The complexity of the algorithm for parameter estimation increases proportionately to the number of possible combinations of the model parameters, reducing the number of parameters was therefore determined to be the most effective method of model simplification. These simplifications were implemented in multiple steps. To answer our research questions the correlations between the TKCs were not of interest and therefore dropped from our model. A subsequent test run of this model specification showed that the three TKCs have identical spatiotemporal process parameters. Combined with the aforementioned assumption of conditional independence and absence of a likelihood hyper-parameter (due to the Poisson likelihood), this enabled modeling the TKCs jointly on one set of parameters. Although they now share one set of hyper-parameters, the latent field is replicated for each TKC independently leading to independent realizations for each TKC. This final simplification reduced the number of model parameters further to a final count of nine parameters. Combined with a reduced number of spatial mesh nodes, these simplifications facilitated the aforementioned final specification of temporal mesh ($k = 12$).

By jointly fitting the three TKCs, our final model specification consists of three calls to INLA in which the posteriors are approximated. All of these calls have been specified using the aforementioned Penalized Complexity priors, ensuring that the posterior probabilities are derived from the data alone [23]. The first two calls consist of fitting the LGCPs on the TKCs and the incidents separately. The computational time of these calls for dataset 1 were 46 and 16 seconds respectively on a Windows 8.1 (64-bit) platform with a quad-core 4.4 GHz Intel i7 processor with 16GB of RAM. The larger spatial mesh and sample size of dataset 2 is reflected in slightly longer computational

times, with 52 and 25 seconds respectively. The resulting approximated posterior values of the individual temporal correlation parameters (ρ) are shown in Table 1.

Table 1. Posterior estimates (mean, standard deviation and 95% credible interval) of the temporal correlation parameter (ρ) of the fitted log-Gaussian Cox processes per dataset

	Dataset 1		Dataset 2	
	TKCs	Incidents	TKCs	Incidents
Mean	.985	.998	.987	.999
σ	.009	.003	.010	.002
$Q_{0.025}$.964	.990	.963	.995
$Q_{0.975}$.997	1	.998	1

These near perfect correlations show that the spatial intensity surfaces ($\lambda(s)$) of both LGCPs remain constant over time. Plots of the latent fields show that the spatial intensity for all four of the point processes is higher near the city center, as displayed in figure 3. The relative risk of events taking place therefore seems to be related to population density.

The approximated posterior values of all of the relevant model parameters of the latent field (θ) were used in the final call to INLA as initial values to speed up the approximations. On the aforementioned system, this resulted in a computational time of two hours and 49 minutes for dataset 1 and four hours and 55 minutes for dataset 2.

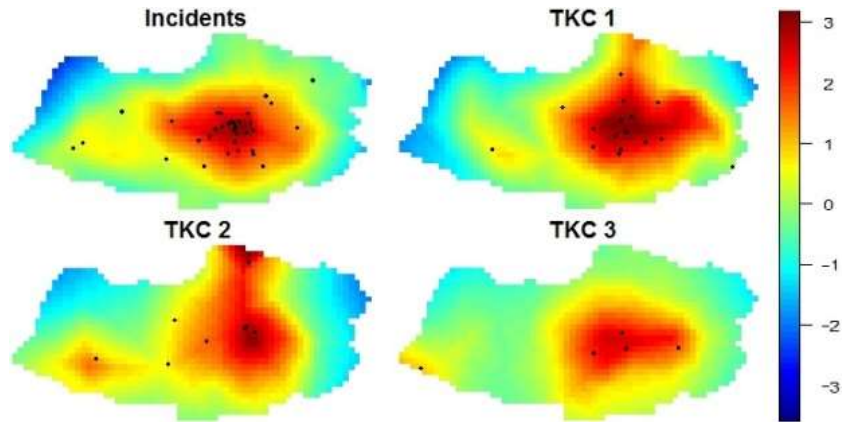


Figure 2. The latent fields of the log-Gaussian Cox processes of the incidents and the three tweet keyword clusters on the first time knot for dataset 1.

To answer our research questions the TKCs were specified as covariates to the incidents. The final model was specified as follows;

$$\eta TKC_m = \beta_0 TKC_m + \xi TKC_m$$

$$\eta Incidents = \beta_0 Incidents + \xi Incidents + \sum_{m=1}^3 \beta TKC_m \xi TKC_m$$

Here η is the linear predictor, ξ the spatiotemporal field, β_0 the intercept and β the correlation coefficient of the respective TKC with the incidents, which indicate the proportion of the spatiotemporal field shared with that of the incidents. The resulting estimated posterior values of the β s are shown in table 2.

Table 2. Posterior estimates (mean, standard deviation and 95% credible interval) of the correlation coefficients of the three tweet keyword clusters to the incidents and sample sizes

	Dataset 1					Dataset 2				
	Mean	σ	$Q_{0.025}$	$Q_{0.975}$	n	Mean	σ	$Q_{0.025}$	$Q_{0.975}$	n
βTKC_1	.366	.106	.165	.581	110	.355	.110	.155	.584	130
βTKC_2	.276	.106	.058	.471	52	.022	.081	-.141	.176	63
βTKC_3	.299	.107	.088	.508	20	.438	.127	.217	.716	15

These results show a great consistency in the approximated values of βTKC_1 between the two datasets, with means of .366 and .355 for dataset 1 and 2 respectively. This extends to their respective 95% credible intervals of [.165, .581] and [.155, .584]. The consistent findings show that the spatiotemporal field of the first tweet keyword cluster correlates significantly with that of the incidents. These results support our first sub-hypothesis, proving that the spatiotemporal distribution of tweets containing keywords related to physical violence is correlated positively with the spatiotemporal distribution of violent crime incidents. From this it can be concluded, with respect to our central research question, that the spatiotemporal distribution of certain tweets correlates positively with the spatiotemporal distribution of violent crime incidents. The approximated values of βTKC_2 and βTKC_3 however show a great inconsistency. For βTKC_2 the mean value estimated from dataset 2 (.022) even lies outside of the 95% credible interval derived from dataset 1 [.058, .471]. The latter of which indicates a significant correlation, which is however contradicted by the 95% credible interval of dataset 2 [-.141, .176]. Although the approximated values for βTKC_3 are significant in both datasets, with 95% credible intervals of [.088, .508] and [.217, .716], they differ greatly from each other ($\Delta X=.139$). The obtained results for βTKC_2 and βTKC_3 are too inconsistent to be considered reliable. With respect to the sub-questions whether the spatiotemporal distributions of tweets containing keywords related to agitation or aggression (II) and intoxication (III) are correlated with the spatiotemporal distribution of violent crime incidents, the findings are therefore considered to be inconclusive.

7 Discussion

The goal of this study was to explore the potential for predictive policing models based on Twitter data within the city of Amsterdam, the Netherlands. To this end a novel approach to Bayesian inference was used to assess the correlations between the spatiotemporal distributions of violent crime incidents and that of tweets containing predefined, hypothetically relevant keywords. This paper discussed the implementation of spatiotemporal log-Gaussian Cox processes with first-order autoregressive dynamics which have been jointly modeled and correlated using the SPDE approach offered by the R-INLA package. This approach enables using more complex models than traditional (MCMC) methods of Bayesian inference due to its computational efficiency. The R-INLA package offers an unprecedented level of flexibility and versatility, which is reflected in the multiple simplifications made to our model and the option to expand upon it further by adding terms specifying any type of random effect conceivable. To model the spatial and temporal dependency of the analyzed point processes, the applied statistical methods are considered an optimal approach. The excellent capability of dealing with uncertainty inherent to Bayesian inference makes it most suitable to account for the stochastic nature of the log-Gaussian Cox processes. Despite the small dataset this paper proves its feasibility for modeling the spatiotemporal relation between twitter messages and violent crime incidents, a topic previously unexplored within the Netherlands.

References

- [1] Achrekar, H., Gandhe, A., Lazarus, R., Yu, S., and Liu, B. (2011). Predicting flu trends using twitter data. *Proceedings of the IEEE Conference on Computer Communications Workshops*, 702–707.
- [2] Aramaki, E., Maskawa, S. and Morita, M. (2011). Twitter catches the flu: detecting influenza epidemics using Twitter. *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 1568–1576.
- [3] Asur, S. and Huberman, B.A., (2010). Predicting the future with social media. *Proceedings of the Web Intelligence and Intelligent Agent Technology Conference*, 492–499.
- [4] Blangiardo, M. and Cameletti, M., (2015). Spatial and Spatio-temporal Bayesian Models with R-INLA. West Sussex: Wiley.
- [5] Bollen, J., Mao, H. and Zeng, X. (2011). Twitter mood predicts the stock market. *Journal of Computational Science*, 2(1), 1–8.
- [6] Cameletti, M., Lindgren, F., Simpson, D., and Rue, H., (2013). *Advances in Stat.An.* , vol. 97, issue 2, pp. 109–131.
- [7] Gelfand, A.E., Schmidt, A.M. and Sirmans, C.F., (2002). Multivariate spatial process models: conditional and unconditional Bayesian approaches using coregionalization. Storrs, CT: Center for Real Estate and Urban Economic Studies, University of Connecticut.
- [8] Gerber, M.S., (2014). Predicting crime using Twitter and kernel density estimation. *Decision Support Systems*, 61(1), 115–125.
- [9] Howard, P.N. and Duffy, A., (2011). Opening closed regimes, what was the role of social media during the arab spring? *Project on Information Technology and Political Islam*, 1–30.
- [10] Illian, J.B., Sørbye, S.H., and Rue, H., (2012). A toolbox for fitting complex spatial point process models using integrated nested laplace approximation (inla). *Annals of Applied Statistics*, 6(4), 1499–1530.
- [11] Krainski, E.T., Lindgren, F., Simpson, D. and Rue, H. (2015). The R-INLA tutorial: SPDE models (draft). Available from <http://r-inla.org>
- [12] Lindgren, F. and Rue, H. (2015). Bayesian spatial modelling with R-INLA. *Journal of Statistical Software*. 63(19), 1–25.

- [13] Lindgren, F., Rue, H. and Lindstrom, J. (2011). An explicit link between Gaussian fields and Gaussian Markov random fields: The SPDE approach (with discussion). *Journal of the Royal Statistical Society (Series B)*, 73(4), 423–498.
- [14] Martins, T.G., Simpson, D., Lindgren, F. and Rue, H. (2013). Bayesian computing with INLA: new features. *Computational Statistics & Data Analysis*, 67, 68–83.
- [15] Møller, J., Syversveen, A.R. and Waagepetersen, R.P. (1998). Log Gaussian Cox processes. *Scandinavian journal of statistics*, 25(3), 451–482.
- [16] Pearsal, B. (2010). Predictive Policing: The Future of Law Enforcement? *National Institute of Justice Journal*, 266, 15–19.
- [17] Rue, H., Martino, S. and Chopin, N., (2009). Approximate Bayesian inference for latent Gaussian models using integrated nested Laplace approximations (with discussion). *Journal of the Royal Statistical Society (Series B)*, 71(2), 319–392.
- [18] Sanders, E. and Bosch, A. van den (2013). Relating political party mentions on Twitter with polls and election results. *Proceedings of the 13th Dutch-Belgian Workshop on Information Retrieval*.
- [19] Schmidt, A.M., and Gelfand, A.E., (2003). A Bayesian coregionalization approach for multivariate pollutant data. *Journal of Geophysical Research: Atmospheres*, 108(24).
- [20] Simpson, D.P., Illian, J., Lindgren, F., Sørbye, S.H., and Rue, H. (2011). Going off grid: Computationally efficient inference for log-Gaussian Cox processes. *ArXiv e-prints*.
- [21] Simpson, D.P., Lindgren, F. and Rue, H. (2012a). In order to make spatial statistics computationally feasible, we need to forget about the covariance function. *Environmetrics*, 23(1), 65–74.
- [22] Simpson, D. P., Lindgren, F., and Rue, H., (2012b). Think continuous: Markovian Gaussian models in spatial statistics. *Spatial Statistics*, 1, 16–29.
- [23] Simpson, D.P., Rue, H., Martins, T.G., Riebler, A., and Sørbye, S.H., (2014). Penalising model component complexity: A principled, practical approach to constructing priors. *ArXiv e-prints*.
- [24] Sloan, L. and Morgan, J., (2015). Who Tweets with Their Location? Understanding the relationship between demographic characteristics and the use of geoservices and geotagging on twitter. *PLoS ONE*, 10(11).
- [25] Tjong Kim Sang, E. and Bos, J. (2012). Predicting the 2011 Dutch Senate Election Results with Twitter. *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, 53–60.
- [26] Tobler, W.R., (1970). A computer movie simulating urban growth in the Detroit region. *Economic Geography*, 2(46), 234–240.
- [27] Van Noort, R. Kunneman, F. and Bosch, A. van den. (2016). Predicting civil unrest by categorizing Dutch Twitter events. In: *Proceedings of the 28th Benelux Conference on Artificial Intelligence, BNAIC 2016* (eds. T. Bosse and B. Bredeweg), Springer CCIS.
- [28] Wackernagel, H. (1998). *Multivariate geostatistics: An introduction with applications* (2nd ed.). New York, NY: Springer-Verlag.
- [29] Wang, X., Gerber, M.S. and Brown, D.E. (2012). Automatic crime prediction using events extracted from twitter posts. In N. Agarwal, K. Xu & N. Osgood (Eds.) *Social Computing, Behavioral-Cultural Modeling and Prediction* (pp. 231–238). New York, NY: Springer.
- [30] Weerkamp, W. and Rijke, M. de (2012). Activity prediction: A twitter-based exploration. *SIGIR Workshop on Time-aware Information Access*.
- [31] Zhang, X., Fuehres, H., and Gloor, P.A. (2011). Predicting Stock Market Indicators Through Twitter: I hope it is not as bad as I fear. *Social and Behavioral Sciences*, 26, 55–62.

Constructions at Work!

Visualising Linguistic Pathways for Computational Construction Grammar

Sébastien Hoorens ^a

Katrien Beuls ^a

Paul Van Eecke ^{ab}

^a *Artificial Intelligence Lab, VUB, Pleinlaan 2, 1050 Brussels, Belgium*

^b *Sony Computer Science Laboratories, 6, Rue Amyot, 75005 Paris, France*

Abstract

Computational construction grammar combines well-known concepts from artificial intelligence, linguistics and computer science into fully operational language processing models. These models allow to map an utterance to its meaning representation (*comprehension*), as well as to map a meaning representation to an utterance (*formulation*). The processing machinery is based on the unification of usage-patterns that combine morpho-syntactic and semantic information (*constructions*) with intermediate structures that contain all information that is known at a certain point in processing (*transient structures*). Language processing is then implemented as a search process, which searches for a sequence of constructions (*a linguistic pathway*) that successfully transforms an initial transient structure containing the input into a transient structure that qualifies as a goal. For larger grammars, these linguistic pathways become increasingly more complex, which makes them difficult to interpret and debug for the human researcher. In order to accommodate this problem, we present a novel approach to visualising the outcome of constructional language processing. The linguistic pathways are visualised as graphs featuring the applied constructions, why they could apply, with which bindings, and what information they have added. The visualisation tool is concretely implemented for Fluid Construction Grammar, but is also of interest to other flavours of computational construction grammar, as well as more generally to other unification-based search problems of high complexity.

1 Introduction

A crucial first step in natural language understanding and production is the implementation of a system that can reliably map between an utterance and its meaning representation. The utterance is used for transferring a meaning representation from one human or artificial agent to another, and the meaning representation is used for interpreting the utterance in a given grounded or textual context [17]. Computational construction grammar is a field of study that aims to tackle this challenge by combining insights from linguistics, artificial intelligence and computer science into fully operational, bidirectional language processing models. Linguistically, the models are inspired by the basic principles of construction grammar [4, 5, 7, 3, 6], in particular the tight integration of syntax and semantics, the lexicon-grammar continuum and the use of multiple perspectives, such as phrase structure, functional structure, case structure and information structure. From artificial intelligence and computer science, the models borrow core concepts, such as problem solving through search [11, 12] and the unification of feature structure representations [13, 14, 10].

Computational construction grammars implement language processing as a search process, with operators, called *constructions*, expanding intermediate structures until a solution is found. For larger grammars, these intermediate structures can easily consist of “dozens of units and hundreds of features” [19]. Each intermediate structure has been shaped by a whole range of

constructions of different types, each construction contributing units or features to the analysis. As constructions fit together like the pieces of a puzzle, it often happens that certain preconditions of a construction have been contributed by one earlier construction, and others by a different one. Such constructional dependencies can be hard to grasp, yet it is important that they are correctly captured by the grammar engineer.

In order to enhance the human readability of computational construction grammar analyses, we introduce a tool that clearly visualises the processing results as a graph, featuring the applied constructions, why they could apply, with which bindings, and what information they have contributed. The tool is designed to help grammar engineers develop and debug grammars, as well as to help regular users inspect and interpret the results of a computational construction grammar analysis. In this way, the tool will help to address one of the biggest challenges faced by computational construction grammars, namely scaling up for attaining larger coverage.

There are multiple computational construction grammar implementations available, of which *Embodied Construction Grammar (ECG)* [2, 1] and *Fluid Construction Grammar (FCG)* [15, 16] are the most advanced projects. Our visualisation tool has concretely been implemented for Fluid Construction Grammar, but the proposed visualisations should be easily transferable to other computational construction grammar implementations and can also be applied to other unification-based search problems of high complexity. This generality is demonstrated in this paper by applying the tool to a planning problem.

The paper is structured as follows. First, we introduce the necessary basics of bidirectional language processing in Fluid Construction Grammar. Then, we present the design and implementation of the visualisation tool and demonstrate the tool in a use case. Finally, we show that the tool can also be used to visualise other problems than language processing, by demonstrating its application to a planning problem. The paper is supported by an interactive web demonstration, which can be accessed via <https://www.fcg-net.org/demos/visualising-pathways>.

2 Bidirectional Language Processing using FCG

2.1 Language Processing as a Problem Solving Process

Fluid Construction Grammar (FCG, <https://www.fcg-net.org>, [15, 16]) is an open-source computational construction grammar formalism and implementation, designed for mapping between utterances and their meaning representation. It is a bidirectional formalism in the sense that it uses the same grammar and processing mechanisms for both *comprehension*, i.e. mapping from a form to a meaning representation, and *formulation*, i.e. mapping from a meaning representation to a form. FCG implements language processing as a problem solving process [18], consisting of the following main components:

- *Transient Structures*. Transient structures are the state representations in the search problem. A transient structure is a feature structure that contains all information (morpho-syntactic, semantic, pragmatic, ...) that is known at a certain point in processing.
- *Initial Transient Structure*. The initial transient structure is the root of the search problem and is computed directly from the input. In comprehension, the initial transient structure contains a representation of the strings in the input and of the internal ordering between the strings. In formulation, the initial transient structure contains the predicates of the meaning representation that needs to be formulated.
- *Constructions*. Constructions are the operators in the search problem. A construction can apply to a transient structure if it *matches* it, i.e. if there are no conflicts when unifying the *conditional part* of the construction with the transient structure. If the construction matches the transient structure, it can expand the transient structure by *merging* the information from its contributing part into it.
- *Goal Tests*. Goal tests compute whether a given transient structure qualifies as a solution.

The task of the FCG system is to find a sequence of constructions that can expand the initial transient structure into a transient structure that qualifies as a goal. This sequence of

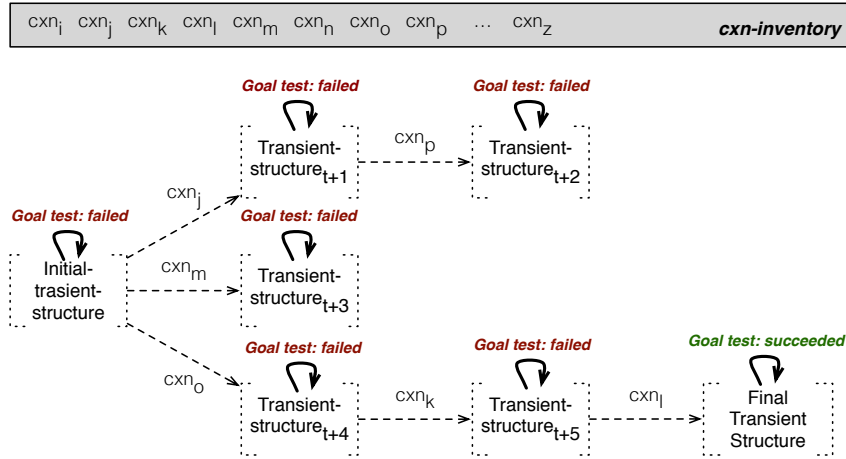


Figure 1: FCG implements language processing as a search process, in which constructions (cxns for short) expand an initial transient structure until a solution is found. A sequence of construction applications that leads to a solution is called a *linguistic pathway*.

construction applications is called a *linguistic pathway*. In order to make the search process computationally feasible, different heuristics and optimization strategies are used, which fall however outside the scope of this paper. A schematic representation of how FCG implements language processing as a problem solving process is shown in Figure 1. For studying FCG in more detail, we recommend playing around with the interactive web service at <https://www.fcg-net.org/fcg-interactive>.

2.2 Bidirectional Constructions

Constructions are, linguistically speaking, usage-patterns that combine morpho-syntactic and pragmato-semantic information. Constructions can be very concrete, for example in the case of morphological and lexical constructions, but can also be more abstract, for example in the case of argument structure constructions. Computationally speaking, constructions are feature structures which always have the same basic design, as exemplified in Figure 2. Constructions consist of a *contributing part*, left of the arrow, and a *conditional part*, right of the arrow. The conditional part consists of 1 or more units, and the contributing part of 0 or more. The units on the conditional part of a construction each consist of two parts, separated by a horizontal line. The upper part is called the *formulation lock* and the lower part the *comprehension lock*. When the construction is used in comprehension, the comprehension lock of the units on the conditional part will be matched with the units of the transient structure. Matching is a subset unification process, which checks whether the conditions stated in the locks are compatible with the transient structure. If matching succeeds, the rest of the construction, i.e. the formulation lock and the contributing part, will be merged into the construction. Merging is an other unification process, which, in addition to matching, adds features from the construction that do not yet occur in the transient structure to the transient structure. When the construction is used in formulation, it will be the formulation locks that are matched and the comprehension locks and contributing part that will be merged. The active locks, i.e. the comprehension locks in comprehension and the formulation locks in formulation, can thus be thought of as the *preconditions* of the construction, and the non-active locks and the contributing part as *postconditions*. The fact that the construction is structured into this lock system, facilitates an efficient implementation of bidirectional language processing.

An example construction is shown in Figure 2. The conditional part of this **determinednoun-cxn** consists of three units and the contributing part of one unit. Conventionally, (logic) variables are preceded by a question mark. In comprehension as well as in formulation, this construction adds a noun to an existing noun phrase unit, with the additional constraint that the noun needs

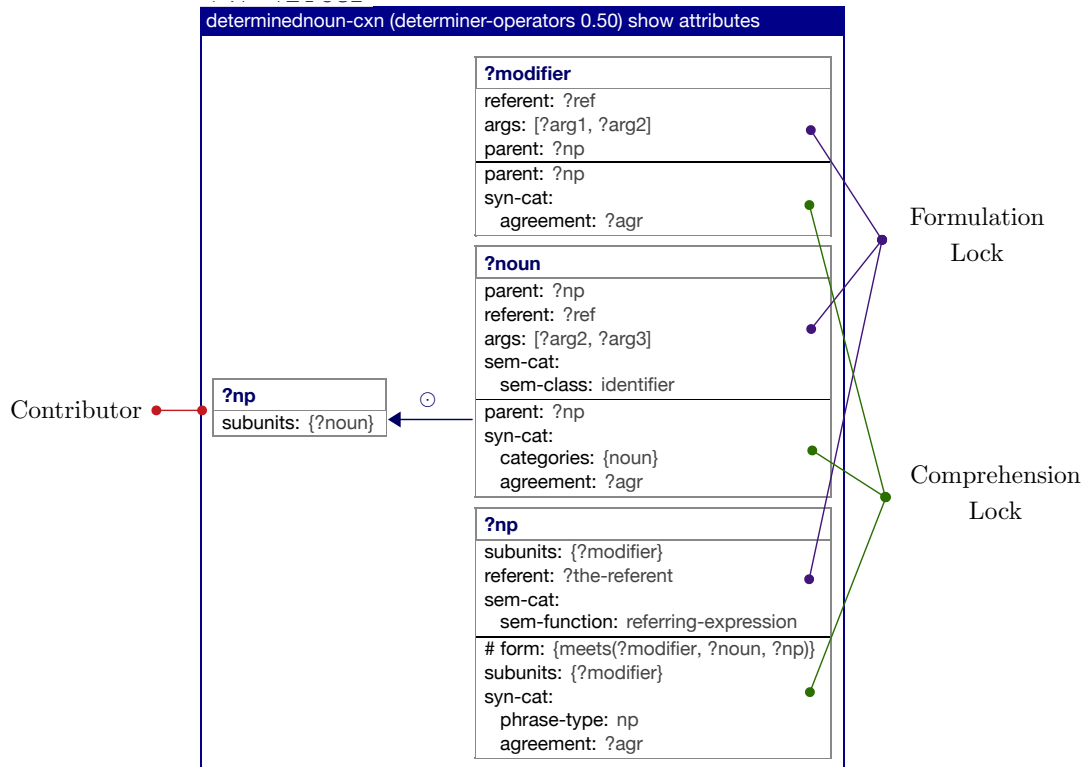


Figure 2: A determinednoun-cxn with labels highlighting its *comprehension lock*, *formulation lock* and *contributor*.

to be right adjacent to a modifier that is already part of the noun phrase.

2.3 Constructional Dependencies

For more complex grammars and inputs, the number of applied constructions, units and features can become quite large. Structures containing a thousand features over sixty units, contributed by a hundred constructions are no exception. These large structures are difficult to debug and interpret and it is highly non-trivial to tell why a construction could apply, i.e. which previous constructions contributed the necessary features that satisfy the locks of the construction. Moreover, from a linguistic point of view, the construction application process is often more informative than the final transient structure, as it reveals which constructions have applied with which inputs and outputs.

At this moment, FCG's standard visualisation system [8] visualises the construction application process as a tree structure, drawing one node for each construction application. Each node shows the transient structure before construction application, the construction itself, and the transient structure after construction application, as well as the bindings from matching. This standard visualisation is shown in Figure 4 for the input utterance *the cat*, as analysed by the Basic English Grammar [20]. The tree structure is linear here, as the comprehension process did not require any search. Each green box represents the application of a construction, and contains the transient structure after the application of that construction. The figure should be read from left to right, and shows how the transient structure is gradually build up. For space reasons, the transient structure before construction application, the construction itself, the matching bindings and the features inside the units are collapsed here, but they can be explored in full detail in the web demonstration supporting this paper.

Although the standard visualisation has proven its worth in grammar engineering for over a decade now, it does not provide insight in the complex puzzle of constructional dependencies. It does not transparently display which earlier constructions have provided the necessary features

that satisfy the lack of a later construction. An earlier effort at integrating the idea of construction networks into FCG focused mainly on their benefits in optimizing the search space, rather than visualising the constructional dependencies as such [22, 21]. The new visualisation that is proposed in our paper specifically tackles the problem of visualising linguistic pathways with a special focus on these dependencies.

3 Visualising Linguistic Pathways

The main asset of the new visualisation is that it gives insight into how a solution, in this case a certain feature structure that resulted from the construction application process, came into being. To explain the specifics of the new visualisation, we work again with the example utterance *the cat*. We comprehend the utterance using the Basic English Grammar [20]. The initial transient structure consists only of the input strings and word order information. Then the search process begins, and six constructions apply, gradually extending the transient structure. These constructions are the **the-cxn**, the **cat-noun-morph-cxn**, the **cat-noun-lex-cxn**, the **determiner-operation-cxn**, the **determined-noun-cxn** and the **singular-cxn**. The final transient structure is shown in Figure 3 and the standard visualisation of the application process of the six constructions in Figure 4.

The final transient structure contains three units: an NP unit and two lexical units as its children. The highlighted text in the figure shows certain variable equalities (e.g. **referent**) and important features such as **agreement**, whose value is the same in all three units. These features have been contributed by the six different constructions, and certain constructions could only apply after a combination of other construction had contributed the necessary features. How these constructions depend on each other to build the transient structure that we see here, is exactly what our tool visualises.

This section first explains the design of the visualisation tool (3.1) and then discusses how it has been integrated into Fluid construction Grammar (3.2).

3.1 Design

The graph in Figure 5 shows the constructional dependencies between the six constructions that were active in the analysis of the example utterance *the cat*. It consists of three main components:

1. *Constructions* are represented as rectangular, blue node clusters. In each node cluster, the name of the construction is shown at the top in white text on a blue background (e.g. **determinednoun-cxn**). The clusters are laid out from left to right, in their order of application. Constructions more to the left are responsible for earlier expansions of the transient structure.
2. *Units of constructions* make up the nodes of the graph and are shown as rectangular boxes within the constructions. These boxes contain the names of the units (e.g. **?definite-article** or **?modifier**). The unit names are variables that are bound to units in the transient structure during construction application. When building up the transient structure shown in Figure 3 for instance, **?definite-article**, **?determiner** and **?modifier** will all be bound to **the-179**. **?cat-unit** and **?noun** will be bound to **cat-45** and **?np** will be bound to **np-828**. Units of constructions can have a white or a green background. White units were already present in the transient structure before the construction had applied, and were thus added by an earlier construction application. Green units are added by the construction during its application.
3. *Constructional dependencies* are represented by labelled edges between units. The direction of the edges follows a chronological order: the construction of the unit at the source applied before the construction of the unit at the target. Hence, the unit at the target (usually) depends on information from the unit at the source. The labels on the edges contain conditional information that was present in the target construction's formulation

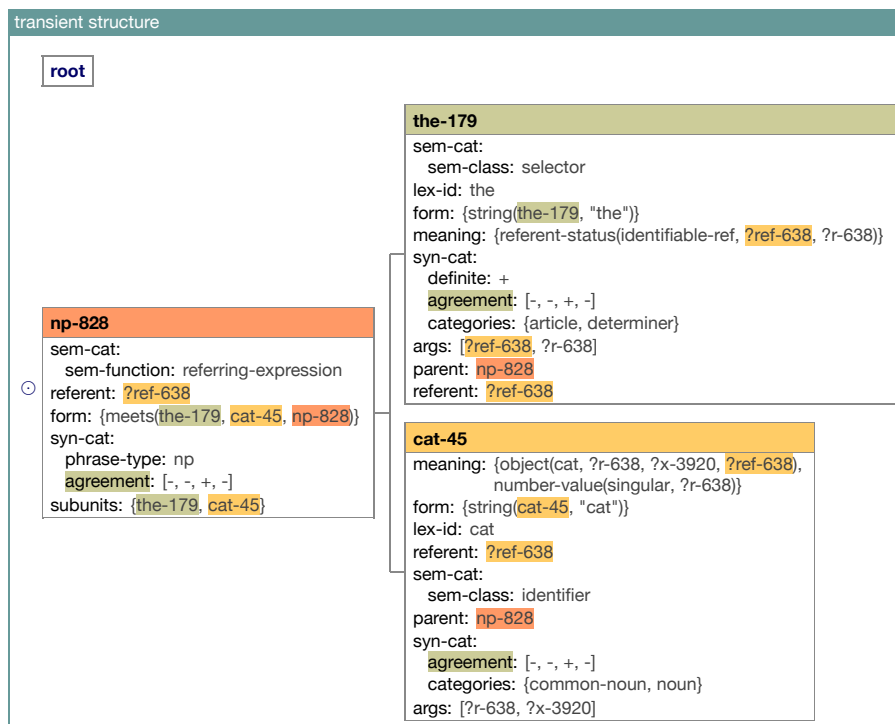


Figure 3: Resulting transient structure after comprehending *the cat*. Six different constructions have shaped this transient structure: the the-cxn, the cat-noun-morph-cxn, the cat-noun-lex-cxn, the determiner-operation-cxn, and the determined-noun-cxn.

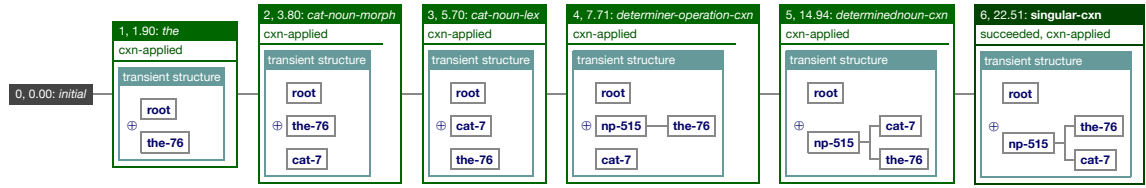


Figure 4: The construction application process for the utterance *a cat* using the Basic English Grammar [20], as visualised by FCG’s standard visualisation library. The green boxes show how the application of the different constructions sequentially expand the transient structure.

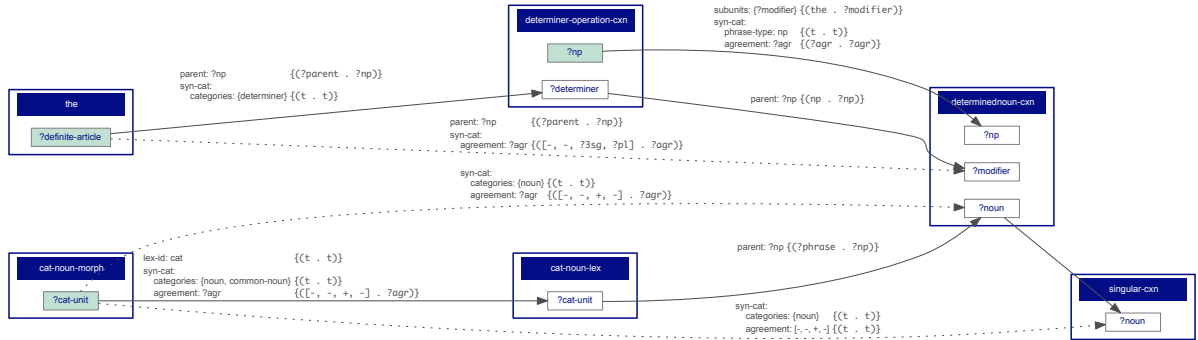


Figure 5: The construction application process for the utterance *a cat* using the Basic English Grammar [20], as visualised by the tool introduced in this paper. The dependencies between the constructions and their units are clearly indicated with arrows.

or comprehension lock, depending on the direction of processing. Every label consists of two columns:

- (a) The first column contains all features that were contributed by the source construction and were used by the target construction's lock for matching.
- (b) The second column is the binding list that resulted from the matching operation, with on the left the source feature value and on the right the target feature value to which it could be bound. If the binding looks like $(t \cdot t)$, it means that two atomic values were unified, e.g. $np \sqcup np$.

Figure 5 includes two types of edges. Solid arrows indicate dependencies between directly adjacent nodes and thus reflect the chronology of construction applications, while dotted arrows display longer-range dependencies. Let's focus on the **determined-noun-cxn** in the example, and more in particular on its **?noun** unit. A solid arrow connects the **?cat-unit** from the **cat-noun-lex** construction to this unit and the label on the arrow indicates that the latter gets its **parent** feature from there. The fact that the arrow is solid marks that the no other construction has affected this unit between the **cat-noun-lex** construction and the **determined-noun-cxn**. A dotted arrow connects the **?cat-unit** from the **?cat-noun-morph** construction to the **?noun** unit of the **determined-noun-cxn** and the label indicates that it gets its **categories** and **agreement** features from there. The dottedness of the arrow shows that an other construction has affected the unit between the **cat-noun-morph** construction and the **determined-noun-cxn**, namely the **cat-noun-lex** construction.

3.2 Integration into FCG

The visualisation tool has been integrated into the FCG web interface and monitoring system [8, 9], and will be included in FCG's next release. The graph with constructional dependencies as shown in Figure 5 is automatically added to the visualisation of processing results when the following configuration option is set:

```
(set-configuration (visualization-configuration cxn-inventory)
                   :constructional-dependencies t)
```

The graph can be customised to the user's preferences by setting a few additional configuration options in the visualisation configuration of the construction inventory.

- `(:labeled-paths nil/no-bindings/t)`

This option defines what information is shown on the arrows that indicate the dependencies. `nil` leaves the arrows empty, `t` shows the matched features and their bindings (as shown in Figure 5) and `no-bindings` shows the matched features without bindings.

- `(:colored-paths nil/t)`

When this option is set to `t`, each directed path in the graph is drawn in a different colour. This means that the units in the construction that have been bound to the same unit in the transient structure and the arrows connecting these units will have the same colour.

- `(:trace-units '(list-of-unit-names))`

Having the matching features and their bindings on all arrows can make larger graphs quite crowded. Therefore, it is sometimes better to only include the features and bindings of the specific paths that the grammar engineer is interested in. These paths can be specified with using the `:trace-units` option.

For space reasons, we could not include figures showing these different options into the printed version of this paper, but they are all shown next to each other in the web demonstration supporting this paper.

4 Demonstration

We will now demonstrate the visualisation tool with a somewhat larger example, in which the linguistic pathway grows into a more intricate web of unit dependencies. We will analyse the comprehension process of a single sentence, but the web demonstration that accompanies this paper includes many more (and longer) examples, also for production. Because the graphs become very detailed when a larger number of constructions are used in the analysis, we chose to leave out the labels (i.e. the matching features and bindings) on the edges that link the unit nodes. The complete constructional dependency graphs are included in the on-line web demo.

The example sentence, *the cat will jump*, contains a noun phrase, a modal auxiliary and an intransitive main verb. When analysed with the current version of the Basic English grammar ([20] and interactively consultable at www.fcg-net.org/fcg-interactive), 17 constructions are needed to analyse the sentence in comprehension. The visualisation of the constructional dependencies is shown in Figure 6.

The constructional dependencies graph contains 17 blue node clusters, one for each construction that has been applied. On the left side of the graph, we can see that there are four entry points, i.e. node clusters with no incoming edges. The comprehension locks of these four constructions, *jump-morph*, *will-morph*, *the* and *determiner-operation-cxn*, only require strings that are found in the input utterance and do not need any features added by previous construction applications. They create one lexical unit each, which is shown on a green background. After the *the* construction has applied and created a *?definite-article* unit, the *determiner-operation-cxn* can match its *?determiner* unit on this unit and create a new *?np* unit. The *cat-noun-lex* and *jump* constructions can respectively match on the units created by the *jump-morph* and *cat-noun-morph* constructions, adding information to these units, but not creating any new ones.

Then, we can see that the lexical units for the modal auxiliary and the lexical main verb are both affected by the *marked-modality-cxn*, which creates a new *?vp* unit. This *?vp* unit is then modified by the *vp-cxn*, *non-perfect-cxn* and the *non-progressive-cxn*. The *determinednoun-cxn* adds information into lexical units for the noun and the definite article, as well as to the *?np* unit. The lexical unit for the noun is further affected by the *singular-cxn*.

After that, the subject and verb come together. The *subject-verb-cxn* matches on the *?np* and *?vp* units, and creates a new *?clause* unit. These three units are then further extended by the *declarative-main-clause-verb-cxn* and the *intransitive-cxn*.

From this short description, it becomes already clear that constructional dependencies are a complex matter, and yet, we have only described the information conveyed by the boxes and the solid arrows in the graph. The visualisation tool allows the grammar engineer and user to quickly grasp how the grammar has analysed an input utterance and spot possible bugs or inconsistencies, which was much more difficult using FCG’s standard visualisation. The graphs are designed to be studied on computer screens, where the users can zoom in and out on parts of the network. Therefore, they tend to get too large to be printed on paper quite soon.

5 Beyond Language Processing

Although the visualisation tool introduced in this paper was specifically designed for visualising linguistic pathways yielded by computational construction grammar analyses, it can easily be applied to other domains as well. In order to demonstrate the generality of the tool, we will now show an example of its application to a planning problem. As the basic building blocks of FCG, in particular the exploration of a search space, are so general, the planning problem could be implemented in the same formalism.

The goal in our planning problem is having pancakes. The initial state is a kitchen’s basic equipment (a stove, pans, bowls, a whisk, cutlery, ...) and basic ingredients (eggs, butter, milk, flour, tabasco, ham, ...). The operators are actions such as taking eggs, collecting the ingredients that are needed for pancake, and putting butter in a pan and putting the pan on the stove. The planning problem consists in finding a series of subsequent actions that form a path from the initial state to a state in which there are delicious pancakes.



The graph containing the constructional dependencies for the pancake problem is shown in Figure 7. We can see that in total, 11 actions have been performed to make the pancakes. 7 actions were *take-...* actions that could be independently performed based on the initial state. Then, a *collect-pancakes-ingredients* action matched on the *?flour*, *?eggs* and *?milk* units, creating a new *?pancake-ingredients* unit. The *make-pancake-dough* action then matched on the *?pancake-ingredients*, *?whisk* and *?bowl* units and created a new *?pancake-dough* unit. The *prepare-pan* action matches on the *?pan*, *?butter* and *?stove* units and creates a new *?prepared-pan* unit. Finally, the *make-pancakes* action matches on the *?prepared-pan* unit and the *pancake-dough* unit and creates a new *pancakes* unit. The arrows are labelled with the features on which the actions match, and the bindings of these features. When looking at the *make-pancakes* action for example, we can see that it matches on a *?pancake-dough* unit that does not have *baked* among its properties (marked in red in the figure). The *make-pancakes* construction will add *baked* to the properties of this unit, and indeed, pancake dough can only be baked a single time.

This visualisation is of course only one possibility for visualising a planning process. It does not replace other, complementary visualisations, such as the explored search space. The worth of this visualisation is its focus on the dependencies between the actions, and on which previous actions have facilitated the performance of later actions. It is particularly useful when actions are complex and depend on many different factors affected by other actions.

6 Future Work

At this moment, we use the tool to visualise the dependencies between the construction that were active in the comprehension or formulation process of a single utterance. For the future, we plan to comprehend a corpus of sentences and gradually build up a large graph containing the dependencies between all grammatical constructions in the grammar. The edges in the graph can then be coded with frequency information about (parts of) linguistic pathways and capture systematic dependencies between groups of constructions. This would allow us to automatically build and visualise a usage-based constructional dependency graph that evolves over time, for a complete grammar.

7 Conclusion

In this paper, we have introduced a novel approach to visualising the outcome of constructional language processing. The linguistic pathways are visualised as graphs, featuring the applied constructions, why they could apply, with which bindings, and what information they have contributed. By revealing the dependencies between the applied constructions, these graphs are very helpful when developing and debugging grammars, while also enhancing the human readability of construction grammar analyses. The tool has been concretely implemented in Fluid Construction Grammar, but the proposed visualisations should be easily transferable to other construction grammar implementations. In order to highlight that the applicability of the tool is not limited to language processing problems, we have also demonstrated how the tool can be used to visualise planning problems.

Acknowledgements

We are particularly indebted to Luc Steels, Remi van Trijp and Yannick Jadoul for their support and valuable feedback during the development of the tool.

References

- [1] B. Bergen and N. Chang. Embodied Construction Grammar in simulation-based language understanding. In *Construction Grammar(s): Cognitive and Cross-Language Dimensions*. John Benjamins, 2005. 234

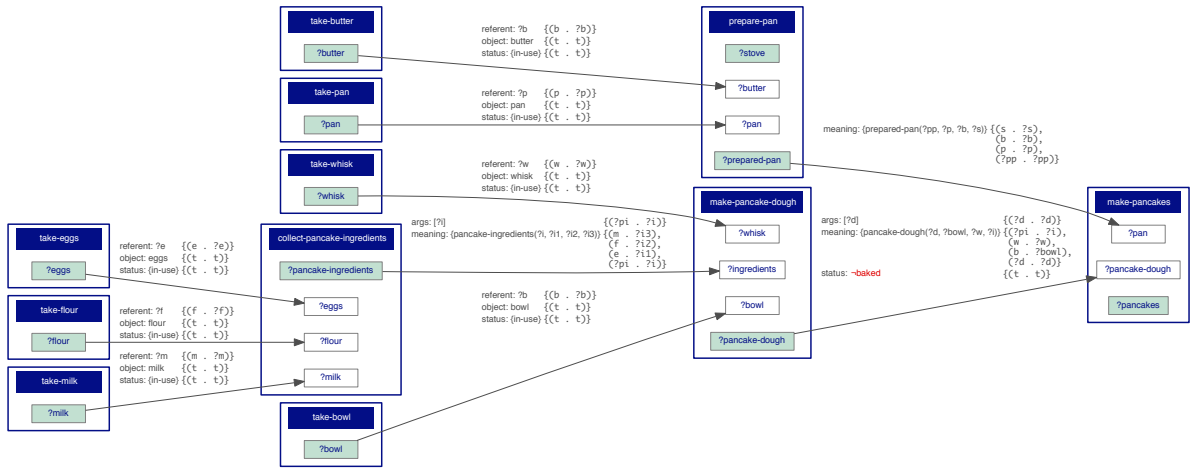


Figure 7: Using the new visualisation tool for a (hierarchical) planning problem.

- [2] N. Chang, J. Feldman, R. Porzel, and K. Sanders. Scaling cognitive linguistics: Formalisms for language understanding. In *Proc. 1st International Workshop on Scalable Natural Language Understanding*, 2002.
- [3] W. Croft. *Radical Construction Grammar: Syntactic theory in typological perspective*. Oxford University Press, Oxford, 2001.
- [4] C.J. Fillmore, P. Kay, and M.C. O'Connor. Regularity and idiomaticity in grammatical constructions: The case of let alone. *Language*, 64(3):501–538, 1988.
- [5] A.E. Goldberg. *Constructions: A construction grammar approach to argument structure*. University of Chicago Press, Chicago, 1995.
- [6] A.E. Goldberg. *Constructions at work: The nature of generalization in language*. Oxford University Press, Oxford, 2006.
- [7] P. Kay and C.J. Fillmore. Grammatical constructions and linguistic generalizations: the What's X doing Y? construction. *Language*, 75(1):1–33, 1999.
- [8] M. Loetzsch. Tools for grammar engineering. In L. Steels, editor, *Computational Issues in Fluid Construction Grammar*. Springer Verlag, Berlin, 2012.
- [9] M. Loetzsch, J. Bleys, and P. Wellens. Understanding the dynamics of complex lisp programs. In *Proceedings of the 2nd European Lisp Symposium*, pages 59–69, Milano, Italy, May 2009.
- [10] A. Martelli and U. Montanari. An efficient unification algorithm. *ACM Transactions on Programming Languages and Systems (TOPLAS)*, 4(2):258–282, 1982.
- [11] A. Newell, J.C. Shaw, and H.A. Simon. Empirical explorations of the logic theory machine: A case study in heuristic. In *Papers Presented at the February 26-28, 1957, Western Joint Computer Conference: Techniques for Reliability*, IRE-AIEE-ACM '57 (Western), pages 218–230, New York, NY, USA, 1957. ACM.
- [12] N. Nilsson. *Problem-Solving Methods in Artificial Intelligence*. McGraw-Hill Pub. Co., 1971.
- [13] J.A. Robinson. A machine-oriented logic based on the resolution principle. *J. ACM*, 12(1):23–41, 1965.
- [14] J.A. Robinson. Computational logic: The unification computation. *Machine intelligence*, 6:63–72, 1971.
- [15] L. Steels, editor. *Design Patterns in Fluid Construction Grammar*. John Benjamins, Amsterdam, 2011.
- [16] L. Steels. Basics of Fluid Construction Grammar. *Constructions and Frames*, 9(2), 2017.
- [17] L. Steels, J. De Beule, and P. Wellens. Fluid Construction Grammar on real robots. *Language Grounding in Robots*, pages 195–213, 2012.
- [18] L. Steels and P. Van Eecke. Insight grammar learning using pro- and anti-unification. Forthcoming.
- [19] L. Steels and E. Szathmáry. Fluid Construction Grammar as a biological system. *Linguistics Vanguard*, 12(1), 2016.
- [20] R. van Trijp. A computational construction grammar for English. In *The AAAI 2017 Spring Symposium on Computational Construction Grammar and Natural Language Understanding Technical Report*, number SS-17-02, pages 266–273, Stanford, 2017. Association for the Advancement of Artificial Intelligence.

- [21] P. Wellens. Organizing constructions in networks. In L. Steels, editor, *Design Patterns in Fluid Construction Grammar*, pages 181–201. John Benjamins Publishing Company, 2011.
- [22] P. Wellens and J. De Beule. Priming through constructional dependencies: a case study in fluid constructions grammar. In *Proceedings of the 8th International Conference on the Evolution of Language (EVLANG 8)*, pages 344–351, Singapore, 2010. World Scientific.

Generalization of an Upper Bound on the Number of Nodes Needed to Achieve Linear Separability

Marjolein Troost✉, Katja Seeliger, Marcel van Gerven

Radboud University
Donders Institute for Brain, Cognition and Behaviour
Nijmegen, The Netherlands
`marjolein.troost@student.ru.nl`

Abstract. An important issue in neural network research is how to choose the number of nodes and layers such as to solve a classification problem. We provide new intuitions based on earlier results by [1] by deriving an upper bound on the number of nodes in networks with two hidden layers such that linear separability can be achieved. Concretely, we show that if the data can be described in terms of N finite sets and the used activation function f is non-constant, increasing and has a left asymptote, we can derive how many nodes are needed to linearly separate these sets. For the leaky rectified linear activation function, we prove separately that under some conditions on the slope, the same number of layers and nodes as for the aforementioned activation functions is sufficient. We empirically validate our claims.

1 Introduction

Artificial neural networks perform very well on classification problems. They are known to be able to linearly separate almost all input sets efficiently. However, it is not generally known how the artificial neural networks actually obtain this separation so efficiently. Therefore, it is difficult to choose a suitable network to separate a particular dataset. Hence, it would be useful if, given a dataset used for training and a chosen activation function, one can analytically derive how many layers and nodes are necessary and sufficient for achieving linear separability on the training set. Even though an analytical solution is still far away, some steps in the right direction have already been taken.

An et al. [1] showed for rectified linear activation functions that the number of hidden layers sufficient for linearly separating any number of (finite) datasets is 2 (follows from universality as well) and that the number of nodes per layer can be determined using disjoint convex hull decompositions. Yuan et al. [6] have provided estimates for the number of nodes per layer in a two-layer network based on information-entropy. Fujita [5] has done the same based on statistics by adding extra nodes one by one. Another approach by Kůrková [4] is to calculate how well a function can be approximated using a fixed number of nodes. Baum [7] has shown that a single-layer network can approximate a random dichotomy with only N/d units for an arbitrary set of N points in general position in d

dimensions. He also makes the link to the Vapnik-Chervonenkis dimension of the network. In this work we do not use statistics to achieve an estimate of the number of nodes but rather simple algebra to obtain an absolute upper bound, following An et al. [1] and Baum [7]. However, we will obtain this bound for multiple activation functions and arbitrary finite sets.

It is well-known that two-layer neural networks are universal approximators (e.g. [2, 3] or more recently [8]). However, even though we know there should exist a network that can linearly separate two arbitrary finite sets, we do not know which one it is. Choosing the wrong kind of network can lead to severe overfitting and reduced performance on the test set [6]. Therefore, it is useful to have an upper bound on the number of nodes. The upper bound can aid in choosing an appropriate network for a task. With this in mind, we aim to give a theoretical upper bound on the size of a network with two hidden layers in terms of nodes, that is easily computable for any finite input sets that need to be separated.

The rest of this work is organized as follows: in Section 2 we repeat some of the definitions from [1] and we give a direct extension of two of their theorems for which their proof does not need to be changed. In Section 3 we present our main theorem, which generalizes the two theorems from Section 2 to a larger class of activation functions. In Section 4 we add some corollaries and refer to an extension to multiple sets that is given in [1], we also provide an algorithm to estimate the upper bound on the number of nodes. We show simulation results that support our claims in Section 5 and conclude with some final remarks in Section 6.

2 Definitions and Basic Results

We want to emphasize that the following definitions and theorems (Definition 1, Theorems 3 and 5 and Corollary 12) are due to An et al.[1] and are repeated here for convenience. We took the liberty of adapting some of these definitions for clarity and giving slightly stronger versions of their Theorems 4 and 5 in Theorems 3 and 5 which follow directly from the proof given by [1].

Throughout the article, we will use the following notation and conventions: all sets are finite. We use f to denote a non-constant activation function that is always applied element-wise to its argument. So

$$f((x_1, x_2, \dots, x_n)^T) = (f(x_1), f(x_2), \dots, f(x_n))^T. \quad (1)$$

\mathbb{R} is the set of real numbers. We define the convex hull of a set as the set of all convex combinations of the points in the set. In set notation:

$$C(X) = \left\{ \sum_{i=1}^{|X|} \alpha_i x_i \mid \forall i \ \alpha_i \geq 0, \ \sum_{i=1}^{|X|} \alpha_i = 1 \right\}. \quad (2)$$

We will now first define what is meant by a disjoint convex hull decomposition.

Definition 1 Let $X_k, k \in \{1, \dots, m\}$, be m disjoint, finite sets in \mathbb{R}^n . A decomposition of X_1, \dots, X_m , $X_k = \bigcup_{i=1}^{L_k} X_k^i$, with $L_k \geq 1$ is called a *disjoint convex hull decomposition* if the unions of the convex hulls of X_k^i ,

$$\hat{X}_k \triangleq \bigcup_{i=1}^{L_k} C(X_k^i), \quad (3)$$

are still disjoint. I.e. for all $k \neq l$: $\hat{X}_k \cap \hat{X}_l = \emptyset$. See for an illustration Figure 1B.

Since we are interested in finite sets, we can always define a disjoint convex hull decomposition (just take every point as a singleton). Such a decomposition is not unique. We would like to find a decomposition with the smallest L_k s. However, it is not necessary for the following that the decomposition is minimal. For two sets we also use the terminology described in Definition 2. This definition can easily be extended to multiple sets by applying it pairwise.

Definition 2 If $C(X_1) \cap C(X_2) = \emptyset$, X_1 and X_2 are called **linearly separable**. If $C(X_1) \cap X_2 = \emptyset$ or $X_1 \cap C(X_2) = \emptyset$, X_1 and X_2 are called **convexly separable**. If all disjoint convex hull decompositions of X_1 and X_2 satisfy $\min(L_1, L_2) > 1$, X_1 and X_2 are called **convexly inseparable**.

We start by giving a generalization of Theorem 4 from [1]. Instead of considering a rectified linear classifier activation function, we consider the more general class of functions that satisfy $f(x) = 0$ for $x \leq 0$ and $f(x) > 0$ for $x > 0$. We will call these functions **semi-positive**. Notice that they can be any function of $x > 0$ as long as they remain positive. This generalization is straightforward and the proofs do not need to be adapted but are given here for easy reference.

Theorem 3 Let X_1 and X_2 be two convexly separable sets, with a finite number of points in \mathbb{R}^n . Say, $C(X_1) \cap X_2 = \emptyset$ and $X_2 = \bigcup_{j=1}^{L_2} X_2^j$ with $L_2 \in \mathbb{N}$, $X_2^j \subseteq X_2$ such that $C(X_1) \cap C(X_2^j) = \emptyset$ for each j . Let $w_j^T x + b_j$ be linear classifiers of X_2^j and X_1 such that for all j

$$w_j^T x + b_j \leq 0 \quad \forall x \in X_1 \quad (4)$$

$$w_j^T x + b_j > 0 \quad \forall x \in X_2^j. \quad (5)$$

Let $W = [w_1, \dots, w_{L_2}]$, $b = [b_1, \dots, b_{L_2}]^T$ and $Z_k = \{f(W^T x + b) \mid x \in X_k\}, k \in \{1, 2\}$. Here f is a semi-positive function that is applied component-wise. Then Z_1 and Z_2 are linearly separable. For this we need L_2 affine transformations.

Proof. For all $x \in X_1$ we have that $w_j^T x + b_j \leq 0$. So $Z_1 = \{f(W^T x + b) \mid x \in X_1\} = \{(f(w_j^T x + b_j))_j \mid x \in X_1\} = \{0\}$. Now, for an $x \in X_2$, there exists a j such that $x \in X_2^j$. So, there exists a j such that $w_j^T x + b_j > 0$. Therefore, each $z \in Z_2$ has components greater or equal to zero and at least one component that is strictly greater than zero. This means $C(Z_1) \cap C(Z_2) = \emptyset$. We used L_2 transformations to create Z_1 and Z_2 .

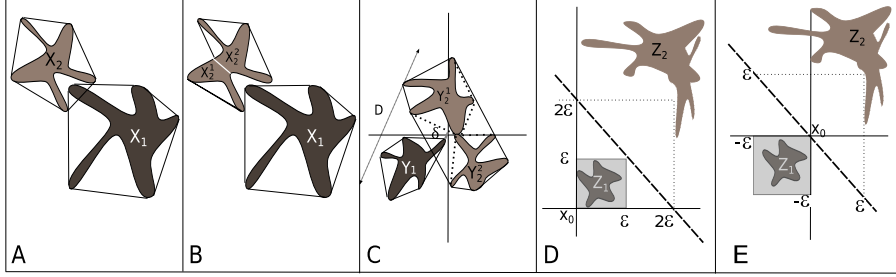


Fig. 1. This figure illustrates the proof of Theorems 7 and 11. **(A)** We see the two original sets and their convex hulls (outline). **(B)** X_2 is separated in two parts such that the convex hull of each part is linearly separable from the convex hull of X_1 . **(C)** We then apply a linear transformation such that all points x in X_1 end up below x_0 (the origin) and all points in X_2^2 have first coordinate above x_0 and all points in X_2^1 have the second coordinate above x_0 . By again drawing the convex hulls we can determine the minimal distance between the convex hull of Y_1 and the convex hulls of Y_2^1 and Y_2^2 . **(D)** Then we apply f . Z_1 will become enveloped by a regular hypercube, and Z_2 will lie outside a hypercube with edges that are $L_2 = 2$ times as long. The separating plane is drawn as a dashed line. **(E)** Equivalently, a translated picture is used in the proof of the leaky rectified linear activation function. Instead of Figure 1 D, we now have Z_1 below the axis. ϵ is chosen to be the same as the diameter D of the set in Figure 1 C.

The initial sets that the network needs to separate are denoted by X_k , see Figure 1 A. After applying a linear classifier to the initial sets, these will be denoted by Y_k such that after applying the transformation $w_j^T x + b_j$ on all $x \in X_1$, we get Y_1 , see Figure 1 C. When we apply the activation function to elements in Y_k , we denote the resulting set by Z_k , shown in Figures 1 D and 1 E. This means that a neural network with a single hidden layer with L_2 nodes, can transform X_k into Z_k . The following theorem is a generalization of Theorem 5 from [1]. Again, this is straightforward and does not require any changes to the proof. The theorem will make use of the following Lemma.

Lemma 4 *Two finite sets are linearly separable if and only if there exists a one-dimensional projection that maps the sets to linearly separable sets.*

Proof. Suppose we have two sets that are linearly separable. Let l be the hyperplane that separates the data. Project the data on the axis that is orthogonal to the hyperplane. By this, l will be collapsed into a point that lies at the threshold between the two separated sets. If we have a one-dimensional projection of the two sets, and a threshold t , let m be the hyperplane orthogonal to the projection axis containing t . Then the sets will be linearly separated by m .

Theorem 5 Let X_1 and X_2 be finite and convexly inseparable. Let $w_{ij}^T x + b_{ij}$ be linear classifiers of X_2^j and X_1^i such that for all i, j

$$w_{ij}^T x + b_{ij} \leq 0 \quad \forall x \in X_1^i \quad (6)$$

$$w_{ij}^T x + b_{ij} > 0 \quad \forall x \in X_2^j. \quad (7)$$

Let $W_i = [w_{i1}, \dots, w_{iL_2}]$ and $b_i = [b_{i1}, \dots, b_{iL_2}]$. Let $W = [W_1, \dots, W_{L_1}]$, $b = [b_1^T, \dots, b_{L_1}^T]^T$ and $Z_k = \{f(W^T x + b) \mid x \in X_k\}$ for $k \in \{1, 2\}$. Also, let $Z_1^i = \{f(W^T x + b) \mid x \in X_1^i\}$. Here f is again semi-positive. Then Z_1 and $C(Z_2)$ are disjoint, so Z_1 and Z_2 are convexly separable. For this we need $L_1 L_2$ nodes.

Proof. Define $Z_{2i} = \{f(W_i^T x + b_i) \mid x \in X_2\}$ and $Z_{1i}^t = \{f(W_i^T x + b_i) \mid x \in X_1^t\}$. Notice that these sets are projections of Z_2 and Z_1^t . Apply Theorem 3 on X_1^i , X_2 and their images Z_{1i}^i and Z_{2i} under the transformation $f(W_i^T \cdot + b_i)$. Then we have

$$C(Z_{1i}^i) \cap C(Z_{2i}) = \emptyset \quad i \in \{1, \dots, L_1\}. \quad (8)$$

With Lemma 4, we then also have that

$$C(Z_1^i) \cap C(Z_2) = \emptyset \quad i \in \{1, \dots, L_1\}. \quad (9)$$

Since $Z_1 \subset \bigcup_{i=1}^{L_1} C(Z_1^i)$, we have $Z_1 \cap C(Z_2) = \emptyset$. Therefore Z_1 and Z_2 are convexly separable. We needed L_1 linear transformations to separate a single part of X_2 from all parts of X_1 . So in total we need $L_1 L_2$ transformations to create Z_1 and Z_2 .

From Theorem 5 and 3 we can conclude that any two sets that are disjoint, can be made linearly separable by a network with two hidden layers that applies the function f as above and has $L_2 L_1$ and L_1 nodes per respective layer.

3 Main Result

We can generalize Theorems 3 and 5 to a larger set of activation functions. We need the following lemma for that. For simplicity we define $0/0 = 0$.

Lemma 6 For a given $\delta > 0$ and a fixed $L_2 > 0$, let $f : \mathbb{R} \rightarrow \mathbb{R}$ be increasing with a left asymptote to zero and $\inf_{x_0} \frac{f(x_0)}{f(x_0 + \delta)} < \frac{1}{L_2}$. Then $\exists x_0 \in \mathbb{R}, \epsilon > 0$ such that $\forall x \leq x_0 : f(x) \in [0, \epsilon]$ and $\forall x \geq x_0 + \delta : f(x) > L_2 \epsilon$.

Proof. Choose x_0 and ϵ such that $f(x_0) = \epsilon$ and $\frac{f(x_0)}{f(x_0 + \delta)} < \frac{1}{L_2}$. Let $x \leq x_0$. Then $f(x) \leq f(x_0) = \epsilon$. Let $x \geq x_0 + \delta$, then $f(x) \geq f(x_0 + \delta) > f(x_0) L_2 = \epsilon L_2$.

Lemma 6 puts a constraint on the speed with which the function f increases (near $-\infty$). We need $f(x_0 + \delta) \geq L_2 f(x_0)$. So if we move by δ , the value of the function will be multiplied by L_2 . Notice that this is a very rapidly growing function. If a function does not satisfy this constraint, we find there is a

minimum distance δ needed between the $C(Y_1)$ and $C(Y_2^1) \cup C(Y_2^2)$. Lemma 6 also implies that the function should have a left asymptote to zero, however, we can shift an activation function with a different left asymptote such that this holds and then shift it back later using Corollary 13. This way, the lemma, and therefore Theorems 7 and 8, holds for all commonly used activation functions. We will compute the distance δ for the sigmoid, hyperbolic tangent, rectified linear function and leaky rectified linear function in Corollary 15.

We would further like to note that we will from now on define $\delta = \min_j \delta_j$ where

$$\delta_j = \inf_{x,y} \{\|x - y\| \mid x \in C(Y_1), y \in C(Y_2^j)\} \quad (10)$$

is the smallest distance between the convex hulls of two sets.

Theorem 7 *Let X_1 and X_2 be two convexly separable sets, with a finite number of points in \mathbb{R}^n . So $C(X_1) \cap X_2 = \emptyset$ and $X_2 = \bigcup_{j=1}^{L_2} X_2^j$ with $L_2 \in \mathbb{N}$, $X_2^j \subseteq X_2$ such that $C(X_1) \cap C(X_2^j) = \emptyset$ for each j . Let $f : \mathbb{R} \rightarrow \mathbb{R}$ be increasing with a left asymptote to zero and define δ as in Equation 10, such that*

$$\inf_{x_0} \frac{f(x_0)}{f(x_0 + \delta)} < \frac{1}{L_2}. \quad (11)$$

For x_0 satisfying this inequality, let $w_j^T x + b_j$ be linear classifiers of X_2^j and X_1 such that for all j

$$\sup_{x \in X_1} \{w_j^T x + b_j\} = x_0, \quad (12)$$

$$w_j^T x + b_j \geq x_0 + \delta \quad \forall x \in X_2^j. \quad (13)$$

Let $W = [w_1, \dots, w_{L_2}]$, $b = [b_1, \dots, b_{L_2}]^T$ and $Z_k = \{f(W^T x + b) \mid x \in X_k\}$, $k \in \{1, 2\}$. Then Z_1 and Z_2 are linearly separable.

Proof. Choose, using Lemma 6, an $x_0 \in \mathbb{R}$ and $\epsilon > 0$ such that for all $x \leq x_0$ we have $f(x) \leq \epsilon$ and for all $x \geq x_0 + \delta$ we have $f(x) > L_2 \epsilon$. For all $x \in X_1$ we have $w_j^T x + b_j \leq x_0$. So $f(w_j^T x + b_j) \leq \epsilon$. Therefore, Z_1 is contained in a positive hypercube $[0, \epsilon]^{L_2}$. For all $x \in X_2$ there is a j such that $x \in X_2^j$. For $x \in X_2^j$ we have that $w_j^T x + b_j \geq x_0 + \delta$. So $f(w_j^T x + b_j) > L_2 \epsilon$ for at least one coordinate, and all other coordinates are larger than 0. Therefore, $Z_2 \subseteq [0, \infty)^{L_2} \setminus [0, L_2 \epsilon]^{L_2}$, see Figure 1 D. The convex hulls of these two sets can be separated by the hyperplane $\sum_{i=1}^{L_2} x_i = \epsilon L_2$. This is because the convex hull of Z_1 is contained in the hypercube with edges ϵ and the convex hull of Z_2 is bounded by the separating hyperplane.

Theorem 8 *Let X_1 and X_2 be finite and convexly inseparable. Let $f : \mathbb{R} \rightarrow \mathbb{R}$ be increasing with a left asymptote to zero, define δ as in Equation 10 such that $\inf_{x_0} \frac{f(x_0)}{f(x_0 + \delta)} < \frac{1}{L_2}$. For x_0 satisfying this inequality, let $w_{ij}^T x + b_{ij}$ be linear*

classifiers of X_2^j and X_1^i such that for all i, j

$$\sup_{x \in X_1^i} \{w_{ij}^T x + b_{ij}\} = x_0, \quad (14)$$

$$w_{ij}^T x + b_{ij} \geq x_0 + \delta \quad \forall x \in X_2^j. \quad (15)$$

Let $W_i = [w_{i1}, \dots, w_{iL_2}]$ and $b_i = [b_{i1}, \dots, b_{iL_2}]$. Let $W = [W_1, \dots, W_{L_1}]$, $b = [b_1^T, \dots, b_{L_1}^T]^T$ and $Z_k = \{f(W^T x + b) \mid x \in X_k\}$ for $k \in \{1, 2\}$. Also, let $Z_1^i = \{f(W^T x + b) \mid x \in X_1^i\}$. Then Z_1 and Z_2 are convexly separable.

Proof. Define $Z_{2i} = \{f(W_i^T x + b_i) \mid x \in X_2\}$ and $Z_{1i}^t = \{f(W_i^T x + b_i) \mid x \in X_1^t\}$. Notice that these sets are projections of Z_2 and Z_1^t . Apply Theorem 7 on X_1^t , X_2 and their images Z_{1i}^t and Z_{2i} under the transformation f . Then we have

$$C(Z_{1i}^t) \cap C(Z_{2i}) = \emptyset \quad i \in \{1, \dots, L_1\}. \quad (16)$$

With Lemma 4, we then also have that

$$C(Z_1^i) \cap C(Z_2) = \emptyset \quad i \in \{1, \dots, L_1\}. \quad (17)$$

Since $Z_1 \subset \bigcup_{i=1}^{L_1} C(Z_1^i)$, we have $Z_1 \cap C(Z_2) = \emptyset$. Therefore Z_1 and Z_2 are convexly separable.

So we see that both Theorems 3 and 5 can be generalized to increasing functions with a left asymptote to zero. We still need two layers with $L_1 L_2$ and L_1 nodes respectively. However, we also need a minimal separation δ between the convex hulls of the two sets (in Euclidean distance) after applying the first linear transform. We formalize this in the following theorem:

Theorem 9 *Given finite disjoint sets X_1 and X_2 with a disjoint convex hull decomposition with L_1 and L_2 sets in the partitions, and given an increasing activation function f with a left asymptote to zero, we can linearly separate X_1 and X_2 using an artificial neural network with an input layer, a layer with $L_1 L_2$ hidden nodes, a layer with L_1 hidden nodes and an output layer.*

Proof. We can assume X_1 and X_2 are convexly inseparable and have linear classifiers $w_{ij}^T x + b_{ij}$ as in Theorem 5. The corresponding δ is always greater than zero, and scales with (w, b) . Since $f \neq 0$ we can scale δ such that $\inf_{x_0} \frac{f(x_0)}{f(x_0 + \delta)} < \frac{1}{L_2}$. Then apply Theorem 8. We need $L_1 L_2$ affine transformations for separating the L_2 parts of X_2 and the L_1 parts of X_1 . Then f is applied to all transformations. A neural network can do this by learning the weights and biases of the affine transformations and then applying f . Now we have L_2 pairs of convexly separable sets, which can be linearly separated using Theorem 7. For each j we need to find an affine plane that separates X_2^j from X_1 . This means we have to learn L_2 affine transformations before applying f , which can be done by a neural network with L_2 nodes. Now we have two linearly separable sets, which can be separated by using a linear classifier as the output layer, which proves the theorem.

Note that this proof implies that we can separate X_1 and X_2 independent of the distance between Y_1 and Y_2 , so independent of δ . The learning algorithm should be able to scale the weights and biases such that the sets can be separated no matter how small δ was originally.

We can also prove a similar theorem for the leaky rectified linear activation function, which does not have a left asymptote to zero. However, we need to prove Lemma 10 first. The diameter of a set A is defined as $\text{diam}(A) = \sup\{\|x - y\| \mid x, y \in A\}$.

Lemma 10 Suppose $D = \text{diam}(Y_1 \cup Y_2)$, δ as in Equation 10, and $f(x) = c_2x$ for $x \geq 0$ and $f(x) = c_1x$ for $x \leq 0$, where $c_2 > c_1$. Then $\mu(\delta, D) \triangleq \inf_{x_0} \frac{f(x_0) - f(x_0 - D)}{f(x_0 + \delta) - f(x_0 - D)}$ is reached at $x_0 = 0$.

Proof. We have four cases:

(a) $x_0 < 0, x_0 + \delta < 0$, then $x_0 - D < 0$:

$$\mu(\delta, D) = \inf_{x_0} \frac{c_1x_0 - c_1x_0 + c_1D}{c_1x_0 + c_1\delta - c_1x_0 + c_1D} \quad (18)$$

$$= \inf_{x_0} \frac{1}{\frac{\delta}{D} + 1} = \frac{1}{\frac{\delta}{D} + 1} \quad (19)$$

(b) $x_0 < 0, x_0 + \delta \geq 0$, then $x_0 - D < 0$:

$$\mu(\delta, D) = \inf_{x_0} \frac{(c_1 - c_1)x_0 + c_1D}{(c_2 - c_1)x_0 + c_2\delta + c_1D} \quad (20)$$

$$= \inf_{x_0} \frac{1}{\frac{(c_2 - c_1)x_0}{c_1D} + \frac{c_2\delta}{c_1D} + 1} \quad (21)$$

$$= \frac{1}{\frac{c_2\delta}{c_1D} + 1} \quad (22)$$

for x_0 increasing to zero.

(c) $x_0 \geq 0, x_0 - D < 0$, then $x_0 + \delta \geq 0$:

$$\mu(\delta, D) = \inf_{x_0} \frac{c_1x_0 - c_1x_0 + c_1D}{(c_2 - c_1)x_0 + c_2\delta + c_1D} \quad (23)$$

$$= \inf_{x_0} \frac{\frac{(c_2 - c_1)x_0}{c_1D} + 1}{\frac{(c_2 - c_1)x_0}{c_1D} + \frac{c_2\delta}{c_1D} + 1} \quad (24)$$

which is an increasing function on the interval $[0, D)$. Therefore the infimum will be at $x_0 = 0$.

(d) $x_0 \geq 0, x_0 - D \geq 0$, then $x_0 + \delta > 0$:

$$\mu(\delta, D) = \inf_{x_0} \frac{c_2x_0 - c_2x_0 + c_2D}{c_2x_0 + c_2\delta - c_2x_0 + c_2D} \quad (25)$$

$$= \inf_{x_0} \frac{1}{\frac{\delta}{D} + 1} = \frac{1}{\frac{\delta}{D} + 1} \quad (26)$$

Since cases (a) and (d) are equal, and since $c_2/c_1 > 1$ we see that the infimum is assumed at the value $x_0 = 0$.

Now we are ready to prove the following theorem for leaky rectified linear functions. Because of Lemma 10 we can assume $x_0 = 0$.

Theorem 11 *Suppose we have X_1 and X_2 as in Theorem 3. Define δ as in Equation 10. Let $f(x) = c_1x$ for $x \leq 0$ and $f(x) = c_2x$ for $x \geq 0$ be increasing with*

$$\frac{-f(-D)}{f(\delta) - f(-D)} < \frac{1}{L_2}. \quad (27)$$

Then Z_1 and Z_2 as defined in Theorem 3 are linearly separable.

Proof. Let $\epsilon = -f(-D)$. Then $\forall -D \leq x \leq 0$ we have $-\epsilon \leq f(x) \leq 0$. And $\forall x > \delta$ we have $f(x) > (L_2 - 1)\epsilon$. For all $x \in X_1$ we know $-D \leq w_j^T x + b_j \leq 0$. Therefore for all $x \in Y_1$ we have $-\epsilon \leq f(x) \leq 0$ and for all $x \in Y_2$ we have that $f(x) \geq -\epsilon$ and there exists a j such that $f(x_j) > (L_2 - 1)\epsilon$. See Figure 1 E. Therefore the convex hulls of Z_1 and Z_2 are disjoint.

We will not prove a version of Theorem 5 for the leaky rectified linear activation function because this is straightforward and the proof is the same as the proof of Theorem 5. But we can conclude that for leaky rectified linear activation functions a network that consists of two layers and L_2 and L_1L_2 nodes respectively, can achieve linear separability. If δ is not large enough, there are two options now: the network could learn to scale the function appropriately, or this could be done manually by increasing the fraction c_2/c_1 . In the next section we will explore some consequences of these results. We will also provide a way to calculate L_1 and L_2 .

4 Corollaries

We can generalize the results from Section 3 to any number of sets (Corollary 12), by using the similar result in Sections 3.4 and 3.5 from [1] as a foundation. After stating this result we will show that we can apply any translation to the function f in the above theorems while retaining their validity (Corollary 13). Then we will provide a cheap way to estimate L_1 and L_2 in the disjoint convex hull decomposition (Algorithm 1) and we will calculate δ , for the most commonly used activation functions (Corollary 15).

Corollary 12 *For any number of sets, the above holds, with adjusted L_1 and L_2 . By using the result from Section 3.4 and 3.5 on multiple sets from [1] as a foundation, it is easy to see that the same reasoning will apply to Theorems 3, 5, 7, 8 and 11.*

We can generalize the theorems still a little more by showing that they also hold for translated versions of the activation function that satisfies the constraints.

Corollary 13 Suppose $f(-\infty) = c$, f is increasing and we have $\inf_{x_0} \frac{f(x_0) - c}{f(x_0 + \delta) - c} \leq \frac{1}{L_2}$. Then Theorem 7 still holds. Moreover, for left and right translations of f , Theorem 7 still holds.

Proof. Define $g = f - c$. Then $g(-\infty) = 0$ and $\inf_{x_0} \frac{g(x_0)}{g(x_0 + \delta)} \leq \frac{1}{L_2}$. Therefore the theorem holds for g . Adding a constant to the linear separable sets Z_1 and Z_2 does not affect their separability. So the theorem holds for f .

Let $g(x) = f(x + c)$ be a translated version of f . If we apply $g(y) = g(w_j^T x + b_j) = f(w_j^T x + b_j + c)$ we see that we could just subtract c from x_0 to get back to the original theorem. Therefore left and right translation of functions is allowed.

We need a way to estimate L_1 and L_2 for arbitrary datasets. Since it is difficult to decompose the sets in a high dimensional space, we found a way to do it in a low dimensional space. This will allow for a rough upper bound on L_1 and L_2 but does not guarantee that the smallest disjoint convex hull decomposition can be found.

Lemma 14 If we have a disjoint convex hull decomposition of a projection of our dataset, this partition will also form a disjoint convex hull decomposition of the original dataset.

Proof. Suppose $P(X_1)$ and $P(X_2)$ are n -dimensional projections of X_1 and X_2 . Assume we have disjoint convex hull decompositions $\widehat{P(X_1)} = \bigcup_{j=1}^{L_1} C(P(X_1)^j)$ and $\widehat{P(X_2)} = \bigcup_{j=1}^{L_2} C(P(X_2)^j)$ such that $\widehat{P(X_1)} \cap \widehat{P(X_2)} = \emptyset$. Now take $\widehat{X_1} = \bigcup_{j=1}^{L_1} C(X_1^j)$ such that $P(X_1)^j = P(X_1^j)$ and $\widehat{X_2} = \bigcup_{j=1}^{L_2} C(X_2^j)$ such that $P(X_2)^j = P(X_2^j)$. Then we see that $C(P(X_1^j)) \cap C(P(X_2^i)) = \emptyset$. Therefore $P(C(X_1^j)) \cap P(C(X_2^i)) = \emptyset$. So we can conclude that $C(X_1^j) \cap C(X_2^i) = \emptyset$. So also $\widehat{X_1} = \bigcup_{j=1}^{L_1} C(X_1)^j$ and $\widehat{X_2} = \bigcup_{j=1}^{L_2} C(X_2)^j$ are a disjoint convex hull decomposition.

We can estimate the number of sets in the convex hull decomposition using Lemma 14 as follows: Take a random projection of the datasets, preferably a one-dimensional projection. Then find the disjoint convex hull decomposition of this projection alone. This is easy in one dimension as it can be done by counting how often one switches from one set to the other when traversing through the projection. This number is an upper bound for L_1 and L_2 , but a very coarse one, as we are using a random projection. So it is necessary to repeat this for many more random projections and minimize for L_1 and L_2 . We use this procedure in algorithm 1 to find a reasonable estimate for L_1 and L_2 .

We can prove that this algorithm will actually give a disjoint convex hull decomposition. The algorithm has complexity $\mathcal{O}(n^2)$. The worst-case scenario for the while-loop contributes a factor n and computing the inner-products also contributes a factor n . It is fair to mention the algorithm also depends on the dimension of the data and on the number of random projections that is used. Both can be quite large. The number of random projections needs to be significantly larger than the dimension to get good results. Also notice that adding *count* to L_1 and L_2 in lines 19 and 20 is naïve and can easily be improved.

Algorithm 1 Estimating L1 and L2

```

1: Input: two sets, X and Y, of n-dimensional data,
2:     the sets are finite and disjoint.
3: while size of overlap is smaller than size of previous overlap
4:   count the number of times we do this
5:   calculate mx  $\leftarrow$  mean of X
6:   my  $\leftarrow$  mean of Y
7:   for x in X, y in Y
8:     project x and y on my-mx
9:   calculate cx  $\leftarrow$  maximum of the projection of X
10:  cy  $\leftarrow$  minimum of the projection of Y
11:  create overlapX  $\leftarrow$  all x with projection at least cy
12:  overlapY  $\leftarrow$  all y with projection at most cx
13:  replace X by overlapX, Y by overlapY
14: for t in 1:10000
15:   create random vector
16:   for x,y in overlapX and overlapY
17:     calculate px  $\leftarrow$  projection of x on random vector
18:     py  $\leftarrow$  projection of y on random vector
19:   count the number of set changes from px to py
20:   L1  $\leftarrow$  number of set changes+count
21:   L2  $\leftarrow$  number of set changes+1+count
22: minimize L1 and L2

```

Corollary 15 *For a dataset with convex hull decomposition in L_2 sets, recall that the minimal distance between the $C(Y_1)$ and $C(Y_2^j)$ is $\delta = \inf_j \delta_j$, see Equation 10. For the sigmoid the minimal δ needed for separation equals $\ln(L_2)$. For a shifted hyperbolic tangent the minimal δ equals $\frac{1}{2} \ln(L_2)$. For the ReLU the minimal δ equals 0. For the leaky rectified linear activation function $\delta = (L_2 + 1) \frac{c_2}{c_1} D$. Or equivalently, $\frac{c_2}{c_1} = \frac{\delta}{D(L_2 - 1)}$, then we are able to separate any two sets with a leaky rectified linear activation function.*

Proof. Note that proving that the limit becomes smaller than $1/L_2$ implies that the infimum also becomes smaller than $1/L_2$. For practical purposes we will use the limit in this proof.

Sigmoid. The sigmoid function is written as $\sigma(x) = \frac{e^x}{1+e^x}$. If we calculate

$$\frac{\sigma(x_0)}{\sigma(x_0 + \delta)} = \frac{e^{x_0}}{1 + e^{x_0}} \frac{1 + e^{x_0} e^\delta}{e^{x_0} e^\delta} = \frac{e^{-\delta} + e^{x_0}}{1 + e^{x_0}} \quad (28)$$

and then take the limit $x_0 \rightarrow -\infty$ we see that equation 28 goes to $e^{-\delta}$. To get this smaller than $1/L_2$ we need $\delta > \ln(L_2)$.

Hyperbolic tangent. We start by writing a shifted hyperbolic tangent $\tanh(x) + 1$ out in terms of exponentials. If we calculate

$$\frac{\tanh(x_0) + 1}{\tanh(x_0 + \delta) + 1} = \frac{2}{1 + e^{-2x_0}} \frac{1 + e^{-2x_0} e^{-2\delta}}{2} \quad (29)$$

$$= \frac{1 + e^{-2x_0} e^{-2\delta}}{1 + e^{-2x_0}} \quad (30)$$

and then take the limit $x_0 \rightarrow -\infty$ we get that equation 29 goes to $e^{-2\delta}$. To get this smaller than $1/L_2$ we need $\delta > \frac{1}{2} \ln(L_2)$. So also for the hyperbolic tangent we have with Corollary 13 that $\delta > \frac{1}{2} \ln(L_2)$.

Rectified linear function. We did not need any δ in the proof for the rectified linear function, so the minimal δ equals zero.

Leaky rectified linear activation function. With Lemma 10 we get:

$$\frac{-f(-D)}{f(\delta) - f(-D)} = \frac{Dc_1}{\delta c_2 + Dc_1}, \quad (31)$$

where f denotes the leaky rectified linear function. To get this smaller than $\frac{1}{L_2}$ we need $\delta = (L_2 + 1) \frac{c_2}{c_1} D$.

5 Simulation Results

We tested the ideas in Sections 3 and 4 empirically. We trained several networks with different sizes and activation functions on the first two classes (number classes 0 and 1) of the MNIST dataset [9]. We calculated the minimal distance between these two sets and found $\delta = 3.96$. This is a sufficient distance for any of the activation functions we used, which means the network is able to use weights close to 1. Next we estimated L_1 and L_2 . For this dataset with more than 12000 data points, we found $L_1 = 6$ and $L_2 = 6$. That would mean that a network with 36 nodes in the first layer and 6 nodes in the second would be sufficient to linearly separate the data in the two sets.

Several hidden layer sizes were tested. All networks had a depth of three, hence four layers of nodes. An input layer with 784 nodes, two hidden layers with the sizes mentioned before, and an output layer with 2 nodes which acts as a classifier. Linear separability, as discussed in this paper, precisely means that this output layer can classify the input sets perfectly.

We compared the ReLU, sigmoid, leaky ReLU and tanh networks trained for 150 epochs using stochastic gradient descent optimization. For the leaky ReLU the slope was set to the standard value of 0.2. We implemented the linear classifier multi-layer perceptron in the neural network framework Chainer v2.0 [10]. We regard the training capabilities of this framework as a black box sufficient for our simulation needs. The results are displayed in Figure 2. Indeed as expected, the network with the hidden layer sizes estimated based on the proposed

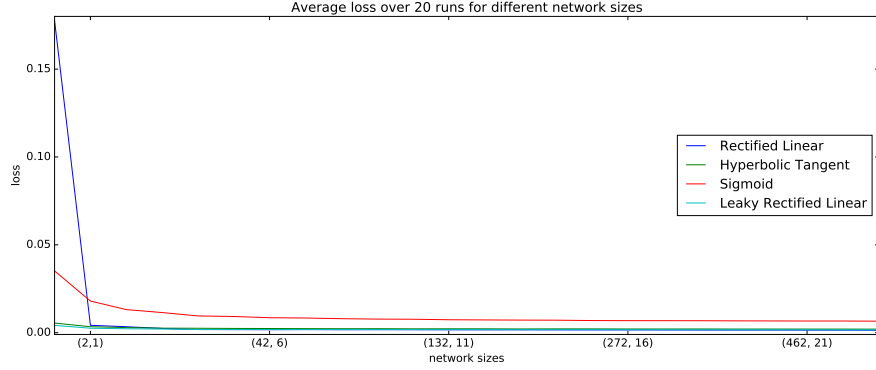


Fig. 2. This figure shows the averaged loss over 20 runs. The activation functions we used are the ReLU, the tanh, the sigmoid and the leaky ReLU. 25 networks of different sizes were trained using the above functions as activation functions. The numbers in brackets on the x-axis denote the sizes of the hidden layers. The size of the input layer was 784, the size of the output layer was 2. The network was trained with 150 epochs and a batch size of 150.

theoretical analysis $((36, 6))$ performs very well. We see clearly that the losses barely decrease for larger networks. The error is not yet zero for the predicted network but this may be explained by the imperfect training. The ReLU network performs bad for the smallest network. This may be explained by the fact that the ReLU maps a lot of information to zero even though it has the smallest δ of the tested activation functions. The sigmoid consistently has a larger loss than the other functions. This is not necessarily predicted by the theory since the sigmoid's δ is only a factor 2 larger than the hyperbolic tangent's δ . Also interesting is the very good performance of the leaky ReLU network. This could be caused by not mapping a lot of information to zero like the ReLU as well as having two options to compensate for the δ .

All activation functions seem to imply that there exists a slightly smaller network that can achieve linear separability on the test set. A better algorithm for determining L_1 and L_2 can probably confirm this.

6 Discussion

The practical contribution of this article is heuristic. It is widely believed that deep neural networks need less nodes in total than shallow neural networks to solve the same problem. Our theory presents an upper bound on the number of nodes that a shallow neural network will need to solve a certain problem. Therefore, a deep neural network will not need more nodes. The theory does not give an optimal architecture, nor a minimum on the number of nodes. Still it

is useful to have an inkling about the correct network size for solving a certain problem.

Contrary to what An et al. [1] claim, their theory does not show why ReLU networks have a superior performance. We extended their theory to all commonly used activation functions. Only the leaky rectified linear networks seem to be at a disadvantage, but test results show the opposite. We think the differences between the functions may be caused by the scaling that needs to be done during learning. The linear functions and also the hyperbolic tangent are very easy to scale. Tweaking the sigmoid to the best slope can be quite difficult.

Some issues which we have not addressed in this article are worth mentioning. For example, we cannot make any statements about generalization performance of the networks. Of course, it is generally known that a network with too many parameters will not generalize well. So it is wise to use a network that is as small as possible, or even a bit smaller. This paper contributes an estimate for the number of nodes that is an absolute maximum. It should never be necessary to use more nodes than this estimate. We do not give a necessary number of nodes but rather an upper bound. A bound that is necessary and sufficient would be optimal, but this is a much harder problem to solve.

Another problem is that we do not know what will happen if we use too few nodes. The number of nodes that we estimated will guarantee linear separability. If the number of nodes is too small to achieve linear separability, performance on the training set will be reduced, but it is difficult to say anything about performance on the test set. We also do not know what will happen to the number and distribution of nodes as we increase the number of layers. An extension of the theory to an arbitrary number of layers would be very interesting.

Furthermore, in the simulations we cannot guarantee that the learning algorithm achieves zero error, even though it is possible in theory. The reason is that the algorithm does not always find the absolute minimum. Therefore it is hard to judge from the results whether the predicted network size is performing as expected.

Even though we already find small L_1 and L_2 , more elaborate simulations could use another algorithm to find the convex hull decomposition. Random projections are cheap to use, but they will always find a pair L_1, L_2 such that $L_2 = L_1 + 1$. (We found $L_1 = L_2$ since no random projections were necessary.) This is a serious constraint because the first layer of the network consists of $L_1 L_2$ nodes, and will therefore always be very large if L_1 and L_2 are similar size. An idea would be to use a method that uses higher dimensional projections. It is also not guaranteed that Algorithm 1 performs well on other input sets. A better algorithm might perform well on all types of input sets.

The results show a stunning performance of the leaky ReLU activation. More research is needed to understand why this is the case. There clearly is more to the performance of a neural network than revealed in this article. Still, it is an important result to have an estimation of sufficient network sizes for certain activation functions. It would also be interesting to see the effect of the slope of

a leaky ReLU and the distance between the datasets on the performance of the network.

This paper provides a heuristic explanation why ReLU and perhaps leaky ReLU networks are easier to train than tanh and sigmoid networks. We give an upper bound on the number of nodes that is needed to achieve linear separability on the training set for feedforward networks with two hidden layers. It is still unclear how this generalises to more layers, which poses an interesting question for further research. Furthermore, our theory does not yet address convolutional networks, however it does represent a foundation for exploring their superior performance in an extension of this work.

References

1. An, S., Boussaid, F., Bannamoun, M.: How can deep rectifier networks achieve linear separability and preserve distances? 32nd International Conference on Machine Learning **37**, 514-523 (2015).
2. Hornik, K., Stinchcombe, M., White, H.: Multilayer feedforward networks are universal approximators. Neural Networks **2**, 359-366 (1989).
3. Arteaga, C., Marrero, I.: Universal approximation by radial basis function networks of Delsarte translates. Neural Networks **46** 299-305 (2013).
4. Kůrková, V., Kainen, P.C., Kreinovich, V.: Estimates of the number of hidden units and variation with respect to half-spaces. Neural Networks **10.6** 1061-1068 (1997).
5. Fujita, O.: Statistical estimation of the number of hidden units for feedforward neural networks. Neural Networks **11** 851-859 (1998).
6. Yuan, H.C., Xiong, F.L., Huai, X.Y.: A method for estimating the number of hidden neurons in feed-forward neural networks based on information entropy. Computer and electronics in agriculture **40** 57-64 (2003).
7. Baum, E.B.: On the capabilities of multilayer perceptrons. Journal of Complexity **4** 193-215 (1988).
8. Sonoda, S., Murata, N.: Neural network with unbounded activation functions is universal approximator. Applied and Computational Harmonic Analysis **43.2** 233-268 (2017).
9. LeCun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-Based Learning Applied to Document Recognition. Proceedings of the IEEE **86.11** 2278-2324 (1998).
10. Tokui, S., Oono, K., Hido, S., Clayton, J.: Chainer: a Next-Generation Open Source Framework for Deep Learning. Proceedings of Workshop on Machine Learning Systems (LearningSys) in The Twenty-ninth Annual Conference on Neural Information Processing Systems (NIPS) (2015).

Socially smart software agents entice people to use higher-order theory of mind in the Mod game

Kim Veltman¹, Harmen de Weerd^{1,2}, and Rineke Verbrugge¹

¹ Institute of Artificial Intelligence, University of Groningen

² Professorship User Centered Design, Hanze University of Applied Sciences

Abstract. In social settings, people often need to reason about unobservable mental content of other people, such as their beliefs, goals, or intentions. This ability helps them to understand, to predict, and even to influence the behavior of others. People can take this ability further by applying it recursively. For example, they use second-order theory of mind to reason about the way others use theory of mind, as in ‘Alice believes that Bob does not know about the surprise party’. However, empirical evidence so far suggests that people do not spontaneously use higher-order theory of mind in strategic games. Previous agent-based modeling simulations also suggest that the ability to recursively apply theory of mind may be especially effective in competitive settings. In this paper, we use a combination of computational agents and Bayesian model selection to determine to what extent people make use of higher-order theory of mind reasoning in a particular competitive game, the Mod game, which can be seen as a much larger variant of the well-known rock-paper-scissors game.

We let participants play the competitive Mod game against computational theory of mind agents. We find that people adapt their level of theory of mind to that of their software opponent. Surprisingly, knowingly playing against second- and third-order theory of mind agents entices human participants to apply up to fourth-order theory of mind themselves, thereby improving their results in the Mod game. This phenomenon contrasts with earlier experiments about other strategic one-shot and sequential games, in which human players only displayed lower orders of theory of mind.

1 Introduction

Theory of mind, the ability to reason about unobservable mental content such as beliefs and goals of others [17], plays an important role in many social interactions. Indeed, many researchers have claimed that a higher proficiency in theory of mind is related with pro-social behavior [11], social competences [13], and negotiation skills [24]. But while some experiments show impressive theory of mind skills in adults, people are typically slow to take advantage of their theory of mind ability in strategic settings [10,2,25,9]. In this paper, we let participants

interact with artificial theory of mind agents to determine to what extent these agents can encourage and train people in their use of theory of mind.

The human ability for theory of mind is not limited to reasoning about the goals, desires, and beliefs concerning world facts of others. People can also use their theory of mind to reason about the way others use theory of mind. For example, people use *second-order theory of mind* to understand a sentence such as “Alice *knows* that Bob *knows* that Carol is throwing him a birthday party”, by reasoning about what Alice knows about what Bob knows. Experimental results from story comprehension tasks show that people are able to use orders of theory of mind beyond second-order theory of mind as well. In these tasks, adults perform much better than chance on story comprehension questions that explicitly involve theory of mind reasoning up to the fourth order [12,19].

In this article, we let participants play the Mod game against artificial theory of mind agents, and estimate their level of theory of mind reasoning using random-effects Bayesian model selection [18]. Simulation studies show that the ability to make use of higher-order theory of mind can be particularly effective in competitive settings [8,23,4,6]. In particular, results from agent simulations in a variant of the Mod game suggest that both first-order and second-order theory of mind can greatly benefit players, while the use of orders of theory of mind beyond the second hardly provide additional benefits [23]. However, results in the matching pennies game show that in this simple competitive game, many people rely on simpler, behavior-based strategies when engaging with artificial agents [20]. In this paper, we consider an extension of the matching pennies game known as the Mod game. The Mod game has a structure that is similar to that of matching pennies, but due to the larger number of possible actions, the Mod game should make it easier for participants to distinguish between strategies. This may help people to detect the use of theory of mind in their opponent, and encourage them to make use of higher orders of theory of mind themselves.

For this research, we let human participants play against virtual agents that we developed to determine the effectiveness of making use of increasingly higher orders of theory of mind [22]. This allows us to precisely control and monitor the mental content of the opponents faced by the participants, including their application of theory of mind. By letting the participants play against virtual agents, we can analyze the participant data in a more controlled setting. This allows us to diagnose the human behavior. But that’s not the only benefit of using virtual agents: By letting human participants train with virtual agents that are programmed to reason in a certain fashion, we can stimulate people to improve their own reasoning. For example, when a participant plays against a higher-order theory of mind agent, which makes it harder for the participant to win the game, this might be an incentive for the participant to actually try harder and also apply higher-order theory of mind, since the agent is consistent in using the higher orders of theory of mind.

The remainder of this paper is structured as follows. Section 2 and Section 3 introduce the Mod game and a range of strategies that agents and humans could use. In Section 4, we describe how random-effects Bayesian model selection may

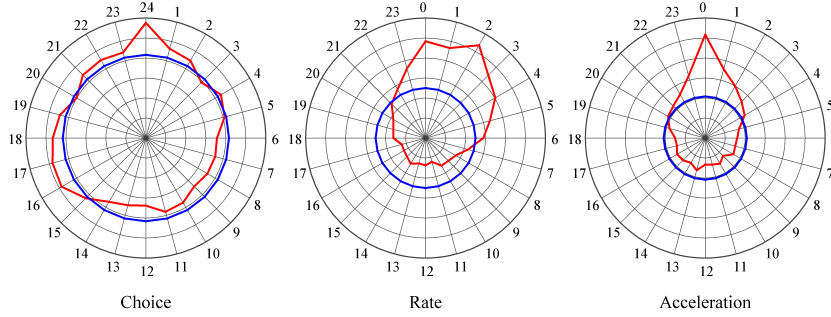


Fig. 1: Histograms over 24 choices, rates, and accelerations of human behaviour in the Mod game. In each graph, the blue curve shows the expected results from random behaviour, while the red curve shows the participant behaviour (reconstructed from [7]).

help to gauge agents' and participants' reasoning strategies from their behavior. Section 5 delineates our experiment in which human participants played the Mod game against agents that used different orders of theory of mind. In Section 6, the experimental results are presented and in Section 7 we close the article and conclude that virtual agents can indeed be used to support people in using higher orders of theory of mind in a competitive game such as Mod.

2 Mod Game

The Mod game is an n -player generalization of rock-paper-scissors [23]. It has been introduced by Frey and Goldstone [8] as a way to reveal patterns in individual theory of mind strategies. In the Mod game, n players simultaneously choose a number in the range $\{1, \dots, m\}$, for $m > n > 1$. Players gain one point for every opponent that has chosen the number that is exactly one lower than their own choice. For example, a player that has chosen the number 4 gains a point for every opponent that has chosen number 3. The only exception to this rule is that players that have chosen number 1 gain one point for every player that has chosen number m , hence the name 'mod' game; for the goal '+1 mod m '. To visualize the rules of the game to human participants, actions are arranged in a circle (see Figure 2 for $m = 24$).

Note that in the Mod game, each action is dominated by some other actions, similar to games such as rock-paper-scissors. In fact, the Mod game is equivalent to a non-zero-sum version of rock-paper-scissors for $n = 2$ and $m = 3$. The Mod game has a mixed-strategy Nash equilibrium in which each action is chosen with equal probability. When all players play according to this randomizing strategy, none of the players has an incentive to change his or her strategy.

However, participant behavior in repeated Mod games deviates from the Nash equilibrium, as depicted in Figure 1 (reconstructed from [7]). The figure shows the aggregate participant data (red line in left graph) and the idealized randomizing behavior (blue line) over 100 rounds of play and $m = 24$. Participant choices (red line in left-most graph) appear to be approximately random, with a slight bias towards 24. However, *participant rates*, defined as the difference in choice between two subsequent rounds, shows a clear deviation from the Nash equilibrium. As Figure 1 shows, participants (red line in the middle graph) are less likely to select numbers that are 7 to 21 ahead of their previous choice. Instead, they are most likely to choose a number that is 0 to 4 higher than their previous choice. If participants were to play according to the Nash equilibrium, however, each rate should be equally likely (blue line). *Participant acceleration*, which is defined as the change in participant rate, shows a similar effect. Figure 1 (red line in right-most graph) shows that participants tend to vary little in their rate. That is, a participant who chose a number in the last round that was 2 higher than the number in the round before that is mostly likely to choose the number that is 2 higher in the current round than his choice in the previous round. In addition, Figure 1 also shows that participants that vary their acceleration do so by a small amount. Again, Nash equilibrium play would result in each acceleration being equally likely (blue line).

In our experiment, participants play a specific variant of the Mod game with $m = 24$ actions and $n = 2$ player. We follow [7] in this choice, as choosing m relatively large allows us to better distinguish reasoning strategies behind participants' behavior. For the remainder of the paper, we will only consider this specific variant of the Mod game.

3 Strategies in the Mod game

The Mod game, as outlined in Section 2, can be played using a variety of strategies. In this section, we describe a number of these strategies. These include strategies based on the use of theory of mind, as well as simple behavior-based strategies. Table 1 shows several possible Mod game strategies, which we selected for the following reasons. The *ToM* strategies correspond to the main focus of this research. The Win-Stay-Lose-Shift strategy appears to be used often and has been selected because of its popularity.

The self- and other-regarding strategies were chosen to see whether participants may use a simpler strategy, based on either their own or the opponent's recent behavior. It seems that some participant data could be explained purely by choosing a rate (differences between the players' previous and current number) by which they change their number based on either their own or the opponent's previous choice. These self- and other-regarding strategies include the strategies where a participant consistently chooses 1, 2, 3 or 4 higher than he/she or the opponent did in the previous round. In this section, we describe these strategies in detail. To avoid confusion, in the remainder, we will refer to focal agents as if they were male, while we will refer to their opponents as if they were female.

Table 1: We consider eight possible strategies for playing the Mod game, including four behavior-based strategies and four theory of mind strategies.

Strategies	Behavior-based	ToM
Other regarding	✓	
Self regarding	✓	
Win-Stay-Lose-Shift	✓	
ToM_0	✓	
ToM_1		✓
ToM_2		✓
ToM_3		✓
ToM_4		✓

3.1 Behavior-based strategies

While participants may benefit from using theory of mind in the Mod game, the Mod game can be played using strategies that do not model unobservable mental content of others. In our Bayesian RFX-BMS analysis, we therefore also consider a number of behavior-based strategies. A player that uses a behavior-based strategy responds to actions observed in the last round of play only.

Self-regarding strategy An agent that follows a *self-regarding* strategy reacts to the action it has performed in the previous round, while it ignores all actions of the opponent. The self-regarding strategy depends on two free parameters. The drift parameter k determines the amount of drift a player experiences in every round, so that an agent that follows a self-regarding strategy with drift k tends to choose the number that is k higher (modulo 24) than the action he performed in the previous round.

In addition, the choice probability p determines the strength of the self-regarding tendency. For example, an agent that follows a self-regarding strategy with $k = 2$ selects the action that is 2 higher than his previous choice with probability p , while each other action has a probability $\frac{1}{23}(1-p)$ of being selected.

Other-regarding strategy The *other-regarding* strategy is similar to the self-regarding strategy, except that an agent that follows the other-regarding strategy reacts to the previous action of his opponent rather than his own previous action. Like the self-regarding strategy, the other-regarding strategy relies on a drift parameter k and choice probability p . An agent that follows an other-regarding strategy with drift k selects the action that is exactly k higher (modulo 24) than his opponent's action in the previous round with probability p . Each other action is selected with probability $\frac{1}{23}(1-p)$. Note that if an agent plays according to an other-regarding strategy with $k = 1$, the agent tends to play the action that would have won in the previous round.

Win-Stay, Lose-Shift (WSLS) strategy An agent that follows the *win-stay, lose-shift* (WSLS) strategy bases his current decision on the outcome of the

previous round. If the agent won the previous round, he will repeat his previously chosen action with probability p , while each of the other 23 action is selected with probability $\frac{1}{23}(1-p)$. However, if the agent did not win the previous round, he will repeat his previously chosen action with probability $1-p$, while each of the other 23 action is selected with probability $p/23$. The single parameter p is a free parameter.

3.2 Theory of mind strategies

In addition to behavior-based strategies, we include agents that are capable of using theory of mind while playing the Mod game. Each additional order of theory of mind provides an agent with an additional prediction of his opponent's behavior, so that a ToM_k agent has $k+1$ models of opponent behavior [23]. These agents have been previously used to determine the extent of participants' use of higher-order theory of mind in the simpler matching pennies setting [20]. Below, we briefly describe these agents. A full mathematical model of these agents can be found in [23].

Zero-order theory of mind A zero-order theory of mind (ToM_0) agent has no theory of mind at all, and is therefore unable to attribute mental content to others. In particular, a ToM_0 agent cannot form the belief that his opponent is trying to obtain a high score. Instead, the ToM_0 agent forms zero-order beliefs about the actions the opponent will play in future rounds of the game.

In our agent model, a ToM_0 agent forms beliefs $b^{(0)}$ about the actions of the opponent. For each number $i = 1, \dots, 24$, the ToM_0 agent has a belief $b^{(0)}(i)$ that represents what he believes to be the likelihood that most of his opponents will select to play that number. The ToM_0 agent acts on these beliefs by choosing the number that maximizes his score. For example, if a ToM_0 agent strongly believes that number 4 will be selected by his opponent, the agent should choose to play number 5 himself.

After every round, the ToM_0 agent updates his zero-order beliefs to reflect the actual outcome. An agent-specific learning speed $\lambda \in [0, 1]$ determines the relative influence of the current observation on the agent's beliefs. For example, a ToM_0 agent with zero learning speed ($\lambda = 0$) does not update his beliefs at all. Such an agent selects the same action in every round. A ToM_1 agent with the maximal learning speed ($\lambda = 1$), on the other hand, completely replaces his zero-order beliefs after each observation, and forgets all information obtained from previous rounds³.

To account for small deviations between participant choices and the ToM_0 agent strategy, we make use of the so-called 'softmax' probabilistic policy [3,20]. That is, in addition to the learning speed λ , the ToM_0 agent strategy has an addition parameter β that controls the magnitude of behavioral noise.

³ Note that a ToM_0 agent with learning speed $\lambda = 1$ is identical to an agent following an other-regarding strategy with $k = 1$.

First-order theory of mind Unlike the ToM_0 agent, a first-order theory of mind (ToM_1) agent reasons about the goals of others, and therefore believes that his opponent may be trying to maximize her score as well. To predict the behavior of his opponent, the ToM_1 agent attributes his own thought process to her. That is, a ToM_1 agent considers the possibility that, while the agent reacts to the actions of his opponent, the opponent is reacting to the actions of the agent. For example, if the agent has played 4 in the previous round, he believes that his opponent is more likely to play 5. After all, that is what the ToM_1 agent would do himself if he has observed his opponent playing 4 in the previous round.

Although the ToM_1 agent models his opponent as being able to use zero-order theory of mind, agents in our setup do not know the extent of the abilities of their opponent for certain. Rather, a ToM_1 agent has two models of opponent behavior, one based on zero-order theory of mind and one on first-order theory of mind. Through repeated interaction, a ToM_1 agent learns which of his models best describes the behavior of his opponent. Based on this information, a ToM_1 agent may therefore choose to play as if he were a ToM_0 agent, and ignore the predictions of his first-order theory of mind.

Note that the ToM_1 agent strategy does not introduce any new free parameters. Like the ToM_0 agent strategy, there are two parameters: the learning speed λ and the magnitude of behavioral noise β .

Higher orders of theory of mind For each additional order of theory of mind k , an agent generates an additional prediction of opponent behavior, by attributing his own $(k - 1)$ st-order theory of mind thought process to his opponent. For example, a ToM_2 agent models his opponents as ToM_1 agents, in addition to his zero-order and first-order theory of mind models of opponent behavior. As a result, a ToM_k agent has $k + 1$ hypotheses for the action that will be chosen by his opponent, with corresponding predictions. Based on the accuracy of these predictions, the ToM_k agent can therefore choose to behave according to $k + 1$ patterns of behavior.

4 Random-effects Bayesian model selection

In this paper, we attempt to encourage participants in their use of theory of mind through interactions with artificial theory of mind agents. To determine what level of theory of mind reasoning a participant is engaging in at different points throughout the experiment, we make use of a technique known as group-level random-effects Bayesian model selection (RFX-BMS), introduced by Stephan and colleagues [18]. Unlike fixed-effects Bayesian model selection, which assumes that there is one strategy that best describes the actions of all participants, random-effects Bayesian model selection models participants as individuals who may differ in the strategy they use while playing the game. Strategies are treated as random effects that occur with an unknown but fixed probability

in the population. A group of participants represents a random sample drawn from these strategies.

Random-effects Bayesian model selection estimates what distribution of strategies best fits the experimental data. Each strategy s generates pieces of evidence $p(y|s)$ representing the probability that choosing actions according to strategy s will result in some observed data y . Using these model evidences and the participant data, random-effects Bayesian model selection estimates the relative frequencies of strategies in the general population.

To determine to what extent a participant makes use of theory of mind while playing the Mod game, we compare the observed behavior of each participant with the predicted behavior of computational agents following the strategies described in Section 3. That is, the model evidence $p(y|s)$ generated by a given model is the probability that that model will perform the same action as the participant, given the history of moves observed by the participant. This combination of our theory of mind agents and RFX-BMS has been previously used to accurately recover the level of theory of mind reasoning of Devaine’s Bayesian theory of mind agents [3], which indicates that this method can overcome biases in the designer’s choice of how to implement theory of mind.

5 Experimental Setup

To determine whether interacting with artificial theory of mind agents encourages the use of theory of mind, we let human participants play the Mod game against artificial theory of mind agents. Participants played the two-player Mod game with 24 actions, as described in Section 2.

5.1 Participants

Sixteen participants were included in this study, of which eight were male and eight were female, all students and all over the age of 18 ($M = 21.5$, $SD = 2.3$). The experiment was conducted in English, and all participants were sufficiently skilled in reading and understanding the English language.

Before starting with the experiment, all participants gave informed consent about partaking and about the use of the data obtained by the experiment for the purpose of this study.

5.2 Experimental design

Each participant played the Mod game against four different opponents: a ToM_1 agent, a ToM_2 agent, a ToM_3 agent, and an agent whose order was randomized each round. This randomizing agent would randomly select to respond as if it were a first-order, second-order, or third-order theory of mind agent in each round. That is, during a block of twenty rounds, the randomizing agent would randomly select a reasoning strategy twenty times.

Note that participants never played against a ToM_0 agent. During a pilot study, we discovered that ToM_0 agents and ToM_1 agents exhibit the same behavior when playing the Mod game against a human participant. The ToM_0 agent believes that the best predictor for a participant's future behavior is this/her behavior in the previous round. As a result, the ToM_0 agent tends to select the number that is 1 higher than the number last chosen by the participant.

The ToM_1 agent, on the other hand, believes that the opponent wants to win the game. By taking the perspective of the opponent, the ToM_1 agent believes that the opponent will choose the number that is 1 higher than his own last choice. For example, suppose that the agent chose 23 in the last round and the participant played 24. In this case, the ToM_1 agent will believe that the opponent is going to play 24 again, since the ToM_1 agent believes that the participant is a ToM_0 agent who believes that the agent is going to play 23 again. Following this reasoning, the agent decides to play 1, which is exactly 1 higher than the participant's previous choice (24). However, this yields the same result as the ToM_0 agent, who also chose to play the number that was 1 higher than the participant's previous choice. Due to this effect, we decided not to include the ToM_0 agent in our experiment.

Each block consisted of twenty rounds of the Mod game, in which participants played against the same opponent for all twenty rounds. Between blocks, the opponent was changed to a ToM agent of a different order. The order in which participants faced the different opponents was randomly drawn from four possible sequences: $[?, 1, 2, 3]$; $[3, ?, 1, 2]$; $[2, 1, 3, ?]$; and $[1, ?, 3, 2]$, where the question mark (?) represents a randomizing agent, whose order of theory of mind reasoning was randomized each round. The different sequences were chosen to rule out the effect of sequence on the performance. Participants were informed about the ToM order of the opponent they were playing against, except in the blocks in which they faced the randomizing agent. During the rounds against the randomizing agent, the order of the opponent was not shown to the participants, in order to see if the participants' behavior also changed if the ToM order of the opponent was not known.

Each participant faced the four different opponents in two of the aforementioned sequences, so every participant played 2 repetitions each of 4 different opponents. In total, every participant played 40 rounds against every opponent. A certain opponent was played against for twenty rounds before the agent's order changed. The number of twenty rounds per opponent was chosen because people typically need many trials before showing higher-order reasoning behavior [9].

5.3 Procedure and materials

Participants first read some short information about theory of mind and the different orders that were used in this experiment (ToM_1 , ToM_2 , and ToM_3). Note that while this procedure may have primed participants to make use of theory of mind strategies, evidence from Marble Drop experiments shows that even in this case, participants may have difficulties implementing higher-order theory of mind strategies [10,16]. Participants then read an explanation of the



Fig. 2: Interface of the Mod24 Game experiment.

experiment itself, including the rules of the Mod game and an explanation of the interface. Before the experimental rounds started, the participants completed three test rounds to confirm they understood the interface. As the participants started with the experimental trials, they saw an interface with twenty-four buttons, numbered from 1 to 24, placed in a circle (see Figure 2). The placement of the buttons was constant throughout the whole experiment. The interface also showed what level of theory of mind the agent used (except during the randomizing agent blocks). The participants could also see how many rounds they had already played, how many rounds they would play against the same opponent, the current score of both players, and the chosen actions of both players in the previous round of play.

At the end of each block, participants were informed that they would continue playing against a new opponent. After four blocks, participants could take a break before continuing with the next four blocks. Once all eight blocks were finished, another pop-up was shown, informing the participants that the experiment was finished. Upon finishing the experiment, the participants were thanked for their cooperation and received payment. Each participant was equally compensated for their effort: the reward was not dependent on the points obtained during the experiment.

6 Results

In this section, we present the results of the experiment described in Section 5. To determine to what extent participants made use higher-order theory of mind reasoning, we applied RFX-BMS analysis (see Section 4) to the choices of both the participants as well as their computer opponents.

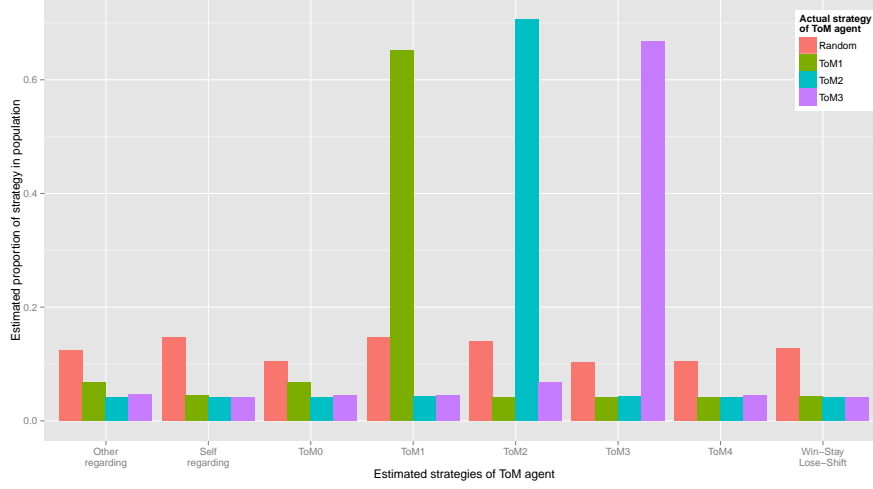


Fig. 3: Estimated strategies of the agents based on their choices throughout the experiment.

6.1 Agent behavior

Figure 3 shows the estimated strategies of all four agents in our experiment. By analyzing the choices of the computer opponents, we make sure that the behavior exhibited by the agents in interaction with human participants can be accurately distinguished as theory of mind strategies by our RFX-BMS analysis.

As Figure 3 shows, the RFX-BMS estimation correctly classifies agent behavior of the ToM_1 , ToM_2 , and ToM_3 agents as consistent with the corresponding order of theory of mind reasoning. Moreover, the randomizing agent is classified as using a strategy that is equally consistent with all strategies. This shows that the RFX-BMS estimation can accurately distinguish different order of theory of mind reasoning, and also considers the randomizing agent to be unpredictable.

6.2 Participant behavior

No significant influence of the sequence on the participant scores was found ($F = 0.483, p = 0.487$). That is, there is no reason to believe that the sequence in which the agents appeared affected the performance of the participants in any way. In the remainder, we therefore present results that are aggregated across the different sequences.

Figure 3 shows the estimated strategies of the participants in our experiment. Since we expect that participants adjust their strategy to their opponent, we analyze participant behavior for each of the four agent opponents separately.

The RFX-BMS analysis for the blocks in which participants played against a ToM_1 agents are indicated by green bars in Figure 4. The figure shows that a

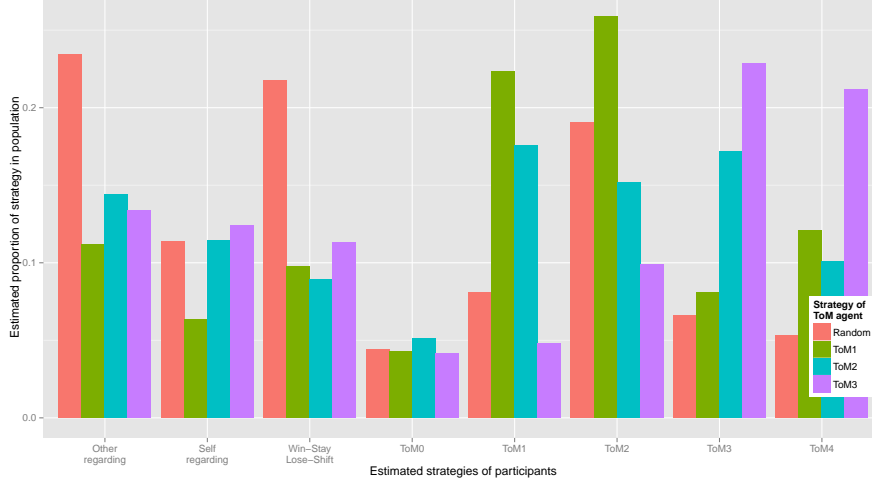


Fig. 4: Estimated strategy use of participants in the Mod game across the four different opponent types.

large proportion of participants are classified as making use of theory of mind. When playing against a ToM_1 agent, an estimated 25.8% of the participants made use of a ToM_2 strategy, while another estimated 22.4% of the participants used first-order theory of mind. It seems that, according to the participants, the ToM_2 strategy was the best strategy. This also makes sense in relation to the theory, if you want to win a ToM game, it is most beneficial to think exactly one order of theory of mind higher than your opponent does. That is, when playing against a ToM_1 opponent, it is best to make use of second-order theory of mind.

When playing against a ToM_2 agent, participants mainly played according to first-order, second-order, or third-order theory of mind, as indicated by the blue bars in Figure 4. Note that it would be optimal for participants to reason using third-order theory of mind, while following first-order theory of mind would mean that the computer opponent consistently outsmarts the participant.

The purple bars of Figure 4 show the estimates strategies of participants while playing against a ToM_3 agent. According to our RFX-BMS analysis, the strategies that best explained the largest percentages of the population are the ToM_3 and ToM_4 strategies, that represent an estimated 22.9% and 21.2% of the population respectively. Interestingly, the higher the order of theory of mind reasoning of the opponent, the longer participants took to react. In addition, those participants that were classified as using a higher order of theory of mind reasoning also obtained higher scores on average.

Note that for each of the ToM_1 , ToM_2 , and ToM_3 opponent agents, theory of mind strategies were estimated to explain the largest proportions of participant choices. However, this is not true when participants play against the randomizing

ToM agent, as shown by the red bars in Figure 4. While the behavior of a sizable proportion of participants is consistent with the *ToM*₂ strategy, participant behavior is generally better described as other-regarding or as a win-stay, lose-shift strategy. That is, unlike when participants play against the *ToM*₀, *ToM*₁, or *ToM*₂ agent, participant behavior against the random *ToM* opponent is better explained by behavior-based strategies than it is by theory of mind strategies.

7 General discussion and conclusion

In many social settings, people rely on the use of theory of mind to guide their actions. However, experimental evidence shows that adults more readily show signs of theory of mind reasoning in story comprehension tasks than in strategic games and that children learn to apply theory of mind in games much later than in story tasks [12,19,5,1,16].

In this paper, we let participants play a specific competitive game known as the Mod game [8,21] against artificial theory of mind agents, to determine to what extent participants use theory of mind reasoning. Our results show that, when playing against agents that use increasingly higher orders of theory of mind, a larger proportion of participants is estimated to make use of higher-order theory of mind. Participants that played against a first-order theory of mind agents relied mostly on first-order or second-order theory of mind themselves, while participants that played against a third-order theory of mind agent were better described as using third-order or fourth-order theory of mind. Moreover, when faced against a randomizing agent, who randomly selected to play as if it were a first-order, second-order, or third-order at the start of each round, participants were better described as relying on simpler behavior-based strategies.

In strategic games, participants are typically found to rely on low orders of theory of mind, and to be slow to adjust their level of theory of mind reasoning to more sophisticated opponents [10,2,25,9]. Earlier empirical research suggests that the use of first-order and second-order theory of mind in games can be facilitated by creating a believable story or insightful visual representation around an abstract problem [15,3], by creating a clear competition or negotiation setting [9,24], or by providing stepwise training from games that require zero-order *ToM* to second-order *ToM* games [14].

Impressively, our results in the Mod game suggest that participants even make use of an unprecedented *fourth-order* theory of mind reasoning when playing against a higher-order theory of mind opponent in the Mod game, even though they only faced each opponent for twenty consecutive rounds of play. Moreover, participant that were classified as using higher order of theory of mind tended to obtain higher scores. In future work, it would be interesting to determine to what extent participants exhibit the same behavior when facing more than one opponent (i.e., for $n > 2$).

Interestingly, our results in the Mod game show higher levels of theory of mind reasoning than results of similar experiments with the matching pennies game [20,3]. One possible explanation is that adults in the matching pennies

experiment played 60 rounds in which they were given 1300 ms each to make a decision. In contrast, in our experiment, participants took an average 3700 ms to respond in the easiest blocks. That is, participants in the matching pennies experiment may not have had enough time to engage in higher-order theory of mind reasoning, and relied on simpler behavior-based strategies instead.

Alternatively, while the Mod game is arguably more difficult than matching pennies, it is easier to distinguish strategies from one another in the Mod game than in matching pennies. In matching pennies, players can only choose two possible actions. As a result, the choice of a particular action gives little information about the underlying strategy. In our Mod game experiment, however, there are 24 possible actions, which allows players to more easily interpret the actions of their opponent and draw conclusions about the underlying strategy.

The surprisingly high level of theory of mind reasoning of participants can also be partially explained by the representation of the game. In our specification of theory of mind agents, we assumed that the zero-order theory of mind agent reasons about opponent actions. However, because the actions are meaningfully arranged in a circle, it is also possible to define a zero-order theory that reasons about the change of actions (i.e. rate) of opponents rather than their actual choices. A first-order theory of mind agent in this former specification would have similar behavior to a zero-order theory of mind agent in the latter specification. That is, the representation of the zero-order theory of mind model is important in determining at which order of theory of mind participants are reasoning [15].

Conclusion

Since the estimated level of theory of mind reasoning of participants increases with the increasing order of theory of mind reasoning of the artificial opponent, our results suggest that artificial agents can indeed encourage people to make use of higher order of theory of mind reasoning, thereby providing people with better results.

References

1. Arslan, B., Verbrugge, R., Taatgen, N., Hollebrandse, B.: Teaching children to attribute second-order false beliefs: A training study with feedback. In: CogSci. pp. 108–113 (2015)
2. Camerer, C., Ho, T., Chong, J.: A cognitive hierarchy model of games. *Quarterly Journal of Economics* 119(3), 861–898 (2004)
3. Devaine, M., Hollard, G., Daunizeau, J.: The social Bayesian brain: Does mentalizing make a difference when we learn? *PLoS Computational Biology* 10(12), e1003992 (2014)
4. Devaine, M., Hollard, G., Daunizeau, J.: Theory of mind: Did evolution fool us? *PloS ONE* 9(2), e87619 (2014)
5. Flobbe, L., Verbrugge, R., Hendriks, P., Krämer, I.: Children’s application of theory of mind in reasoning and language. *Journal of Logic, Language and Information* 17(4), 417–442 (2008)

6. Franke, M., Galeazzi, P.: On the evolution of choice principles. In: Szymanik, J., Verbrugge, R. (eds.) *Proceedings of the Second Workshop Reasoning About Other Minds: Logical and Cognitive Perspectives*. CEUR Workshop Proceedings, vol. 1208, pp. 11–15 (2014)
7. Frey, S.: *Complex collective dynamics in human higher-level reasoning: A study over multiple methods*. Ph.D. thesis, Indiana University (2013)
8. Frey, S., Goldstone, R.L.: Cyclic game dynamics driven by iterated reasoning. *PloS ONE* 8(2), e56416 (2013)
9. Goodie, A.S., Doshi, P., Young, D.L.: Levels of theory-of-mind reasoning in competitive games. *Journal of Behavioral Decision Making* 25(1), 95–108 (2012)
10. Hedden, T., Zhang, J.: What do you think I think you think?: Strategic reasoning in matrix games. *Cognition* 85(1), 1–36 (2002)
11. Imuta, K., Henry, J.D., Slaughter, V., Selcuk, B., Ruffman, T.: Theory of mind and prosocial behavior in childhood: A meta-analytic review. *Developmental Psychology* 52(8), 1192–1205 (2016)
12. Kinderman, P., Dunbar, R.I., Bentall, R.P.: Theory-of-mind deficits and causal attributions. *British Journal of Psychology* 89(2), 191–204 (1998)
13. Liddle, B., Nettle, D.: Higher-order theory of mind and social competence in school-age children. *Journal of Cultural and Evolutionary Psychology* 4(3–4), 231–244 (2006)
14. Meijering, B., van Rijn, H., Taatgen, N., Verbrugge, R.: I do know what you think I think: Second-order theory of mind in strategic games is not that difficult. In: *CogSci*. pp. 2486–2491 (2011)
15. Meijering, B., Van Maanen, L., Van Rijn, H., Verbrugge, R.: The facilitative effect of context on second-order social reasoning. In: *CogSci*. pp. 1423–1428 (2010)
16. Meijering, B., Taatgen, N.A., van Rijn, H., Verbrugge, R.: Modeling inference of mental states: As simple as possible, as complex as necessary. *Interaction Studies* 15(3), 455–477 (2014)
17. Premack, D., Woodruff, G.: Does the chimpanzee have a theory of mind? *Behavioral and Brain Sciences* 1(04), 515–526 (1978)
18. Stephan, K.E., Penny, W.D., Daunizeau, J., Moran, R.J., Friston, K.J.: Bayesian model selection for group studies. *Neuroimage* 46(4), 1004–1017 (2009)
19. Stiller, J., Dunbar, R.I.: Perspective-taking and memory capacity predict social network size. *Social Networks* 29(1), 93–104 (2007)
20. de Weerd, H., Diepgrond, D., Verbrugge, R.: Estimating the use of higher-order theory of mind using computational agents. *The B.E. Journal of Theoretical Economics* (2017)
21. de Weerd, H., Verbrugge, R., Verheij, B.: Theory of mind in the Mod game: An agent-based model of strategic reasoning. In: *Proceedings ECSI*. pp. 128–136 (2014)
22. de Weerd, H., Verheij, B.: The advantage of higher-order theory of mind in the game of limited bidding. In: *Proceedings Workshop ‘Reasoning about other Minds’, CEUR Workshop Proceedings*, vol. 751, pp. 149–164 (2011)
23. de Weerd, H., Verbrugge, R., Verheij, B.: How much does it help to know what she knows you know? An agent-based simulation study. *Artificial Intelligence* 199–200, 67–92 (2013)
24. de Weerd, H., Verbrugge, R., Verheij, B.: Negotiating with other minds: The role of recursive theory of mind in negotiation with incomplete information. *Autonomous Agents and Multi-Agent Systems*, 31, 250–287 (2017)
25. Wright, J.R., Leyton-Brown, K.: Beyond equilibrium: Predicting human behavior in normal-form games. In: *Proceedings of the Twenty-Fourth Conference on Artificial Intelligence*. pp. 901–907 (2010)

Recommending Treatments for Comorbid Patients Using Word-Based and Phrase-Based Alignment Methods

Elie Merhej¹, Steven Schockaert², T. Greg McKelvey^{4*}, and
Martine De Cock^{1,3}

¹ Ghent University, Ghent, Belgium

{`elie.merhej`,`martine.decock`}@ugent.be

² Cardiff University, Cardiff, United Kingdom

`schockaerts1@cardiff.ac.uk`

³ University of Washington Tacoma, Tacoma, USA

`mdecock@u.washington.edu`

⁴ KenSci, Seattle, USA

`Greg@KenSci.com`

Abstract. The problem of finding treatments for patients diagnosed with multiple diseases (i.e. a comorbidity) is an important research topic in the medical literature. In this paper, we propose a new data driven approach to recommend treatments for these comorbidities using word-based and phrase-based alignment methods. The most popular methods currently rely on combining specific information from individual diseases (e.g. procedures, tests, etc.), then aim to detect and repair the conflicts that arise in the combined treatments. This proves to be a challenge especially in the cases where the studied comorbidities contain large numbers of diseases. In contrast, our methods rely on training a translation model using previous medical records to find treatments for newly diagnosed comorbidities. We also explore the use of additional criteria in the form of a drug interactions penalty and a treatment popularity score to select the best treatment in the case where multiple valid translations for a single comorbidity are available.

1 Introduction

The number of comorbid patients keeps rising [7]. Since clinical guidelines focus on treating every disease separately, simply combining these guidelines to find treatments for comorbid patients may introduce unwanted conflicts in the combined treatments. For example, a patient diagnosed with Duodenal Ulcer (DU) is required to stop the use of any anti-inflammatory medicine, including aspirin. On the other hand, a patient diagnosed with a Transient Ischemic Attack (TIA) is required to take aspirin as part of the treatment. When a patient is diagnosed with both DU and TIA, combining the individual treatments of these

* University of Washington Occupational and Environmental Medicine Fellow

diseases introduces a conflict around the use of aspirin. Therefore, a system that can automatically suggest treatments for comorbid patients is a valuable aid for clinicians.

Existing algorithmic approaches aim to combine the important information found in the clinical guidelines of the individual diseases of a comorbidity. First, they make use of computer interpretable guidelines with dedicated languages [6,3] that capture all the essential information of clinical guidelines, such as tests, procedures, etc. Then, they try to detect the different types of conflicts that arise between the procedures inside the combined treatment. Finally, they provide techniques to resolve these detected conflicts in order to find a valid treatment for a given comorbidity.

In this paper, we tackle the problem of recommending treatments for comorbid patients from a very different angle, namely by treating it as a Statistical Machine Translation (SMT) problem. The main idea is to look at a set of diagnoses (i.e. a comorbidity) as a source sentence that we want to translate into a set of procedures (i.e. a treatment), considered as a target sentence. In other words, every diagnosis in a comorbidity is considered as a “word” in a “sentence”, and similar to the methods used to automatically translate a text sentence from one language to another, we aim to translate a comorbidity into a treatment. In order to do that, we train the translation models on a corpus of medical records. As we will show in Section 4, we also take advantage of the ability of the translation system to find multiple treatment recommendations for a single comorbidity, which allows us to make use of additional criteria, such as a drug interactions penalty and a treatment popularity score to recommend the most optimized treatment.

The remainder of this paper is structured as follows: In Section 2, we discuss the advantages and drawbacks of various approaches that aim to find treatments for comorbidities. In Section 3, we provide some background on statistical machine translation systems. In Section 4, we propose a number of different algorithms used for treatment recommendations. We first demonstrate how a nearest neighbour baseline method (1-NN) can be used to solve our problem, then we present our methods based on word-based and phrase-based alignment. In Section 5, a comparative experimental evaluation on approximately 45K patient records from the MIMIC-III database, shows that our SMT approaches comfortably outperform 1-NN. Finally, we conclude with directions for future work in Section 6.

2 Related Work

Several *expert knowledge driven* approaches for recommending treatments for comorbid patients have already been proposed. Mainly, these approaches try to combine treatments of individual diseases, and repair the conflicts that arise in the combined treatments. In [16], general models for representing computer interpretable guidelines that express evidence as causation beliefs are presented. These models aim to detect the different types of interactions found in these

guidelines and specify their severities. In our previous work [11] based on [17,15], mitigation operators are used to detect and repair conflicts in combined treatments. When applied, these mitigation operators offer alternative treatments based on the information that they encode.

The existing expert knowledge driven methods require the availability of clinical guidelines encoded in a machine readable way. Such computer interpretable guidelines are not readily available except for a few of the most common diseases. This is reflected by the fact that, in the literature, the knowledge driven methods are only evaluated for a handful of specific comorbidities. The data driven approach that we propose in this paper can be directly applied to any given comorbidity, without the need for any expert knowledge encoded in a knowledge representation language. Hence, our approach is far more widely applicable than existing methods. In particular, to the best of our knowledge, there is no existing method that can be directly applied to recommend treatments for all the comorbidities that we evaluate our approach on in Section 5. The comorbidities considered in Section 5 are not hand picked, as is usually the case in work on recommending treatments for comorbid patients. Instead, we apply our approach to *all* the comorbidities that occur in a database of hospital discharge records.

The analysis of medical records has shown to have potential in developing and optimizing clinical treatment regimens. With the large amounts of available clinical data, there is a growing need to develop methods for automatically mining and analyzing this data. In [12], a method is proposed that aims at exploiting the rich information in doctor orders to improve clinical treatments. In [13], a system is implemented to use previously collected medical data to automatically identify the co-occurrence of patient events. These prior works are not aimed at recommending treatments for comorbid patients. To the best of our knowledge, we are the first to propose a fully *data driven* approach to this end.

Concerning the specific technique we employ (see Section 3), recently, neural network based models for machine translation have emerged as a popular alternative for SMT [2,5]. Such models avoid the need for an explicit alignment between the source and target sentences, and generally consist of an encoder network, which derives a vector representation for the source sentence, and a decoder network, which maps that vector onto a sentence from the target language. While such models have achieved state-of-the-art performance, they have two drawbacks, which are important for our purposes. First, they tend to need a large amount of training data, which means that they would not be suitable, in our context, for generating recommendations for rare (combinations of) diseases, whereas it is precisely in such rare cases that a recommendation system might be most helpful to a doctor. Second, alignment based models allow us to generate explanations as to why a certain treatment is proposed (e.g. normally disease A is treated using procedure P, but because disease B is also present, procedure Q is preferred). Generating supporting explanations from neural network approaches, on the other hand, is known to be a challenging problem.

3 Word-Based and Phrase-Based Alignment

The field of Machine Translation (MT) is concerned with automatically translating the meaning of a sentence (i.e. sequence of words) $s = [s_0, \dots, s_i]$ of a source language to another sentence $t = [t_0, \dots, t_j]$ of a target language. Statistical Machine Translation (SMT) systems learn how to make such translations in a purely data driven fashion, by comparing a large number of sentences from the source language with their corresponding translation in the target language. This only requires access to a sentence-aligned corpus, i.e. SMT systems figure out automatically which words or phrases from each source sentence correspond to which words or phrases from each target sentence. This process is called alignment, and plays a crucial role in our approach for recommending treatments.

The problem of translating a sentence s to t can be expressed, using the Bayes rule, as follows:

$$\operatorname{argmax}_t \Pr(t|s) = \operatorname{argmax}_t \Pr(t)\Pr(s|t) \quad (1)$$

In other words, given any sequence of words s in the source language, we want to find the sequence of words t in the target language which maximizes $\Pr(t|s)$. In (1), the prior probability $\Pr(t)$ is the language model probability, which models how natural the sequence t is in the target language, irrespective of the source sentence. On the other hand, the probability $\Pr(s|t)$ is the translation model probability, which models how likely s is as a translation of t .

3.1 Word-Based Alignment

As mentioned above, a key challenge for SMT is to identify, in a given sentence-aligned training corpus, which words from each sentence correspond to which words from their translation. In word-based alignment models, the probability that a given word s_i from the source sentence matches the word t_j from the target sentence is assumed to be independent of the other words in these sentences. Different ways of modelling the alignment probability can be found in [4,14].

3.2 Phrase-Based Alignment

One main disadvantage of word-based alignment models is that the context of a word is not taken into account when trying to find a suitable translation. For many words, the translation depends heavily on the surrounding words that occur in the same sentence. For example, when translating a sentence from English to French, the meaning of the word “right” in the English sentence “You are right” is totally different than in the sentence “Turn to the right”. Hence, different translations of the same word are expected. Indeed, the English word “right” in the first sentence is translated to the French word “raison”, while the same English word “right” in the second sentence is translated to the French word “droite”. The basis of phrase-based alignment is to decompose the input sentence from the source language into phrases (i.e. natural sequences of words),

find a translation for every phrase, then re-order these phrase translations and combine them to produce the target sentence.

A popular method for finding a phrase-based alignment is to learn the phrase translations from a corpus that has already been aligned using a word-based translation model [10]. In particular, the method relies on two word alignments: from source language to target language and from target language to source language. The two word alignments are then combined by doing an intersection of the aligned words, and then finding additional words that are not present in the intersection using different heuristics. Then, all aligned phrase pairs that are consistent with the combined word alignment are collected. Consistency is defined such that the words of a phrase pair are only aligned to each other, and are not aligned to words that are not inside the phrase pair. The extracted phrases and their translations constitute a phrase translation table. A phrase-based decoder (usually based on beam search) is finally used to generate the output sentence by re-ordering phrase translations. More details about this method can be found in [10].

N-best Phrase-based Alignment A phrase-based alignment model is generally expected to output the most likely translation of an input sentence. However, some applications benefit from having a set of alternative translations. A common method to find the n-best translations of an input sentence is to apply a phrase-based translation model to generate candidate translations. Subsequently, the probability score from the phrase-based translation model can be combined with additional features, where available, to produce a list of the n-best translations [9].

In Section 4, we use all methods described above, i.e. word-based alignment, phrase-based alignment, and n-best phrase-based alignment, to recommend treatments for comorbid patients.

4 Treatment Recommendation Methods

In this paper, we use alignment based translation models to map lists of diagnoses to lists of procedures. To train the translation model, we assume the availability of a large database of medical records, showing for each clinical event of each patient (e.g. a hospital admission) what diagnoses were made, what procedures were proposed by the providers and which drugs were prescribed. Some examples of such records are shown in Table 1. In medical records, the diagnoses and procedures of every admission are usually encoded using a standard encoding. For example in Table 1, the ICD-9 encoding is used, where the diagnosis codes “V3001” and “74783” refer to “Single liveborn, born in hospital, delivered by cesarean section” and “Persistent fetal circulation” respectively, and the procedure codes “9604” and “9671” refer to “Insertion of endotracheal tube” and “Continuous invasive mechanical ventilation” respectively. The patients in Table 1 are considered comorbid because they have a variety of different diagnoses that each require treatment.

Admission_ID	Diagnoses	Procedures	Drugs
...
196807	V3001, 74783, 7700, 7756, 7761, 77181, V053	9604, 9671, 9390, 0331, 9955	Heparin, Ampicillin_Sodium, Gentamicin, Pediatric_Vitamins, Potassium_Chloride, Sodium_Bicarbonate, Sodium_Chloride
100589	3940, 9982, 9971, 49320, 4239, E8788, 2449, 53081, 4019	3596, 370, 3723, 8856, 8872	Acetaminophen, Magnesium_Hydroxide, Atropine_Sulfate, Clonazepam, Hydrochlorothiazide, Levothyroxine_Sodium, Mirtazapine, Morphine_Sulfate, Nitroglycerin, Oxazepam, Oxycodone_Acetaminophen, Pantoprazole, Potassium_Chloride, Prochlorperazine, Simethicone, Venlafaxine, Vioxx, Zebeta
...

Table 1. Extract from medical records containing diagnose, procedure and drug information for every hospital admission.

An important difference between our medical treatment recommendation setting and the standard machine translation setting relates to the role of word ordering. While in standard settings, word ordering plays a critical role, the order in which diagnoses and procedures are ordered in our setting may be arbitrary. However, in practice, these orderings are not completely arbitrary, in the sense that the most important diagnoses and procedures are often listed first. As we will see in Section 5, this can be exploited by the translation model. For evaluation purposes, however, the task that we consider is to predict an (unordered) set of procedures.

We now explain the methods we propose for recommending a suitable treatment for a patient given their diagnosed comorbidity. While WBT, PBT and NPBT below are based on SMT approaches, we also consider a nearest neighbour baseline approach (1-NN) that utilizes treatments of previously diagnosed comorbidities in a simple way to find the best treatment for a new comorbidity case. We compare the effectiveness of these approaches in Section 5.

4.1 1-NN: 1-Nearest Neighbor with Jaccard Similarity (Baseline)

This method serves as a baseline approach to find a treatment for a newly diagnosed set of diseases, given a database that contains records of previous comorbidities, as well as their corresponding treatments. The idea behind this method is to find the most similar set of diagnoses across all records in the database, and to use the corresponding treatment for this case as the recommended treatment. To measure similarity, we use the Jaccard measure, which is a standard measure of similarity between sets, and is defined as $J(C, X) = |C \cap X| / |C \cup X|$ for two sets of diagnoses C and X . In the case where multiple records are equally similar, the first record found is then used.

4.2 WBT: Word-Based Translation

This is the first and most basic alignment method that we use to translate a given set of diagnoses C into a set of procedures T . A detailed explanation about this method is found in Section 3. When using word-based translation, every diagnosis in C is treated as a “source word”. It gets individually translated

into a procedure, i.e. a “target word”. Every translation is also given a corresponding translation probability. Essentially, there are two steps to training the translation model: finding the most likely alignment for every source word, and computing a probability table given the alignment. In practice, an Estimation Maximization (EM) algorithm can be used where the probability from a given alignment is initially estimated, and then the alignment is improved based on the new probabilities. This iterative process converges to give us the final probability table [4]. Finding the recommended treatment T then comes down to choosing for each diagnosis, the most likely translation (i.e. procedure) from the final probability table. When using this model, we are effectively ignoring any interactions between diagnoses and procedures. In other words, we are not taking into account conflicts that arise due to comorbidity.

4.3 PBT: Phrase-Based Translation

The main limitation of the word-based model in our context is the fact that interactions between procedures (and between procedures and diseases) are being ignored. This can be addressed by using a phrase-based translation model. Instead of translating every word of a sentence separately, this model groups sequential words together into phrases, then aims to find translations for these phrases. Finding the translation of a sentence consists then of combining these phrase translations.

In our case study, this translation model will solve the problem of arising conflicts between diagnoses (and procedures) of a comorbidity in the following way. When two or more “conflicting” diseases require a special treatment, the model will detect this instance provided that there are enough records in the training data in which these conflicting diseases occur. If the exact set of diagnoses has not been observed before, the model will try to split the set into subsets (i.e. phrases) that it can adequately translate, which contain as many diseases as possible. By doing so, all the conflicts between the diseases within each subset will be avoided effectively. However, any interactions between diseases that belong to different subsets will be unaddressed.

4.4 NPBT: N-Best Phrase-Based Translation With Cost Minimization

When using PBT, the recommended treatment is chosen as the most likely translation of the given list of diagnoses. This strategy is optimal in cases where we have no other information. When it comes to recommending procedures, however, we can take advantage of existing databases that describe the interactions between the drugs used during these procedures [1], among others. To this end, we use the phrase-based translation model to find the n -best translations, and then use the external knowledge to help select the best translation among these. In particular, we look to assign a penalty cost for every possible treatment, and then minimize this cost to find the best treatment.

NPBT+Drug: Drug Interactions Penalty The first type of information that we use to score treatments is the number and severity of drug interactions that are present in them. In previous work, we already found the usefulness of taking into account such interactions when recommending treatments [11]. In order to create a drug interactions penalty, we take into account the number of drug interactions as well as the severity of every drug interaction that is found inside a treatment.

In order to find the drug interactions penalty of a treatment, we need to translate it into a set of drugs. To translate procedures into drugs, we can proceed in the same way as for translating diagnoses into procedures, by learning a phrase-based alignment model from the procedures and drugs in each record of our database (cfr. Table 1). Now, given a comorbidity C , we can translate C into a set of procedures T , which can in turn be translated into a set of drugs R . Note that when translating from procedures to drugs, we also use the phrase-based translation model to find the n -best translations.

To obtain the drug interactions penalty of a treatment, we need to use a drug interactions database. Let $I = \{(a_0, b_0, \beta_0), \dots, (a_m, b_m, \beta_m)\}$ be a drug interactions database where a_i and b_i are two drugs that are known to interact, and β_i is an integer that represents the severity level of that drug interaction. The drug interaction penalty p of a treatment T is then calculated as $p = \sum \{\beta \mid (a, b, \beta) \in I \wedge \{a, b\} \subseteq R\}$ i.e. the sum of the severity levels of all drug interactions found in the set of drugs R obtained by translating T . Since we consider n plausible translations for every treatment T , we get n drug interactions penalties. The translation that yields the lowest drug interactions penalty is then considered the best procedures-to-drugs translation of T . In the case where multiple translations have the lowest drug interactions penalty, the most probable translation is then considered.

The treatment that has the lowest drug interactions penalty from the n best diagnoses-to-procedures translations of a comorbidity is then considered the best treatment. In the case where multiple treatments have the same lowest drug interactions penalty, the one that is found to be the most probable translation is then considered the best treatment.

NPBT+Drug+Pop: Drug Interactions Penalty With Procedure Popularity In practice, there may be multiple valid procedures to treat a given disease. Some of these tend to be more popular than others, e.g. because they are cheaper or have a lower risk. While we do not have access to any explicit knowledge about the popularity of different treatments, such popularity scores can be estimated from the database of patient records.

This method of selecting the best valid treatment extends NPBT+Drug in the following way. In the case where multiple treatments have the same lowest drug interactions penalty cost, the treatment with the highest popularity score is now preferred as the best treatment. The popularity score of a treatment is calculated by adding the popularity scores of all the procedures in the treatment. The popularity score of a procedure is calculated by counting the number of times

this procedure has been prescribed in the database of medical records. Similar to NPBT+Drug, when multiple treatments have the same lowest drug interactions penalty and the same highest treatment popularity score, the one that resulted from the more probable phrase-based translation is then considered the best treatment.

5 Experiments and Results

To train and evaluate our methods, we use the MIMIC-III database [8] of medical records, from which we extract the diagnoses, procedures and drugs used per hospital admission. The diagnoses and procedures are in the form of ICD9 codes, while the drugs are referred to by their short name. Note that the drug names have been inspected and cleaned manually to be consistent with the drug names in the drug interactions database. We only consider the admissions that contain comorbidities (i.e. at least 2 diagnoses), and all the information needed for the treatment recommendation methods (no empty fields for diagnoses, procedures or drugs). From the MIMIC-III database, we get 44,223 different admissions. The comorbidity size (i.e. the number of diagnoses) in every admission varies from 2 to 39. A graph containing the number of admissions for every comorbidity size is shown in Fig. 1.

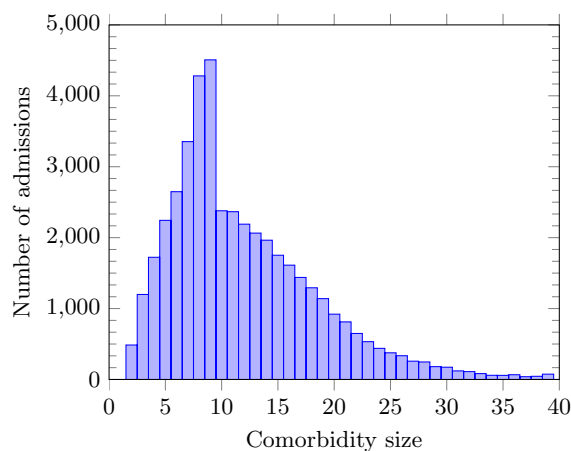


Fig. 1. The number of admissions for every comorbidity size taken from the MIMIC-III database.

We divide this database into two disjoining sets: a training set and a testing set. The training set consists of 80% of the total number of admissions, while the testing set consists of 20% of the total number of admissions. From the MIMIC-III database, we get 35,378 admissions in the training set and 8,845 in

the testing set. The task now becomes: using the training set of admissions, find treatments for the comorbidities in the testing set. When applying a treatment recommendation method, the training dataset is used to train the translation model and the language model. These models are then applied to translate the testing dataset. In our implementation, we use the Moses decoder [9], one of the most popular state-of-the-art decoders to create the word-based and phrase-based alignment models.

In NPBT, we use the drug interactions database described in [1]. We cleaned this database by cleaning the drug names and removing duplicated interactions. We extracted from this database two metrics that describe the severity of a drug interaction: severity level and contraindication. There are 5 possible severity levels (1-5), which increase the drug interactions penalty of a treatment by 1-5 respectively. Additionally, a contraindication is a boolean which, when set to true, indicates that the drug interaction should be avoided at all costs. To reflect this severity measure, the drug interactions penalty of a treatment is increased by 100 when a drug interaction with a contraindication is detected. In NPBT+Drug and NPBT+Drug+Pop, we start from the best $n=10$ translations for both translations needed in these methods (from diagnoses to procedures and from procedures to drugs).

To evaluate every recommended treatment, we use the F_1 score metric, which we calculate as follows. Let A be the set of recommended procedures by a given model and B the actual set of procedures from the testing database. We write $|A|$ and $|B|$ for the number of procedures in the sets A and B respectively. The F_1 score is given by $F_1 = 2 \times (\text{precision} \times \text{recall}) / (\text{precision} + \text{recall})$, with $\text{precision} = |A \cap B| / |A|$ and $\text{recall} = |A \cap B| / |B|$. The precision and recall can also be expressed using true positives (TP), false positivies (FP) and false negatives (FN) in the following way: $\text{precision} = \text{TP} / (\text{TP} + \text{FP})$ and $\text{recall} = \text{TP} / (\text{TP} + \text{FN})$. Since we want to evaluate every method based on all the recommended treatments, we use the micro-average and macro-average of the F_1 score. The micro-average F_1 score consists of individually averaging the TP, FP and FN of all the sets, then calculating the F_1 score using these averages. The macro-average F_1 score consists of averaging the precision and recall of all the sets, then calculating the F_1 score using these averages.

	Average Precision	Average Recall	Mirco-average F1 score	Macro-average F1 score
1-NN	0.334	0.303	0.281	0.318
WBT	0.255	0.531	0.323	0.345
PBT	0.268	0.564	0.348	0.364
NPBT+Drug	0.304	0.595	0.389	0.403
NPBT+Drug+Pop	0.321	0.607	0.414	0.420

Table 2. The average precision, average recall, micro-average and macro-average F_1 scores for every treatment recommendation method.

The results are shown in Table 2. From the table, we notice that the word-based translation approach (WBT) already performs better than the baseline approach. However, using a phrase-based translation model (PBT) leads to a further improvement of the results. Using NPBT gives the most accurate treatment recommendations, and in particular NPBT+Drug+Pop, where the procedure popularity score is used in addition to the drug interactions penalty to select the best treatment out of the n-best valid translations. Note that the Moses decoder allows up to 100-best translations for every input sentence. We also evaluated method NPBT+Drug+Pop when using n=20 and n=100, but we found the differences in F_1 score to be negligible.

As previously mentioned, word ordering plays an important role during the training of the alignment model. In the MIMIC-III database, the diagnoses and procedures are ordered based on their priorities, from highest to lowest. In other words, the primary diagnosis and the primary procedure of an admission are listed first, then the remaining ones are listed from the most important to the least important one. We compare this default setting (ordered by priority) with two different word ordering settings: Keep Primary Then Sort (KPTS) and Keep Primary Then Random (KPTR). In KPTS, the primary diagnosis and procedure of the source and target sentences in the training dataset are listed first, but the remaining ones are sorted alpha-numerically in increasing order. In KPTR, the primary diagnosis and procedure are also listed first, but the remaining ones are randomly shuffled. We train two new phrase-based translation models using KPTS and KPTR respectively and apply PBT to find the treatment recommendations. The results are shown in Table 3. From the table, we notice that the recommendation method is less accurate when using KPTS and KPTR compared to the default word ordering (ordered by priority) found in the medical database.

	Average Precision	Average Recall	Mirco-average F1 score	Macro-average F1 score
Ordered by Priority	0.268	0.564	0.348	0.364
KPTS	0.261	0.565	0.347	0.357
KPTR	0.263	0.536	0.338	0.354

Table 3. The average precision, average recall, micro-average and macro-average F_1 scores for PBT using different word orderings during the training process.

6 Conclusion

In this paper, we presented the first fully data driven method to find treatments for patients that are diagnosed with comorbidities. Instead of trying to combine clinical guidelines and trying to repair the conflicts that arise in the combined

treatments, we used word-based and phrase-based alignments to find direct mappings from a comorbidity to a recommended treatment. To improve this translation based approach, we also detect drug interactions and calculate procedure popularity scores to select the best treatment out of different valid translations. Contrary to manual approaches that aim to prescribe treatments in an evidence based way, this approach allows us to take advantage of previous medical records to recommend treatments for newly diagnosed comorbidities. From our experimental results, we found that using the method NPBT+Drug+Pop, which uses a drug interactions penalty and a procedure popularity score to find the best translation, give the most accurate treatment recommendations. The next step in this research would be to do an error analysis by looking into divergences between recommended treatments and treatments that were actually prescribed, and use this information to improve the treatment recommendation methods. There can be many reasons for why a recommended treatment diverges from a prescribed treatment: (1) the recommendation is wrong, (2) the recommendation is slightly different from the prescribed treatment, in the sense that the recommended procedures are very similar to the prescribed procedures, though not identical. A more coarse grained evaluation, where related procedures are grouped together (i.e. using ICD-grouping software) would bring this to light. (3) The recommendation is a valid alternative, and possibly even better than the treatment that was given in practice. This could be an indication of inefficiency, error, or even fraud from the provider.

References

1. Ayvaz, S., Horn, J., Hassanzadeh, O., Zhu, Q., Stan, J., Tatonetti, N.P., Vilar, S., Brochhausen, M., Samwald, M., Rastegar-Mojarad, M., et al.: Toward a complete dataset of drug–drug interaction information from publicly available sources. *Journal of biomedical informatics* 55, 206–217 (2015)
2. Bahdanau, D., Cho, K., Bengio, Y.: Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473* (2014)
3. Boxwala, A.A., Peleg, M., Tu, S., Ogunyemi, O., Zeng, Q.T., Wang, D., Patel, V.L., Greenes, R.A., Shortliffe, E.H.: Glif3: a representation format for sharable computer-interpretable clinical practice guidelines. *Journal of biomedical informatics* 37(3), 147–161 (2004)
4. Brown, P.F., Pietra, V.J.D., Pietra, S.A.D., Mercer, R.L.: The mathematics of statistical machine translation: Parameter estimation. *Computational linguistics* 19(2), 263–311 (1993)
5. Cho, K., Van Merriënboer, B., Bahdanau, D., Bengio, Y.: On the properties of neural machine translation: Encoder-decoder approaches. *arXiv preprint arXiv:1409.1259* (2014)
6. Fox, J., Johns, N., Rahmanzadeh, A.: Disseminating medical knowledge: the proforma approach. *Artificial intelligence in medicine* 14(1), 157–182 (1998)
7. Jakovljevic, M., Ostojic, L.: Comorbidity and multimorbidity in medicine today: challenges and opportunities for bringing separated branches of medicine closer to each other. *Psychiatr Danub* 25(Suppl 1), 18–28 (2013)

8. Johnson, A.E., Pollard, T.J., Shen, L., Lehman, L.w.H., Feng, M., Ghassemi, M., Moody, B., Szolovits, P., Celi, L.A., Mark, R.G.: Mimic-iii, a freely accessible critical care database. *Scientific data* 3 (2016)
9. Koehn, P., Hoang, H., Birch, A., Callison-Burch, C., Federico, M., Bertoldi, N., Cowan, B., Shen, W., Moran, C., Zens, R., et al.: Moses: Open source toolkit for statistical machine translation. In: *Proceedings of the 45th annual meeting of the ACL on interactive poster and demonstration sessions*. pp. 177–180. Association for Computational Linguistics (2007)
10. Koehn, P., Och, F.J., Marcu, D.: Statistical phrase-based translation. In: *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*. pp. 48–54. Association for Computational Linguistics (2003)
11. Merhej, E., Schockaert, S., McKelvey, T.G., De Cock, M.: Generating conflict-free treatments for patients with comorbidity using asp. In: *8th International workshop on Knowledge Representation for Health Care (KR4HC'16); held in conjunction with HEC 2016: Health-exploring complexity: an interdisciplinary systems approach*. pp. 93–100 (2016)
12. Sun, L., Liu, C., Guo, C., Xiong, H., Xie, Y.: Data-driven automatic treatment regimen development and recommendation. In: *KDD*. pp. 1865–1874 (2016)
13. Titus, A., Faill, R., Das, A.: Automatic identification of co-occurring patient events. In: *Proceedings of the 7th ACM International Conference on Bioinformatics, Computational Biology, and Health Informatics*. pp. 579–586. ACM (2016)
14. Vogel, S., Ney, H., Tillmann, C.: Hmm-based word alignment in statistical translation. In: *Proceedings of the 16th conference on Computational linguistics-Volume 2*. pp. 836–841. Association for Computational Linguistics (1996)
15. Wilk, S., Michalowski, W., Michalowski, M., Farion, K., Hing, M.M., Mohapatra, S.: Mitigation of adverse interactions in pairs of clinical practice guidelines using constraint logic programming. *Journal of biomedical informatics* 46(2), 341–353 (2013)
16. Zamborlini, V., Hoekstra, R., Da Silveira, M., Pruski, C., ten Teije, A., van Harmelen, F.: Generalizing the detection of internal and external interactions in clinical guidelines. In: *HEALTHINF*. pp. 105–116 (2016)
17. Zhang, Y., Zhang, Z.: Preliminary result on finding treatments for patients with comorbidity. In: *Knowledge Representation for Health Care*, pp. 14–28. Springer (2014)

Reasoning about Norms Revision

Fabiano Dalpiaz, Mehdi Dastani, and Davide Dell’Anna

Department of Information and Computing Sciences, Utrecht University,
Princetonplein 5, 3584 CC Utrecht, The Netherlands

Abstract. Norms with sanctions have been widely employed as a mechanism for controlling and coordinating the behavior of agents without limiting their autonomy. The norms enforced in a multi-agent system can be revised in order to increase the likelihood that desirable system properties are fulfilled or that system performance is sufficiently high. In this paper, we provide a preliminary analysis of some types of norm revision: relaxation and strengthening. Furthermore, with the help of some illustrative scenarios, we show the usefulness of norm revision for better satisfying the overall system objectives.

1 Introduction

Modern software systems execute in highly dynamic environments [1]. Typical dynamic settings are open environments, where many (heterogeneous) autonomous software components coexist, interact and can join and leave as they please. Dynamicity and uncertainty can be caused by many different (often unpredictable) factors: events from the operating environment, behaviors emerging from the effects of software executions and their interactions, the change of external norms [2] and the impact of this on software behaviour, etc. [3].

In order to guarantee desirable overall system level properties, in the multi-agent systems research field, norms with sanctions have been proposed as a means to control and coordinate the behavior of autonomous agents without limiting their autonomy [4, 5].

In many cases, however, it is infeasible for a system designer to anticipate all the possible states that the software systems and its operating environment can reach during execution [3], and to define adequate norms for each of them. Furthermore, system objectives are themselves in constant motion [6, 7]: new goals arise while others are dropped, the desired qualities vary, and the relative priority of the objectives evolves.

A static predefined set of norms may often result at runtime inadequate to guarantee the overall system objectives in various contexts [8, 9].

Dynamic update of norms at runtime is therefore one of the key factors to build a versatile normative multi-agent system, able to accommodate a heterogeneous population of agents and capable of ensuring the overall system objectives within a dynamic and uncertain environment [10].

In this paper we provide a preliminary analysis of some main types of *norm revision* that can be applied to a running system with the goal of better satisfying the overall system objectives.

We define the concept of norms revision as the act of replacing a set of norms with a new set. We specialize norms revision in three different types: relaxation, strengthening and generic alteration, which differ for the type of relationship that the revised set has with the original one. Finally, we provide a series of practical examples to clarify the different types of revisions and their possible applications.

This paper should be considered as the initial step of a research on runtime supervision of normative systems. We do not report on technical nor experimental results, but rather lay down the foundations of our future research.

The remainder of this paper is structured as follows: Section 2 frames our work within the existing literature, Section 3 discusses the problem of norms revision, Section 4 provides a series of examples of revisions of norms and possible applicative scenarios. Section 5 ends the paper with concluding remarks and future work.

2 Background

Numerous papers over the years have focused on deciding and proving the correctness of normative systems by model checking formulas describing desired properties such as liveness or safety properties [11, 12, 13]. Despite their elegance, these approaches do not fully cope properly with the dynamicity of today's complex systems, where the behavior of the system may change over time due to changes in the participating agents, their behaviors, or the active norms.

As of today, there is no generally agreed formal methodology to reason about the dynamics of norms and their impact on system specification. However some formal frameworks emerged in recent years to cope with these problems. In the rest of this section we discuss these related works on norm dynamics.

Knobbout *et al.* [10] propose a dynamic logic to formally characterize the dynamics of state-based and action-based norms. Both in Knobbout's works [10, 14] and in Alechina *et al.*'s [13], norm change is intended as norm addition. Taking these approaches as our baseline, we aim to investigate further types of norms update in order to extend the existing framework for dynamic normative systems.

Aucher *et al.* [15] introduce a dynamic context logic in order to describe the operations of contraction and expansion of theories by introducing or removing new rules. Governatori *et al.* [16] investigate from a legal point of view the application of theory revision to reason about legal abrogations and annulments. The types of revision presented in this paper can be related to theory revision, but we take a multi-agent systems standpoint, in which norms revision should be studied in terms of its impact on agents autonomy and we leave for future work the study of the impact of a revision on the existing normative system.

Norms update has also been studied from the perspective of approximation [17], where an approximated version of a norm is formally obtained to cope with imperfect monitors for the original norm. The concept of approximation is similar to our notion of relaxation, however it is defined with respect to a specific

monitor: an approximated norm is synthesized from the original one in order to maximize the violations detectable by the available imperfect monitor. Here we assume perfect monitors and we focus instead on the relationship between different norms in terms of behaviors they allow or prohibit.

Whittle and colleagues [3] present early studies on relaxation of requirements of a software system. They define a requirements language for self-adaptive systems that allows to specify, with opportune operators, relaxed versions of a requirement during the requirement elicitation phase. While they focus on a language point of view, useful for the acquisition and specification of norms, in this paper we assume the norms are already specified and we provide an analysis of revision from a normative point of view based on a formal model of multi-agent system that abstracts from the language used to define the norms.

3 Norms Revision

The class of models that we use to study and describe normative multi-agent systems are transition systems. These models consist of states and transitions between states. The idea is that such model describes all possible transitions of states that can occur within the system as a result of the actions that the agents perform.

Definition 1 (Transition System). *Given a set of atomic propositions $Props$, a transition system is a tuple $M = (S, T, s_0, V)$ where S is a set of states, $T \subseteq S \times S$ is a transition relation, $s_0 \in S$ is a distinguished initial state and V is a valuation function $V : S \rightarrow Props$ associating propositions with states (i.e., defining which propositions hold in a state).*

Given a transition system $M = (S, T, s_0, V)$, a *path* r through M is a sequence s_0, s_1, s_2, \dots of states such that $s_i T s_{i+1}$ for $i = 0, 1, \dots$

Each path can be seen as a possible behavior of the system. We consider some of the behaviors as desired and others as undesired and we use the concept of *norm* to identify these behaviors.

In this paper we consider conditional norms with sanctions and deadlines, as they are commonly used in the existing literature on normative multi-agent systems [18, 13] and they have proven to be a reasonable compromise between expressiveness and ease of reasoning [17].

Definition 2 (Conditional Norms). *Given a propositional language L_N , let $cond$, ϕ and d be boolean combinations of propositional variables from L_N and let san be a propositional atom. A conditional norm n is represented by a tuple $(cond; Z(\phi); d; san)$ where Z can be either O (obligation) or F (prohibition).*

Given a conditional norm, $cond$ represents the condition that must be satisfied in a state of M in order to detach the norm. A detached norm persists as long as it is not obeyed or violated or the deadline d is not reached (a state where d holds is encountered). Conditional norms are evaluated on paths of the transition system. In this paper we omit a formal definition of a violation, which is strictly

dependent on the language used to express them. In the following we denote a violation formula for a conditional norm n by $v(n)$ (see previous publication [17] for a formal definition for PLTL formulae). A norm n is violated on a path r if, and only if, $v(n)$ holds in some state on r . We denote by $Viol(M, n)$ the set of paths through M that violate the norm n . Fig. 1 sketches the main components of a runtime supervision framework that continuously monitors the execution of a multi-agent system (MAS), evaluates its behavior against the currently enforced norms, and intervenes by deciding which norms should be revised. Norm violation is monitored (and sanctioned) by the **Monitoring and sanctioning** component. In this paper we ignore the techniques of monitoring and evaluation of norms, for which many works can be found in literature (e.g. [19]), and we assume there exists a perfect monitor for each norm. The aim of this paper is to provide a study of the possible revisions of norms and of the possible conditions when a revision may be useful. The study is a starting point to build a **Norm update** component able to revise at runtime the currently enforced norms in order to ensure the achievement of the overall objectives. We assume that the **Monitoring and sanctioning** component stores at runtime information about the obedience or violation of the norms and about the operating contexts in which they are evaluated (e.g. the hour of the day, etc.). This information, together with information about the satisfaction of the overall systems objectives, is used by the **Norms update** component in order to decide if and how to revise the currently enforced norms.

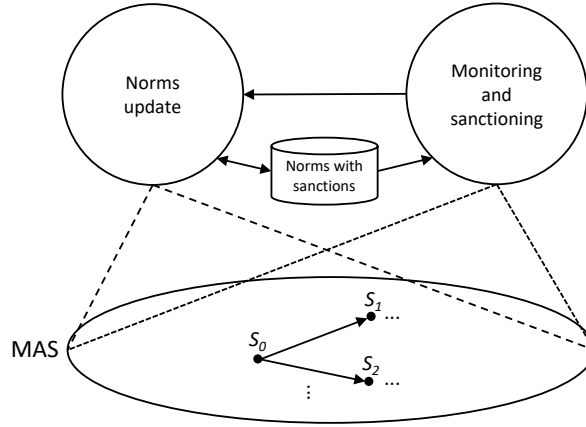


Fig. 1. The main components of the proposed runtime supervision framework aimed at revising norms enforced into a normative MAS. Dotted lines represent the scope of the components.

Norm enforcement in multi-agent systems can be done in two possible ways: regimentation and sanctioning [13]. Regimentation makes bad behaviors impossible, i.e., it makes some of the paths of M inaccessible. With regimentation, violations of the norms are not possible, however the autonomy of the agents is reduced. Sanctioning norms violation is instead a means to discourage the violation. Sanctions are essentially treated like fines. They penalize agents when

they bring the system on an undesired path, however they leave autonomy to the agents, even if their resources are reduced if they violate the norms.

A regimented norm n can be described by $Viol(M, n)$ (i.e., by the set of paths of M that violate the norm), while a norm n with a sanction is described by both $Viol(M, n)$ and a propositional sanction atom that is asserted in case of violation of n . We assume in the following to be able to compare two sanctions (e.g., consider the case of numeric sanctions such as money).

Given a set of norms $N = \{n_1, \dots, n_k\}$, we denote by $Viol(M, N)$ the set of all paths through M each violating at least one norm in N (i.e., $Viol(M, N) = Viol(M, n_1) \cup \dots \cup Viol(M, n_k)$), and by San a conjunction of all sanctions associated with the norms in N (i.e., $San_N = san_1 \wedge \dots \wedge san_k$).

Given a pair of sets of norms, N_1 and N_2 , we denote by

- $N_1 < N_2$: the fact that $Viol(M, N_1) \supset Viol(M, N_2)$. In this case, we say that N_1 is a set of norms more restrictive than N_2 , or that N_2 is more relaxed than N_1 .
- $N_1 \equiv N_2$: the fact that $Viol(M, N_1) = Viol(M, N_2)$. We say that the sets of norms N_1 and N_2 are equally restrictive.

Note that the relationships between norms above reported can only be satisfied by pairs of sets of norms such that $Viol(M, N_1) \subseteq (\supseteq) Viol(M, N_2)$.

Definition 3 (Norms Revision). *Let N be a set of norms, a revision of N is a replacement of N with a new set N^R . N^R may be a more/equivalently/less restrictive set of norms or an alternative set where no relationship can be stated.*

We distinguish two main types of revision: relaxation and strengthening, and we consider all the other types of revisions as regular alteration.

Definition 4 (Revision, Relaxation, Strengthening). *A relaxation of a set of norms N is a revision of N with a new set N^R such that $N^R > N$. A strengthening of a set of norms N is a revision of N with a new set N^R such that $N^R < N$. Any other revision of N with a new set N^R such that either $N^R \equiv N$ or no relationship can be stated are regular alterations of N .*

Given a pair of norms, n_1 and n_2 such that $n_1 = (cond_1; Z(\phi_1); d_1; san_1)$ and $n_2 = (cond_2; Z(\phi_2); d_2; san_2)$,

1. n_2 is a relaxation of n_1 if and only if at least one of the following holds (all else being equal):
 - (a) $cond_2$ is a stricter formula $cond_1$. Note that with stricter (less strict) formula we mean a formula that is satisfied in strictly less (more) paths of M . A stricter formula for a condition of a conditional norm makes therefore the norm applicable in fewer paths than the original.
 - (b) ϕ_2 is a less strict formula than ϕ_1 if $Z = O$ (obligation), or ϕ_2 is a stricter formula than ϕ_1 if $Z = F$ (prohibition). A less strict obligation (or a stricter prohibition) makes the norm violated in fewer paths, which means that more agents' behaviors are allowed.

- (c) d_2 is a less strict formula than d_1 . Note that this is valid since we only accept propositional formulae, therefore a less strict deadline for a conditional norm means that it can be withdrawn in more states.
- 2. n_2 is a strengthening of n_1 if and only if n_1 is a relaxation of n_2 .
- 3. n_2 is a regular alteration of n_1 if it is neither a relaxation nor a strengthening

Note that while a revision of a single norm can be analyzed in terms of all its components, revision of a set of norms involving more than one norm can only be compared in terms of the set of paths of M and the sanctions. A revision N^R of a set N obtained by relaxing some of the norms and strengthening some others can be compared to N only in terms of the resulting set of behaviors allowed in M , since $Viol(M, N^R)$ can be either a subset or a superset of $Viol(M, N)$ or even a disjoint set. However if the revised norms are all relaxed (strengthened) the resulting revised set is a relaxed (strengthened) set of N .

Notably, increasing or decreasing sanctions associated to norms, under the assumption that such sanctions are comparable, is an alternative means to influence the behavior of agents that goes beyond relaxation or strengthening of norms. Thus, increasing the sanctions San_N associated with a set of norms N is a way to bring the system to the desired paths defined by N , while decreasing the sanctions is a way to accept undesired paths. In the next section we provide an intuitive explanation of this concept with the help of illustrative examples.

4 Illustrative Examples

In this section we provide some illustrative examples of the different types of revision of norms and some illustrative scenarios where to revise norms may be useful to ensure the overall system objectives. Note that in this paper we are not interested in how the norms that we use are acquired. Moreover, to consider only relevant norms, we assume that we are provided with information about the rationale behind norms, that relates them to the overall system objectives. Finally, we assume every parameter of the norms as perfectly monitorable.

4.1 Relaxation

Consider norm n_1 : “cars speed in road r shall be kept below 15km/h, otherwise €10k fine”, formally (for brevity assume car and road as implicit) $n_1 = (inRoad, F(speedAbove15), \top, 10keuros)$, where \top provides “always” interpretation. Possible examples of relaxations of n_1 are the following (possibly combined):

- $r_1 = ((inRoad \wedge trafficHigh), F(speedAbove15), \top, 10keuros)$.
 r_1 differs from n_1 only for the condition: $(inRoad \wedge trafficHigh)$ is a stricter formula than $inRoad$. Since the condition is more specific and the rest of parameters is analogous (case 1a of Definition 4 holds), r_1 is applicable in fewer paths of M , therefore fewer paths are violations and more behaviors are allowed in the system.

- $r_2 = (inRoad, F(speedAbove20), \top, 10euros)$.
 r_2 differs from n_1 only for the prohibition (case 1b): *speedAbove20* is stricter than *speedAbove15* because fewer paths are considered prohibited.
- $r_3 = (inRoad, F(speedAbove15), firstHalfCompleted, 10euros)$.
 r_3 differs from n_1 only for the deadline (case 1c): *firstHalfCompleted* is less strict than \top (“always”, e.g. *firstHalfCompleted* \wedge *secondHalfCompleted*) since the norm remains valid only when the car is in the first half of the road. Again more behaviors are allowed.

Example. Consider a simple scenario where a population of autonomous vehicles can take a road r where their speed and the traffic level can be perfectly monitored. The designer of the system M wants to be sure that “no cars stay in road r (10km long) for more than 35 minutes”. Norm n_1 is currently enforced and at runtime it appears to be too strict for achieving the overall objective: if it is obeyed it prevents the fulfilment of the objective. Since n_1 is too strict, one (or a combination) of the following relaxations may be useful to improve the performance of the system:

- replace n_1 with r_1 , r_2 or r_3 . In case of r_1 , choosing a stricter condition allows agents to perform more actions without incurring in sanctions (agents are allowed to drive above 15km/h unless the traffic level is high). Since obeying to n_1 prevents the fulfilment of the objective, reducing the conditions where it applies means allowing behaviors previously forbidden. An analogous explanation can be provided also for r_2 and r_3 .

Notice that an analogous effect can be obtained by replacing n_1 with another norm $s_1 = (inRoad, F(speedAbove15), \top, 5euros)$, differing from n_1 only for the sanction: *5euros* < *10euros*. Reducing the fine incentives more agents to violate n_1 . Since the objective is achieved when n_1 is violated we increase the chances the objective is achieved.

4.2 Strengthening

Consider a norm $n_2 = (inRoad, O(speedbelow50), outOfRoad, 5euros)$. Possible examples of strengthening of n_2 are the following (possibly combined):

- $r_5 = ((inRoad \vee aroundTheRoad), O(speedbelow50), (outOfRoad \wedge 1kmFarAway), 5euros)$.
 r_5 differs from n_2 for both the condition and the deadline: $(inRoad \vee aroundTheRoad)$ is less strict than *inRoad*, while $(outOfRoad \wedge 1kmFarAway)$ is stricter than *outOfRoad*. Since the condition is less specific, r_5 is detached in more paths, and since the deadline is more specific, r_5 remains valid in more cases, therefore more paths violate it and less behaviors are allowed.
- $r_6 = (inRoad, O(typeScooter), outOfRoad, 5euros)$.
 $O(typeScooter)$ is stricter than $O(speedbelow50)$: while n_2 accepts all executions involving vehicles (scooters or not) that keep the speed below 50km/h, r_6 accepts only executions involving scooters (assumed here to have maximum speed of 50km/h). Again less behaviors are allowed.

Example. Consider a scenario where a population of autonomous vehicles can take a road r where their speed, type of vehicle and noise they produce can be perfectly monitored. There is an overall system objective “*vehicle’s noise in the neighborhood is below x db*”. Norm n_2 is currently enforced and it is proven that violation of n_2 prevents the achievement of the overall objective. Norm n_2 appears at runtime to be too weak to ensure the overall system objective (e.g. it is often violated). One (or a combination) of the following strengthening of n_2 may therefore be useful to improve the performance of the system:

- replace n_2 with r_5 . Choosing a less strict condition makes the norm valid in more cases. Deincentivating agents to drive faster than 50km/h also around the road may prevent vehicles to accelerate immediately out of the road, which may be harmful for the achievement of the overall objective of keeping the noise in the neighborhood low.
- replace n_2 with r_6 . Assuming scooters have maximum speed of 50km/h, if the new norm is obeyed by agents, the goal is ensured.

Notice one more time that an analogous effect can be obtained by replacing n_2 with another norm $s_2 = (inRoad, O(speedbelow50), outOfRoad, 10keuros)$, differing from n_2 only for the sanction: $10keuros > 5euros$. Increasing the fine incentives more agents to obey n_2 . Since the objective is achieved when n_2 is obeyed we increase the chances to achieve it.

4.3 Alteration

Consider norm n_3 : “*if two cars $c1$ and $c2$ are at the opposite ends of the narrowed road, $c1$ shall move and $c2$ shall wait, otherwise 10k euros fine to $c2$* ”, formally $n_3 = ((firstEnd(c1) \wedge secEnd(c2)), O(move(c1) \wedge wait(c2)), \top, c2_10keuros)$. Possible examples of alteration of n_3 are the following:

- $r_8 = ((firstEnd(c1) \wedge secEnd(c2)), O(wait(c1) \wedge move(c2)), \top, c1_10keuros)$.
- $r_9 = (inRoad(c1), O(speedbelow50), \top, c1_10keuros)$.

Both r_8 and r_9 define a set of underused paths that is neither a subset or a superset of the paths defined by n_3 . Therefore they cannot be considered neither a relaxation nor a strengthening. Notice that r_9 is reported as an example of a generic alteration, which reflects the definition we provided, even if it is not strictly related to n_3 .

Example. Consider a scenario of a narrowed road and cars coming from both directions. The overall objective is “*keep cars queue size in both directions below a threshold t* ”. Norm n_3 is currently enforced and it proved to be useful to achieve the objective when traffic is low in both directions. However, when traffic is high in the direction of car $c1$ the queue in the opposite direction grows too much. Norm n_3 appears therefore to be not good to ensure the overall system objective in such case. An alteration such as replacing n_3 with r_9 may be useful to improve the performance of the system. Notice that a generic alteration

involves a new norm that either is not strictly related to the existing one or it is equally restrictive, therefore it is hard to predict the outcome of the enforcement of the new norm. However, assuming the considered norms are related to the same overall system objective, enforcing an alteration in the system may lead to different (hopefully better) performances.

5 Conclusion and Future Work

In this paper we proposed a preliminary study of some types of revision of norms that can be enacted on a running multi-agent system in order to enhance its performance. We defined the concept of norms revision as a replacement of a set of norms with a new set and we formally defined the notions of relaxation and strengthening, which introduce supersets and subsets of the allowed behaviors, respectively. We provided some examples of revisions and some application scenarios where a revision of the current set of enforced norms may be useful in order to achieve the overall system objectives.

The analysis of different types of norms update provided in this paper is a first step in order to extend existing works for dynamic normative systems (see [14, 10]) with revision of norms. As future work we plan to provide a formal semantics for the concepts here introduced and to develop a syntactically sound and complete reasoning system. We also want to consider the effects of a revision in terms of update of the normative system and to analyse the relation of the revisions here presented with standard operators from the literature (e.g. from AGM framework). Finally we plan to provide a formal discussion about the correlation between the enforced norms and the fulfilment of the overall system objectives and to develop techniques to automatically reason at runtime about this correlation and to automatically suggest and perform a revision. These developments are meant to be part of an adaptive runtime supervision framework that continuously monitors the execution of a normative multi-agent system and intervenes on it in order to improve its performance.

References

- [1] Sommerville, I., Cliff, D., Calinescu, R., Keen, J., Kelly, T., Kwiatkowska, M.Z., McDermid, J.A., Paige, R.F.: Large-scale complex IT systems. *Commun. ACM* **55**(7) (2012) 71–77
- [2] Rosemann, M., Recker, J., Flender, C.: Contextualisation of business processes. *IJBPM* **3**(1) (2008) 47–60
- [3] Whittle, J., Sawyer, P., Bencomo, N., Cheng, B.H.C., Bruel, J.: RELAX: a language to address uncertainty in self-adaptive systems requirement. *Requir. Eng.* **15**(2) (2010) 177–196
- [4] Boella, G., van der Torre, L.W.N.: Regulative and constitutive norms in normative multiagent systems. In: *Principles of Knowledge Representation and Reasoning: Proceedings of the Ninth International Conference (KR2004)*, Whistler, Canada, June 2-5, 2004. (2004) 255–266

- [5] Bulling, N., Dastani, M.: Norm-based mechanism design. *Artif. Intell.* **239** (2016) 97–142
- [6] Zowghi, D., Gervasi, V.: On the interplay between consistency, completeness, and correctness in requirements evolution. *Information & Software Technology* **45**(14) (2003) 993–1009
- [7] Ernst, N.A., Borgida, A., Jureta, I., Mylopoulos, J.: An overview of requirements evolution. In: *Evolving Software Systems*. (2014) 3–32
- [8] Ali, R., Dalpiaz, F., Giorgini, P., Souza, V.E.S.: Requirements evolution: From assumptions to reality. In: *Enterprise, Business-Process and Information Systems Modeling - 12th International Conference, BPMDS 2011, and 16th International Conference, EMMSAD 2011, held at CAiSE 2011, London, UK, June 20-21, 2011. Proceedings*. (2011) 372–382
- [9] Letier, E., van Lamsweerde, A.: Reasoning about partial goal satisfaction for requirements and design engineering. In: *Proceedings of the 12th ACM SIGSOFT International Symposium on Foundations of Software Engineering, 2004, Newport Beach, CA, USA, October 31 - November 6, 2004*. (2004) 53–62
- [10] Knobbout, M., Dastani, M., Meyer, J.C.: A dynamic logic of norm change. In: *ECAI 2016 - 22nd European Conference on Artificial Intelligence, 29 August-2 September 2016, The Hague, The Netherlands - Including Prestigious Applications of Artificial Intelligence (PAIS 2016)*. (2016) 886–894
- [11] Dastani, M., Grossi, D., Meyer, J.C., Tinnemeier, N.A.M.: Normative multi-agent programs and their logics. In: *Normative Multi-Agent Systems*, 15.03. - 20.03.2009. (2009)
- [12] Knobbout, M., Dastani, M.: Reasoning under compliance assumptions in normative multiagent systems. In: *International Conference on Autonomous Agents and Multiagent Systems, AAMAS 2012, Valencia, Spain, June 4-8, 2012 (3 Volumes)*. (2012) 331–340
- [13] Alechina, N., Dastani, M., Logan, B.: Reasoning about normative update. In: *IJCAI 2013, Proceedings of the 23rd International Joint Conference on Artificial Intelligence, Beijing, China, August 3-9, 2013*. (2013) 20–26
- [14] Knobbout, M., Dastani, M., Meyer, J.C.: Reasoning about dynamic normative systems. In: *Logics in Artificial Intelligence - 14th European Conference, JELIA 2014, Funchal, Madeira, Portugal, September 24-26, 2014. Proceedings*. (2014) 628–636
- [15] Aucher, G., Grossi, D., Herzig, A., Lorini, E.: Dynamic context logic. In: *Logic, Rationality, and Interaction, Second International Workshop, LORI 2009, Chongqing, China, October 8-11, 2009. Proceedings*. (2009) 15–26
- [16] Governatori, G., Rotolo, A.: Changing legal systems: legal abrogations and annulments in defeasible logic. *Logic Journal of the IGPL* **18**(1) (2010) 157–194
- [17] Alechina, N., Dastani, M., Logan, B.: Norm approximation for imperfect monitors. In: *International conference on Autonomous Agents and Multi-Agent Systems, AAMAS '14, Paris, France, May 5-9, 2014*. (2014) 117–124
- [18] Tinnemeier, N.A.M., Dastani, M., Meyer, J.C., van der Torre, L.W.N.: Programming normative artifacts with declarative obligations and prohibitions. In: *Proceedings of the 2009 IEEE/WIC/ACM International Conference on Intelligent Agent Technology, IAT 2009, Milan, Italy, 15-18 September 2009*. (2009) 145–152
- [19] Bulling, N., Dastani, M., Knobbout, M.: Monitoring norm violations in multi-agent systems. In: *International conference on Autonomous Agents and Multi-Agent Systems, AAMAS '13, Saint Paul, MN, USA, May 6-10, 2013*. (2013) 491–498

Distribution-driven Regression Ensemble Construction for Time Series Forecasting

Florian Wimmerauer, Evgueni Smirnov, Matúš Mihalák

Maastricht University, P.O. Box 616 6200 MD Maastricht, The Netherlands,
`f.wimmerauer@student.maastrichtuniversity.nl`

Abstract. This paper introduces a two-stage approach on selecting members of an ensemble generating accurate forecasts of a time series with a small amount of forecasts. In the first stage, models trained on similarly-distributed data are selected based on time series stationarity, more specially, the local stationarity of sub-series. The second stage identifies the most diverse models among those selected in the first stage to compose the final ensemble. Diversity is measured either on pair-wise basis or over the complete ensemble. In both case, multiple novel diversity metrics are introduced. Additionally, the presented approach is highly modular and does not presuppose the type of comprising models of the ensemble. The experiments show that the proposed approach outperforms the base line when predicting both synthetic and real-world time series.

Keywords: Time Series Forecasting, Ensemble Forecasting, Distribution Similarity

1 Introduction

In machine learning, a forecasting or prediction task is the estimation of future events based on previous observations. Forecasting has a wide range of applications, such as weather forecasts, stock value predictions, and fine-grained forecasts of energy consumption in specific geographic areas. A special field of forecasting which has drawn significant research interest is the forecast of time series. A time series is an indexed series of data points where each index typically corresponds to a point in time. Unlike traditional data sets where the ordering of data points do not matter, a time series is ordered.

With the increasing computation power and advances in machine learning techniques, the task of time series forecasting has become more approachable. In the pursuit of higher prediction accuracy, ensemble forecasts have gained attention of both researchers and practitioners. An ensemble is a collection of models, where multiple individual predictions are combined into one single prediction with the aim of obtaining better forecasting performance than an individual model otherwise would. A key challenge in constructing ensembles is to choose a set of models that yield desirable forecasting performance when combined. An

important feature of an ensemble is the diversity among its members. The ensemble diversity can be seen as the extent of disagreement among the models. In an extreme case, if all models in the ensemble would produce identical forecasts, no performance gain could be achieved.

This paper proposes a novel framework of ensemble construction for time series forecasting from a collection of trained models. Firstly, the members of the ensemble are selected based on their relevance to the prediction task. That is, those models trained on distributions different from the encountered one are disregarded. Secondly, among the relevant models, only the most diverse ones compose the final ensemble. Last but not least, new measures of diversity for regression ensembles are proposed. It is noteworthy that this framework is highly modular and does not presuppose the type of comprising models of the ensemble.

The proposed ensemble construction approach is based on a collection of trained models, each of which concerns a part (sub-series) of a longer time series. The relative position of the sub-series in the full time series is known. When new prediction tasks are encountered, firstly, previous parts of the full time series that have the same distribution with the current sub-series are detected. Within the set of sub-series sharing the same distribution, the most diverse ones are chosen, and their corresponding trained models form the final ensemble. In this process, the selection of sub-series with distribution and the measurement of diversity are based on statistical tests rather than techniques specific to the trained models. Therefore, the proposed approach is model-independent and thus can be applied to ensemble construction of any time series without structural modifications.

The presented approach is evaluated on a synthetic time series of known underlying distribution and a real-world time series consisting of a store’s sales values in five-minute buckets. For the latter, every day over the past year, models are trained based on data from the previous seven days with the goal of predicting the sales value of the next day.

This section is followed by a short survey of related work, preliminaries and introduction to the relevant concepts. In section 4, applicable methods are defined and the new measure of diversity for regression ensembles is developed. Afterwards, the experiments and results are presented in section 5. Finally, the paper closes with the conclusions in section 6.

2 Related Work

Time series forecasting can be conducted via various approaches. Statistical learning models such as the autoregressive integrated moving average (ARIMA) are widely discussed [17]. More recently, there has been a raise in machine learning approaches such as regression trees [15], regression support vector machines [16] and deep learning models [1, 24]. Notably, deep learning models became very applicable since recurrent neural networks (RNN) such as long short-term memory recurrent neural networks (LSTM) [8] were introduced. In contrast to standard feed-forward neural networks, LMTSs are capable of ‘remembering’ previously-encountered data and therefore learn time- or sequence-based con-

cepts. With the increasing computation power, the model architecture became deeper, achieving higher predictive power [24]. RNNs nowadays often outperform their traditional counterparts on time series forecasting tasks [1].

A shortcoming of RNNs is the computational load and therefore time consumption. In case not all training data is available upfront and the models need to be adapted in later stages, RNNs have to be retrained on the full set of data. Retraining on the newly available data alone will lead to the loss of acquired knowledge on the older data [23]. This issue can be resolved by constructing an ensemble of models trained on different parts of the time series [23]. Ensemble learning in general is a highly active field of research, because well-chosen ensembles can increase the performance compared to single-model approaches [4, 10, 13, 21]. Error reduction via ensemble prediction can only be achieved when the ensemble is diverse, meaning that the members of the ensemble disagree with each other to a certain extent [4, 10, 13, 21]. It can be trivially seen that an ensemble of 1,000 models all producing the same forecasting result cannot lead to any performance gain over a single model. Measuring or even ensuring the diversity in an ensemble is so far mostly done for classification problems [12]. For regression problems, often specialised approaches are introduced [4, 13]. One way to increase the diversity in an ensemble is to train the models on different subsets of the data [21].

Ensemble construction for a longer time series faces additional challenges due to the possible changes in the underlying distribution of the data. In this case, a traditional approach for model selection is to compute the distances between sub-series by dynamic time warping [27] and selecting the least distant models.

3 Preliminaries

When training models with supervised machine learning techniques, one should ensure that the distribution of the data on which the models are trained has the same distribution with the data to be processed at production time [20]. A model trained on data from one distribution is not assured to make accurate predictions on data from a completely different distribution [20]. The same principle holds when constructing an ensemble from a collection of trained models. Otherwise, members of the ensemble might contribute a higher error due to incorrect assumptions on the underlying distribution. In the context of this paper, namely time series forecasting, this means that only the models trained on similarly-distributed data as the data to be forecasted are suitable to contribute to the final ensemble forecast.

Testing whether two data sets have the same underlying distribution is a well-researched topic [14]. From a time series point of view, the statistical properties of the data are closely related to stationarity. A stationary time series is defined as a time series whose statistical properties do not change over time [17]. When it comes to stationary times series, one distinguishes between first order stationary and second order stationary time series.

In a first order stationary time series, all moments of all degrees such as mean and variance do not change throughout the complete time series. This means that for a time series $T = t_1 \dots t_L$ with length L , the joint statistical distribution $X_{t_1} \dots X_{t_l}$ is the same as the joint statistical distribution $X_{t_1+\tau} \dots X_{t_l+\tau}$ for all l and τ where $l + \tau \leq L$. In contrast to this strict definition, in a second order stationary time series, the mean and the variance are constant and the autocovariance between X_t and $X_{t+\tau}$ solely depends on the lag τ . With the first order stationarity being a mostly overly strict assumption for real-world data sets, the upcoming sections focus on the second order stationarity. A test for second order stationarity of a time series was introduced by Priestley and Subba Rao [25]. In this test, an analysis of variance is performed on the logarithm of a time-varying spectral estimate at a set of times and frequencies in the Fourier spectrum.

For a long time-series, second order stationary cannot be assumed on the complete series. A more reasonable assumption is that the long time series might contain second order stationary parts and that the complete time series is locally stationary. To test a time series for local second order stationarity and to localise these parts within the time series, a test introduced by Nason [18] can be used. This test is closely related to the work of Priestley and Subba Rao [25], but analyses the evolutionary wavelet spectrum [19] instead of the Fourier spectrum of the time series.

4 Methods

After introducing the general framework, this section details the methods used to construct an ensemble. Firstly presented are approaches for selecting models trained on similarly-distributed data. Secondly, various ensemble diversity measures are discussed. Finally, three optimisation methods for selecting an ensemble with the highest diversity are introduced.

4.1 Sub-Series with Similar Underlying Distribution

Two methods are introduced to construct an ensemble of models with distribution similarity. The former is based on the time series property of stationarity, whereas the latter uses traditional statistics techniques.

Time Series Stationarity Properties As mentioned in section 3, a stationary time series has the same underlying distribution throughout the complete series. This section focuses on testing second order stationarity. To this end, and to additionally localise the stationary parts within the time series, a test introduced by Nason [18] is used. The choice for this test was based on, besides the reported runtime of $\mathcal{O}(l \log l)$ for a time series of length l , the fact that it does not assume a normal distribution as the underlying distribution of the time series.

The test is based on evaluating the stability of the expected value $\beta_j(k/T)$ of the wavelet periodogram

$$I_\ell(k/T) = I_{j,k} = \left(\sum_{t=1}^T X_t \psi_{j,k-t} \right)^2 \quad (1)$$

over a finite set of wavelet scales ℓ . In general, $\ell \in \mathbb{N}$ but since T is finite, $\ell = 1, \dots, (J = \log_2 T)$. X_t is the time series over time $t = 1, \dots, T$, $\{\psi_{j,k}\}_{j,k}$ is a set of non-decimated discrete wavelets, $k = 1, \dots, T$ and $j \in \mathbb{N}$ is the scale of the wavelet. The tested time series is stationary if and only if for all scales ℓ , $\beta_\ell(z)$ is a constant function for rescaled time $z = k/T \in (0, 1)$

To test if the $\beta_\ell(z)$ is a constant function, the Haar wavelet coefficients of time series are analysed. These coefficients are given as

$$v_{i,p}^{(\ell)} = \int_0^1 \beta_\ell(z) \psi_{i,p}^H(z) dz \quad (2)$$

for Haar wavelet scale $i = 1, \dots, J$, $p = 1, \dots, 2^i - 1$, with $\{\psi_{i,p}^H(t)\}_{i,p}$ being the Haar wavelets.

The null hypothesis H_0 is that the quantity $\beta_\ell(z)$ is a constant function of z and hence $v_{i,p}^{(\ell)} = 0$ for all ℓ, i, p , as $\int \psi(z) dz = 0$ is a defining wavelet property. The set of test statistic \mathcal{S} is:

$$\mathcal{S}_{i,p}^{(\ell)} = \hat{v}_{i,p}^{(\ell)} \hat{\sigma}_{i,p}^{(\ell)-1} \quad (3)$$

As proposed by von Neuman et al. [29], an estimate for $\hat{\sigma}_{i,p}^{(\ell)2}$ is the variance of the estimated wavelet periodogram:

$$\text{var}(\beta_\ell) = \int_{-\pi}^{\pi} f(\omega) \left| \hat{\psi}_\ell(\omega) \right|^2 d\omega \quad (4)$$

In this case the classical stationary spectrum $f(\omega)$ can be estimated by the regular periodogram in equation (1).

To evaluate the test statistic, the empirical wavelet periodogram values $I_{\ell,k} = I_\ell(k/T)$ for $k = 1, \dots, T$ are analysed. With this, the Haar wavelet coefficients are estimated for scale ℓ by

$$\hat{v}_{i,p}^{(\ell)} = 2^{i/2} \left(\sum_{r=0}^{2^{i-1}-1} I_{\ell, 2^i p - r} - \sum_{q=2^{i-1}}^{2^i-1} I_{\ell, 2^i p - q} \right) \quad (5)$$

Based on \mathcal{S} being a set of test statistics for multiple H_0 , multiple hypothesis tests must be performed. For combining these multiple hypotheses, the false discovery rate method [3] is used, because this method is less conservative than for example the Bonferroni correction [7].

The above-mentioned test on a time series not only provides knowledge about stationarity but also locates non-stationarity. For each test statistic in \mathcal{S} , with

ℓ, i and p , the corresponding part of the time series and thus the non-stationary parts where the related H_0 's were rejected can be identified. This can, in the context of this paper, find trained models that are not suitable for contributing to the ensemble forecast. Therefore, only the parts where none of the H_0 's are rejected qualify for contributing trained models to the ensemble.

Statistics Properties After discussing finding the time series parts with the same underlying distribution via local stationarity tests, the focus is shifted from time series properties to traditional statistical analyses of the corresponding training data of the models. Under the assumption that each model is trained on a sub-series of the time series, the distribution of the training data of each model is analysed.

The underlying distributions are compared between the current training data on hand and the corresponding data where existing models were trained. Two different tests are used, namely the two-sample Kolmogorov-Smirnov test (KS test) [14] and the two-sample Anderson-Darling test (AD test) [2]. The two-sample KS test tests whether two distributions differ. The null hypothesis H_0 is that this is not the case. Like in all KS tests, each instance is weighted equally. However, many distributions differ primarily in their tails, which poses a challenge to the KS test. The Anderson-Darling test is specially suited to detect differences in the tails of distributions. In this case, a two-sample AD test is required. [22] extended the standard AD test to a two sample version. Similar to the two-sample KS test, the two-sample AD test examines whether two underlying distributions differ by testing the null hypothesis H_0 that the two samples are from the same distribution.

4.2 Diversity

Once the appropriate models are selected, the diversity of the resulting ensemble needs to be evaluated. Firstly, techniques for evaluating the diversity of a pair of models are discussed, followed by diversity measures on the complete ensemble. Analogous to the methods in section 4.1, the diversity is only measured based on distributions, making the approach applicable regression tasks in general.

Pair-Wise Diversity The diversity between trained models can be measured in various ways. Let $m_u, m_v \in M$ be two models trained on the time series $Y^u = [y_1^u, y_2^u, \dots, y_{n_Y}^u]$ and $Y^v = [y_1^v, y_2^v, \dots, y_{n_Y}^v]$ respectively, where the training series has n_Y steps. Straightforward diversity measures include the covariance, the correlation coefficient and entropy [5]. The covariance between Y^u and Y^v is defined as

$$\text{cov}(Y^u, Y^v) = \frac{1}{n^2} \sum_{i=1}^{n_Y} \sum_{j=i+1}^{n_Y} (y_i^u - y_j^u)(y_i^v - y_j^v) \quad (6)$$

A smaller covariance indicates less association between models and therefore higher diversity. As the magnitude of covariance is in general hard to interpret,

the correlation coefficient, the normalisation of the covariance, may serve as a better diversity measure than the covariance. The correlation coefficient between Y^u and Y^v is defined as

$$\text{corr}(Y^u, Y^v) = \frac{n_Y \sum y_i^u y_i^v - \sum y_i^u - \sum y_i^v}{\sqrt{n_Y \sum y_i^u - (\sum y_i^u)^2} \sqrt{n_Y \sum y_i^v - (\sum y_i^v)^2}}. \quad (7)$$

Here \sum abbreviates $\sum_{i=1}^{n_Y}$. Since diversity is inversely proportional to correlation, ensembles with lower correlation coefficients are preferred.

Besides the above-mentioned measurements, the entropy of a system provides an indication of diversity. The entropy between Y^u and Y^v is defined as

$$\text{ent}(Y^u, Y^v) = -\frac{1}{2} \ln(\sigma_u^2 \sigma_v^2 (1 - \text{corr}(Y^u, Y^v)^2)), \quad (8)$$

where $\text{corr}(Y^u, Y^v)$ is the correlation coefficient as defined above in equation (7).

In addition to the conventional approaches, a novelty in this paper is to measure diversity via statistical tests. Recall that the aforementioned two-sample KS test and the two-sample AD test examine whether two given samples come from the same underlying distribution under the null-hypothesis H_0 that they do. This means that if the computed p-value is high or higher than a critical value, one fails to reject H_0 . In this light, a lower p-value implies higher diversity.

Ensemble Diversity Based on all pair-wise diversity measures of the n_E models, measuring the complete ensemble diversity requires all training series of the models. Let all n_E models $m_j \in E$ be trained on the $n_Y \times n_E$ values

$$Y = \begin{pmatrix} y_1^{[1]} & y_1^{[2]} & \cdots & y_1^{[n_E]} \\ y_2^{[1]} & y_2^{[2]} & \cdots & y_2^{[n_E]} \\ \vdots & \vdots & \ddots & \vdots \\ y_{n_Y}^{[1]} & y_{n_Y}^{[2]} & \cdots & y_{n_Y}^{[n_E]} \end{pmatrix} \quad (9)$$

Extending the previously introduced pair-wise diversity measure to the entire ensemble can be conveniently achieved by averaging over all pairs. For instance, the ensemble covariance can be defined based on pairwise covariance as follows:

$$\text{cov}_E = \frac{1}{n_E(n_E - 1)} \sum_{i=1}^{n_E} \sum_{\substack{j=1 \\ j \neq i}}^{n_E} \text{cov}(Y^{[i]}, Y^{[j]}). \quad (10)$$

Moreover, the averaging of pair-wise diversity such as equation (10) can be extended to account for different weights.

Whereas measurements like correlation coefficients and entropy can be averaged, combining p-values is less straightforward. In this case, the p-values $P = [p_{1,1}, p_{1,2}, \dots, p_{n_E, n_E}]$ are combined using Stouffer's z-score method explained by [30]. Stouffer's z-score method was chosen over other methods such

as Fishers's method, because Stouffer's z-score method allows the extra degrees of freedom with weights $W = [w_{1,1}, w_{1,2}, \dots, w_{n_E, n_E}]$ which specify the relative importance of pairwise p-values. The combined p-value is:

$$p_N = 1 - \Phi \left(\frac{\sum_{i=1}^{n_Y} \sum_{j=1}^{n_Y} w_{i,j} \Phi^{-1}(1 - p_{i,j})}{\sqrt{\sum_{i=1}^{n_Y} \sum_{j=1}^{n_Y} w_{i,j}^2}} \right) \quad (11)$$

where p_{n_Y} is the p-value of the model pair and Φ, Φ^{-1} are the standard normal cumulative distribution function and its inverse respectively. In addition to combining the p-values of the pair wise test, k-sample testes such as the k-sample AD test [28] can be performed.

A new diversity measure based on disagreement measure introduced by [12] is defined as follows: For each of the n_Y series points per model training series, the standard deviation in Y is computed. Then an $n_Y \times n_E$ error matrix ε is built from Y where $e_i^{[j]} \in [0, 1], i = 1, \dots, n_Y, j = 1, \dots, n_E$, where $e_i^{[j]} = 0$ if $e_i^{[j]}$ falls within a margin of one standard deviation around the mean of the forecast value of all models. Otherwise $e_i^{[j]} = 1$. Based on this, the new disagreement measure of the ensemble is

$$dis = \frac{\sum_{i=1}^{n_Y} \sum_{j=1}^{n_E} e_i^{[j]}}{n_Y n_E} \quad (12)$$

4.3 Selecting Most Diverse Ensemble

The methods in section 4.1 provide possibilities to select n_M feasible models $m_k \in M$ from the pool of models. With the methods in section 4.2 the diversity of an ensemble can be measured. This section introduces methods to select an ensemble $E \subset M$ with n_E models from the feasible n_M models such that the diversity d_E is maximised.

One of the most straightforward but computational intensive possibility for selecting the best n_E models is to enumerate all $\binom{n_M}{n_E}$ unique combinations of size n_E out of the n_M feasible models and select the combination with the highest diversity d_E . For larger n_M this becomes infeasible. In these cases, an approximation of the highest diversity d_E can be found for example with simulated annealing [9].

To find the optimal subset of n_E models with the pair-wise diversity measures, a mixed integer linear programming (MILP) approach is chosen. The underlying idea is to select the models such that the minimum of the pair-wise diversities is maximised. To model this as an MILP, firstly a variable d_E representing the diversity of the ensemble is introduced. Secondly, n_E binary variables x_1, \dots, x_{n_E} representing if a model m_k is selected or not are added to the program. Finally the following optimisation problem, as similarly introduced by Kuby [11], must be solved:

$$\begin{aligned}
& \text{maximise} && d_E \\
& \text{subject to} && \sum_{i=1}^{n_E} x_i = n_E, \\
& && d_E \leq M(2 - x_i - x_j) + d_{ij}, \ 1 \leq i \leq j \leq n_M \\
& && \forall x_i \in \{0, 1\} \\
& && d_E \geq 0
\end{aligned}$$

For the above-mentioned NP-complete MILP [6], a linear time approximation can be achieved by selecting the both most distant models and iteratively adding the most distant remaining model to the growing ensemble until n_E models are selected. Due to its greedy nature, this solution might not reach the optimum. As shown in the proof by Ravi et al. [26], for problems where the triangular inequality holds, this method is a 2-approximation of the optimal solution.

5 Experiments and Results

In order to evaluate the performance of the proposed methods, multiple experiments are conducted. Section 4.1 introduced three different approaches for model selection based on distribution similarity and 17 approaches on selecting the most diverse models, the combination of which result in 51 setups. For each experiment, all possible forecasts for the sub-sequences are performed and diversity mean (div_μ), diversity standard deviation (div_σ), mean squared forecast error (MSE), the mean MSE (MSE_μ) and the MSE standard deviation (MSE_σ) are computed. The experiments were conducted on two time series. The first one is a synthetic time series, generated from three different beta-distributions with known parameters. The synthetic time series is divided into 150 sub-series each coming from one of the three underlying distributions. The models to be selected are a collection of LSTMs, where each model is trained per sub-sequence. The second time series is a real-world time series representing the sales values of a store in five minutes buckets. Every day, 20 LSTMs are trained on data from the past seven days with different time lags. The data set spans over 324 continuous days. On both data sets, a fixed 10% of available models are selected into the ensemble.

In the first experiment, the three approaches for selecting sub sequences with the same distribution are compared. Table 1 presents the selection accuracy and the false positive (FPR) rate. The FPR measures the proportion of models in the ensemble that are not trained on the correct distribution and therefore can lead to higher prediction error.

	Accuracy $_{\mu}$	Accuracy $_{\sigma}$	FPR $_{\mu}$	FPR $_{\sigma}$
LS	0.97	0.05	0.001	0.005
KS	0.96	0.06	0.03	0.06
AD	0.95	0.07	0.02	0.03

Table 1: Performance of distribution similarity selection with local stationarity (LS), Kolmogorov-Smirnov test (KS) and Anderson-Darling test (AD)

As expected, the local-stationarity-based approach outperforms the traditional statistical measures. The higher FPR of the KS test suggests that the previously-mentioned shortcomings in detecting differences in the tails of the distributions do play a role.

For the comparison of different selection methods, three base lines are introduced. Base line A does not perform any selection and builds an ensemble of all available models. Base line B performs the distribution-based model selection but does not consider diversity. In base line C, the most diverse models are selected, but distribution similarity is not taken into account. Performance measures in terms of the mean squared error (MSE) are shown in table 2.

	Synthetic			Real-world		
	MSE $_{\mu}$	MSE $_{\sigma}$	MSE $_{median}$	MSE $_{\mu}$	MSE $_{\sigma}$	MSE $_{median}$
Base line A	231.63	204.84	128.85	659.10	248.40	585.64
Base line B	74.41	22.32	67.73	614.94	281.67	557.49
Base line C	203.91	122.42	190.21	630.42	264.36	562.41
Selected	69.24	20.75	69.02	603.27	246.46	556.93

Table 2: Performance in terms of MSE of the base lines and of the best selection

The performance differences among the base lines clearly suggest the importance of basing the models on similar distributions while training and forecasting. The same tendency can be observed in the real-world time series, although the difference is less salient due to more similar distributions.

Tables 3 and 4 show results obtained by the three optimisation methods for models pre-selected by diversity similarity evaluated by the local stationarity.

	Approximation				MILP			
	div_μ	div_σ	MSE_μ	MSE_σ	div_μ	div_σ	MSE_μ	MSE_σ
cov	6907.36	2191.24	79.12	22.44	6694.47	2059.24	71.01	22.02
corr	0.98	0.07	69.76	21.40	0.96	0.01	69.52	19.26
ent	7.14	0.25	71.89	20.62	7.24	0.23	68.77	20.74
$p\text{-ks}_2$	0.08	0.02	69.63	20.78	0.28	0.08	69.7	20.82
$p\text{-ad}_2$	0.13	0.20	69.24	20.75	0.18	0.03	69.87	20.10

Table 3: Performance of pair-wise diversity selections on local stationarity preselected synthetic time series

	div_μ	div_σ	MSE_μ	MSE_σ
cov	6354.32	1989.95	79.07	22.06
corr	0.95	0.07	71.61	22.76
ent	7.38	0.29	70.32	21.84
$p\text{-ks}_2$	0.05	0.01	70.13	20.59
$p\text{-ad}_2$	0.04	0.01	69.47	20.53
$p\text{-ad}_k$	0.02	0.01	71.28	21.01
dis	0.75	0.05	69.43	20.97

Table 4: Performance of combinatoric diversity selection on local stationarity preselected synthetic time series

For the MILP approach and the combinatorial optimisation, the covariance performs worse than the other diversity measures, whereas the correlation coefficient yields stable results for all method on both experiment data sets. The performance difference could be attributed to the correlation coefficient being a normalised measure whereas the covariance being on a less workable scale. Throughout all tests, the two-paired AD test and the correlation coefficient yield stable results. The newly proposed diversity measure for the complete ensemble yields the best results, but due to the computation of the standard deviation for each point of the time series, it is the slowest of all tested approaches.

The differences in the optimal solutions for pair-wise diversity measure or a measure on the complete ensemble are not this prominent. In the tested cases, the pair-wise optimisation perform slightly better than the measurements on the complete ensemble. Additionally, the pair-wise evaluations are less computational intensive.

The proposed two-stage selection on diversity and distribution equality outperforms all three base lines. To evaluate if the proposed methods are performing significantly better than the base lines, two-sample t-tests are performed. At a

significance level of $\alpha = 0.02$, the proposed method achieves significantly lower MSE than base line A and C. Although the performance gain over base line B is not statistically significant, the numerical differences in the results are visible.

After evaluating the performance of the methods from the forecasting accuracy and diversity measure point of view, the performance of the approximation, the MILP method and the combinatorial approach are tested. Figures 1, 2 and 3 show a two dimensional scaling of the pair-wise distances between 17 different real-world time series measured with the described diversity metrics. A smaller random subset of the time series was chosen for the sake of clearer visualisation. It is noteworthy that the distances in the two dimensional visualisation is an approximation of the actual distances for most of the measures, as entropy is the only diversity measure where triangle inequality holds. It can be seen that both the pair-wise approaches of approximation and MILP give different selections. Visually, the MILP selection appears to be more diverse than the approximation selection. This is in line with the assumption in section 4.2 that the greedy approach might not be optimal. Even though one could have expected that the MILP approach and the combinatorial approach yield the same result, it can be seen that this is not the case. This happens because the objective is different. The MILP approximates the diversity by pair-wise measurements whereas the combinatorial approach takes the final ensemble diversity into account. Taking the entropy as measurement, MILP and combinatorial approach made the same selection.

Additionally, it can be seen that the shapes of the distance visualisations of the correlation and covariance are similar, which is as expected. Moreover, the MILP method made marginally different selections for correlation and covariance. The correlation coefficient is shown to outperform the covariance as a diversity measure for all tested cases.

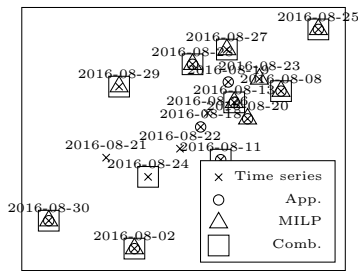


Fig. 1: Estimated two dimensional visualisation of covariance diversity selections

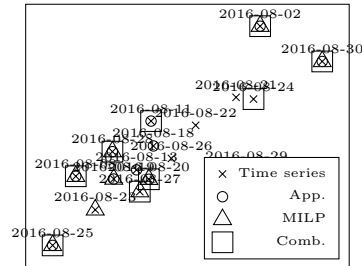


Fig. 2: Estimated two dimensional visualisation of correlation coefficient diversity selections

Finally, figure 4 shows the increase in accuracy throughout the selection and forecast process. The figure shows the range where the forecast could reside when

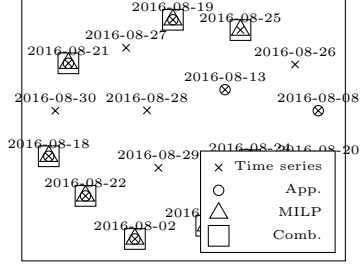


Fig. 3: Estimated two dimensional visualization of entropy diversity selections

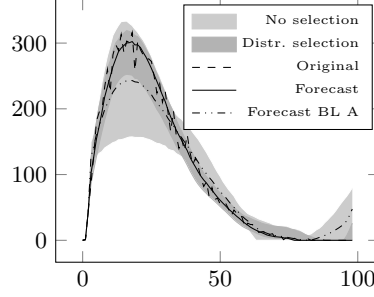


Fig. 4: Both stages of selection with final forecast of one sub-series

no selection is performed, the shrunk area which results from the diversity-based selection and the final forecast with the actual values.

6 Conclusion

This paper has shown that selecting specific models as members for an ensemble from a pool of trained models can increase the performance of the forecast. An increase in performance can be gained if models are selected based on the distributions of their training series. It is also important to select a set of diverse models for the final forecast. With the proposed methods, accurate results can be achieved by performing a smaller amount of forecasts than an ensemble of all available models would need. Testing for these distributions is challenging in the field of time series. It was shown that analysing stationary of time series yields good result, so does traditional statistical tests which consider differences in the tails of the underlying distributions.

Multiple measurements for diversity in forecasting ensembles based on the training series were introduced. Diversities in ensembles can be measured both on pair-wise basis and over the complete ensemble at once. In the tested series, the pair-wise measurement was faster in computation and marginally better in performance. The newly proposed ensemble-level measurement yielded highly promising results.

Lastly, optimising based on the size of the ensemble and the weights of its members could lead to an additional increase in performance and opens an interesting field for further research. It can be concluded that the underlying distribution of time series is an important feature in ensemble construction for time series forecasting.

Bibliography

- [1] Abou-Nasr, M.: Time series forecasting with recurrent neural networks nn3 competition (2007)
- [2] Anderson, T.W., Darling, D.A.: A test of goodness of fit. *Journal of the American statistical association* 49(268), 765–769 (1954)
- [3] Benjamini, Y., Hochberg, Y.: Controlling the false discovery rate: a practical and powerful approach to multiple testing. *Journal of the royal statistical society. Series B (Methodological)* 57(1), 289–300 (1995)
- [4] Brown, G., Wyatt, J.L., Tiño, P.: Managing diversity in regression ensembles. *Journal of Machine Learning Research* 6(9), 1621–1650 (2005)
- [5] Dutta, H.: Measuring diversity in regression ensembles. In: *IICAI*. vol. 9, p. 17 (2009)
- [6] Erkut, E.: The discrete p-dispersion problem. *European Journal of Operational Research* 46(1), 48–60 (1990)
- [7] Glickman, M.E., Rao, S.R., Schultz, M.R.: False discovery rate control is a recommended alternative to bonferroni-type adjustments in health studies. *Journal of clinical epidemiology* 67(8), 850–857 (2014)
- [8] Hochreiter, S., Schmidhuber, J.: Long short-term memory. *Neural computation* 9(8), 1735–1780 (1997)
- [9] Khachatryan, A.G., Semenovskaya, S.V., Vainstein, B.: A statistical-thermodynamic approach to determination of structure amplitude phases. *Sov. Phys. Crystallogr* 24, 519–524 (1979)
- [10] Krogh, A., Vedelsby, J.: Neural network ensembles, cross validation, and active learning. *Advances in neural information processing systems* 7, 231–238 (1995)
- [11] Kuby, M.J.: Programming models for facility dispersion: The p-dispersion and maxisum dispersion problems. *Geographical Analysis* 19(4), 315–329 (1987)
- [12] Kuncheva, L.I., Whitaker, C.J.: Ten measures of diversity in classifier ensembles: limits for two classifiers. In: *A DERA/IEE Workshop on intelligent Sensor Processing* (Ref. No. 2001/050). pp. 10–16. IET (2001)
- [13] Kuncheva, L.I., Whitaker, C.J.: Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy. *Machine learning* 51(2), 181–207 (2003)
- [14] Law, A.M., Kelton, W.D.: *Simulation modeling and analysis*, vol. 2. McGraw-Hill New York (1991)
- [15] Meek, C., Chickering, D.M., Heckerman, D.: Autoregressive tree models for time-series analysis. In: *Proceedings of the 2002 SIAM International Conference on Data Mining*. pp. 229–244. SIAM (2002)
- [16] Müller, K., Smola, A., Rätsch, G., Schölkopf, B., Kohlmorgen, J., Vapnik, V.: Using support vector machines for time series prediction. *Advances in kernel methods—support vector learning* pp. 243–254 (1999)

- [17] Nason, G.P.: Stationary and non-stationary time series. *Statistics in Volcanology. Special Publications of IAVCEI* 1, 000–000 (2006)
- [18] Nason, G.P.: A test for second-order stationarity and approximate confidence intervals for localized autocovariances for locally stationary time series. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 75(5), 879–904 (2013)
- [19] Nason, G.P., Von Sachs, R., Kroisandt, G.: Wavelet processes and adaptive estimation of the evolutionary wavelet spectrum. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 62(2), 271–292 (2000)
- [20] Ng, A.: *Machine Learning Yearning - Technical Strategy for AI Engineers, In the Era of Deep Learning* (2017)
- [21] Oliveira, M., Torgo, L.: Ensembles for time series forecasting. In: *ACML* (2014)
- [22] Pettitt, A.N.: A two-sample anderson–darling rank statistic. *Biometrika* 63(1), 161–168 (1976)
- [23] Polikar, R., Upda, L., Upda, S.S., Honavar, V.: Learn++: An incremental learning algorithm for supervised neural networks. *IEEE transactions on systems, man, and cybernetics, part C (applications and reviews)* 31(4), 497–508 (2001)
- [24] Prasad, S.C., Prasad, P.: Deep recurrent neural networks for time series prediction. *CoRR* abs/1407.5949 (2014)
- [25] Priestley, M.B., Rao, T.S.: A test for non-stationarity of time-series. *Journal of the Royal Statistical Society. Series B (Methodological)* 31(1), 140–149 (1969)
- [26] Ravi, S.S., Rosenkrantz, D.J., Tayi, G.K.: Heuristic and special case algorithms for dispersion problems. *Operations Research* 42(2), 299–310 (1994)
- [27] Salvador, S., Chan, P.: Toward accurate dynamic time warping in linear time and space. *Intelligent Data Analysis* 11(5), 561–580 (2007)
- [28] Scholz, F.W., Stephens, M.A.: K-sample anderson–darling tests. *Journal of the American Statistical Association* 82(399), 918–924 (1987)
- [29] Von Sachs, R., MacGibbon, B.: Non-parametric curve estimation by wavelet thresholding with locally stationary errors. *Scandinavian Journal of Statistics* 27(3), 475–499 (2000)
- [30] Whitlock, M.C.: Combining probability from independent tests: the weighted z-method is superior to fisher’s approach. *Journal of evolutionary biology* 18(5), 1368–1373 (2005)

A Hierarchical Bayesian Network for the Optimization of SRM Assays

Jérôme Renaux¹, Andrea Argentini^{2,3,4}, and Jan Ramon⁵

¹ Department of Computer Sciences, KU Leuven, Leuven, Belgium
`jerome.renaux@cs.kuleuven.be`

² VIB-UGent Center for Medical Biotechnology, Ghent, Belgium

³ Department of Biochemistry, Ghent University, Ghent, Belgium

⁴ Bioinformatics Institute Ghent, Ghent University, Ghent, Belgium

⁵ Inria Lille - Nord Europe, Lille, France

Abstract. Many experimental processes in biomedical sciences consist of several sequential steps. Predictions regarding the final output of these processes can be made based on the initial input by learning a mapping between the two and considering the corresponding process as a black box. This simple approach can be improved upon by opening the black box and performing inference about all the steps of the process as well as the relationships between them. This level of reasoning allows to answer a broader range of more refined queries, to potentially achieve better predictions and to gain insights into the workings of the process of interest. We present such an approach applied to mass spectrometry proteomics in the form a sequential architecture of probabilistic models trained to solve an important problem in that field.

1 Introduction

Many experimental processes in biomedical sciences can be decomposed in a series of sequential operations. Such architectures are often hierarchical in the sense that the product of one operation is a function of its input, which in turn becomes the input for the next operation. The accumulation of transformations taking place during the different intermediate operations lead from the initial input of the system to its final output. We will refer to such architecture as hierarchical processes. Examples of hierarchical experimental processes can be found in applications of mass spectrometry, drug design, and gene sequencing. [1–3]

When such systems are modelled, they are often treated as black box models, where machine learning methods can be used to learn a mapping from the initial input to the final output, without extensive consideration regarding the intermediate operations. This simple approach can be improved upon by opening the black box and performing inference about all the steps of the process, as well as the relationships between them. This level of reasoning should allow answering a broader range of more refined queries, potentially achieving better predictions and gaining insights into the mechanisms of the process of interest.

Moreover, this explicit modelling of the internal operations of the process of interest leads to a more interpretable model, better able to convey the aforementioned new insights. The interpretability of a machine learning model refers to the extent to which its predictions can be explained in terms of the importance and interactions of the features. Decision trees constitute an example of interpretable models. During the learning phase, the features are selected according to their discriminative power and ranked accordingly in the construction of the tree [4]. This reasoning is immediately visible when looking at the learned decision tree, which can be read as a workflow chart and understood even by users who are not familiar with machine learning. On the other hand, neural networks for example do not provide this interpretability; the weights in the intermediate layers of a network provide little insight about the learning process. Interpretability is particularly relevant in biomedical applications, where explicit models of the inner workings of the different aspects of the process under study can be as important as the final predictions. More generally, it may even become a legal requirement in a much wider range of applications, as specified in the recent EU General Data Protection Regulation.[5]

The goal of opening the black box requires more advanced inference techniques in order to be able to capture relational dependencies between intermediate steps, to deal with the nondeterministic nature of these in-between steps (as often encountered when attempting to model biological phenomena) and to cope with the fact that the results of all intermediate steps are not always completely observable. The combination of uncertainty and need for relational modelling make this application fall in the context of Statistical Relational Learning (SRL) [6], which allows to represent the process of interest as a probabilistic graphical model and provides techniques to perform efficient inference over such structures. At the same time, the multi-layered architecture of the processes discussed here also invite a comparison with deep learning, which has proven to perform very well on a multitude of machine learning tasks.

Bayesian networks (BN) are a family of directed probabilistic graphical models. The nodes in the graph correspond to random variables that can correspond to both observable and unobservable (latent) steps in a process. The edges indicate directed conditional dependencies. Probability functions can be learned for each node, taking as input the output of the parent nodes and providing probabilistic outputs in return. While this formulation can directly match the structure of hierarchical processes, in practice the grounded networks obtained from many experimental applications can be extremely large, requiring generalizing strategies such as lifted inference.

Deep Neural Networks (DNN) [7] consist in multiple layers of linear and non-linear transformations, where each layer extracts new, abstract features in an unsupervised fashion. They present several advantages, such as their generally high performance and their ability to function well even when the important features and the structure of their interactions are not known. However, the intermediate features extracted within each layer tend to be highly uninterpretable, and may not necessarily match directly the intermediate steps of the hierarchi-

cal process being modelled. Each layer becomes a black box on its own, which prevents an explicit analysis of the learning taking place, as well as the incorporation of domain knowledge. This family of learners has been very successful on image processing and signal analysis tasks, where these shortcomings have little importance due to the nature of the inputs. However, these constitute important issues in the context of biomedical applications. In addition, DNN do not handle missing values and noise well and do not provide straightforward ways to model uncertainty. Finally, they require vast datasets and are very computationally expensive in the learning phase.

We propose an approach specifically tailored to reason about hierarchical experimental designs, which combines relevant aspects of the previously discussed approaches while addressing the shortcomings we have mentioned. This approach consists in setting up a cascade of interpretable, probabilistic models, which form a hierarchical Bayesian network [8]. The structure of the network directly matches the structure of the hierarchical process to model. The random variables correspond to the intermediate operations of that process, and are modelled by interpretable models such as decision trees or random forests. The predictions made by one model constitute newly extracted features which become input features for the next model in the sequence, in addition to base features that are specific to each individual step of the process and relevant only to them (these offer the opportunity to incorporate information about domain knowledge). In addition to becoming inputs for subsequent mappings, the predictions of each model accumulate until the end of the cascade, in order to be usable in conjunction for additional predictions.

In the following sections, we illustrate this approach in the context of its application to mass spectrometry proteomics.

2 Problem statement

Mass spectrometry (MS) is an analytical technique used to analyze biological samples. It is used extensively in drug design, medical diagnosis, food analysis and forensics. Proteomics correspond to the study of proteins and their functions. To that end, mass spectrometry is often used to characterize the proteic content of biological samples [1]. The mass spectrometry process consists in a series of transformations applied to proteins in order to be able to detect and quantify their presence. The input is a sample containing a certain amount of proteins, and the output is a mass spectrum, indicating the distribution of the detected masses. This distribution can be used to infer the initial composition of sample. Due to the succession of intermediate steps occurring in such an experiment, this setting matches the formulation of hierarchical experimental processes.

In a typical MS experiment, a high amount of molecules is processed, of which very few are relevant. This is a situation of low signal to noise ratio. In addition, due to the resource constraints related to the instruments, it is not possible to monitor all the molecules that go through the experiment. This calls for an

optimization of the monitoring time, a problem that is referred to as Selected Reaction Monitoring (SRM) [9].

More specifically, this selective monitoring is carried out through the use of two filtering steps based on the masses of the molecules at two different stages. The determination of the filtering to operate is done by specifying transitions, tuples of three real positive values corresponding to the two masses to filter and the point in time at which the filtering must take place. Informally, this corresponds to specifying what to monitor and when.

In practice, multiple proteins of interest are monitored during one experiment, by specifying multiple transitions. A set of transitions of interest is called an assay. Given a set \mathcal{P} of target proteins, the objective is to elaborate an optimal SRM assay to detect them. More formally, find the composition of the optimal assay A to be monitored in a mass spectrometry experiment in order to maximize the probability of detecting the proteins in \mathcal{P} . This can be formulated in terms of the optimization problem formalized in (1).

$$\text{optimalAssay} = \arg \max_{a \in \mathcal{A}_P} f(a) \quad f : \mathcal{A}_P \rightarrow [0, 1] \quad . \quad (1)$$

Where \mathcal{A}_P is the set of all possible SRM assays and the objective function $f(a)$ maps an assay to a value between 0 and 1, corresponding to the accuracy of the identifications achieved with the selected assay.

This main problem involves the following sub-problems. First, because assays are lists of transitions, the set of all possible transitions needs to be known. In other words, a function mapping a set of proteins to a set of observable transitions (together with their properties) is required.

Second, enumerating all possible assays is to be avoided because of the combinatorial explosion inherent to considering all possible combinations of transitions. Therefore, it is necessary to explore the space of possible assays in a more efficient way, through a heuristic search. This requires extracting additional features which can be used to identify good transitions and guide the search.

In the next section, we show how these two problems are addressed in the context of the proposed inference approach.

3 Method

3.1 Enumerating the possible transitions

The first part of the approach is the enumeration of all the possibly observable transitions from the set \mathcal{Prot} of all existing proteins of a given species. This set of all possible transitions will henceforth be referred to as \mathcal{T} . This operation needs only be done once and provides the basis from which optimal assays can be computed for an arbitrary amount of queries regarding any set \mathcal{P} of target proteins from the same species.

Enumerating \mathcal{T} is achieved using a hierarchical Bayesian network architecture designed to correspond to the multiple stages of a typical mass spectrometry (MS) experiment. We refer the reader to [9] for more information on these stages;

the description that follows pertain to the way they were modelled and doesn't discuss the biological aspects in details.

Protein cleavage is modelled using CP-DT [10], which models the cleaving behavior of the trypsin enzyme using random forests. The prediction of the retention time of the cleaved proteins is achieved using the Elude tool from the Percolator suite [11]. The distribution of electrical charges that the protein fragments can take is modelled using random forests trained on experimental data mined from the PRIDE database [12]. Fragmentation patterns and ion intensity are predicted with the help of two random forest models [13, 14] trained on MS-LIMS data [15]. Finally, prior knowledge about the abundance of proteins within a given proteome is incorporated as prior probabilities, obtained when available from the PaxDB protein abundance database [16]. As all these steps have multiple possible outcomes and are nondeterministic, the actual outputs of the models consist in probability distributions over the possible predictions.

For each possible protein of a species (the list is obtained from the Uniprot database [17]), these models are run sequentially to make predictions about the transitions that will occur and their chemical properties. The sequential ordering of the steps in a MS experiment results in the fact that the outcome of one stage strongly depends on the outcomes of previous stages. This is reflected in our architecture by the fact that the predictions made by one model become additional features for the next model. The information flow between models and the probabilistic nature of their predictions make the architecture correspond to a Bayesian Network. The cleavage and fragmentation models allow to list the possible transitions, while the other models allow to enumerate the combinations of the different other chemical properties being considered, thus yielding the entire set of observable transitions \mathcal{T} . In that respect, the hierarchical Bayesian networks acts as a function mapping a set of proteins \mathcal{P} to a set of transitions \mathcal{T} .

Transitions are weighted by their probability of occurrence, derived from the probabilities predicted by each intermediate node in the Bayesian network.

3.2 Heuristic search using isolation scores

The knowledge of \mathcal{T} can be used to determine the composition of an optimal assay for a set of target proteins $\mathcal{P} \subsetneq \text{Prot}$. The search for the optimal assay is guided by a heuristic called the isolation score.

The transitions that can only be generated by the target proteins are first selected. They constitute the set $\mathcal{C}_{\mathcal{P}}$ of candidates transitions from which we attempt to constitute the optimal assay. For each transition, a measure of its quality referred to as its isolation score is computed. The intuition behind the selection of good transitions is that they should be reliably observable (high probability of occurrence and high detection intensity) and must overlap as little as possible with co-occurring transitions from non-target proteins in terms of mass.

The isolation score is designed to reflect these properties and can be computed by considering the spatial organization of transitions in a four-dimensional space.

The four dimensions of that space correspond to the key properties of ideal transitions mentioned in the previous paragraph, namely:

1. Mass at the first filtering step (Q1),
2. Mass at the second filtering step (Q3),
3. Probability of occurrence,
4. Predicted detection intensity.

For each candidate transition in \mathcal{C}_T , its neighborhood in the transition space is defined as the set of all transitions whose Q1 and Q3 values are within one unit from the candidate’s own Q1 and Q3 values (the first two axes listed above). The isolation score of the candidate is then computed as a weighted count of the neighboring transitions. The lower the score, the better, indicating that the candidate is more isolated in the transition space. Neighboring transitions are weighted by their probability and intensity. The rationale is that the proximity of rarely observed or low-intensity neighbors is less problematic than the vicinity of highly intense, highly probable ones.

Equation (2) illustrates the computation of the isolation score, capturing the heuristic used to determine the quality of a transition.

$$i = \sum_{t \in N} p_t * I_t . \quad (2)$$

Where i is the isolation score, N the set of transitions in the neighborhood, and p_t and I_t the predicted probability and intensity of transition t , respectively.

The computation of the neighborhood of a transition in the transition space is accomplished efficiently by storing all the transitions in a R-tree, a data structure optimized for spatial queries such as neighborhood counts [18].

Once a score has been attributed to each candidate transition, the list of candidates is sorted from best (lowest score) to worst (highest score), and the n best transitions can be selected to constitute the optimal assay (n being an instrument-related parameter.)

3.3 Validation

Our approach was validated against three datasets listing empirically curated transitions for human proteins. Table 1 summarizes the dataset that were used¹. The transitions in these datasets, which we we’ll refer to as reference transitions, have been validated through lab experiments as allowing to detect their target proteins reliably and accurately.

However, there is no strong guarantee that the listed transitions in these assays are the optimal ones. There may exist equally good or better transitions for

¹ The SRM Atlas dataset is actually a test subset of the entire SRM Atlas database. Some data from SRM Atlas was used to train some of the models of the architecture presented here. The training data was randomly selected from the entire database. Similarly, this test subset consists in 2500 randomly selected proteins which have not been used in the training set.

Table 1. Description of the three test datasets

Dataset	Number of proteins	Average number of transitions
SRMatlas [19]	2500	6.2
Transcription factors dataset [20]	96	4.2
Cervical cancer dataset [21, 22]	36	5.1

the same target proteins that have not been reported. Therefore, the possibility exists that optimal or close to optimal transitions that are not present in the reference databases may be predicted as optimal by our approach. Because of this, instead of attempting to find a hard correspondence between the predicted transitions and the reference transitions, we exploited the fact that the output of our approach consists in a ranked list of the candidate transitions, based on their isolation score. This allows to use ranking performance metrics and to accommodate for possible missing values in the reference data. In the remainder of this section, we describe the two ranking metrics that we used to measure the performance of our approach.

Average Median Rank The Average Median Rank (AMR) corresponds to the average across all the proteins of a dataset of the median rank of the reference transitions within the ranked list of the candidate transitions. The intuition behind this measure is that the reference transitions should have low isolation scores. Therefore, the candidate transitions which correspond to reference transitions should have low ranks in the list of all candidates.

One AMR is computed for each dataset and represents the performance of the approach on the dataset it is computed on. For each protein in the dataset, the candidate transitions are enumerated. That list of candidates contains all possible transitions for the target protein, including the reference ones. The R-tree storing \mathcal{T} is queried to compute the isolation score of each candidate. The candidates are then ranked according to that score. The median of the ranks of the reference transitions among the 30 top candidates is computed and stored. Finally, the average of all the median ranks over the test set is computed. See Table 2 in the next section for the lower and upper bounds that the AMR can take on each dataset.

Normalized Discounted Cumulative Gain The Discounted Cumulative Gain (DCG) [23] is used in the field of information retrieval to assess the effectiveness of search engine algorithms. It is primarily used in a setting where the objects of interest are documents, which can have varying degrees of relevancy with respect to a query. The DCG captures the usefulness (or gain) of the ranked list of retrieved documents in response to the query, by taking into account both their relevance and their positions in the list. This measure was designed to penalize the low ranking of highly relevant documents. The formula to compute the DCG is displayed in Equation 3.

$$DCG_p = \sum_{i=1}^p \frac{rel_i}{\log_2(i+1)} \quad (3)$$

Where $rel_i \in \mathbb{N}$ is the relevance score of document i and p is the rank up to which documents are considered (e.g. DCG_{10} is the DCG computed over the 10 top ranked documents). The same formula can be used when the relevance of documents is binary ($rel_i \in \{0, 1\}$) [24]. The DCG can be normalized, resulting in the Normalized Discounted Cumulative Gain (NDCG) as shown in Equations 4 and 5. NDCG values range from 0 to 1. The closer to 1, the better the performance.

$$nDCG_p = \frac{DCG_p}{IDCG_p} \quad (4)$$

$$IDCG_p = \sum_{i=1}^{|REL|} \frac{2^{rel_i} - 1}{\log_2(i+1)} \quad (5)$$

This operation corresponds to dividing the DCG by the ideal DCG (IDCG), which would be obtained if the documents were perfectly ranked in descending order of relevance.

This measure, although initially designed to evaluate search engines, can be applied to the setting discussed in this article as well, if transitions are considered as "documents". The candidate transitions constitute the list of documents obtained as a result of a query. Each transition can be considered as relevant or not (or having a rel value of 0 or 1) depending on its presence in the reference database. Consequently, a NDCG can be computed for each protein, and averaged across all proteins to provide one NDCG per dataset.

4 Results

Table 2 summarizes important AMR values computed for each of the three test sets. The result is the AMR score obtained by our approach.

The lower bound corresponds to the performance that a perfect predictor would achieve by consistently ranking the n reference transitions of every target protein at the n first positions. If for example 5 transitions are available as references, and they constitute the top 5 of the predictions, the median of the ranks would be the median of the numbers 1 to 5, which would be 3.

The upper bound on the other hand indicates the performance of the worst possible predictor that would consistently rank the n reference transitions at the n last positions out of the 30 selected candidates.

Table 3 reports the $NDCG_{30}$ scores obtained on the three datasets.

To further investigate the usefulness of the isolation score as an important feature extracted by the network, we used it in a classical binary classification setting, to distinguish reference transitions from non-reference transitions.

Table 2. Significant AMR values and results for each test set

Dataset	Lower bound	Upper bound	Result
SRMATlas	3.6	27.4	10
Transcription factors dataset	2.6	28.4	5.5
Cervical cancer dataset	3	27.9	4

Table 3. NDCG results for each test set

Dataset	NDCG ₃₀
SRMATlas	0.45
Transcription factors dataset	0.78
Cervical cancer dataset	0.82

The non-reference transitions were randomly selected in equal number as the reference ones and paired with the latter in terms of probability and intensity. We then trained two random forest models to discriminate between these two classes. Model A exclusively used the chemical properties of the transitions as features, while model B used the isolation score in addition. The results of both models are indicated in Table 4.

Table 4. Classification performance of transitions with and without the use of the isolation score

Dataset	Model A accuracy	Model B accuracy
SRMATlas	0.65	0.89

5 Conclusion

The obtained results seem to indicate that complex hierarchical experimental processes can be successfully modelled using a hierarchical Bayesian architecture. This architecture allows to enumerate the possible intermediate and final outcomes of an experiment, as well as to engineer a new feature (the isolation score) which correlates well with the solution of a specific optimization problem. This feature can be designed by combining explicit background knowledge about the desirable properties of the solution with intermediate features generated by the network.

SRM is an important problem in the field of proteomics, because of the value of knowing good transitions for a protein, or, conversely, the cost of having to run multiple experiments to find some. This problem and the kind of reasoning about mass spectrometry experiments that it requires, opens the way to other

inference tasks, such as the optimization of other aspects of an experiment (like the parameters of the instruments), or the interpretation of mass spectra to infer the full composition of a sample, for examples. More generally, the architecture we propose should be flexible enough to perform inference about the various kinds of multistage processes encountered in experimental settings, where the observed data is generated by the interactions of many different factors on multiple levels. The cascade of models presents an interesting similarity with deep learning, in the form of the new features computed at each stage based on the input of the previous ones. An important difference however lies in the highly probabilistic nature of this design, since it corresponds to a specialization of the Bayesian network approach. Another key difference is the fact that the intermediate stages are interpretable, as opposed to neural network models where the inner layers generate very abstract features. The white-box quality of this deep structure makes it particularly appropriate for experimental design settings in the life sciences, where insight into the underlying processes at work can be almost as important as the final predictions.

References

1. Aebersold, R., Mann, M.: Mass spectrometry-based proteomics. *Nature* **422**(6928) (2003) 198–207
2. Vyas, B., Singh, M., Kaur, M., Silakari, O., Bahia, M.S., Singh, B.: Pharmacophore and docking-based hierarchical virtual screening for the designing of aldose reductase inhibitors: synthesis and biological evaluation. *Medicinal Chemistry Research* **25**(4) (2016) 609–626
3. Weber, J.L., Myers, E.W.: Human whole-genome shotgun sequencing. *Genome research* **7**(5) (1997) 401–409
4. Quinlan, J.R.: Induction of decision trees. *Machine learning* **1**(1) (1986) 81–106
5. Goodman, B., Flaxman, S.: Eu regulations on algorithmic decision-making and a right to explanation. In: *ICML workshop on human interpretability in machine learning (WHI 2016)*, New York, NY. <http://arxiv.org/abs/1606.08813> v1. (2016)
6. De Raedt, L., Kersting, K.: Statistical relational learning. In: *Encyclopedia of Machine Learning*. Springer (2011) 916–924
7. LeCun, Y., Bengio, Y., Hinton, G.: Deep learning. *Nature* **521**(7553) (2015) 436–444
8. Gyftodimos, E., Flach, P.A.: Hierarchical bayesian networks: A probabilistic reasoning model for structured domains. In: *Proceedings of the ICML-2002 Workshop on Development of Representations*, The university of New South Wales (2002) 23–30
9. Lange, V., Picotti, P., Domon, B., Aebersold, R.: Selected reaction monitoring for quantitative proteomics: a tutorial. *Molecular systems biology* **4**(1) (2008) 222
10. Fannes, T., Vandermarliere, E., Schietgat, L., Degroeve, S., Martens, L., Ramon, J.: Predicting tryptic cleavage from proteomics data using decision tree ensembles. *Journal of proteome research* **12**(5) (2013) 2253–2259
11. Moruz, L., Staes, A., Foster, J.M., Hatzou, M., Timmerman, E., Martens, L., Käll, L.: Chromatographic retention time prediction for posttranslationally modified peptides. *Proteomics* **12**(8) (2012) 1151–1159

12. Vizcaíno, J.A., Côté, R.G., Csordas, A., Dienes, J.A., Fabregat, A., Foster, J.M., Griss, J., Alpi, E., Birim, M., Contell, J., et al.: The proteomics identifications (pride) database and associated tools: status in 2013. *Nucleic acids research* **41**(D1) (2012) D1063–D1069
13. Degroeve, S., Martens, L.: Ms2pip: a tool for ms/ms peak intensity prediction. *Bioinformatics* **29**(24) (2013) 3199–3203
14. De Grave, K., Van den Bulck, A., Touzé, S., Fannes, T., Ramon, J.: Prediction of peptide fragment ion intensity: A priori partitioning reconsidered. In: *IMSC 2014: 20th International Mass Spectrometry Conference*. Volume 20. (2014) 507–507
15. Helsen, K., Colaert, N., Barsnes, H., Muth, T., Flikka, K., Staes, A., Timmerman, E., Wortelkamp, S., Sickmann, A., Vandekerckhove, J., et al.: ms.lims, a simple yet powerful open source laboratory information management system for ms-driven proteomics. *Proteomics* **10**(6) (2010) 1261–1264
16. Wang, M., Weiss, M., Simonovic, M., Haertinger, G., Schrimpf, S.P., Hengartner, M.O., von Mering, C.: Paxdb, a database of protein abundance averages across all three domains of life. *Molecular & Cellular Proteomics* **11**(8) (2012) 492–500
17. Consortium, U., et al.: Uniprot: the universal protein knowledgebase. *Nucleic acids research* **45**(D1) (2017) D158–D169
18. Guttman, A.: R-trees: A dynamic index structure for spatial searching. Volume 14. *ACM* (1984)
19. Picotti, P., Clément-Ziza, M., Lam, H., Campbell, D.S., Schmidt, A., Deutsch, E.W., Röst, H., Sun, Z., Rinner, O., Reiter, L., et al.: A complete mass spectrometric map for the analysis of the yeast proteome and its application to quantitative trait analysis. *Nature* **494**(7436) (2013) 266
20. Stergachis, A.B., MacLean, B., Lee, K., Stamatoyannopoulos, J.A., MacCoss, M.J.: Rapid empirical discovery of optimal peptides for targeted proteomics. *Nature methods* **8**(12) (2011) 1041–1043
21. Van Ostade, X., Dom, M., Van Raemdonck, G.: Ipa analysis of cervicovaginal fluid from precancerous women points to the presence of biomarkers for the precancerous state of cervical carcinoma. *Proteomes* **2**(3) (2014) 426–450
22. Van Raemdonck, G.A., Tjalma, W.A., Coen, E.P., Depuydt, C.E., Van Ostade, X.W.: Identification of protein biomarkers for cervical cancer using human cervicovaginal fluid. *PloS one* **9**(9) (2014) e106488
23. Järvelin, K., Kekäläinen, J.: Cumulated gain-based evaluation of ir techniques. *ACM Transactions on Information Systems (TOIS)* **20**(4) (2002) 422–446
24. Croft, W.B., Metzler, D., Strohman, T.: Search engines: Information retrieval in practice. Volume 283. Addison-Wesley Reading (2010)

Deep Colorization for Facial Gender Recognition

Jonathan Hogervorst, Emmanuel Okafor, and Marco Wiering

Institute of Artificial Intelligence and Cognitive Engineering,
Faculty of Science and Engineering, University of Groningen, The Netherlands
`jonathan@hogervorst.info, {e.okafor, m.a.wiering}@rug.nl`

Abstract. Recent research suggests that colorization models have the capability of generating plausible color versions from grayscale images. In this paper, we investigate whether colorization prior to gender classification improves classification performance on the FERET grayscale face dataset. For this, we colorize the images using an existing Lab colorization model, both with and without class rebalancing, and our novel HSV colorization model without class rebalancing. Then we construct gender classification models on the grayscale and colorized datasets using a reduced GoogLeNet convolutional neural network. Several models are trained using different loss functions (cross entropy loss, hinge loss) and gradient optimization solvers (Nesterov’s Accelerated Gradient Descent, Stochastic Gradient Descent), initialized using both random and pre-trained weights. Finally, we compare the gender classification accuracies of the models when applied to the face image color variants. The best performances are obtained by models initialized using pre-trained weights, and models using colorization without class rebalancing.

Keywords: Deep Convolutional Neural Networks · Gender recognition · Face image analysis · Computer vision · Automatic colorization

1 Introduction

Gender recognition is an interesting problem with various applications, like video surveillance and authentication. Furthermore, gender plays an important role in human interaction. Gender recognition can therefore also be a useful method of improving human-computer interaction [20]. While gender recognition is a simple task for humans, it is difficult to create a system that can perform this task with sufficient performance. Additionally, face images may only be available in grayscale. For example, simple surveillance cameras and night vision cameras often do not provide color images, and historic photo and video material is also not available in color. Gender recognition may be more difficult on grayscale images because color images contain more information that may aid recognition.

Much research has been done into face gender recognition systems, applying various kinds of techniques [4, 24]. A successful approach is the use of support vector machines on face images [9, 15]. Recently, convolutional neural networks (CNNs) have been tried as an approach to gender classification as well [7, 20].

Research has examined the development of systems that can predict color information given a grayscale image using different approaches. One such approach is the transfer of color from a source image [13, 19], requiring one or more color source images featuring a scene similar to the image to be colorized. Another approach is colorization of grayscale images using color scribbles [8, 21]. While such systems do not need similar source images, they require the user to scribble appropriate colors onto the grayscale image. Recent research yielded colorization models which use large datasets of color images [3, 23]. These systems colorize images automatically, without any user input.

Contributions. In this paper, we investigate the influence of colorization prior to gender classification on the FERET grayscale face dataset [12]. The images are colorized using a colorization CNN [23] in Caffe [6]. We use the provided Lab colorization model, which predicts CIE *Lab* color values given a grayscale input image. We use the provided models with and without class rebalancing, which compensates for the fact that some color areas in the *ab* output space occur less often in photos. Furthermore, we adapt the CNN to predict HSV color values and train our own HSV colorization model without class rebalancing on the ImageNet dataset [14]. We then train gender classification models on the various face image color variants. For this, we use a reduced version of the GoogLeNet CNN [16] in Caffe. We compare two classification loss functions: cross entropy loss and hinge loss. Additionally, we compare two gradient optimization solvers: Nesterov’s Accelerated Gradient Descent (NAGD) [10] and Stochastic Gradient Descent (SGD). Gender classification models are trained both from scratch, using random Xavier initialization [5], and fine-tuned from a pre-trained GoogLeNet model. We compare classification accuracies of the eight resulting gender model variants for the four image color variants using ten-fold Monte Carlo cross-validation (MCCV).

Outline. In Section 2, we present our method: first, we train an HSV colorization model in addition to the existing Lab colorization models (Section 2.1); then, we colorize the grayscale images to new, colorized versions of the dataset using the colorization models (Section 2.2); and lastly, we use the colorized images in our gender classification models (Section 2.3). In Section 3, we provide our results. Finally, in Section 4, we discuss our conclusions.

2 Methods

2.1 Colorization

To perform colorization, we used a colorization CNN [23] implemented in the deep learning framework Caffe [6]. This CNN was created with the goal of predicting plausible color versions of grayscale images given as input. The CNN employs the CIE *Lab* color space: it predicts the *a* and *b* channels of an image, given its *L* channel. The *L* channel contains information about the image’s lightness or brightness — it thus provides a grayscale representation of the image.

The colorization CNN is unique due to its tailored loss function. Image color prediction is a multimodal problem [3]: objects in an image often have multiple

plausible colors. By using a loss function like the Euclidean loss, the optimal solution would be the mean of all plausible color values, resulting in grayish colors [23]. To prevent this, the colorization CNN predicts a probability distribution over the possible colors. For this, the in-gamut ab output space is divided into 313 bins, over which the probabilities are calculated. In addition, the loss function applies class rebalancing. This is done to compensate for the fact that some color values in the ab output space occur less often in photos. Class rebalancing reweighs the loss of pixels based on the rarity of the color during training, ensuring that the model can still predict less common color values.

The Lab colorization model was trained on the training set of the ImageNet dataset [14], containing 1.3M color images, in 450k iterations [23]. The ImageNet dataset consists of color images of many different types of objects collected from the Internet. We used the trained Lab models with¹ and without class rebalancing². More information about the loss function and training parameters of the Lab colorization model can be found in [23].

Moreover, we adopted the colorization CNN to create our own HSV colorization model. In this CNN, we predict the H and S channels of an image given its V channel. Similar to the L channel in the Lab color space, the V channel in the HSV color space provides a grayscale representation of an image. We applied binning over the HS output space using 18 equally sized bins in both directions, yielding 324 bins in total. We did not apply class rebalancing. We trained our HSV colorization model on the training set of the ImageNet dataset for 300k iterations using the unmodified training parameters of the Lab colorization CNN. This took ~ 14 days on a single core of an NVIDIA Tesla K40 GPU.

2.2 Dataset

We employed a selection of 1242 images of the grayscale face image dataset FERET [12]. This dataset consists of photos of subjects from several angles with different facial expressions. Our selection consisted of frontal images and images in which subjects had their face slightly turned. Our images were cropped to squares with the faces aligned in the middle and contained one color channel. From our selection of 1242 images, we put 255 aside as test set. We randomly split the remaining images into a training set of 789 images (80%) and a validation set of 198 images (20%). This random splitting procedure was repeated to create ten training/validation set pairs for MCCV.

We then colorized the images. To colorize an image using the Lab colorization models, we loaded it using the Caffe [6] function `io.load_image()`, automatically converting it to three-channel RGB data in grayscale. Then we converted it to Lab using the scikit-image [18] function `color.rgb2lab()` and provided the L channel as input to the colorization model. The output of the model, being the predicted ab color values, was combined with the calculated L channel. The resulting Lab image was then converted to RGB using `color.lab2rgb()` and

¹ `colorization_release_v2.caffemodel`

² `colorization_release_v2_norebal.caffemodel`

stored. We repeated this for all images. Similarly, each image was colorized using our HSV colorization model: we converted it to HSV using `color.rgb2hsv()`; provided the V channel to the model; combined the V channel with the predicted HS channels; converted it to RGB using `color.hsv2rgb()`; and stored it.

We thus had four variants of the dataset: original grayscale, Lab colorized with class rebalancing, Lab colorized without class rebalancing, and HSV colorized without class rebalancing. Some example images are shown in Fig. 1.



Fig. 1. Example images from the FERET face dataset [12] used in our research. From left to right: original grayscale, Lab colorized with class rebalancing, Lab colorized without class rebalancing, and HSV colorized without class rebalancing.

2.3 Gender Recognition

Network We performed gender classification using a reduced version of the GoogLeNet CNN [16] implemented in Caffe [6]. GoogLeNet contains Inception modules, each consisting of six convolution layers and one pooling layer, outputting the concatenation of its contained layers. The original CNN is 27 layers deep and contains nine Inception modules. It yielded an excellent performance on the ImageNet Large-Scale Visual Recognition Challenge 2014 [16].

We noticed that the extensive GoogLeNet structure would not be required for our gender classification task. Therefore we reduced it, removing six of the nine Inception layers. The resulting reduced GoogLeNet still provided good performance for our task, while saving $\sim 40\%$ computation time during training. The structure of our reduced GoogLeNet is shown in Fig. 2.

Loss Function Loss functions indicate the difference between predicted values and target values in supervised learning. During training, the loss is minimized by the CNN to get the predictions closer to the target values, improving the model's performance. The original GoogLeNet contains a softmax classification layer which employs the cross entropy loss function (`SoftmaxWithLossLayer` in Caffe). We also applied this in our reduced CNN. Additionally, we created a variant of our CNN using the hinge loss function (`HingeLossLayer`). More information can be found in [1].

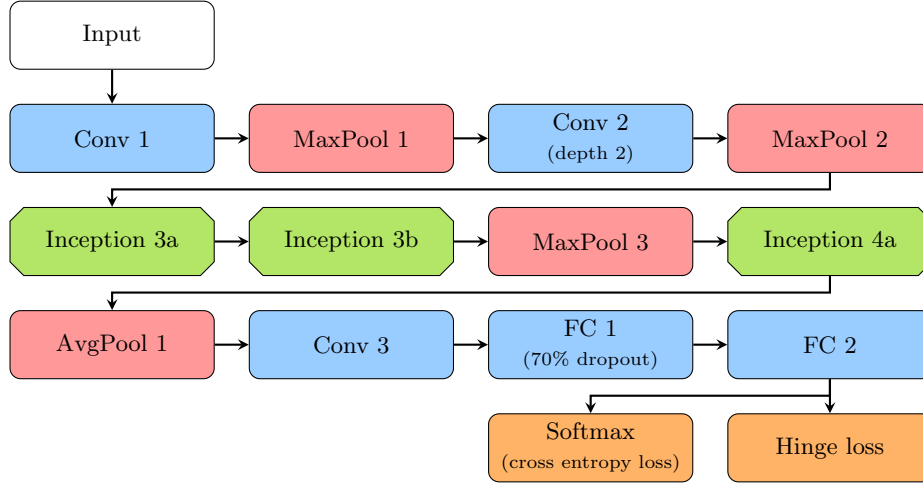


Fig. 2. Structure of our reduced GoogLeNet CNN for gender classification

Gradient Solver During training, the model’s parameters are updated to minimize the loss. The gradient solver specifies how the CNN updates its parameters. We employed two different gradient optimization solvers: NAGD [10] and SGD. More information can be found in [2].

Initialization The Caffe implementation of GoogLeNet employs Xavier initialization [5]. This algorithm initializes the model’s weights from a distribution around zero, with a variance based on the number of input/output neurons in the CNN. We employed this technique when training our models from scratch.

In addition to training from scratch, we also trained models by fine-tuning from pre-trained weights. In this case, the weights in our models were initialized using weights from the pre-trained Caffe GoogLeNet model³, which was trained on the ImageNet dataset similar to [16].

Training Parameters When training our models, we set the base learning rate to 0.001 (`base_lr` parameter in Caffe’s `solver.prototxt` file). After each step of 10k training iterations (`stepsz`; `lr_policy=step`) we lowered the learning rate by multiplying it with 0.96 (`gamma`). Furthermore we used a momentum of 0.9 (`momentum`) and a weight decay factor of 0.0002 (`weight_decay`).

We trained gender classification models on the ten training sets for each color variant of our dataset. We trained each scratch model for 30k iterations, which took ~ 1.5 hour on a single core of an NVIDIA Tesla K40 GPU. We trained each fine-tuned model for 20k iterations, which took ~ 1 hour on the same GPU.

³ `bvlc_googlenet.caffemodel`

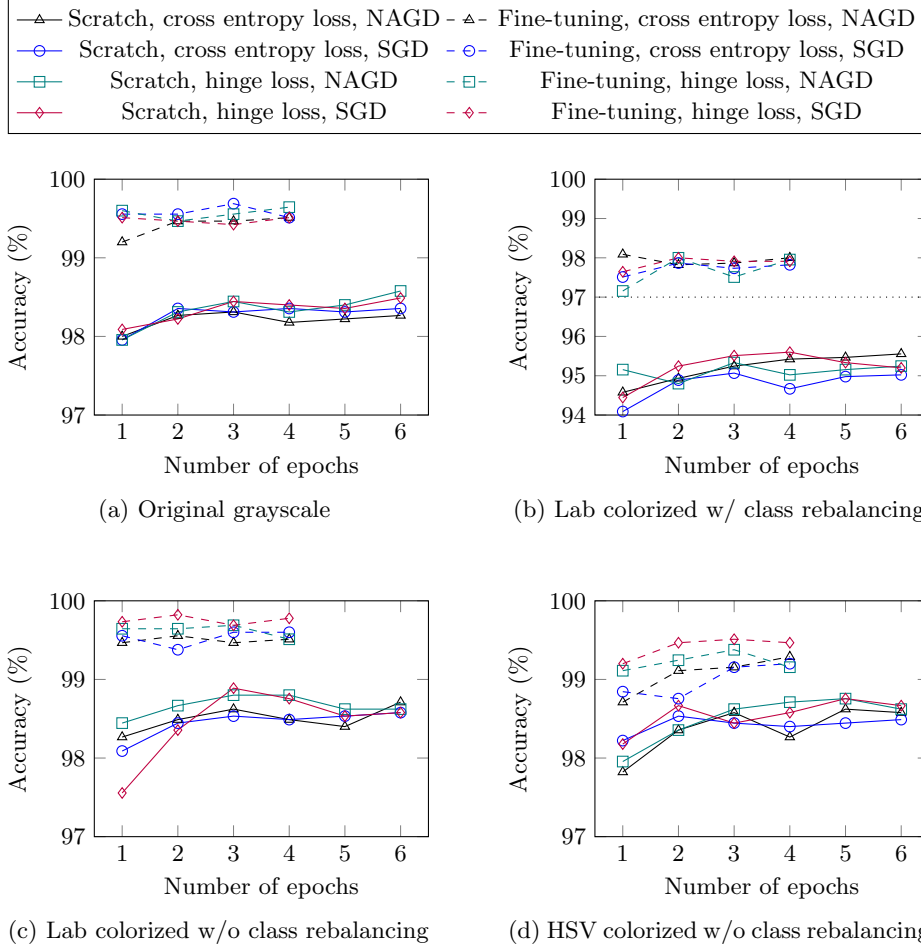


Fig. 3. Average gender classification accuracy of ten-fold MCCV on the test set after training for a number of epochs (1 epoch \equiv 5k iterations). The scratch CNN models were trained for 30k iterations, the fine-tuned CNN models for 20k iterations.

3 Results

The gender classification learning curves of our models after various numbers of training iterations are shown in Fig. 3. The scratch models, which used Xavier weight initialization, show significant improvements after the first and second training epoch; after that, the accuracies slightly improve further. The same applies to some of the fine-tuned models, which were initialized using pre-trained weights. Other fine-tuned models only show slight improvements from the first training epoch on. Because the models keep improving until the final training epoch, the models at the final training epoch will be used for further reporting.

Table 1. Average gender classification accuracy in % (\pm standard deviation) of ten-fold MCCV on the test set after training for 30k (scratch) or 20k (fine-tuning) iterations.

Training	Loss Function	Gradient Solver	Grayscale	Lab Color'ed w/ class rebal.	Lab Color'ed w/o class rebal.	HSV Color'ed w/o class rebal.
From scratch (Xavier init.)	Cross entropy loss	NAGD	98.27 ± 0.50	95.56 ± 0.66	98.71 ± 0.46	98.58 ± 0.55
		SGD	98.36 ± 0.40	95.02 ± 0.48	98.58 ± 0.62	98.49 ± 0.80
	Hinge loss	NAGD	98.58 ± 0.56	95.24 ± 0.63	98.62 ± 0.58	98.62 ± 0.42
		SGD	98.49 ± 0.41	95.20 ± 0.91	98.58 ± 0.62	98.67 ± 0.40
Fine-tuning (pre-trained weights)	Cross entropy loss	NAGD	99.51 ± 0.24	98.00 ± 0.46	99.51 ± 0.31	99.29 ± 0.22
		SGD	99.51 ± 0.42	97.82 ± 0.37	99.60 ± 0.24	99.20 ± 0.33
	Hinge loss	NAGD	99.64 ± 0.27	97.96 ± 0.80	99.51 ± 0.42	99.16 ± 0.42
		SGD	99.51 ± 0.31	97.91 ± 0.56	99.78 ± 0.30	99.47 ± 0.27

The gender classification test accuracies of our various models are reported in Table 1. The results show that the maximum classification accuracy is obtained by the fine-tuned model using hinge loss and SGD, applied on the Lab colored version of the dataset without class rebalancing, yielding 99.78%.

When we compare the accuracies of our models on the Lab and HSV colored images without class balancing, they show a significant increase over the models with class rebalancing. This implies that colorization without class rebalancing works better for this study. Furthermore, the models without class rebalancing only show a slight improvement over some of the original grayscale image models.

In general, our fine-tuned models showed more promising performance than their scratch counterparts on all versions of the dataset we examined. The results of the models using the different loss functions or gradient solvers show only very small differences — no conclusions can be drawn from this.

4 Conclusions

This study comprised a comprehensive evaluation using different variants of a reduced GoogLeNet CNN to determine the gender classification performance on original and colorized versions of the FERET grayscale face dataset. The results show that class rebalancing in colorization does not improve the gender classification performance in this study since all models without class rebalancing yielded better results than the models with class rebalancing. Additionally, fine-tuning from pre-trained weights seems to be the best approach in this study: all fine-tuned models yielded better performance than their scratch counterparts, while also requiring less training time.

In general, the gender classification accuracies from our models were all very high and similar. While we observed some improvements in models using colorization, they were in the order of tenths of a percentage point. To attribute sig-

nificant improvement to colorization prior to gender classification, there should be a greater difference in accuracy between the grayscale and colorized models.

A reason for the high classification accuracies could be that the model used some individual characteristics of subjects, rather than gender characteristics only. Because subjects were photographed from multiple angles for the FERET dataset, different images of the same face could have ended up in different sets, like our training and test set. While such images were not identical, they may have allowed the model to use face-specific characteristics.

Research has shown that color information improves the performance of face recognition systems [17]. Moreover, psychological research suggests that humans use color information in gender recognition if other information is unavailable [11, 22]. Our dataset consisted of high quality frontal whole face photos. Our approach of colorization prior to gender classification might be better applicable to more complex datasets. In that case, the grayscale baseline results would be worse, and colorization could bring actual improvement.

Future work should examine multi-orientation data augmentation using colorized images, which may improve the performance further.

Acknowledgments. The authors would like to thank the Center for Information Technology of the University of Groningen for their support and for providing access to the Peregrine high performance computing cluster.

References

1. Berkeley Vision and Learning Center: Caffe documentation. <http://caffe.berkeleyvision.org/doxygen/> (2017), accessed: 2017-03-17
2. Berkeley Vision and Learning Center: Caffe tutorial: Solver. <http://caffe.berkeleyvision.org/tutorial/solver.html> (2017), accessed: 2017-03-17
3. Charpiat, G., Hofmann, M., Schölkopf, B.: Automatic image colorization via multimodal predictions. In: Forsyth, D., Torr, P., Zisserman, A. (eds.) ECCV 2008. LNCS, vol. 5304, pp. 126–139. Springer, Berlin, Heidelberg (2008)
4. Chellappa, R., Wilson, C.L., Sirohey, S.: Human and machine recognition of faces: a survey. In: Proceedings of the IEEE. vol. 83, pp. 705–741. IEEE (1995)
5. Glorot, X., Bengio, Y.: Understanding the difficulty of training deep feedforward neural networks. In: Teh, Y.W., Titterton, M. (eds.) 13th International Conference on Artificial Intelligence and Statistics. JMLR Workshop and Conference Proceedings, vol. 9, pp. 249–256. PMLR (2010)
6. Jia, Y., Shelhamer, E., Donahue, J., Karayev, S., Long, J., Girshick, R., Guadarrama, S., Darrell, T.: Caffe: Convolutional architecture for fast feature embedding. In: 22nd ACM International Conference on Multimedia. pp. 675–678. ACM, New York (2014)
7. Levi, G., Hassner, T.: Age and gender classification using convolutional neural networks. In: 2015 IEEE Conference on Computer Vision and Pattern Recognition Workshops. pp. 34–42. IEEE (2015)
8. Levin, A., Lischinski, D., Weiss, Y.: Colorization using optimization. In: Hart, J.C. (ed.) ACM SIGGRAPH 2004. ACM Transactions on Graphics, vol. 23, pp. 689–694. ACM, New York (2004)

9. Moghaddam, B., Yang, M.H.: Gender classification with support vector machines. In: IEEE International Conference on Automatic Face and Gesture Recognition. pp. 306–311. IEEE (2000)
10. Nesterov, Y.: A method of solving a convex programming problem with convergence rate $O(1/k^2)$. Soviet Mathematics Doklady 27(2), 372–376 (1983)
11. Nestor, A., Tarr, M.J.: Gender recognition of human faces using color. Psychological Science 19(12), 1242–1246 (2008)
12. Phillips, P.J., Wechsler, H., Huang, J., Rauss, P.J.: The FERET database and evaluation procedure for face-recognition algorithms. Image and Vision Computing 16(5), 295–306 (1998)
13. Reinhard, E., Adhikhmin, M., Gooch, B., Shirley, P.: Color transfer between images. IEEE Computer Graphics and Applications 21(4), 34–41 (2001)
14. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A.C., Fei-Fei, L.: ImageNet large scale visual recognition challenge. Int J Comput Vis 115(3), 211–252 (2015)
15. Shan, C.: Learning local binary patterns for gender classification on real-world face images. Pattern Recognition Letters 33(4), 431–437 (2012)
16. Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., Rabinovich, A.: Going deeper with convolutions. In: 2015 IEEE Conference on Computer Vision and Pattern Recognition. pp. 1–9. IEEE (2015)
17. Torres, L., Reutter, J.Y., Lorente, L.: The importance of the color information in face recognition. In: 1999 International Conference on Image Processing. vol. 3, pp. 627–631. IEEE (1999)
18. van der Walt, S., Schönberger, J.L., Nunez-Iglesias, J., Boulogne, F., Warner, J.D., Yager, N., Goullart, E., Yu, T.: scikit-image: image processing in Python. PeerJ 2 (2014)
19. Welsh, T., Ashikhmin, M., Mueller, K.: Transferring color to greyscale images. In: ACM SIGGRAPH 2002. ACM Transactions on Graphics, vol. 21, pp. 277–280. ACM, New York (2002)
20. van de Wolfshaar, J., Karaaba, M.F., Wiering, M.A.: Deep convolutional neural networks and support vector machines for gender recognition. In: 2015 IEEE Symposium Series on Computational Intelligence. pp. 188–195. IEEE (2015)
21. Yatziv, L., Sapiro, G.: Fast image and video colorization using chrominance blending. IEEE Transactions on Image Processing 15(5), 1120–1129 (2006)
22. Yip, A.W., Sinha, P.: Contribution of color to face recognition. Perception 31(8), 995–1003 (2002)
23. Zhang, R., Isola, P., Efros, A.A.: Colorful image colorization. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9907, pp. 649–666. Springer, Cham (2016)
24. Zhao, W., Chellappa, R., Phillips, P.J., Rosenfeld, A.: Face recognition: A literature survey. ACM Computing Surveys 35(4), 399–458 (2003)

Predicting Chaotic Time Series using Machine Learning Techniques

Henry Maathuis*, Luuk Boulogne*, Marco Wiering, and Alef Sterk

University of Groningen, Groningen, The Netherlands,
maathuishenry@gmail.com, lhboulogne@gmail.com, m.a.wiering@rug.nl and
a.e.sterk@rug.nl

Abstract. Predicting chaotic time series can be applied in many fields, e.g. in the form of weather forecasting or predicting stocks. This paper discusses several neural network approaches to perform a regression prediction task on chaotic time series. Each approach is evaluated on its sequence prediction ability on three different data sets: the intermittency map, logistic map and a six-dimensional model. In order to investigate how well each regressor generalizes, they are compared to a 1 Nearest Neighbor baseline. In previous work, the Hierarchical Mixture of Experts architecture (HME) has been developed. For a given input, this architecture chooses between specialized neural networks. In this work, these experts are Multilayer Perceptrons (MLPs), Residual MLPs, and Long Short-Term Memory neural networks (LSTMs). The results indicate that a Residual MLP outperforms a standard MLP and an LSTM in sequence prediction tasks on the logistic map and the 6-dimensional model. The standard MLP performs best in a sequence prediction task on the intermittency map. With the use of HMEs, we successfully reduced the error in all the above mentioned time series prediction tasks.

Keywords: Dynamical Systems, Neural Networks, Hierarchical Mixture of Experts

1 Introduction

Time series prediction is a task that involves the use of historical information to predict the future states of a system. Such tasks are often partitioned in one-step predictions and the prediction of a sequence of states. Many disciplines are concerned with time series prediction. These disciplines range from weather stations forecasting days or even weeks ahead to stock brokers and traders predicting the course of stocks. Most problems in nature, including prediction problems, deal with nonlinear or chaotic data, which poses a challenge. In the field of Machine Learning, a large quantity of work on predicting time series that involves using Neural Networks (NNs) such as Multi-Layer Perceptrons (MLP) and Radial Basis Function NNs (RBF-NN) has already been done, including work on

*These authors contributed equally to this work.

weather forecasting [1, 2]. Other research has been conducted with other Supervised Learning models such as Support Vector Machines or Recurrent Neural Networks [3–5].

To measure how well a system performs on a prediction task it is often compared to a baseline. 1 Nearest Neighbor regression is used as a baseline in this research. This baseline is a powerful tool when large amounts of data are available (see Section 4.1). However, since temporal data of any kind might not always be available in abundance, it is interesting to see whether an NN can generalize better than the baseline.

Three types of chaotic time series are discussed in Section 2. Different kinds of NNs: MLPs, Residual MLPs and Long Short-Term Memory (LSTM), are considered in this work and their predictive performance on chaotic data is investigated. These NNs are also embedded within a larger architecture, the Hierarchical Mixture of Experts (HME). This architecture allows expert NNs to specialize in certain input regions so that the problem complexity is reduced for the separate experts.

The nature of the dynamical systems with which the data sets for this work are produced makes the behavior of these systems state dependent. Allowing for an HME system where agents specialize in certain types of states could improve the system performance. In this research we are interested to see whether the HME architecture is useful for improving the prediction accuracy on chaotic time series.

2 Data Sets

For the convenience of the reader we first give a general overview of dynamical systems. Next, we give the concrete examples used in our machine learning experiments.

2.1 Dynamical Systems

Dynamical systems are mathematical models for everything that evolves in time. Simple examples are springs and pendulum clocks. More complicated examples are the celestial bodies comprising the solar system or the atmosphere which produces everyday weather. These systems are deterministic in the sense that the present state of the system completely determines its future. In other words, probability does not play a role in the evolution of a system. See [6] for an extensive account.

Many dynamical systems arise in the form of *iterated maps*. Let D be some domain and consider a map $f : D \rightarrow D$. For an initial condition $x_0 \in D$ we can iterate the map f by setting $x_{n+1} = f(x_n)$. This gives the following time series:

$$x_0, f(x_0), f(f(x_0)), f(f(f(x_0))), \dots$$

A weather forecasting model, for example, fits in this framework. If an initial condition x_0 represents today's weather, then by solving the governing differential

equations of atmospheric physics we can compute a prediction for tomorrow's weather $f(x_0)$.

Since the seminal work of the mathematician and meteorologist E.N. Lorenz [7] it is well known that deterministic systems can be unpredictable: small errors in the initial condition x_0 may lead to large errors in predictions for the future. This phenomenon, which is colloquially known as *chaos*, hampers long-term weather forecasts and stimulated the development of mathematical research on nonlinear dynamics and chaos theory.

2.2 The Logistic Map and Intermittency Map

The most familiar, and perhaps simplest, example of a dynamical system with chaotic dynamics is the *logistic map* which is given by:

$$f : [0, 1] \rightarrow [0, 1], \quad f(x) = rx(1 - x), \quad (1)$$

where $0 < r \leq 4$ is a parameter. This map is a simple model for population growth which also takes overpopulation into account. Increasing the parameter r leads to a period doubling cascade, and for $r = 4$ the dynamics have been proven to be chaotic.

Another example of a 1-dimensional dynamical system is the so-called *intermittency map* [8] which is given by:

$$f : [0, 1] \rightarrow [0, 1], \quad f(x) = \begin{cases} x(1 + (2x)^\alpha) & \text{if } 0 \leq x \leq \frac{1}{2}, \\ 2x - 1 & \text{if } \frac{1}{2} < x \leq 1, \end{cases} \quad (2)$$

where $0 < \alpha < 1$ is a parameter. This map has a neutral fixed point at $x = 0$, which causes the time series to spend long times near $x = 0$. This effect becomes stronger when α tends to 1. For the experiments in our paper, this value is fixed to 0.5.

2.3 Atmosphere Data

A classical problem in the theory of atmospheric circulation is the characterization of the recurrent flow patterns observed at midlatitudes in northern hemisphere winters. The prime motivation for studying this phenomenon is to understand the persistence and predictability of atmospheric motion beyond the time scales of baroclinic synoptic disturbances (2 to 5 days). It is expected that insight in the nature of this so-called low-frequency variability will lead to significant progress in extended range weather forecasting [9].

Classical theories associate recurrent large-scale flow patterns with stationary states of the atmospheric circulation, which correspond to equilibria in the dynamical equations of atmospheric motion [10]. Small-scale weather acts then as a random perturbation inducing fluctuations around equilibria and transitions between states. From the perspective of nonlinear dynamical systems this scenario

has been explained in terms of *intermittency* which means that a system alternates between regimes of chaotic and regular behavior such as nearly steady or periodic dynamics. The intermittency map given by Equation (2) exhibits this type of dynamics. Intermittency is observed in various forms in atmospheric models [11, 12]; also see [13] for an overview.

The simplest model for the midlatitude atmospheric circulation is the barotropic vorticity equation for flow over an orography profile (i.e., mountains):

$$\frac{\partial}{\partial t} \Delta \psi = -J(\psi, \Delta \psi + \beta y + \gamma h) - C \Delta(\psi - \psi^*), \quad (3)$$

where ψ is the stream function which describes the atmospheric velocity field, ψ^* is the forcing, β controls the Coriolis force, and h is the orography profile. The differential operators Δ and J are defined as $\Delta f = f_{xx} + f_{yy}$ and $J(f, g) = f_x g_y - f_y g_x$, respectively. For parameter settings and boundary conditions, see [11, 14].

A *low-order model* can be derived from Equation (3) by means of spectral discretisation. The idea is to expand the stream function ψ in a truncated Fourier series with time-dependent coefficients. An orthogonal projection then gives a system of ordinary differential equations. A particular low-order model consisting of 6 ordinary differential equations was studied in [11] who found intermittent transitions between two states representing a westerly and a blocked flow respectively which resemble the patterns found in the real atmosphere. The dynamics consists of three recurrent episodes: (i) transitions from westerly to blocked flows, typically taking 30 days, (ii) transitions from blocked to westerly flows, typically taking 40–80 days, and (iii) spiraling behavior around the westerly regime, typically lasting more than 200 days. The dynamics of the intermittency map given by Equation (2) can be seen as a prototype for this more complicated scenario.

From the 6-dimensional model derived from Equation (3), see [11, 14] for explicit equations, we generated a discrete time series by numerical integration. We sampled the continuous time series by intervals of 6 hours.

2.4 Splitting the Data Sets

The total length of the resulting discrete time series for each type of dynamical system is 12,001 data points. The first 12,000 are used as input and the last 12,000 as the corresponding output. The resulting data sets are divided into three segments of equal length, which are used respectively as training, test and validation sets for our systems. For the atmosphere data, the training, validation and test sets each translate to approximately 3 years of data.

3 System Design

In this work, two different Neural Network (NN) regressors are considered to learn the behavior of the dynamical systems described in Section 2. These are the well established feedforward Multi-Layered Perceptron (MLP) and the recurrent

Long Short-Term Memory NN (LSTM). Each of these NNs are evaluated on their ability to predict sequences and they are compared to a simple but robust baseline (1 Nearest Neighbor).

To obtain good performance, we allow a regressor to express different behavior for different parts of the data set domain by using the structure of the Hierarchical Mixture of Experts [15] ensemble technique (HME) for our regressor, explained in Section 3.3.

3.1 (Residual) Multilayer Perceptron

The MLP is a very popular NN used in a lot of different tasks in which a nonlinear problem is to be solved. An MLP consists of an input layer, output layer and an arbitrary amount of hidden layers. An MLP maps an input vector to an output vector: $f : R^{input} \rightarrow R^{output}$. More specifically, the output is obtained by a combination of the input and the weights associated between the neurons in the NN. After obtaining the output for each neuron, the result is transformed by feeding it to an activation function. This allows the system to learn nonlinear decision boundaries. The parameters of the model are learned by the use of an optimizer. An optimizer describes how the weights in an NN are updated according to the gradient obtained by using the backpropagation algorithm. Backpropagation uses a loss function with respect to the weights to compute the gradient of the output error and propagates this back through the NN. Such a loss function $l(d, y)$ computes a similarity measure between the desired output d and the actual output y .

A closely related NN is the MLP that utilizes residual learning blocks. This NN uses residual learning blocks instead of the default hidden layers. Apart from this learning block, this NN is identical to the plain MLP. Residual learning blocks allow the system to learn not only from the output of the layer itself, but also from its input. In a standard setup the output of an NN can be represented as:

$$y = f(x). \quad (4)$$

Here x is the input and y is the mapped output. However in the case of a residual learning block, the output is calculated as:

$$y = f(x) + x. \quad (5)$$

Empirical evidence has shown that residual learning blocks can reduce the error of the NN and allow for easier optimization [16]. Figure 1(a) shows a schematic overview of the residual learning block.

3.2 Long Short-Term Memory

In MLPs a layer of neurons only contains forward connections to subsequent layers. Layers in Recurrent Neural Networks (RNNs) however, also have neurons that maintain connections to themselves and preceding layers. When predicting

the next time step in a series, these connections allow the past to be taken into account. This is because the activation of neurons is not only dependent on the input at the current time step, but also on input from earlier time steps [17]. An issue with RNNs is that the error gradients vanish after being propagated back through many layers or time steps. [18]. To overcome this problem, a Long Short-Term Memory NN (LSTM) has been introduced [19, 18]. LSTMs can be implemented by replacing the nodes in one or more hidden layers of an NN with so-called memory cells. These cells contain an internal state that is maintained using gates. The memory cell is depicted in Figure 1(b). Here, we give a brief description of the memory cell. A more extensive explanation of an LSTM and the memory cell is presented in [17].

A gate in a memory cell is defined as a neuron with a sigmoid activation function. In a gated connection from neuron A to neuron B , the activation of a gate is multiplied with the activation of A to obtain the input for B . Since the activation of a gate lies within the interval $(0,1)$, it can be viewed as indicating the percentage of activation of A that flows through to B .

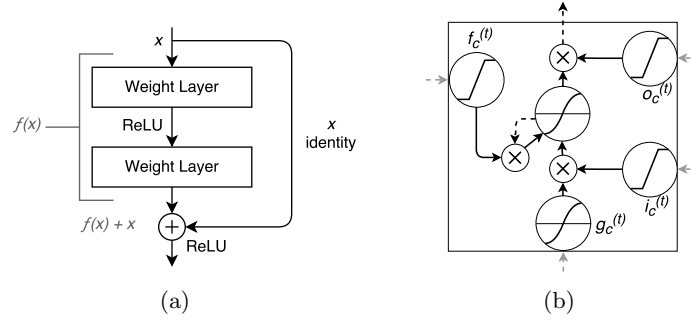


Fig. 1: (a): A residual learning block as described in [16]. (b): The LSTM memory cell extended with forget gates [20], after [17]. On the neurons, the corresponding activation function (hard sigmoid [21] or hyperbolic tangent) is depicted. The multiplication of the output of nodes is depicted as a circle with the multiplication symbol. All arrows can be regarded as connections with a weight fixed at one. Filled arrow: Connection within one time step. Black dashed arrow: Connection to the next time step. Gray dashed arrow: Input connection. This is a connection from the previous time step to the current one.

Including the forget gate, which is described in [20], a memory cell contains three gates. For memory cell c at time step t , these are an input gate $i_c^{(t)}$, a forget gate $f_c^{(t)}$ and an output gate $o_c^{(t)}$. Besides gates, the cell has an input neuron $g_c^{(t)}$ and an internal state $s_c^{(t)}$. It produces an activation $h_c^{(t)}$, which is the activation of c in the hidden layer that is used as input for subsequent layers. We compute a forward pass through a hidden layer of LSTM cells with input $x^{(t)}$, as described

by:

$$\begin{aligned}
\mathbf{g}^{(t)} &= \tanh(W^{gx}\mathbf{x}^{(t)} + W^{gh}\mathbf{h}^{(t-1)} + \mathbf{b}_g), \\
\mathbf{i}^{(t)} &= \hat{\sigma}(W^{ix}\mathbf{x}^{(t)} + W^{ih}\mathbf{h}^{(t-1)} + \mathbf{b}_i), \\
\mathbf{f}^{(t)} &= \hat{\sigma}(W^{fx}\mathbf{x}^{(t)} + W^{fh}\mathbf{h}^{(t-1)} + \mathbf{b}_f), \\
\mathbf{o}^{(t)} &= \hat{\sigma}(W^{ox}\mathbf{x}^{(t)} + W^{oh}\mathbf{h}^{(t-1)} + \mathbf{b}_o), \\
\mathbf{s}^{(t)} &= \mathbf{g}^{(t)} \odot \mathbf{i}^{(t)} + \mathbf{s}^{(t-1)} \odot \mathbf{f}^{(t)}, \\
\mathbf{h}^{(t)} &= \tanh(\mathbf{s}^{(t)}) \odot \mathbf{o}^{(t)},
\end{aligned} \tag{6}$$

after [17]. Here, element-wise multiplication is indicated with the symbol \odot . Furthermore, W^{ij} denotes the weight matrix from j to i , \mathbf{b}_i denotes the bias for i , \tanh denotes the hyperbolic tangent and $\hat{\sigma}$ denotes a variant of the hard sigmoid [21], described by:

$$\hat{\sigma}(x) = \max(0, \min(1, 0.2x + 0.5)). \tag{7}$$

3.3 Hierarchical Mixture of Experts

The Hierarchical Mixture of Experts (HME) is a tree-structured architecture for Supervised Learning that was coined by Jordan et al. [15]. Their system, incorporates the idea of Mixture of Experts (MoE) also originating from the authors in earlier research. MoE follows a divide-and-conquer principle in which the problem space is partitioned into local regions.

Each of these local regions are assigned to individual experts and allows the experts to specialize in certain regions. In this architecture an expert could complete any specific task such as a classification or prediction task. In [22] a Mixture of Multi-Layer Perceptron Experts is implemented to forecast the Tehran stock exchange. The Mixture of Experts is also widely used in classification and has seen applications in domains such as gender, ethnic origin, and pose of human faces classification [23].

Empirical evidence has shown that HME has several advantages over the plain MoE architecture. HME often outperforms MoE which is attributed to the fact that a HME partitions the data both locally and globally, providing different resolution scales.

The MoE architecture consists of a manager (or gate) and a variable amount of experts. The manager network maintains a softmax output where the amount of output neurons equals the amount of experts. Each output unit represents how much the output of that corresponding expert should contribute to the final prediction. Using a softmax output allows us to divide the contribution of each expert and force the total contribution given each expert to partition unity. For an HME architecture with a depth of two, the i^{th} weighted output in the bottom layer is described by:

$$\mu_i = \sum_j g_{j|i} \mu_{ij}, \tag{8}$$

where μ_{ij} is the output of the j^{th} expert that contributes to μ_i and $g_{j|i}$ is the weight given to μ_{ij} . The output of the whole system μ is computed using the outputs in the bottom layer weighted by the top layer manager:

$$\mu = \sum_i g_i \mu_i. \quad (9)$$

The HME architecture consists of a variable amount of managers which each manage either multiple experts or the output of multiple linearly combined expert blocks. This definition allows the system to obtain an arbitrary recursion depth as seen in Equations (8) and (9). Figure 2 shows a general HME model of depth two. In this figure, x denotes the input which is identical for every manager and expert network. The individual gates and experts are trained end-to-end with backpropagation.

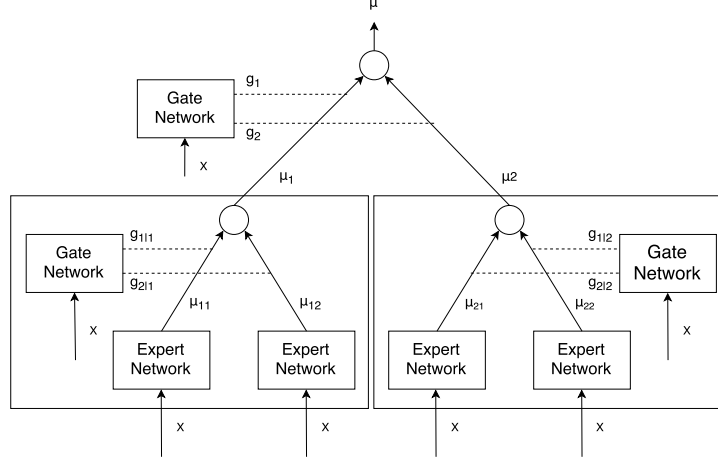


Fig. 2: Hierarchical Mixture of Experts model of depth two. Reprinted from [15].

Expert Networks. In the HME structure, multiple expert networks are present. These networks are full fledged regressors that are trained dependently of each other. In our setup the expert types that are considered are MLPs, Residual MLPs and LSTMs. For each of these types the hyperparameters associated with the network that yielded the lowest validation loss are used.

Gating Networks. The strength of the HME architecture lies in the managers that determine to what extent the individual agents contribute to the final prediction. In this research, MLPs, Residual MLPs and LSTM are tested as managers using the hyperparameters that yielded the lowest validation loss in the

single learner tasks. LSTMs are notorious to learn from experience and therefore we hypothesize that LSTM managers could provide better results than MLP or Residual MLP managers.

4 Results

4.1 1 Nearest Neighbor

1 Nearest Neighbor (1NN) is used as a baseline measure for the systems described in Section 2. The baseline matches an input with the training input it received earlier and returns the corresponding training output. Starting at this corresponding training output, it follows the trajectory already present in the training space. This follows from the way the training set is constructed, see Section 2.4. The more training inputs are fed to the system, the likelihood of finding a similar training instance to an arbitrary input increases. This means that this baseline can perform well if the system has seen a lot of training instances before. The error of the 1NN baseline reduces over the amount of training instances used.

4.2 Training

The weights of the LSTM memory cells were initialized with the method described in [24]. All other weights were initialized with the method described in [25]. For weight optimization, we used Adam [26]. Training samples were presented to the models with a batch size of four.

Early Stopping. During training, once an NN obtains a better validation accuracy, this NN is saved. To reduce training time, early stopping is implemented by monitoring the validation loss. Training is terminated when either the maximum number of 1000 epochs has been reached or the validation loss did not decrease over 50 epochs.

Hyperparameters. In order to obtain a well performing NN architecture, preliminary tests to determine hyperparameters for the MLP, Residual MLP and LSTM were performed. For each type of NN, the hyperparameters were selected. The selection was done based on the ability to, given a time step in a validation set s_t , make an accurate prediction s'_{t+1} of the next time step s_{t+1} . The performance measure used was the Mean Squared Error, hereafter referred to as loss.

Each of the NNs are tested with different sets of parameters. For all NNs, a parameter sweep was performed over the hidden layer sizes, the learning rate and the learning rate decay. For the MLP and Residual MLP the sweep also included the number of hidden layers and activation functions. Table 1 shows the validation loss for each type of NN and data set together with their best corresponding parameters settings.

Table 1: The best found hyperparameters, which were selected using the validation loss of 1-step prediction. For each data type, the best performing NN type is shown in bold. The loss function is the Mean Squared Error.

Data	NN type	h. layers	h. l. sizes	activation	lr	lr decay	loss
Intermittency	MLP	3	50	PReLU	0.01	0.001	1.60×10^{-4}
	R. MLP	1	200	sigmoid	0.001	0.0001	4.83×10^{-4}
	LSTM	-	50	-	0.01	0.001	4.19×10^{-3}
Logistic	MLP	3	200	PReLU	0.01	0.0001	1.53×10^{-8}
	R. MLP	3	200	sigmoid	0.01	0.0001	6.00×10^{-9}
	LSTM	-	50	-	0.01	0.0001	2.82×10^{-3}
Atmosphere	MLP	3	200	PReLU	0.001	0.0001	8.12×10^{-8}
	R. MLP	1	200	PReLU	0.001	0.0001	2.29×10^{-8}
	LSTM	-	200	-	0.001	0.0001	6.99×10^{-6}

Final NN Architectures. For each dynamical system, we trained 10 NNs of each type with the best found hyperparameters (all rows in Table 1). For each dynamical system, we also trained one type of HME. The NN types and hyperparameters of the best performing NNs (bold rows in Table 1) were used for the experts and managers in these HMEs. The newly constructed models were trained end-to-end. This is needed for the different experts to learn state dependent behavior. We used four experts and three managers as illustrated in Figure 2. This resulted into a total of 40 trained NNs for each of the three dynamical systems.

4.3 Testing

Sequence Prediction Evaluation Method. The 120 NNs and the three baselines, from now on collectively referred to as regressors, were evaluated on their ability to predict the behavior of a dynamical system, given the starting state in the test set s_0 . The sequence that describes the behavior of the dynamical system is generated by first making the regressor predict the next time step s'_1 given s_0 . Subsequent time steps at time $t+1$ are recursively predicted using the prediction made at time t .

For the logistic and intermittency maps, the length l of the predicted sequences is 100 data points. For the atmosphere data, $l = 600$, which corresponds to around five months of data.

To obtain more reliable results, the regressors are evaluated given multiple starting states. Given the test or validation set S the first $4,000 - l$ (the sets consist of 4,000 data points) elements are used exactly once as starting state when evaluating a regressor. The resulting $4,000 - l$ evaluations are averaged elementwise to obtain a single sequence of error measures for each regressor a . This sequence of error measures e_a^S is the mean loss of a , given S .

NN Selection. We define the performance of a regressor a on the validation set val as the mean of all elements of e_a^{val} . Using this performance measure, we selected the single best NN for each dynamical system. Table 2 shows these results.

Table 2: This table shows the lowest mean validation loss on sequence prediction as described in section 4.3 of the single best NN of each NN type for each data set. The best performing NNs are shown in bold.

Data	NN type	Lowest mean loss
Intermittency	HME	0.158152
	Residual MLP	0.327154
	MLP	0.160673
	LSTM	0.164677
Logistic	HME	0.219681
	Residual MLP	0.220254
	MLP	0.222794
	LSTM	14.86825
Atmosphere	HME	0.008987
	Residual MLP	0.010553
	MLP	0.010731
	LSTM	0.027812

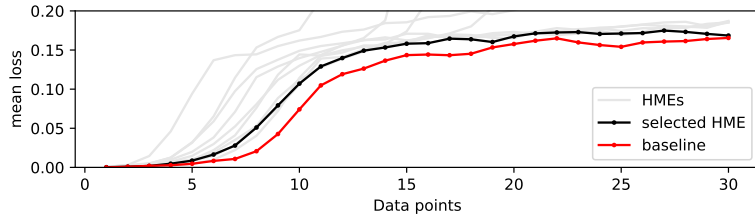
When taking for a each NN of the best NN type and each baseline for every dynamical system, we computed e_a^{test} . Here *test* denotes the test set corresponding to the dynamical system on which a was trained. The losses of the predicted sequences are shown in Figure 3.

5 Discussion

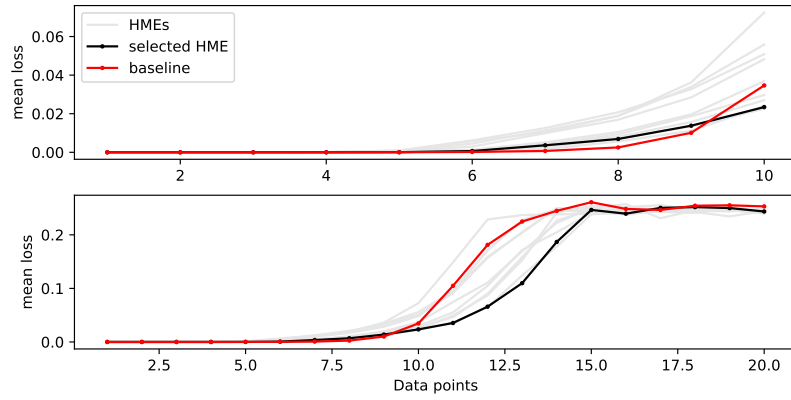
Intermittency Map. The results obtained with the intermittency data show that the sequence prediction performance of the NNs are worse than the baseline regressor. Figure 3(a) shows that the NNs are able to capture the general behavior of the dynamical system, but the predicted sequence deviates a lot from the ground-truth. The loss of the HME rapidly increases after the 7th data point.

Logistic Map. Figure 3(b) shows the sequence prediction ability of the NNs and the baseline on the logistic map. An interesting pattern arises in the domain $[6, 9]$. The baseline performance exceeds the performance of the selected Neural Network within this interval. However one should note that the selected HME performs better in the range $[10, 15]$. After data point 15 there is no evidence that either the baseline or the HME outperforms one another.

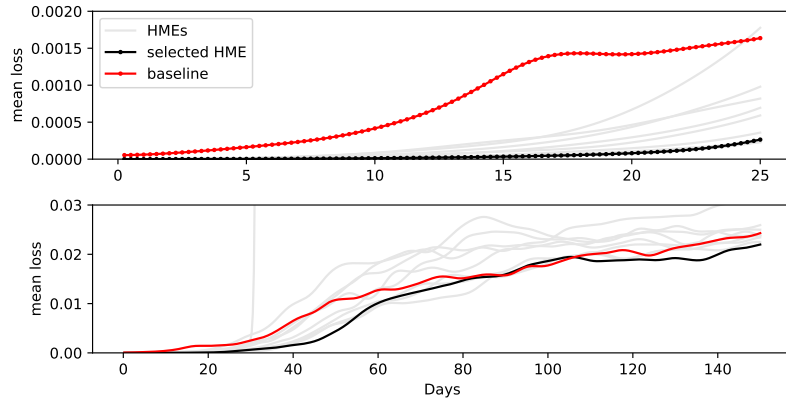
Atmosphere Data. The most interesting results are found in the 6-dimensional model. We found that out of all dynamical systems considered, relative to the



(a)



(b)



(c)

Fig. 3: Sequence prediction on intermittency map (a), logistic map (b) and atmosphere (c) test data.

baseline, the NNs performed best on the atmosphere data. On sequence prediction, an HME with Residual MLPs for managers and experts reliably outperforms the baseline of 1NN for about 24 days (96 time steps) as can be seen in Figure 3(c). Furthermore, the loss remains fairly small during the first 20 days. The HME that performed best on the validation data outperforms the baseline for several months. Figure 4 shows that HMEs are able to capture the general behavior of the atmospheric dynamical system.

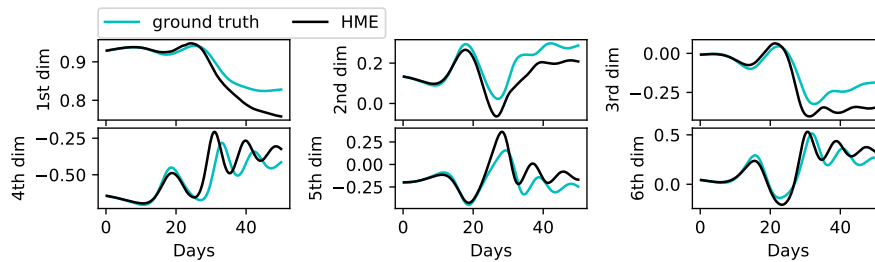


Fig. 4: Example of atmosphere data prediction by an HME over 50 days. The starting state has been chosen from the test set at random.

Relative to the baseline, the NNs perform better on the atmosphere data than on the other data sets. This could be because of the larger dimensionality of the atmosphere data. Because the same number of data points are sampled from each data set, the larger dimensionality causes the atmosphere data to be relatively sparse. This might allow the better interpolation capabilities of the NNs with respect to the baseline to become more apparent on the atmosphere data than on the other data sets.

As can be seen in Figure 3(c) where one of the grey lines becomes almost vertical around day 30, the mean loss of one of the HMEs suddenly becomes enormous. This is probably because this HME made a prediction slightly outside of the domain of the training set and predicted the next datapoint even further outside of the domain. This type of extrapolation causes the error to snowball.

6 Conclusion and Future Work

Conclusion. This work provides an overview of Neural Networks (NNs) in combination with a Hierarchical Mixture of Experts (HME) architecture applied to three different dynamical systems: the logistic map, the intermittency map and a 6-dimensional model which contains patterns that are found in the real atmosphere.

When testing Multi-Layer Perceptrons (MLPs), Residual MLPs and Long Short-Term Memory NNs (LSTMs) on these data sets it was observed that the best results were obtained with the Residual MLP for the logistic map and the

atmosphere data. The best results for the intermittency map were obtained with an MLP.

Compared to the 1-Nearest Neighbor baseline, the NNs used in this work are not suitable for dynamical systems such as the intermittency and logistic map. On the contrary, we found that the time series prediction test on the 6-dimensional data does give promising results. In the first couple of months of the predicted sequence, the baseline is clearly outperformed by the HME.

The results indicate that the HME architecture helps in reducing the generalisation error of dynamical system predictions, since for every data set tested, better results were obtained with using this architecture than without.

Future Work. Although we obtained promising results, there are several ways in which they could be improved. Future research can focus on the use of deeper HME architectures or more extensive hyperparameter studies. Other types of NNs or sequence prediction techniques might also prove useful for the problem at hand. A final suggestion is to investigate different ensemble techniques that might reduce the generalization error.

A drawback of predicting the behavior of a dynamical system step by step as described in Section 4.3 is that, when making a prediction of a time step, the system only uses one preceding step. For feedforward NNs, there is thus no countermeasure for the accumulation of error over time. Performance might thus be increased by using multiple previous predictions as input to the NNs.

Furthermore, although LSTMs did not perform well on these data sets, other types or combinations of recurrent connections might also help to counter this problem. Future research could also indicate whether, for the different data sets, varying the amount of data used influences the performance of NNs with respect to the baseline. It would also be interesting to see how well NNs can make predictions based on noisy training data and whether the models resulting from this research could be used as pre-trained models for training and testing on real world data.

References

1. Maqsood, I., Khan, M.R., Abraham, A.: An ensemble of neural networks for weather forecasting. *Neural Computing & Applications* **13**(2) (2004) 112–122
2. Taylor, J.W., Buizza, R.: Neural Network Load Forecasting With Weather Ensemble Predictions. *IEEE Transactions on Power Systems* **17**(3) (2002) 626–632
3. Gardner, M.W., Dorling, S.: Artificial neural networks (the multilayer perceptron) a review of applications in the atmospheric sciences. *Atmospheric environment* **32**(14) (1998) 2627–2636
4. Smola, A.J., Schölkopf, B.: A tutorial on support vector regression. *Statistics and computing* **14**(3) (2004) 199–222
5. Maier, H.R., Dandy, G.C.: Neural networks for the prediction and forecasting of water resources variables: a review of modelling issues and applications. *Environmental modelling & software* **15**(1) (2000) 101–124
6. Broer, H., Takens, F.: *Dynamical Systems and Chaos*. Volume 172 of Applied Mathematical Sciences. Springer (2011)

7. Lorenz, E.: Deterministic Nonperiodic Flow. *Journal of the Atmospheric Sciences* **20** (1963) 130–141
8. Pomeau, Y., Manneville, P.: Intermittent Transition to Turbulence in Dissipative Dynamical Systems. *Communications in Mathematical Physics* **74** (1980) 189–197
9. Reinhold, B.: Weather Regimes: The Challenge in Extended-Range Forecasting. *Science* **235** (1987) 437–441
10. Charney, J., DeVore, J.: Multiple Flow Equilibria in the Atmosphere and Blocking. *Journal of the Atmospheric Sciences* **36** (1979) 1205–1216
11. Crommelin, D., Opsteegh, J., Verhulst, F.: A Mechanism for Atmospheric Regime Behavior. *Journal of the Atmospheric Sciences* **61** (2004) 1406–1419
12. Sterk, A., Vitolo, R., Broer, H., Simó, C., Dijkstra, H.: New nonlinear mechanisms of midlatitude atmospheric low-frequency variability. *Physica D: Nonlinear Phenomena* **239** (2010) 702–718
13. Broer, H., Vitolo, R.: Dynamical systems modelling of low-frequency variability in low-order atmospheric models. *Discrete and Continuous Dynamical Systems B* **10** (2008) 401–419
14. Sterk, A., Holland, M., Rabassa, P., Broer, H., Vitolo, R.: Predictability of extreme values in geophysical models. *Nonlinear Processes in Geophysics* **19** (2012) 529–539
15. Jordan, M.I., Jacobs, R.A.: Hierarchical Mixtures of Experts and the EM Algorithm. *Neural computation* **6**(2) (1994) 181–214
16. He, K., Zhang, X., Ren, S., Sun, J.: Deep Residual Learning for Image Recognition. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. (2016) 770–778
17. Lipton, Z.C., Berkowitz, J., Elkan, C.: A Critical Review of Recurrent Neural Networks for Sequence Learning. *arXiv preprint arXiv:1506.00019* (2015)
18. Hochreiter, S.: The Vanishing Gradient Problem During Learning Recurrent Neural Nets and Problem Solutions. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems* **6**(02) (1998) 107–116
19. Hochreiter, S., Schmidhuber, J.: Long Short-Term Memory. *Neural computation* **9**(8) (1997) 1735–1780
20. Gers, F.A., Schmidhuber, J., Cummins, F.: Learning to Forget: Continual Prediction with LSTM. *IET Conference Proceedings* (1999) 850–855
21. Courbariaux, M., Bengio, Y., David, J.P.: BinaryConnect: Training Deep Neural Networks with binary weights during propagations. In: *Advances in Neural Information Processing Systems*. (2015) 3123–3131
22. Ebrahimpour, R., Nikoo, H., Masoudnia, S., Yousefi, M.R., Ghaemi, M.S.: Mixture of MLP-experts for trend forecasting of time series: A case study of the Tehran stock exchange. *International Journal of Forecasting* **27**(3) (2011) 804–816
23. Gutta, S., Huang, J.R., Jonathon, P., Wechsler, H.: Mixture of Experts for Classification of Gender, Ethnic Origin, and Pose of Human Faces. *IEEE Transactions on neural networks* **11**(4) (2000) 948–960
24. Glorot, X., Bengio, Y.: Understanding the difficulty of training deep feedforward neural networks. In: *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*. (2010) 249–256
25. He, K., Zhang, X., Ren, S., Sun, J.: Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification. In: *Proceedings of the IEEE international conference on computer vision*. (2015) 1026–1034
26. Kingma, D., Ba, J.: Adam: A Method for Stochastic Optimization. *arXiv preprint arXiv:1412.6980* (2014)

Part III

Type B: Compressed contributions Oral presentation

Reactive Versus Anticipative Decision Making in a Novel Gift-Giving Game

Elias Fernández Domingos¹²³, Juan Carlos Burguillo³, and Tom Lenaerts¹²

¹ AI lab, Computer Science Department, Vrije Universiteit Brussel, Pleinlaan 2, 1050 Brussels, Belgium,

² MLG, Département d'Informatique, Université Libre de Bruxelles, Boulevard du Triomphe CP212, 1050 Brussels, Belgium

³ Department of Telematic Engineering. University of Vigo. 36310-Vigo, Spain
`eliferna@vub.be`, `tlenaert@ulb.ac.be`,

Gift-giving games are often used to model situations related to the presence of trust and fairness in human behavior. The Dictator and Ultimatum games [1] are well known examples that have been thoroughly studied, both analytically and through behavioral experiments, where results contrast with the traditional assumptions of rationality in Game Theory [3]. In a recent paper published in the proceedings of the AAAI-17 conference [2] we highlight the importance of anticipatory reasoning over human decision-making in the context of gift-giving games. Particularly, we study how agents embedded with anticipatory capabilities perform on a variant of the Dictator Game, known as the Anticipation Game (AG), and show how they are able to reproduce the capacity of humans to adapt to context observed in behavioral experiments [5].

The Anticipation Game (AG) takes place between pairs of individuals, i.e., dictators and receivers. The latter must decide whether to accept or reject the interaction with the dictator, based on some information about her actions on past games, which conforms her reputation. If she accepts, the game proceeds as a Dictator Game (DG), where the dictator receives an endowment and is requested to give an amount between zero and the entire endowment to the receiver. However, if she rejects, both players receive a payoff of zero. Therefore, the dictator must anticipate the effects that her actions will have over her reputation. Yet, Game Theory predicts, under assumptions of selfish and rational behavior, that dictators' share should be minimal and, receivers should always accept.

Nonetheless, the experiment performed in [5] has shown that participants deviate from this rational behavior. Concretely, the experiment was divided in three treatments that evaluate how dictators and receivers behave when receivers were given full information about the dictator's past three actions; when this information was only available 50% of the time; and when it was never available. The results show that receivers tend to accept more when they have no information available and dictators reduce considerably their donations – which is the sub-game perfect Nash equilibrium, while in the full information case, the donations get close to half of the endowment and receivers punish unfair dictators with rejection – deviating from rational behavior. The case with partial information stands in the middle of the previous cases. This indicates not only

that humans are neither hyper-rational nor fully selfish, but also that we are able to identify changes in context and alter their strategy based on that. Thus, it reveals the need for an analytical model that loosens this constraint and provides insights into the cognitive mechanisms required to produce the choices observed in experiments.

With this in mind, we investigate how anticipatory reasoning affects human decision-making in the Anticipation Game and whether it's a vector strong enough to drive the emergence of generosity. Hence, we embed an adaptive agent, that should play as dictator, with anticipatory capabilities by taking the definition provided in [4]: *An anticipating system will be defined here as a system containing a predictive model of itself and/or its environment, which allows it to change state at an instant in accord with the models predictions pertaining to a later instant.* In our setup, the predictive model is implemented with a Recurrent Neural Network (RNN) and its predictions are used to evaluate the effects of the available actions into the future and update the strategy of the agent. Afterwards, to determine if anticipation was a relevant cognitive ability in our scenario, we compared this anticipative (forward-looking) model with a reactive (backwards-looking) one. During the simulations, dictator agents that implemented those models, played against a receiver with a fixed strategy for each different context – emulating the treatments of the experiments. Our results demonstrate that only dictators using the anticipative model, were able to account for changes in the context of games and play optimally in each of them [2], which indicates that models that take into account anticipation are more suited to represent human behavior in this sort of scenarios.

References

1. Cooper, D., Kagel, J.H.: Other regarding preferences: a selective survey of experimental results. In: Cooper, D., Kagel, J.H. (eds.) *Handbook of experimental economics*, vol. 2. Princeton: Princeton University Press (2012)
2. Domingos, E.F., Burguillo, J., Lenaerts, T.: Reactive versus anticipative decision making in a novel gift-giving game. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. vol. 5, pp. 4399–4405. San Francisco, USA (2017)
3. Engel, C.: Dictator games: A meta study. *Experimental Economics* 14(4), 583–610 (2011)
4. Rosen, R.: *Anticipatory systems : philosophical, mathematical, and methodological foundations*, vol. 1. Pergamon Press, Oxford (1985)
5. Zisis, I., Di Guida, S., Han, T., Kirchsteiger, G., Lenaerts, T.: Generosity motivated by acceptance-evolutionary analysis of an anticipation game. *Scientific Reports* 5, 18076 (2015)

Evaluating Intelligent Knowledge Systems (Article Abstract)

Neil Yorke-Smith^{1,2}

¹ Delft University of Technology, The Netherlands

² American University of Beirut, Lebanon

`n.yorke-smith@tudelft.nl`

Abstract. The article published in Knowledge and Information Systems examines the evaluation of a user-adaptive personal assistant agent designed to assist a busy knowledge worker in time management. The article examines the managerial and technical challenges of designing adequate evaluation and the tension of collecting adequate data without a fully functional, deployed system. The PTIME agent was part of the CALO project, a seminal multi-institution effort to develop a personalized cognitive assistant. The project included a significant attempt to rigorously quantify learning capability, which the article discusses for the first time, and ultimately the project led to multiple spin-outs including Siri. Retrospection on negative and positive experiences over the six years of the project underscores best practice in evaluating user-adaptive systems. Through the lessons illustrated from the case study of intelligent knowledge system evaluation, the article highlights how development and infusion of innovative technology must be supported by adequate evaluation of its efficacy.

1 Evaluation of the Personalized Time Management (PTIME) Agent

The case study article by Berry et al [1] reports and critiques the *evaluation* of an intelligent knowledge system that learns preferences over an extended period. The domain of application is personal time management, in particular, providing assistance with arranging meetings and managing an individual's calendar. The *Personalized Time Management* (PTIME) calendaring assistant agent increased in usefulness as its knowledge about the user increases. The enabling technologies involved were preference modelling and machine learning to capture user preferences, natural language understanding to facilitate elicitation of constraints, and constraint-based reasoning to generate candidate schedules [2]. Human-computer interaction (HCI) and interface design played central roles.

The PTIME system was part of a larger, seminal project, *Cognitive Assistant that Learns and Organizes* (CALO), aimed at exploring learning in a personalized cognitive assistant. Thus, the primary assessment of PTIME was in terms of its adaptive capabilities, although such a knowledge-based system must necessarily have a certain level of functionality to assist with tasks in time management,

in order to provide a context for learning. At the commencement of the project, however, the degree of robustness and usability required to support evaluation was not immediately obvious. Evaluation was focused almost exclusively on the technology; experiments were designed to measure performance improvements due to learning within a controlled test environment intended to simulate a period of real-life use—rather than in a genuinely ‘in-the-wild’ environment. Technologists such as the majority of the authors are trained primarily to conduct such ‘in-the-lab’ evaluations, but—as argued in the article—many situations require placing the technology into actual use with real users in a business or personal environment, in order to provide a meaningful assessment. In retrospect, the authors suggest that the evaluation methodology of CALO gave too little attention to the usefulness and usability of the technology.

2 Lessons Learned

The six lessons that emerged from the evaluation journey with PTIME are not unfamiliar from other experiences of evaluating (non-adaptive) systems [3]:

1. The contexts of the use of technology, and the competing interests of the stakeholders, must be a primary focus in designing an evaluation strategy.
2. Evaluating one component based on an evaluation of a whole system can be misleading, and vice versa.
3. User-adaptive systems require distinct evaluation strategies.
4. In-the-wild evaluation is necessary when factors affecting user behaviour cannot be replicated in a controlled environment.
5. In-the-wild evaluation implies significant additional development costs.
6. Ease of adoption of the system by users will determine the success or failure of a deployed evaluation strategy.

Summarizing the article, the main lesson from this case study of intelligent knowledge system evaluation is obvious but under-valued: researchers and project managers benefit from familiarity with and adoption of best practice in evaluation methodologies from the start of a technology project.

Acknowledgements This material is based in part upon work supported by the US Defense Advanced Research Projects Agency (DARPA) Contract No. FA8750-07-D-0185/0004. Views are the author(s) and do not necessarily reflect the views of DARPA.

References

1. Berry, P.M., Donneau-Golencer, T., Duong, K., Gervasio, M., Peintner, B., Yorke-Smith, N.: Evaluating intelligent knowledge systems: Experiences with a user-adaptive assistant agent. *Knowledge and Information Systems* 52, 379–409 (2017)
2. Berry, P.M., Gervasio, M., Peintner, B., Yorke-Smith, N.: PTIME: Personalized assistance for calendaring. *ACM Trans. on Intelligent Systems and Technologies* 2(4), 40:1–40:22 (2011)
3. Cohen, P., Howe, A.E.: Toward AI research methodology: three case studies in evaluation. *IEEE Trans. on Systems, Man, and Cybernetics* 19(3), 634–646 (1989)

The Parameterized Complexity of Approximate Inference in Bayesian Networks ^{*}

Johan Kwisthout

Radboud University, Donders Institute for Brain, Cognition and Behaviour,
Nijmegen, The Netherlands j.kwisthout@donders.ru.nl

Abstract. Computing posterior and marginal probabilities constitutes the backbone of almost all inferences in Bayesian networks. These computations are known to be intractable in general [2]; moreover, it is known that *approximating* these computations is also NP-hard [3]. In the original paper we use *fixed-error randomized tractability analysis* [4], a recent randomized analogue of parameterized complexity analysis [5], to systematically address the complexity of (randomized) approximate inference in Bayesian networks. In this extended abstract we will give a brief introduction of the key concepts and results in this paper.

1 Approximate inference in Bayesian networks

As computing posterior probabilities in Bayesian networks is an NP-hard problem [2], one often resorts to approximate inferences. For example, rather than computing the exact value of a posterior probability distribution $\mathbf{P} = \Pr(H \mid E)$, one might settle for a distribution \mathbf{Q} that is easier to compute and that is close to the target distribution \mathbf{P} . One such approximation approach is to *sample* from the distribution, yielding a randomized algorithm that (given sufficient samples) approximates the distribution. In the full paper [1] we investigated the properties of such randomized approximation problems. Given that Bayesian inference is NP-hard, an efficient *general* randomized algorithm can be ruled out, unless $\text{BPP} = \text{NP}$; we are thus focusing on *parameterizing* the problem [5] such that the expected running time of the algorithm is exponential only in the parameter, yet polynomial in the input. Formally, (parameterized) complexity theory is built on decision problems, hence, we are interested in parameters k for which the following¹ decision problem becomes feasible:

Additive-approximated Conditional Probability (AA-CPROB)

Input: A Bayesian network \mathcal{B} with designated non-overlapping subsets of variables \mathbf{H} and \mathbf{E} and corresponding joint value assignments \mathbf{h} to \mathbf{H} and \mathbf{e} to \mathbf{E} ; in addition, error bound ϵ and rational number r .

Question: Is $\Pr(\mathbf{h} \mid \mathbf{e}) \pm \epsilon > r$?

^{*} This is an extended abstract of [1].

¹ Among other variants such as relative approximations and marginal inferences.

2 Fixed-error randomized tractability

The complexity classes PP and BPP are defined as classes of decision problems that are decidable by a randomized algorithm in polynomial time with a particular probability of error; the difference between these two classes is in the bound on the error probability. In particular, a decision problem $\Pi \in \text{PP}$ if and only if there exists a randomized algorithm accepting *Yes*-instances and rejecting *No*-instances of Π with probability strictly larger than $1/2$. In contrast, for BPP these probabilities are polynomially *bounded away* from $1/2$, allowing for effectively ‘boosting up’ the probability of acceptance while still taking only polynomial time. In order to *parameterize* the probability of acceptance—and thus making the time needed to boost the probability of acceptance close to 1 relative to a parameter—we introduce the fixed-error randomized tractability class FERT; informally, a problem in FERT is tractable if the parameter is small and intractable otherwise. This is formalized as follows:

Definition 1 (FERT). *Let Π be a decision problem and let $k\text{-}\Pi$ be a parameterization of Π . We have that $k\text{-}\Pi \in \text{FERT}$ if and only if there exists a randomized algorithm that accepts *Yes*-instances x of Π with probability at least $1/2 + \min(f(k), 1/|x|^c)$ for a constant c and arbitrary function $f : \mathbb{R} \rightarrow \langle 0, 1/2 \rangle$; *No*-instances are accepted with probability at most $1/2 - \min(f(k), 1/|x|^c)$.*

3 Highlighted results

Due to space constraints we just briefly list some interesting results as a teaser to the full paper, where d denotes the in-degree of the network, and D_e formalizes the maximal range of the parameters in the CPTs relative to the evidence:

$$\begin{aligned} \{\text{Pr}(\mathbf{h} \mid \mathbf{e}), \epsilon\}\text{-AA-CPROB} &\in \text{FERT} \text{ but } \{d, \epsilon\}\text{-AA-CPROB} \notin \text{FERT} \text{ and also} \\ \{\text{Pr}(\mathbf{h} \mid \mathbf{e}), d\}\text{-AA-CPROB} &\notin \text{FERT} \\ \{D_e\}\text{-AA-CPROB} &\notin \text{FERT} \quad \text{but } \{D_e, \epsilon\}\text{-AA-CPROB} \in \text{FERT} \end{aligned}$$

In general, this approach allows us to explicate what *does* and what *doesn't* make approximate inference feasible in Bayesian networks. The full paper has a complete overview of known and new results cast into this complexity framework.

References

1. Kwisthout, J.: Approximate inference in Bayesian networks: Parameterized complexity results. *International Journal of Approximate Reasoning* (in press)
2. Cooper, G.F.: The computational complexity of probabilistic inference using Bayesian belief networks. *Artificial Intelligence* **42**(2) (1990) 393–405
3. Dagum, P., Luby, M.: Approximating probabilistic inference in Bayesian belief networks is NP-hard. *Artificial Intelligence* **60**(1) (1993) 141–153
4. Kwisthout, J.: Tree-width and the computational complexity of MAP approximations in Bayesian networks. *Journal of AI Research* **53** (2015) 699–720
5. Downey, R.G., Fellows, M.R.: *Parameterized Complexity*. Springer, Berlin (1999)

On the Problem of Making Autonomous Vehicles Conform to Traffic Law

Henry Prakken^{1,2}

¹ Department of Information and Computing Sciences, Utrecht University, Utrecht, The Netherlands,

h.prakken@uu.nl

² Faculty of Law, University of Groningen, Groningen, The Netherlands

Abstract. Autonomous vehicles are one of the most spectacular recent developments of Artificial Intelligence. Among the problems that still need to be solved before they can fully autonomously participate in traffic is the one of making their behaviour conform to the traffic laws. This paper discusses this problem by way of a case study of Dutch traffic law. First it is discussed to what extent Dutch traffic law exhibits features that are traditionally said to pose challenges for AI & Law models, such as exceptions, rule conflicts, open texture and vagueness, rule change, and the need for commonsense knowledge. Then three approaches to the design of law-conforming AV are evaluated in light of the challenges posed by Dutch traffic law, which includes an assessment of the usefulness of AI & Law models of nonmonotonic reasoning, argumentation and case-based reasoning.

Autonomous vehicles are one of the most spectacular recent developments of Artificial Intelligence. Among the problems that need to be solved is the one of making the behaviour of autonomous vehicles (AV) conform to the traffic laws. Solutions to this problem may well profit from computational models of legal reasoning but so far the field of AI & Law has hardly addressed this issue. The present paper³ puts this topic on the AI & Law research agenda by way of a case study of Dutch traffic law and its implications for the design of fully autonomous self-driving cars. In particular, the challenges are discussed that Dutch traffic law poses for AV and how existing AI & Law techniques are relevant for dealing with these challenges. In the literature on AV design there have to the best of my knowledge so far not been any systematic studies of the problem of making AV conform to traffic law through its design. Therefore, the present study, while still a conceptual one, fills an important gap in the literature.

The problem of making AV conform to traffic law is a special case of the more general problem of making intelligent autonomous systems conform to the relevant laws. The present study of the problem for autonomous vehicles may therefore also have relevance for the study of the more general problem. Computer systems are increasingly being employed in practice with some degree of autonomy. Their behaviour is not fully specified by the programmer but is the result of the implementation of more general cognitive or physical abilities. Such artificially intelligent systems can do things which, when done by humans, are regulated by law. Apart from self-driving cars, some examples are care robots that help sick or elderly people and whose actions can damage

³ The full paper has been published in *Artificial Intelligence and Law* 25 (2017): 341-363.

property or the health of the person (spilling coffee over an iPad, failing to administer medication on time), intelligent fridges that can order food or drinks when the supplies run out and thus have to conform to contract law, and financial trading programs that have to conform to financial regulations.

When such autonomous systems are being used, legal rules cannot any more be regarded as regulating human behaviour, since it is not the humans but the machines who act. This raises the problem of how the autonomous systems can be designed in such a way that their behaviour complies with the law. Note that this question needs to be asked irrespectively of the legal question whether machines can be assigned responsibility in a legal sense. Even if a human remains legally responsible or liable for the actions of the machine, the human faces the problem of ensuring that the machine behaves in such a way that the responsible human complies with the law. Currently, this kind of problem is mainly studied under the heading of ‘machine ethics’. While this may be the appropriate field for studying the related problem of making intelligent autonomous systems behave ethically responsibly, the problem of making them conform to the law arguably belongs to AI & Law.

For AV designers the paper will hopefully create a deeper awareness of the challenges that traffic law poses for AV design. To start with, an account is given of the similarities and differences between the tasks traditionally modelled in AI & Law and the new task of autonomous driving on public roads. Then three different approaches to the design of law-conforming AV, are discussed, namely, the *regimentation* approach (designing the system in a way that guarantees that the system will not exhibit unwanted behaviour), the *reasoning* approach (giving the system the ability to reason about the lawfulness of its own behaviour) and the *training* approach (letting the system acquire the ability to behave legally correctly implicitly by training). The latter is the one currently usually applied in AV design.

Next, the need is identified for formal specification of traffic law as a component of AV design in all three approaches, even in the training approach. As regards the applicability of knowledge representation and reasoning techniques previously developed in AI & Law, the conclusion is that logic-based techniques for rule-based (monotonic or nonmonotonic) reasoning are largely suitable for representing the logical and hierarchical structure of the regulations but cannot deal with the interpretation problems arising from open texture, vagueness and the need for new exceptions. On the other hand, existing argumentation-based techniques for dealing with open texture, vagueness and the need for new exceptions are largely inapplicable, partly since the adversarial setting presumed by these techniques is lacking in the AV problem and partly because the existing case law is too sparse and inconclusive. A more promising approach is to develop standards and guidelines for implementing law-conforming behaviours in a collaborative effort between the government, industry and possibly insurance companies. However, this does not avoid the need for formal representation of the relevant traffic regulations. Finally, the present paper provides reasons for the relevant research communities to focus less on moral algorithms for decision making in emergency situations and more on careful and anticipatory driving behaviour (while not sacrificing traffic efficiency).

The Transitivity and Asymmetry of Actual Causation^{*}

Sander Beckers¹ and Joost Vennekens²

¹ Cornell University

² KU Leuven, Dept. Computer Science @ Campus De Nayer

The problem of actual causation has received a lot of attention in both the philosophy and the AI literature. In a nutshell, the problem is to define when event X is deemed to have caused event Y in the context of a particular story. Typically, it is assumed that this story unfolds according to a given set of causal laws. Coming up with a suitable definition for this concept of actual causation has proven to be quite difficult.

In this paper, we attempt to both explain why this is so difficult, and to offer a method for potentially solving the problem. We do so by focusing on two natural, yet mutually incompatible intuitions, namely that actual causation should be both transitive and asymmetric.

Following the approach of the seminal work by Halpern and Pearl [2], we make use of structural models as our formal tool.

Example 1. An assassin-in-training is on his first mission. Trainee is an excellent shot: if he shoots his gun, the bullet will fell Victim. Supervisor is also present, in case Trainee has a last minute loss of nerve (a common affliction among student assassins) and fails to pull the trigger. If Trainee does not shoot, Supervisor will shoot Victim herself. In fact, Trainee performs admirably, firing his gun and killing Victim.

We can represent this example by means of a structural model that consists of five boolean random variables and two equations that relate these variables:

$$\begin{aligned} VictimDies &:= TraineeHits \vee SupervisorHits \\ SupervisorHits &:= SupervisorShoots \\ TraineeHits &:= TraineeShoots \\ SupervisorShoots &:= \neg TraineeShoots \end{aligned}$$

The story told in the example corresponds to the following assignment of values to these variables:

$$\begin{aligned} TraineeShoots &= TraineeHits = VictimDies = true; \\ SupervisorShoots &= SupervisorHits = false. \end{aligned}$$

In his seminal work, Lewis [3] proposed to define actual causation as the transitive closure of counterfactual dependency. In the case of the above example,

^{*} The full version of this paper was published as [1].

VictimDies is counterfactually dependent on *TraineeHits* (in the context of the story, in which *SupervisorHits* is false, it is the case that if trainee's bullet hadn't hit the victim, the victim would not have died) and *TraineeHits* is counterfactually dependent on *TraineeShoots* (if he hadn't shot, then his bullet wouldn't have hit the victim). Therefore, by transitivity, victim's death was caused by trainee's shooting, even though it does not counterfactually depend on it (if trainee would not have shot, then victim would still have died).

The transitivity of actual causation invoked by Lewis seems intuitively plausible. Indeed, if X caused Y and Y in turn caused Z , then surely it is fair to call also X a cause of Z . However, several convincing counterexamples have emerged, such as the following example by [4].

Example 2. Terrorist, who is right-handed, must push a detonator button at noon to set off a bomb. Shortly before noon, he is bitten by a dog on his right hand. Unable to use his right hand, he pushes the detonator with his left hand at noon. The bomb duly explodes.

$$Bomb := LeftHand \vee RightHand$$

$$LeftHand := DogBite$$

$$RightHand := \neg DogBite$$

Clearly, the dog bite caused the terrorist to use his left hand, and the terrorist's use of his left hand caused the explosion. Nevertheless, it seems too farfetched to call the dog bite a cause for the explosion, as transitivity would mandate. Therefore, this example demonstrates that the intuition of transitivity has its limitations.

In this paper, we claim that the limiting factor is the intuition of asymmetry, i.e., that we should never consider X to have caused Y if, at the same time, we would also have considered $\neg X$ (if $\neg X$ had been the case instead of X) as a cause of the same Y . In the case of the above example, we see that, in the context where the dog would not have bitten the terrorist, there would be counterfactual dependence of *Bomb* on *RightHand* and of *RightHand* on $\neg DogBite$. Therefore, Lewis' account would count both *DogBite* and $\neg DogBite$ as a cause for the same event *Bomb*, which would violate asymmetry.

In the full version of this paper, we argue for the hypothesis that asymmetry is the limiting factor on the transitivity of actual causation, i.e., that actual causation is precisely as transitive as it can be without violating asymmetry.

References

1. Sander Beckers and Joost Vennekens. The transitivity and asymmetry of actual causation. *Ergo, an Open Access Journal of Philosophy*, 2017.
2. J. Halpern and J. Pearl. Causes and explanations: A structural-model approach. part i: Causes. *The British Journal for the Philosophy of Science*, 56:843–87, 2005.
3. D. Lewis. Causation. *Journal of Philosophy*, 70:113–126, 1973.
4. M. McDermott. Redundant causation. *The British Journal for the Philosophy of Science*, 46(4):523–544, 1995.

Multi-View LS-SVM for Temperature Prediction

Lynn Houthuys, Zahra Karevan, and Johan A. K. Suykens

ESAT-STADIUS, KU Leuven. Kasteelpark Arenberg 10 B-3001 Leuven, Belgium

Abstract. In multi-view regression, the input data can be represented in multiple ways or views. The aim is to increase the performance of using only one view by taking into account the information available from all views. We introduce a novel multi-view regression model called Multi-View Least Squares Support Vector Machines (MV LS-SVM). This work was motivated by the challenge of predicting temperature in weather forecasting. Black-box weather forecasting deals with a large number of observations and features and is one of the most challenging learning tasks around. In order to predict the temperature in a city, the historical data from that city as well as from the neighboring cities are taken into account. We use MV LS-SVM to do temperature prediction by regarding each city as a different view. Experimental results on the min. and max. temperature prediction in Brussels, show the improvement of the multi-view method with regard to previous work and that it is competitive to the existing state-of-the-art methods in weather prediction.

1 Introduction

The multi-view [1] method proposed in this paper is called *Multi-View Least Squares Support Vector Machines (MV LS-SVM) Regression*. It is cast in the primal-dual setting typical to Least Squares Support Vector Machines [2] where the separate models for each view are combined in the primal objective function so that information from other views is taken into account during training.

We focus on MV LS-SVM regression for black-box weather forecasting. Accurate weather prediction is a challenging problem that can influence our daily lives in different ways. Lazo et al. reported that the U.S. public obtains more than 300 billion forecasts with a total estimated value of \$31.5 billion each year [3]. In order to forecast a weather condition in a particular city, the historical data of some nearby cities have been taken into account. Instead of simply concatenating the feature vectors of all cities, this paper regards each city as a separate view. By using the novel MV LS-SVM regression method, the temperature influences of each city can be modeled separately, with different parameters for each city, while the coupling term enforces interaction between the views which allows information from all other cities to be taken into account during the training phase. The following is a brief discussion of MV LS-SVM and some results on temperature prediction. For a more in-depth discussion see [4].

2 Multi-View LS-SVM Regression (MV LS-SVM)

The primal formulation of MV LS-SVM consist of multiple LS-SVM regression objectives and a coupling term. This newly introduced term enforces the alignment of the error variables over multiple views so that when training on one

view, the other views are taken into account. Given a number of V views, a training set of N data points $\{y_k, \mathbf{x}_k^{[v]}\}_{k=1}^N$ where $\mathbf{x}_k^{[v]} \in \mathbb{R}^{d^{[v]}}$ denotes the k -th input sample for view v and $y_k \in \mathbb{R}$ the k -th target value, the primal formulation of the proposed model is:

$$\begin{aligned} \min_{\mathbf{w}^{[v]}, \mathbf{e}^{[v]}, b^{[v]}} \quad & \frac{1}{2} \sum_{v=1}^V \mathbf{w}^{[v]T} \mathbf{w}^{[v]} + \frac{1}{2} \sum_{v=1}^V \gamma^{[v]} \mathbf{e}^{[v]T} \mathbf{e}^{[v]} + \rho \sum_{v,u=1; v \neq u}^V \mathbf{e}^{[v]T} \mathbf{e}^{[u]} \\ \text{s.t. } \quad & \mathbf{y} = \Phi^{[v]} \mathbf{w}^{[v]} + b^{[v]} \mathbf{1}_N + \mathbf{e}^{[v]} \quad \text{for } v = 1, \dots, V \end{aligned} \quad (1)$$

where $\mathbf{y} = [y_1; \dots; y_N]$ and $b^{[v]}$ are bias terms, $\gamma^{[v]}$ are positive real constants and $\mathbf{e}^{[v]} \in \mathbb{R}^N$ are error variables. $\Phi^{[v]}$ is defined as $\Phi^{[v]} = [\varphi^{[v]}(\mathbf{x}_1^{[v]})^T; \dots; \varphi^{[v]}(\mathbf{x}_N^{[v]})^T]$ where $\varphi^{[v]} : \mathbb{R}^{d^{[v]}} \rightarrow \mathbb{R}^{d_h^{[v]}}$ are the mappings to a high dimensional feature space related to the v th view. By taking the Lagrangian of the primal problem, deriving the KKT optimality conditions and eliminating the primal variables, the dual problem, a linear system, is obtained.

3 Experiments and conclusion

The data have been collected from Weather Underground [5] and include real measurements for Brussels and 9 neighboring cities. It covers a time period from beginning 2007 to mid 2014 and comprises 198 measured weather variables for each day. The performance of the proposed method is evaluated on min. and max. temperature prediction in two test sets: (i) from mid-Nov. 2013 to mid-Dec. 2013 (Nov/Dec) and (ii) from mid-Apr. 2014 to mid-May 2014 (Apr/May).

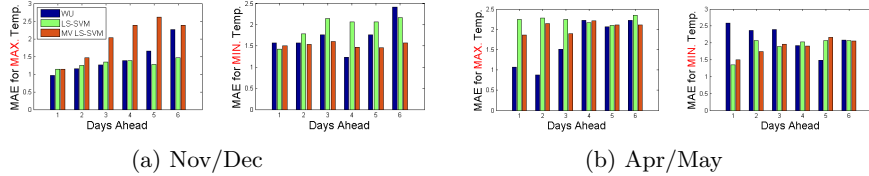


Fig. 1: Comparison of the MAE on temperature prediction between Weather Underground (WU), LS-SVM on the concatenated features and MV LS-SVM.

Figure 1 summarizes the results. We can conclude that the proposed method MV LS-SVM can often outperform LS-SVM for min. and max. temperature prediction. It also shows that MV LS-SVM can outperform WU for min. temperature prediction and is competitive with it for max. temperature in the test set Apr/May. These results suggest the merit of using a multi-view approach instead of simply concatenating the features.

Acknowledgments

The research has received funding from the European Research Council under the European Union Seventh Framework Programme (FP7/2007-2013) / ERC AdG A-DATADRIVE- B (290923). This paper reflects only the authors views and the Union is not liable for any use that may be made of the contained information. Research Council KUL: CoE PFV/10/002 (OPTEC), BIL12/11T; PhD/Postdoc grants Flemish Government: FWO; projects: G.0377.12 (Structured systems), G.088114N (Tensor based data similarity), G0A4917N (Deep restricted kernel machines); PhD/Postdoc grant iMinds Medical Information Technologies SBO 2015 IWT: POM II SBO 100031 Belgian Federal Science Policy Office: IUAP P7/19 (DYSCO, Dynamical systems, control and optimization, 2012-2017).

References

1. Blum, A., Mitchell, T.: Combining labeled and unlabeled data with co-training. *Conference on Learning Theory* (1998) 92–100
2. Suykens, J.A.K., Van Gestel, T., De Brabanter, J., De Moor, B., Vandewalle, J.: *Least Squares Support Vector Machines*. World Scientific (2002)
3. Lazo, J.K., Morss, R.E., Demuth, J.L.: 300 billion served: Sources, perceptions, uses, and values of weather forecasts. *Bulletin of the American Meteorological Society* **90**(6) (2009) 785–798
4. Houthuys, L., Karevan, Z., Suykens, J.A.K.: Multi-view LS-SVM regression for black-box temperature prediction in weather forecasting. *Proceedings of the International Joint Conference on Neural Networks (IJCNN)* (2017) 1102–1108
5. Weather Underground Co.: www.wunderground.com

Regularized Semi-Paired Kernel CCA for Domain Adaptation¹

Siamak Mehrkanoon

Johan A.K. Suykens

KU Leuven, ESAT-STADIUS, Kasteelpark Arenberg 10, B-3001 Leuven, Belgium

Abstract

A Regularized Semi-Paired Kernel Canonical Correlation Analysis (RSP-KCCA) formulation is introduced for learning a latent space for the domain adaptation problem. The optimization problem is formulated in the primal-dual LS-SVM setting where side information can be readily incorporated through regularization terms. A joint representation of the data set across different domains is learned by solving a generalized eigenvalue problem or linear system of equations in the dual. The proposed model is naturally equipped with out-of-sample extension property which plays an important role for model selection. Experimental results are given to illustrate the effectiveness of the proposed approaches on synthetic and real-life datasets.

1 Introduction

Manual labeling of sufficient training data for diverse application domains is a costly, laborious task and often prohibitive. Therefore, designing models that can leverage rich labeled data in one domain and be applicable to a different but related domain is highly desirable. In particular, domain adaptation or transfer learning algorithms seek to generalize a model trained in a source domain (training data) to a new target domain (test data). Depending on the availability of the labeled instances in both domains, three scenarios can be considered, i.e. unsupervised, supervised and semi-supervised domain adaptation models [2]. Unsupervised domain adaptation approaches, do not take label information into consideration when learning the feature representation. On the other hand, supervised domain adaptation approaches, only use labeled data from the source and target domains. In the semi-supervised setting, one learns from labeled source instances as well as a small fraction of the target labeled instances. This setting can have many real world applications, as collecting labeled instances might be costly.

2 Formulation of the method

Consider two training datasets $\mathcal{D}^{(1)}$ and $\mathcal{D}^{(2)}$ as source and target domains with

$$\mathcal{D}^{(1)} = \{ \underbrace{x_1^{(1)}, \dots, x_{n_p}^{(1)}}_{\substack{\text{paired} \\ (\text{labeled/unlabeled}) \\ (\mathcal{D}_{p,ul}^{(1)} \cup \mathcal{D}_{p,l}^{(1)})}}, \underbrace{x_{n_p+1}^{(1)}, \dots, x_{n_l}^{(1)}}_{\substack{\text{unpaired} \\ (\text{unlabeled}) \\ (\mathcal{D}_{up,ul}^{(1)})}}, \underbrace{x_{n_l+1}^{(1)}, \dots, x_{N_1}^{(1)}}_{\substack{\text{unpaired} \\ (\text{labeled}) \\ (\mathcal{D}_{up,l}^{(1)})}} \}, \quad \mathcal{D}^{(2)} = \{ \underbrace{x_1^{(2)}, \dots, x_{n_p}^{(2)}}_{\substack{\text{paired} \\ (\text{labeled/unlabeled}) \\ (\mathcal{D}_{p,ul}^{(2)} \cup \mathcal{D}_{p,l}^{(2)})}}, \underbrace{x_{n_p+1}^{(2)}, \dots, x_{n_2}^{(2)}}_{\substack{\text{unpaired} \\ (\text{unlabeled}) \\ (\mathcal{D}_{up,ul}^{(2)})}} \},$$

where $\mathcal{D}_p^{(1)}$ and $\mathcal{D}_p^{(2)}$ are paired samples whereas $\mathcal{D}_{up}^{(1)}$ and $\mathcal{D}_{up}^{(2)}$ are unpaired samples. Often in a domain adaptation setting, the number of labeled instances in the target domain is limited, compared to that of the source domain. Therefore, here we assume that only a small number of paired labeled data points from both domains are available, i.e. $\{x_1^{(i)}, \dots, x_{n_l}^{(i)}\}$ from $\mathcal{D}_p^{(1)}$ and $\mathcal{D}_p^{(2)}$ are labeled for $i = 1, 2$. Furthermore, the source dataset is equipped with additional unpaired labeled instances during training. Assume that there are Q classes, then the label indicator matrix $Y^{(1)} \in \mathbb{R}^{n_{L1} \times Q}$ for the source

¹The full paper has been accepted for publication in *IEEE-TNNLS*, DOI: 10.1109/TNNLS.2017.2728719

domain is defined as $Y_{ij}^{(1)} = +1$, if the i th point belongs to the j th class and -1 otherwise. Similarly one can define the label indicator matrix $Y^{(2)} \in \mathbb{R}^{n_{L_2} \times Q}$ for the target domain where n_{L_2} denotes the total number of labeled instances in the target domain. The Regularized Semi-Paired KCCA (RSP-KCCA) with centered implicit feature map matrices $\Phi_c^{(1)}$ and $\Phi_c^{(2)}$ in the primal is formulated as follows [2]:

$$\begin{aligned} \max_{w_\ell^{(1)}, w_\ell^{(2)}, r_\ell, e_\ell} \quad & \mu \sum_{\ell=1}^Q e_\ell^T A r_\ell - \frac{1}{2} \sum_{i=1}^2 \sum_{\ell=1}^Q w_\ell^{(i)T} w_\ell^{(i)} - \frac{1}{2} \sum_{\ell=1}^Q e_\ell^T V_1 e_\ell - \frac{1}{2} \sum_{\ell=1}^Q r_\ell^T V_2 r_\ell + \frac{\gamma_3}{2} \sum_{\ell=1}^Q e_\ell^T c_\ell^{(1)} + r_\ell^T c_\ell^{(2)} \\ \text{subject to} \quad & e_\ell = \Phi_c^{(1)} w_\ell^{(1)}, \ell = 1, \dots, Q, \\ & r_\ell = \Phi_c^{(2)} w_\ell^{(2)}, \ell = 1, \dots, Q, \end{aligned}$$

where $c_\ell^{(1)}$ is the ℓ -th column of the matrix $C^{(1)}$ defined as $C^{(1)} = [c_1^{(1)}, \dots, c_Q^{(1)}] = \begin{bmatrix} Y^{(1)} \\ 0_{n_{u_1} \times Q} \end{bmatrix}_{N_1 \times Q}$, and the subscript n_{u_1} denotes the total number of unlabeled instances from source domain. $0_{n_{u_1} \times Q}$ is a zero matrix of size $n_{u_1} \times Q$. Similarly $c_\ell^{(2)}$ is the ℓ -th column of the matrix $C^{(2)}$ defined as $C^{(2)} = [c_1^{(2)}, \dots, c_Q^{(2)}] = \begin{bmatrix} Y^{(2)} \\ 0_{n_{u_2} \times Q} \end{bmatrix}_{n_2 \times Q}$, where the subscript n_{u_2} denotes the total number of unlabeled instances from target domain. Here $V_1 = \gamma_1 P_1 + (1 - \gamma_1) L_1$ and $V_2 = \gamma_2 P_2 + (1 - \gamma_2) L_2$. Matrix A , P_1 and P_2 are defined as follows: $A = \left[\begin{array}{c|c} I_{n_p \times n_p} & 0_{n_p \times n_2 - n_p} \\ \hline 0_{N_1 - n_p \times n_p} & 0_{N_1 - n_p \times n_2 - n_p} \end{array} \right]$, $P_1 = \left[\begin{array}{c|c} I_{n_p \times n_p} & 0_{n_p \times N_1 - n_p} \\ \hline 0_{N_1 - n_p \times n_p} & 0_{N_1 - n_p \times N_1 - n_p} \end{array} \right]$, $P_2 = \left[\begin{array}{c|c} I_{n_p \times n_p} & 0_{n_p \times n_2 - n_p} \\ \hline 0_{n_2 - n_p \times n_p} & 0_{n_2 - n_p \times n_2 - n_p} \end{array} \right]$ and here L_1 and L_2 are the graph Laplacian matrices associated with each of the dataset $\mathcal{D}^{(1)}$ and $\mathcal{D}^{(2)}$.

The applicability of the approach is shown on the Multiple Features Dataset taken from the UCI Machine Learning repository. It contains ten classes, (0 – 9), of handwritten digits with 200 images per class, thus for a total of 2000 images. These digits are represented in terms of six different types of features (heterogeneous features) with different dimensionalities. We first select at random 50% from both source and target data in order to construct the paired training samples. Then 20% of the remaining data from both source and target domains is selected at random to be used as unpaired and unlabeled samples. The final remaining 30% of the source data are labeled in order to construct unpaired and labeled. The same remaining percentage of the target domain data defines the test target dataset, $\mathcal{D}_{\text{test}}^{(2)}$.

Table 1: Comparing the average test accuracy of the proposed RSP-KCCA model with those of mSDA model over 10 simulation runs.

Source domain	mSDA [1]						RSP-KCCA [2]					
	fac	fou	kar	mor	pix	zer	fac	fou	kar	mor	pix	zer
fac	0.9601	0.1840	0.2453	0.3000	0.6217	0.3743	0.9560	0.7683	0.8987	0.6510	0.9233	0.7190
fou	0.6597	0.7953	0.3350	0.3003	0.6123	0.4147	0.8573	0.7993	0.8520	0.6360	0.7853	0.7860
kar	0.6457	0.3170	0.9440	0.2810	0.5483	0.3487	0.9270	0.7780	0.9454	0.6583	0.9540	0.7947
mor	0.5410	0.4083	0.2420	0.6976	0.5167	0.3230	0.7503	0.6857	0.8290	0.7344	0.7920	0.7347
pix	0.6243	0.1733	0.3480	0.2313	0.9644	0.3823	0.9503	0.7817	0.9177	0.6253	0.9590	0.7773
zer	0.6920	0.2697	0.2333	0.2943	0.5770	0.8031	0.8400	0.7837	0.8040	0.6340	0.8773	0.8063

Acknowledgments. The research leading to these results received funding from the European Research Council under the European Union's Seventh Framework Programme (FP7/2007-2013)/ERC AdG A-DATADRIVE-B (290923). This letter reflects only our views; The EU is not responsible for any use that may be made of the information in it. The research leading to these results received funds from the following sources: Research Council KUL: GOA/10/09 MaNet, CoE PFV/10/002 (OPTEC), BIL12/11T; PhD/Postdoc grants; Flemish Government: FWO; PhD/Postdoc grants, projects: G.0377.12 (Structured systems), G.088114N (Tensor based data similarity); IWT: PhD/Postdoc grants, projects: SBO POM (100031); iMinds Medical Information Technologies SBO 2014; Belgian Federal Science Policy Office: IUAP P7/19 (DYSCO, Dynamical systems, control and optimization, 2012-2017). Siamak Mehrkanoon is a postdoctoral fellow of the Research Foundation-Flanders (FWO).

References

- [1] Minmin Chen, Zhixiang Xu, Kilian Weinberger, and Fei Sha. Marginalized denoising autoencoders for domain adaptation. In *in Proceedings of the 29th International Conference on International Conference on Machine Learning (ICML)*, pages 1627–1634, 2012.
- [2] Siamak Mehrkanoon and Johan AK Suykens. Regularized semipaired kernel CCA for domain adaptation. *IEEE Transactions on Neural Networks and Learning Systems*, 2017, DOI: 10.1109/TNNLS.2017.2728719.

Using Values and Norms to Model Realistic Social Agents

Rijk Mercurur, Virginia Dignum, Catholijn Jonker

Delft University of Technology

In the past years a series of papers has highlighted the importance of developing agents which are more socially realistic [7, 6, 5, 10]. Realism, here, means that the agent reproduces human behavior. The aforementioned proponents of social agents furthermore claim that modeling humans with standard agent frameworks (e.g., BDI-agents [13]) filtered according to some ‘social awareness’ modules is not realistic enough. Humans do not first construct a list of possible actions and then realize there are others in the world. Humans use social ingredients, which play a central role in the decision making of the agent. According to Dignum et al. [6], a few possible candidates for such social ingredients are human values, norms, identity, and motives. This paper focuses on two social ingredients: values and norms. Values are understood as ‘what one finds important in life’, for example, privacy, wealth or fairness [11]. Norms are understood as a standard or rule in society [2].

We have looked at a few studies that indeed use norms and values as the main driver of actions [4, 1, 8]. Although these studies help in *constructing* theories on social agents, they do not *check* this theory against independent quantitative data on human behavior. Without this check it is unclear if these theories provide the social realism we need. This raises the question: how useful are value-based and normative agents for modeling *realistic* social agents?

This paper synthesizes and evaluates three theories: a theory on values, a theory on norms and a theory that combines both values and norms. We use the same theory on values as described in Mercurur et al. [9]:

- V.1** There are ten different basic values, those empirically established by Schwartz [12]. Two examples are wealth and fairness.
- V.2** The importance one attributes to a value is correlated with the importance one attributes to other values Schwartz [12]. For example, the values of wealth and fairness are negatively correlated.
- V.3** Humans are heterogeneous in the values they find important.
- V.4** A human considers a subset of these values to determine his or her actions.
- V.5** Values are the prime cognitive determiner for actions.

Our simple theory on norms (e.g., no norm violation) comprises:

- N.1** A statement is a norm if and if only it comprises an actor, action, a deontic elements and a condition that states when it applies.
- N.2** A norm exists, for a particular person, when that persons beliefs other people do or expect that norm.
- N.3** The action a human does is the same as what they perceive as the norm.

And lastly we look at a simple theory that combines norms and values:

VN.1 Some humans always act according to their norms and other humans always act according to their values.

We compared the behavior of humans to that of agents by simulating a psychological experiment: the ultimatum game (UG). In the UG, two players (human or agent) negotiate over a fixed amount of money ('the pie'). Player 1, the *proposer*, demands a portion of the pie, with the remainder offered to Player 2. Player 2, the *responder*, can choose to accept or reject this proposed split. If the responder chooses to 'accept', the proposed split is implemented. If the responder chooses to 'reject', both players get no money. One round comprises of one such demand and reply.

We consider two scenarios: the one-round UG and the multi-round UG. In the one-round UG, humans demand on average 56% of the pie and accept about 81% of the demands [3]. One popular explanation of why humans make these particular demands and accepts is that they have evolved to do this by repeated interaction. This paper checks if simulating our agents to interact for many rounds eventually results in the demands and accepts human make in one-shot games. We found that our theory of norms cannot reproduce these demands and accepts. An intuitive explanation of this is that the normative agents can only copy social norms, but have no incentive to make the precise 56% demands and 81% accepts that humans display. In contrast, our theory on values can reproduce human demands and accepts. Lastly, our theory that combines values and norms can reproduce the human demands and accepts as well. The intuitive explanation here is that this theory allows some agents to act out of their values that will guide the other agents (that act out of norms) towards the realistic demands and accepts.

In the multi-round UG, humans are shown to make on average slightly higher demands and slightly more accepts in the 10th round than in the first. This paper checks if simulating our agents for 10 rounds reproduces this rise in demands and accepts. We found that our theory on values cannot reproduce the learning rate humans display, while our theory on norms can. Norms, in contrast to values, provide a dynamic learning element that makes the agents – just like humans – adaptive. Our theory that combines values and norms can reproduce the learning rate in demands as well. It allows some of these agents to act out of their norms and as such produces similar results as our theory on norms.

We conclude that, for our domain, the theory that combines both values and norms seems the most realistic. First, in contrast to our theory on norms, it can reproduce first-round behavior. Second, in contrast to our theory on values, it can reproduce multi-round behavior. Norms add a realistic adaptive element to the agent behavior, while values provide the static ground-work that realistically models the balance humans find between fairness and wealth. Our study shows a first step in how values and norms can be used to come closer to realistic social agents.

Bibliography

- [1] Atkinson, K., Bench-capon, T.: Value Based Reasoning and the Actions of Others. In: Proceedings of ECAI. pp. 680–688 (2016)
- [2] Bicchieri, C.: The Grammar of Society: The Nature and Dynamics of Social Norms. Cambridge University Press (2005), www.cambridge.org
- [3] Cooper, D.J., Dutcher, E.G.: The Dynamics of Responder Behavior in Ultimatum Games: A Meta-Study. *Experimental Economics* 14(4), 519–546 (2011)
- [4] Dechesne, F., Di Tosto, G., Dignum, V., Dignum, F.: No Smoking Here: Values, Norms and Culture in Multi-Agent Systems. *Artificial Intelligence and Law* 21(1), 79–107 (8 2012), <http://link.springer.com/10.1007/s10506-012-9128-5>
- [5] Dignum, F., Dignum, V., Prada, R., Jonker, C.M.: A Conceptual Architecture for Social Deliberation in Multi-Agent Organizations. *Multiagent and Grid Systems* 11(3), 147–166 (2015)
- [6] Dignum, F., Prada, R., Hofstede, G.: From Autistic to Social Agents. In: Proceedings of the 2014 international conference on Autonomous Agents and Multi-Agent Systems. pp. 1161–1164 (2014), <http://dl.acm.org/citation.cfm?id=2617431>
- [7] Kaminka, G.: Curing robot autism: A challenge. *Proceedings of the 12th International Conference on Autonomous Agents and Multiagent Systems* pp. 801–804 (2013)
- [8] Mercuur, R., Dignum, F., Kashima, Y.: Changing Habits Using Contextualized Decision Making. In: *Advances in Intelligent Systems and Computing*. vol. 528 (2017)
- [9] Mercuur, R., Dignum, V., Jonker, C.: Modeling Social Preferences with Values. In: *Smart Modeling and Simulation for Complex Systems @ IJCAI*. p. (in press) (2017)
- [10] Norling, E.: Don’t Lose Sight of the Forest: Why the Big Picture of Social Intelligence is Essential. In: *Proceedings of the 2016 International Conference on Autonomous Agents & Multiagent Systems*. pp. 984–987. No. May (2016)
- [11] Poel, I.v.d., Royakkers, L.: *Ethics, Technology, and Engineering: An Introduction*. John Wiley & Sons (2011), <http://eu.wiley.com/WileyCDA/WileyTitle/productCd-EHEP002302.html>
- [12] Schwartz, S.H.: An Overview of the Schwartz Theory of Basic Values. *Online Readings in Psychology and Culture* 2, 1–20 (2012)
- [13] Weiss, G.: *Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence*. MIT press, second edi edn. (2013)

Constructing Knowledge Graphs of Depression (Extended Abstract)*

Zhisheng Huang¹, Jie Yang², Frank van Harmelen¹, and Qing Hu¹

¹ VU University Amsterdam, the Netherlands

² Beijing Anding Hospital, China

1 Introduction

Major depressive disorder (MDD) has become a serious problem in modern society. Using antidepressants has been considered the dominant treatment for MDD. Psychiatric doctors confront the challenge to make clinical decision efficiently by gaining a comprehensive analysis over various knowledge resources about depression.

In this paper we propose an approach to constructing a knowledge graph of depression using semantic web technology to integrate those knowledge resources, achieving a high degree of inter-operability. With a single semantic query over integrated knowledge resources, psychiatric doctors can be much more efficient in finding answers to queries which currently require them to explore multiple databases and to make a time-consuming analysis on the results of those searches.

The term “Knowledge Graph” is widely used to refer to a large scale semantic network of entities and concepts plus the semantic relationships among them. Such knowledge graphs are very *a-specific* in terms of the diseases that they cover, and are often prohibitively large, hampering both efficiency for machines and usability for people. In this paper we propose an approach to the construction of *disease-centric knowledge graphs*, which are focussed on a single disease or coherent group of diseases.

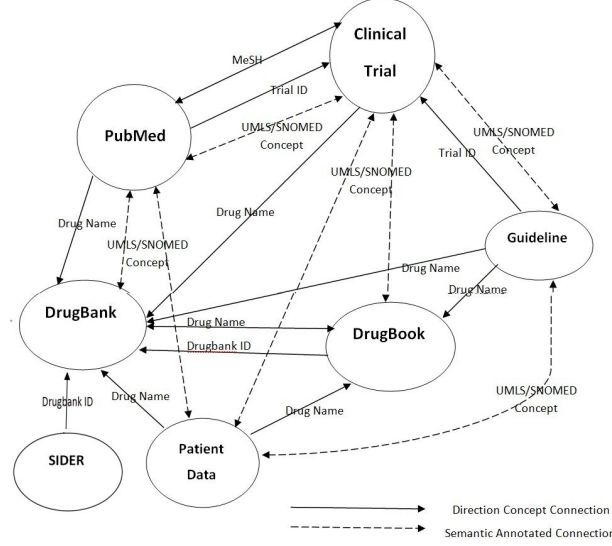
We illustrate our general idea by integrating various knowledge resources about depression (e.g., clinical trials, antidepressants, medical publications, clinical guidelines, etc.). We call the generated knowledge graph *DepressionKG* for short. DepressionKG is represented in RDF/NTriples format [2]. DepressionKG provides a data infrastructure to explore the relationship among various knowledge and data-sources about depression. We show how it provides support for clinical question answering and knowledge browsing.

2 Knowledge Resources and Integration

Following commonly used technology, we construct our knowledge graph as an RDF graph. The current version of DepressionKG (version 0.60) consists of the

* The full version of this paper appears in the proceedings of 2017 International Conference on Health Information Science

RDF representation of the following knowledge resources: i) Clinical trials; ii) PubMed; iii) Guidelines; iv) Drugbank; v) Drugbook; vi) Wikipedia Side Effects of antidepressants; vii) SNOMED CT; viii) Patient Data.



We use the following methods to integrate the various knowledge resources. (i) *Direct Entity identification*; (ii) *Direct Concept identification*; (iii) *Semantic Annotation with an NLP tool*; and (iv) *Semantic Queries with regular expressions*. The figure shows the connectivity of DepressionKG.

3 Conclusions

In this paper, we have proposed an approach to making a knowledge graph of depression, and we have shown how various knowledge resources concerning depression can be integrated for semantic inter-operability. We have provided several use cases for such a knowledge graph of depression. From those use cases, we can see that by using a knowledge graph with its semantic search, it is rather convenient for us to detect relationship which cover multiple knowledge resources. Our conclusions are (i) that it is indeed possible to make disease-centric subgraphs without having to include the entire original graph, and (ii) that realistic clinical queries can still be answered over such disease-specific knowledge graphs without substantial loss of recall.

References

1. S. Ait-Mokhtar, B. D. Bruijn, C. Hagege, and P. Rupi. Intermediary-stage ie components, D3.5. Technical report, EURECA Project, 2014.
2. R. Cyganiak, D. Wood, and M. Lanthaler. RDF 1.1 concepts and abstract syntax, 2014.

Learning Multimodal Transition Dynamics for Model-Based Reinforcement Learning: Abstract.

Thomas M. Moerland, Joost Broekens and Catholijn M. Jonker
Dep. of Computer Science, Delft University of Technology, The Netherlands

1 Introduction

In this work¹ we study how to learn stochastic, multimodal transition dynamics in reinforcement learning tasks. Model-based RL is an important class of RL algorithms that learns and utilizes transition dynamics to enhance data efficiency and target exploration. However, many tasks environments inherently have *stochastic* transition dynamics. We therefore require methods to approximate such complex distributions, while they should also scale to higher-dimensions. In this paper we study conditional variational inference in (deep) neural networks as a principled method to solve this challenge.

2 Conditional Variational Inference

Our goal is to learn a generative model of a (possibly multimodal) distribution $p(y|x)$. We assume the generative process is actually conditioned on some unobserved latent variables z : $p(y|x) = \int p(y|z, x)p(z|x)dz$. The stochastic latent variables z provide the flexibility to predict complex marginal outcome distributions $p(y|x)$. However, the z variables are unobserved and the posterior over z , $p(z|y, x)$, is analytically intractable in most models of interest (for example deep non-linear neural networks).

However, the parameters of this distribution can be efficiently approximated with Stochastic Gradient Variation Bayes (SGVB) [1]. We may first derive a variational lower bound $\mathcal{L}(y|x)$ on our data likelihood $p(y|x)$:

$$\log p(y|x) \geq \mathbb{E}_{z \sim q(\cdot|x, y)}[\log p_\theta(y|z, x)] - D_{\text{KL}}[q_\phi(z|x, y) \| p_\phi(z|x)] = \mathcal{L}(y|x; \theta, \phi) \quad (1)$$

where θ denotes the parameters in a generative network $p_\theta(y|x, z)$, ϕ denotes the parameters in an *inference network* $q_\phi(z|y, x)$ and prior $p_\phi(z|x)$, and D_{KL} denotes the Kullback-Leibler (KL) divergence. The parametric inference network approximates the intractable true posterior over z , while the KL divergence term ensures that the inference network $q_\theta(z|y, x)$ does not diverge too much from the prior $p_\phi(z|x)$. This acts as a regularizer, and ensures that we can at test time (when we do not observe y) sample from $p(z|x)$ instead of $q(z|x, y)$.

¹ Originally published as: Moerland TM, Broekens J, Jonker CM. Learning Multimodal Transition Dynamics for Model-Based Reinforcement Learning. In *Scaling-Up Reinforcement Learning (SURL) Workshop* @ ECML. 2017.

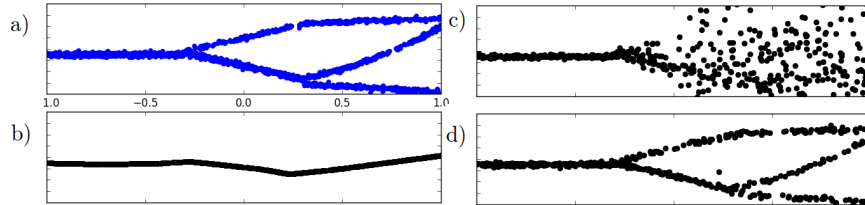


Fig. 1. Comparison of samples from the models produced by multi-layer perceptron (MLP) and conditional variational inference (CVI) networks after training for 30,000 mini-batches. a) Ground truth (artificial) data. b) MLP with deterministic prediction. c) MLP with stochastic inputs. d) CVI with discrete latent z variables.

While the original variational auto-encoder was introduced as a generative model for $p(y)$, the RL setting requires us to condition the entire generative process on the previous state x (i.e. in prior, inference and recognition network). Then, the objective in Eq. 1 can be trained on a single computational graph through the *reparametrization* trick, for details see [1]. For this work we experiment with different types of continuous and discrete latent variables z , for more details see the original paper.

3 Experiments

We illustrate these ideas on an artificial transition function (Fig. 1a) with uni-modal (Fig. 1a, left part), bimodal (middle), and trimodal (right) dynamics. Fig. 1b shows the predictions of a default MLP trained on mean-squared error, which erroneously fits the conditional expectation. Since CVI uses additional noise input to the network (by sampling z), we also compare to a MLP with additional stochastic inputs. Without the inference network, the model is unable to map the input noise to the correct output distribution (Fig. 1c). In contrast, the model with CVI (Fig. 1d) accurately learns the different types of multi-modality, while also correctly predicting the deterministic part of the function. Please refer to the original paper for experiments on reinforcement learning tasks.

4 Conclusion

Our results show that conditional variational inference in deep neural networks successfully predicts multimodal distributions, but also robustly ignores these for deterministic parts of the transition dynamics. Due to the flexibility of neural networks as black-box function approximators, these results are applicable to a variety of RL tasks, and are a key preliminary for model-based RL in stochastic domains.

References

1. Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Simulating Collective Evacuations with Social Elements

Daniel Formolo¹*[0000-0003-2702-5059] and C. Natalie van der Wal¹[0000-0002-0010-8447]

¹Vrije Universiteit Amsterdam, Department of Computer Science, Amsterdam, Netherlands
{d.formolo,c.n.vander.wal}@vu.nl

Abstract. This work proposes an agent-based evacuation model that incorporates social aspects in the behaviour of the agents and validates it on a benchmark. It aims to fill the gap in this research field with mainly evacuation models without psychological and social factors such as group decision making and other social interactions. The model was compared with the previous model, its new social features were analysed and the model was validated. With the inclusion of social aspects, new patterns emerge organically from the behaviour of each agent as showed in the experiments. Notably, people travelling in groups instead of alone seem to reduce evacuation time and helping behaviour is not too costly for the evacuation time as expected. The model was validated with data from a real scenario and demonstrates acceptable results and the potential to be used in predicting real emergency scenarios. This model will be used by emergency management professionals in emergency prevention.

Keywords: Social Contagion, Agent Based Model, Evacuation Simulation.

1 Extended Summary

Most of the evacuation simulation tools consider physical characteristics of the environment, ignoring behaviour and personality of people [13, 10, 9]. Including psychological and social factors in evacuation simulations could make these models more realistic and better in their predictions to ultimately save more lives. For example, observations of actual emergencies show that people tend to be slow to respond to evacuation alarms (taking up to 10 minutes) and take the familiar route out instead of the nearest exit [1, 5]. As part of the EU Horizon 2020 project IMPACT¹, this work proposes and validates an evacuation simulation incorporating these social factors: helping behaviour, groups, age and gender, familiarity, response time and social contagion.

Three existing evacuation simulation models incorporate social and psychological factors, namely FIRESCAP [2], EXODUS² [4] and Multi-Agent Simulation for Crisis Management (MASCM [6]). [9]. These models focus more on the physical constraints and factors such as walking speed, walkways, stairways etcetera to find the optimal flow of the evacuation process. The models include evacuation leaders, evacuees going to the nearest group or social psychological attributes and characteristics for each agent,

¹ <http://www.impact-csa.eu/>

² <http://fseg.gre.ac.uk/index.html>

such as age, sex, breathing rate, running speed. What is missing is group decision making and social interaction, which is included in the proposed model.

The evacuation dynamics were modelled using an agent-based model with the beliefs-desires-intentions and network-oriented modelling approaches [8, 11], and implemented in the Netlogo multi-agent language [12]. Simulation experiments with different factors and levels were designed to determine: (1) if there are significant differences between the improved and initial model version; (2) in how far the model corresponds to reality; (3) the effects of the social features (helping and group formation) on the evacuation time.

Results showed that the current model demonstrates clear improvements over the initial model in terms of Evacuation Time, Falls and Social Influences in the agent's behaviour. Although the experiments presented in this work show the influence of social aspects individually, more experiments have to be conducted to analyse effects of combination of them. The cross relations between social effects and more complex environments might be explored, e.g. environments with pillars or multiple rooms. The validation results were as expected and close to reality: all three model variations performed less than 7,11% to 14,77% (between 42 and 86 seconds) of difference from the benchmark's total evacuation time, which is a good TAT according to [Owen], and the curves of acceptance show values close to the prescribed boundaries, establishing the model's validity. For the future, it is recommended to apply new benchmarks over the model, increasing the confidence about the model's results. Finally, the simulation experiments analysing the influence of Helping and Groups demonstrated interesting patterns useful for future security protocols. Social Contagion effect creates faster evacuation time as expected, because information about the need for evacuation spreads faster than without social contagion. The more people are familiar with the environment: (1) the faster evacuation time and (2) the less falls. These results are a combination of a phased evacuation (less congestion) with more people spread through the environment going to the nearest exits, what leads to less falls as well, and social contagion (the decision to evacuate can spread faster), resulting in faster response and evacuation time. In case of Helping, evacuation time increases only for low crowd density environments. For high crowd density environments the Helping effect is minimised for other effects that grow in importance like blocking of paths due to a number of people. Groups of two people reduce the evacuation time and as more people are added to a group this effect is reduced until it disappears. To conclude, the model has advantages over others that don't consider social effects in collective behaviour to evacuation scenarios, presenting reasonable results and can be used to predict real scenarios. As next steps, new social aspects will be incorporated and more benchmarks will be applied to it.

References

1. Challenger, R., et al. Understanding crowd behaviours, vol. 1: Practical guidance and lessons identified. London: The Stationery Office (TSO) (2010)
2. Feinberg, W. E., Norris R. J. Firescap: A computer simulation model of reaction to a fire alarm. *Journal of Mathematical Sociology* 20.2-3, 247-269 (1995)

3. Galea, E., Deere, S., Filippidis, L., The safeguard validation data set - sgvdsl a guide to the data and validation procedures, fire safety engineering group, university of Greenwich (2012)
4. Gwynne, S., Galea, E. R., Owen, M., Lawrence, P. J., Filippidis, L. A review of the methodologies used in the computer simulation of evacuation from the built environment. *Building and environment*, 34(6), 741-749 (1999)
5. Kobes, M., et al. Building safety and human behaviour in fire: A literature review. *Fire Safety J*, 45(1), 1–11 (2010)
6. Murakami, Y., Minami, K., Kawasoe, T., Ishida, T. Multi-agent simulation for crisis management. In *Knowledge Media Networking, 2002 Proceedings*. 135-139 (2002)
7. Owen M. Galea ER, Lawrence P.J. The Exodus Evacuation Model Applied to Building Evacuation Scenarios. *Journal of Fire Protection Engineering* 8(2):65586 (1996).
8. Rao, A. S., & Georgeff, M. P.: BDI agents: From theory to practice. In: *ICMAS*, Vol. 95, pp. 312-319 (1995)
9. Santos, G., Aguirre, B. A critical review of emergency evacuation simulation models (2004)
10. Templeton, A., Drury, J., Philippides, A. From mindless masses to small groups: Conceptualizing collective behavior in crowd modeling (2015)
11. Treur, J. *Network-Oriented Modeling: Addressing Complexity of Cognitive, Affective and Social Interactions*. Springer (2016)
12. Wilensky, U. NetLogo. <http://ccl.northwestern.edu/netlogo/>. Center for Connected Learning and Computer-Based Modeling, Northwestern University, Evanston (1999)
13. Zheng, X., Zhong, T., Liu, M. Modeling crowd evacuation of a building based on seven methodological approaches. *Building and Environment*, 44(3), 437-445 (2009)

Melody Retrieval and Classification Using Biologically-Inspired Techniques

Dimitrios Bountouridis¹, Dan Brown², Hendrik Vincent Koops¹,
Frans Wiering¹, Remco C. Veltkamp¹

¹ Department of Information and Computing Sciences, Utrecht University

² David R. Cheriton School of Computer Science, University of Waterloo

1 Abstract

Assessing the similarity between two musical items is a complex process that requires human intellect. In addition, Marsden [Marsden2012] argues that similarity involves interpretation, which by itself is a personal creative act. As such, similarity modeling using computational tools, naturally fits to the goals of Artificial Intelligence (AI). In addition, the retrieval and classification of music items is a fundamental problem in AI’s musical counterpart Music Information Retrieval (MIR), and vital to the music recommendation task that underpins the ever-growing digital music industry. Assessing the similarity between two musical items is at the core of both tasks. However, despite the numerous proposed approaches, the problem is far from being solved. The exact mechanics of perceived similarity, is still unknown or incomplete which comes as no surprise considering music’s inherent complex nature. The perceived similarity between two songs is known to be a *subjective* phenomenon, judgments of different individuals can vary significantly. The individual interpretation can be affected also by the *multidimensionality* of music, since similarity between two songs can be a function of timbre, melody, rhythm, structure, or indeed any combinations of those (or other) dimensions. To make matters worse, music similarity is known to be *contextual* thus depending on the circumstances of comparison.

Nevertheless, in the context of symbolically encoded melodies, the most widely used method for similarity assessment has been pairwise alignment via dynamic programming: gaps “-” that represent notes deleted from the melodic sequences are introduced in the sequences, until they have the same length and the amount of “relatedness” between notes at corresponding positions is maximized. Unfortunately, this method suffers from two major drawbacks: first, the dynamic programming technique is slow and fails to efficiently scale to large datasets. Secondly, the final alignment and similarity score are based on pre-defined set of relationships between notes, encoded as a fixed scoring matrix, and pre-defined gap penalties. This fixed scheme is globally applied across the melodies’ length and on all pairs of melodies universally. This contradicts our intuition that certain salient parts of a melody are less likely to change than others in a melody rendition. In addition, the simplistic nature of the substitution matrix clearly cannot accommodate for the proper handling of music uncertainty

and variance (or stability), because the matrix focuses only locally on individual notes. This paper aims to enhance melody retrieval and classification by tackling the two aforementioned issues with the use of bioinformatic techniques. Musical and biological sequences are not as detached as one might think: even as early as the 1950's, it has been observed that they share a number of resembling concepts [Bronson1951].

With regard to efficient classification, we employ BLAST (Basic Local Alignment Search Tool) that uses heuristics to filter out unnecessary comparisons between the query sequence and the target database while it only explores a small part of the dynamic programming space by identifying high-matching substrings. On two datasets of folk and pop music or related melodies grouped into classes, we show that music-unaware BLAST can achieve high classification accuracy, comparable to music-aware methods that use alignment via dynamic programming. However, on the harder retrieval task, the performance of BLAST is relatively low due to its lack of incorporated domain knowledge.

We argue that representations that capture salient and robust properties of musical content, called *profiles* or *prototypes*, are a more intuitive approach to enhance and speed up retrieval. Therefore for our second experiment, we employ an approach with major contributions to the field of computational molecular biology [Durbin1998]: profile hidden Markov models (profile HMMs) [Eddy1998]. These are probabilistic automata that take a multiple sequence alignment (MSA) of related sequences and convert it into a position-specific scoring system that can be used for database searching. Profile HMMs are theoretically more appropriate for assessing the similarity between sequences because they provide a powerful framework for dealing with uncertainty and randomness in sequential data. The scoring and penalizing at each position is dependent on the MSA and not on some arbitrary chosen fixed values, e.g. scoring matrix, gap penalties. We use multiple sequence alignment techniques to reveal the locations of high stability or salience among related melodies and profile HMMs to model the MSA, or in other words, to model the intrafamily similarity. On our two datasets of folk and pop music, we show that profile HMMs outperform the typical alignment or other prototype methods in a retrieval scenario.

In summary, addressing the complexity of music similarity altogether is an almost impossible task. Nevertheless, our experiments show that BLAST and profile HMMs can be reliable and efficient solutions for large-scale melody classification and retrieval respectively, without the incorporation of musical heuristics. As such, they can be generalized to other types of music, datasets and real-life scenarios³.

³ This work appears with the same title in the Proceedings of the 6th International Conference on Computational Intelligence in Music, Sound, Art and Design (2017).

References

- [Marsden2012] Alan Marsden. Interrogating melodic similarity: a definitive phenomenon or the product of interpretation? In: *Journal of New Music Research* 41.4 (2012), pp. 323 - 335.
- [Durbin1998] Richard Durbin, Sean R Eddy, Anders Krogh, and Graeme Mitchison. *Biological sequence analysis: probabilistic models of proteins and nucleic acids*. Cambridge university press, 1998.
- [Eddy1998] Sean R. Eddy. Profile hidden markov models. In: *Bioinformatics* 14.9 (1998), pp. 755 - 763
- [Bronson1951] Bertrand H Bronson. Melodic stability in oral transmission. In: *Journal of the International Folk Music Council* 3 (1951), pp. 50 - 55.

Omniscient Debugging for Cognitive Agent Programs^{*}

Vincent J. Koeman, Koen V. Hindriks, Catholijn M. Jonker

Delft University of Technology, Mekelweg 4 2628CD Delft, The Netherlands
{v.j.koeman,k.v.hindriks,c.m.jonker}@tudelft.nl

For traditional (cyclic) debugging to work, the program under investigation has to be deterministic [3]. Otherwise, reproducing a bug by rerunning the program is likely to fail as it will not hit the same bug again or even hit different bugs [2]. Real-time programs like multi-agent systems are typically not deterministic. Running the same agent system again more often than not results in a different program run or trace, which complicates the iterative process of debugging.

Omniscient debugging is an approach to tackle these issues. Also known as reverse or back-in-time debugging, omniscient debugging is a technique that originates in the context of object-oriented programming (OOP), allowing a programmer to explore arbitrary moments in a program's run by recording the execution. Such a 'time travelling debugger' is regarded as one of the most powerful debugging tools. However, omniscient debugging is still not widely adopted. An important reason for this is that existing (OOP) implementations have a significant performance impact, with slowdown factors ranging from 2 to 300 times. The fact that *agent-oriented programming* (AOP) is based on a higher level of abstraction compared to most other programming languages provides an opportunity to apply omniscient debugging techniques with a significantly lower overhead. The premise here is that tracing for AOP can be based on capturing only high-level decision making events instead of lower-level computational events. Tracing techniques for AOP would thus need tracing of significantly fewer events while still being able to reconstruct all program states, making omniscient debugging for AOP more feasible in practice than for e.g. OOP.

The main contribution of our work is the design of a tracing mechanism for cognitive agent programs that: (i) has a small impact on runtime performance; we show that our technique only has a 10% overhead instead of the much larger factors known for OOP from the literature; (ii) has virtually no impact on program behaviour; we empirically establish that the same tests succeed and fail with or without our tracing mechanism; (iii) can be effectively used for debugging; we propose a visualization technique tailored to cognitive agents and illustrate its application for fault localization. The key question we thus address is whether it is feasible and practical to apply omniscient debugging techniques to AOP without affecting testability.

There are many ways to trace the execution of a program in AOP specifically. Capturing all events, states, and actions performed in a run, i.e., a *full-trace* as

^{*} The full paper was published in the *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence*, pages 265-272, August 2017.

is created in existing solutions for other programming languages, is not feasible in practice for AOP either as it takes 5-20x longer to execute an agent system with such a trace. We therefore started with mechanism that stores a trace that captures as little information as possible but still provides sufficient information for record-replay, i.e., reconstructing any previous state in the programs' run based on replaying the stored *events*. We then extended this minimal trace with information about *state changes* that allows much more efficient reconstruction of a previous state. Finally, we added *source code information* associated with points in a trace to enable a debugger to more effectively explore the trace.

For efficient fault localization, it also needs to be easy for a developer to identify states in a program's execution that are related to the failure under investigation. Moreover, a developer should not get lost in navigating between these states, but always have a sense what point in the execution s/he is evaluating and how the current state affected the execution. Therefore, we adapted the concept of a *space-time view* [1] to cognitive agent programming (see Fig. 1). A space-time view is a table that is structured along space and time dimensions, where the rows in the are composed of the different elements in a state that are traced and the columns entail each step in the agent's execution history.

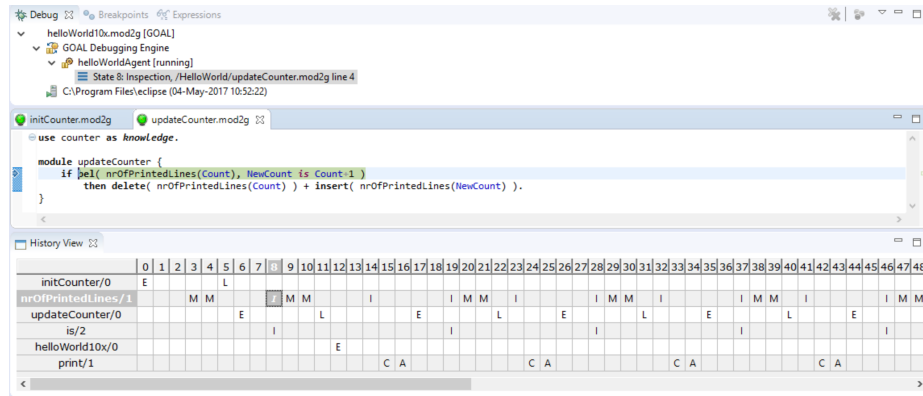


Fig. 1. An example of the space-time view of an execution history.

We believe that our tracing mechanism can provide a starting point for a *history-based explanation mechanism* that can automatically answer questions such as “why did this action (not) happen?”

References

- [1] Azadmanesh, M.R., Hauswirth, M.: Space-time views for back-in-time debugging. Tech. Rep. 2015/02, University of Lugano (May 2015)
- [2] Engblom, J.: A review of reverse debugging. In: System, Software, SoC and Silicon Debug Conference (S4D), 2012. pp. 1–6 (Sep 2012)
- [3] Koeman, V.J., Hindriks, K.V., Jonker, C.M.: Designing a source-level debugger for cognitive agent programs. JAAMAS 31(5), 941–970 (Sep 2017)

Neural Ranking Models with Weak Supervision

Mostafa Dehghani¹, Hamed Zamani², Aliaksei Severyn³,
Jaap Kamps¹, and W. Bruce Croft²

¹ University of Amsterdam

² Google Research

³ University of Massachusetts Amherst

dehghani@uva.nl, zamani@cs.umass.edu,
severyn@google.com, kamps@uva.nl, croft@cs.umass.edu

1 Extended Abstract*

Learning state-of-the-art deep neural network models requires a large amounts of labeled data, which is not always readily available and can be expensive to obtain. To circumvent the lack of human-labeled training examples, unsupervised learning methods aim to model the underlying data distribution, thus learning powerful feature representations of the input data, which can be helpful for building more accurate discriminative models especially when little or even no supervised data is available.

A large group of unsupervised neural models seeks to exploit the implicit internal structure of the input data, which in turn requires customized formulation of the training objective (loss function), targeted network architectures and often non-trivial training setups. Despite the advances in computer vision, speech recognition, and NLP tasks using unsupervised deep neural networks, such advances have not been observed in core information retrieval (IR) problems, such as ranking. A plausible explanation is the complexity of the ranking problem in IR, in the sense that it is not obvious how to learn a ranking model from queries and documents when no supervision in form of the relevance information is available.

To overcome this issue, in this paper, we propose to leverage large amounts of unsupervised data to infer “noisy” or “weak” labels and use that signal for learning supervised models as if we had the ground truth labels. In particular, we use classic unsupervised IR models as a *weak supervision* signal for training deep neural ranking models. Weak supervision here refers to a learning approach that creates its own training data by heuristically retrieving documents for a large query set. This training data is created automatically, and thus it is possible to generate billions of training instances with almost no cost. As training deep neural networks is an exceptionally data hungry process, the idea of pre-training on massive amount of weakly supervised data and then fine-tuning the model using a small amount of supervised data could improve the performance.

The main aim of this paper is to study the impact of weak supervision on neural ranking models, which we break down into the following concrete research questions:

RQ1 Can labels from an unsupervised IR model such as BM25 be used as weak supervision signal to train an effective neural ranker?

RQ2 What input representation and learning objective is most suitable for learning in such a setting?

*This is an extended abstract of Dehghani et al. [1].

RQ3 *Can a supervised learning model benefit from a weak supervision step, especially in cases when labeled data is limited?*

We examine various neural ranking models with different ranking architectures and objectives, i.e., point-wise and pair-wise, as well as different input representations, from encoding query-document pairs into dense/sparse vectors to learning query/document embedding representations. The models are trained on billions of training examples that are annotated by BM25, as the weak supervision signal. Interestingly, we observe that using just training data that are annotated by BM25 as the weak annotator, we can outperform BM25 itself on the test data.

Based on our analysis, the achieved performance is generally indebted to three main factors:

First , defining an objective function that aims to learn the ranking instead of calibrated scoring to relax the network from fitting to the imperfections in the weakly supervised training data.

Second , letting the neural networks learn optimal query/document representations instead of feeding them with a representation based on predefined features. This is a key requirement to maximize the benefits from deep learning models with weak supervision as it enables them to generalize better.

Third , the weak supervision setting makes it possible to train the network on a massive amount of training data.

We further thoroughly analyse the behavior of models to understand what they learn, what is the relationship among different models, and how much training data is needed to go beyond the weak supervision signal. We also study if employing deep neural networks may help in different situations. Finally, we examine the scenario of using the network trained on a weak supervision signal as a pre-training step. We demonstrate that, in the ranking problem, the performance of deep neural networks trained on a limited amount of supervised data significantly improves when they are initialized from a model pre-trained on weakly labeled data.

Our results have broad impact as the proposal to use unsupervised traditional methods as weak supervision signals is applicable to variety of IR tasks, such as filtering or classification, without the need for supervised data. More generally, our approach unifies the classic IR models with currently emerging data-driven approaches in an elegant way.

2 References

- [1] M. Dehghani, H. Zamani, A. Severyn, J. Kamps, and W. B. Croft. Neural ranking models with weak supervision. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '17, pages 65–74, 2017.

Participation Behavior and Social Welfare in Repeated Task Allocations

Qing Chuan Ye¹ and Yingqian Zhang²

¹ Erasmus University Rotterdam, P.O. Box 1738, 3000 DR Rotterdam

² Eindhoven University of Technology, P.O. Box 513, 5600 MB Eindhoven

Overview Task allocation problems have focused on achieving one-shot optimality. In practice, many task allocation problems are of repeated nature, where the allocation outcome of previous rounds may influence the participation of agents in subsequent rounds, and consequently, the quality of the allocations in the long term. In [1], we investigate how allocation influences agents' decision to participate using prospect theory, and simulate how agents' participation affects the system's long term social welfare. We compare two task allocation algorithms in this study, one only considering optimality in terms of costs and the other considering optimality in terms of primarily fairness and secondarily costs. The simulation results demonstrate that fairness incentivizes agents to keep participating and consequently leads to a higher social welfare.

Methodology We consider a multi-round task allocation problem, in which each round has its own subset of jobs. The objective is to maximize the social welfare over all rounds. We assume agents decide for themselves whether they would like to participate in bidding in a certain round or not, using a participation probability, which will be dependent on earlier allocation outcomes.

To model the participation decision using prospect theory, we use the average proportion in the previous round over all agents as the reference point in the editing phase, and a positive (or negative) difference between an agent's proportion in the previous round and the average proportion as a gain (or loss). The intuition is that if an agent feels being treated worse in comparison with others, she might be more uncertain and will care less about participating again, because the time and effort put in the preparation when participating can then be seen as a loss. In order to take into account the experience from previous rounds into the participation probability, we will apply simple exponential smoothing.

Experiments We are interested in the development of the social welfare over multiple rounds and how two different allocation algorithms, a min-cost max-flow algorithm and a fair algorithm, influence the social welfare. Therefore, we conduct a simulation study. We will use a first-price sealed-bid auction where the allocation and the number of jobs bid on from all agents will be made available after each round. We look at several scenarios for both the jobs that are being auctioned (*homogeneous* and *heterogeneous* costs), and the agents that are

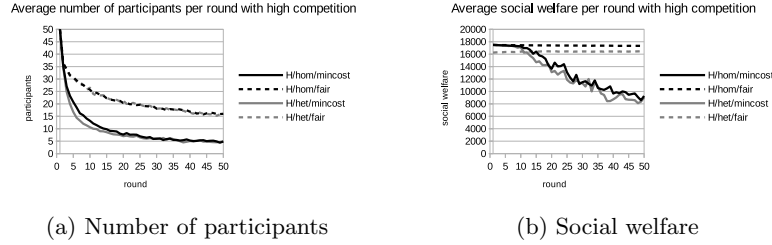


Fig. 1. Average number of participants and social welfare per round over 20 experiments over 50 rounds with high competition.

bidding on the jobs (*low* and *high* competition). For each of the four scenarios, we conduct 20 experiments with 50 rounds, 50 agents and 250 jobs.

We observe that in the low competition case the average number of participants does not differ much between the different cost cases and allocation algorithms over the rounds. This is due to a limited number of bids, resulting in similar allocations regardless of allocation algorithm. Similarly, the average social welfare over the 20 experiments is not very different between the different cases with low competition. In the high competition case, however, the average number of participants when using the fair algorithm is substantially higher than when using the minimum-cost algorithm as the rounds progress (see Fig. 1a). This is due to the jobs being allocated more evenly among agents, which results in higher participation probabilities over all rounds. The social welfare is also substantially higher for the fair algorithm as rounds proceed, and even seems to be stagnant (see Fig. 1b). This is due to the larger number of participants still present, which enables more bids for the algorithm to choose from. The stagnancy stems from the nature of the problem, where it is the maximum social welfare for the problem setting. For the *H/het* scenario, the fair allocation obtains a lower social welfare in the earlier rounds compared to the minimum-cost allocation, but eventually surpasses it, due to the larger number of participants still present.

Allocation algorithms that take into account the participants' behaviors are especially of importance in settings like the sharing economy, which is upcoming in the past years, in which participants share their idle resources and anyone is free to enter and leave as they wish. Therefore, it is important to encourage their participation and to yield an overall higher social welfare, which can be accomplished by ensuring some portion of the market share to the players.

References

1. Q. C. Ye and Y. Zhang, *Participation Behavior and Social Welfare in Repeated Task Allocations*, IEEE International Conference on Agents, pp. 94–97, 2016.

An Empathic Agent that Alleviates Stress by Providing Support via Social Media (Extended Abstract)*

Lenin Medeiros and Tibor Bosse

Behavioural Informatics Group, Vrije Universiteit Amsterdam
De Boelelaan 1081, 1081 HV Amsterdam, NL
{l.medeiros,t.bosse}@vu.nl

Keywords: social media, empathic agents, chatbots, experimentation,
text mining, emotion regulation

In order to help people to cope with everyday stress, *peer support* seems to be a promising means. Online social networks are commonly used for this purpose. Indeed, sharing problems and showing affection are among the most common reasons why people use social media [1].

This paper is part of a project that explores the possibilities of *computer-generated peer support* via online social networks. More specifically, we introduce the concept of ‘artificial friends’ that have the ability to analyze text messages that people share via online social networks, and generate appropriate responses to these messages with the aim of helping them to deal with their ‘everyday problems’ (pretending to be a friend instead of a specialist).

In previous work [2], we conducted a survey via a crowdsourcing platform in order to: 1) identify the most common types of stressful situations shared by people via social media and 2) determine the strategies used by users to support stressed friends in these situations. The resulting data were used to provide a categorization of stressful situations and corresponding support strategies in accordance with the literature mainly inspired by Gross [3] as well as Heaney et al. [4].

Taking this categorization as a point of departure, we are developing a chatbot (an agent) that simulates a friend with the ability to help human users to cope with various stressful situations.

* A version of this extended abstract was published in the proceedings of the *16th International Conference on Autonomous Agents and Multiagent Systems - AAMAS 2017*. The authors would like to state that Lenin Medeiros’ stay at Vrije Universiteit Amsterdam was funded by the Brazilian Science without Borders program. This work was realized with the support from CNPq, National Council for Scientific and Technological Development - Brazil, through a scholarship with reference number 235134/2014-7.

The algorithm behind our agent is based on the results obtained from the previously reported study. The assumption is that the bot receives as input certain messages from users that share stressful personal situations. For each received message, the bot first identifies the type of stressful situation that is involved (for instance, ‘relationship’ or ‘work’). Then, given this stressful situation, the bot selects a type of support strategy that will be used to generate a comforting message. The following support strategies are used: *situation selection* (s.s.), *situation modification* (s.m.), *attentional deployment* (a.d.), *cognitive change* (c.c.), and *general emotional support* (g.e.s.). We know from our previous results how often each strategy was applied to each type of stressful situation. These frequencies are used by the bot as probabilities to select a particular strategy, given a certain situation. For instance, in case an incoming message is classified as a ‘work’ problem, the agent is most likely (with a probability of 44%) to generate a comforting message of the type ‘cognitive change’.

In more detail, the workflow of the application as a whole is as follows. A given user sends a message to the bot, then the bot will try to identify the type of the reported problem and, after this, the bot will select an appropriate support strategy. Finally, after having both the problem and the support strategy identified, the bot will send a support message back to the user. When it is up and running, our bot can be found by searching for ‘stress_support_bot’ on Telegram Messenger App.

A thorough experiment to evaluate the bot’s accuracy as well as its user experience is currently being conducted. Initial results point out that the bot is able to correctly classify incoming messages in the vast majority of cases (over 80%), and that users are generally positive about the appropriateness of the bot’s support messages. Although extensive further testing is obviously required, the expectation underlying this research is that the proposed support agent can help reducing the stress people experience in ‘everyday situations’ by generating tailored response messages, and that this is particularly helpful in cases where users do not receive comforting responses from their human peers.

References

1. Anabel Quan-Haase and Alyson L Young. Uses and gratifications of social media: A comparison of facebook and instant messaging. *Bulletin of Science, Technology & Society*, 30(5):350–361, 2010.
2. Lenin Medeiros and Tibor Bosse. Empirical analysis of social support provided via social media. In *International Conference on Social Informatics*, pages 439–453. Springer, 2016.
3. James J Gross. Emotion regulation: Affective, cognitive, and social consequences. *Psychophysiology*, 39(3):281–291, 2002.
4. Catherine A Heaney and Barbara A Israel. Social networks and social support. *Health behavior and health education: Theory, research, and practice*, 4:189–210, 2008.

Abstract of Expectations Management in Child-Robot Interaction

Mike Ligthart¹, Olivier Blanson Henkemans², Koen Hindriks¹, and Mark A. Neerincx^{1,2}

¹ Interactive Intelligence research group, Delft University of Technology, 2628CD Delft, The Netherlands

`m.e.u.ligthart@tudelft.nl`,

² The Netherlands Organisation for Applied Scientific Research (TNO), The Netherlands

Keywords: Child-Robot Interaction, Social Robotics, Human-Robot Interaction

1 Introduction

Robot applications for children are becoming more mature and are being prepared to be released on the market. The importance to responsibly and appropriately design child-robot interactions increases as these interactions become more elaborate and influential. We need to be especially aware of the consequences of our robot applications when things do not go according to plan. An example is when children expect to have a different interaction with the robot. As we will see later, this actually reduces the effectiveness of the social support the robot can offer.

In our paper ‘Expectation Management in Child-Robot Interaction’, published in the proceedings of RO-MAN 2017 Lisbon, we reflect on several of our user studies and investigated

1. what children expect, within the context of our project, of the interaction with a social robot;
2. what the effect was of misaligned expectations and
3. what we can do to manage expectations.

2 Background

In our paper we do not answer the question how expectations are formed. However, we want to highlight one driving force behind expectation formation and that is anthropomorphizing (ascribing human attributes to) robots [2]. Anthropomorphism is one of the key factors for success of social robots. However it goes wrong when people anthropomorphize too much and expect certain non-existing abilities [3].

In our paper we specifically focus on children in a social/emotional setting rather than a physical setting because 1) most research is focused on physical human-robot interaction (HRI) and 2) child-robot interaction (CRI) is fundamentally different from HRI with adults [1].

Our research is performed in the context of the Personal Assistant for a healthy Lifestyle (PAL) project. The aim is to create a social robot that supports the development of self-management skills of children with diabetes type I [4].

3 What Do Children Expect?

In the orientation phase of the PAL project we performed a co-design session designed to identify the needs, values and expectations of children with diabetes. We analyzed all the collected audio-visual data. We have found that children expect a more unconstrained, substantive and useful interaction with the robot than is possible with the current state-of-the art.

4 Effect of Expectation Misalignment

One of the functions of the PAL-robot is to stimulate the intrinsic motivation of children to keep a digital diabetes diary. We evaluated that functionality with 13 children for 3 weeks. During those 3 weeks we measured their diary adherence, motivation, and perceptions & attitudes towards the robot. During the exit-interviews some children mentioned that they had different expectations while others did not. We compared the aforementioned measures of children with misaligned expectations (4/13) with the others (9/13). We found that children with misaligned expectations added less content to the diary, felt less motivated, felt less related to the robot and viewed it as less sociable. Given the low sample size, the skewed distribution of participants and the ad hoc analysis it is important to do a more thorough study to better understand the consequences and mechanisms of misaligned expectations.

5 Expectation Management Strategies

In our paper we discuss three strategies for the management of expectations in child-robot interaction:

1. *be aware of and analyze children's expectations* — e.g. do not oversell the robot during recruitment and explicitly address the user's expectations during the introduction of the robot.
2. *educate children* — about how robots and artificial intelligence works, what it can and cannot do.
3. *make robots responsible for managing expectations* — e.g. by designing explicit robot behaviors to autonomously detect and adjust misaligned expectations.

6 Conclusion

It is important to manage expectations because it will lead to more effective child-robot interactions and ultimately to a higher quality of life for the children.

References

1. T. Belpaeme, P. Baxter, J. De Greeff, J. Kennedy, R. Read, R. Looije, M. Neerincx, I. Baroni, and M. C. Zelati. Child-robot interaction: Perspectives and challenges. In *Int. Conf. on Social Robotics*, pages 452–459, 2013.
2. T. Fong, I. Nourbakhsh, and K. Dautenhahn. A survey of socially interactive robots. *Robotics and autonomous systems*, 42(3):143–166, 2003.
3. M. Kwon, M. F. Jung, and R. A. Knepper. Human expectations of social robots. In *ACM/IEEE Int. Conf. on Human Robot Interaction*, pages 463–464, 2016.
4. R. Looije. Project website: Personal assistant for a healthy lifestyle, 2014. [accessed on 01-Feb-2017].

Part IV

Type B: Compressed contributions Poster presentation

Chord Label Personalization through Deep Learning of Integrated Harmonic Interval-based Representations

Hendrik Vincent Koops¹, W. Bas de Haas², Jeroen Bransen², and Anja Volk¹

¹ Department of Information and Computing Sciences,
Utrecht University, Utrecht, the Netherlands

`h.v.koops@uu.nl`, `a.volk@uu.nl`

² Chordify, Utrecht, the Netherlands

`bas@chordify.net`, `jeroen@chordify.net`

Abstract

Harmony annotations are at the core of a wide range of studies in music information retrieval and Automatic Chord Estimation (ACE) in particular. Nevertheless, annotator subjectivity makes it hard to derive one-size-fits-all chord labels. Annotators transcribing chords from a recording by ear can disagree because of personal preference, bias towards a particular instrument, and because harmony can be ambiguous perceptually as well as theoretically by definition. These reasons contributed to annotators creating large amounts of heterogeneous chord label reference annotations. For example, on-line repositories for popular songs (e.g. Ultimate Guitar, Chordify) often contain multiple, conflicting versions. Approaches that aim to integrate conflicting versions can be used to create a unified view that can outperform individual sources. Nevertheless, this approach is built on the intuition that one single correct annotation exists that is best for everybody, on which ACE systems are almost exclusively trained and evaluated.

In this paper, we propose a first solution to the problem of finding appropriate chord labels in multiple, subjective heterogeneous reference annotations for the same song. We propose an automatic audio chord label estimation and personalization technique using the harmonic content shared between annotators. We create an harmonic bird’s-eye view from different reference annotations, by integrating their chord labels at the level of harmonic intervals. More specifically, we introduce a new feature that captures the shared harmonic interval profile of multiple chord labels, which we deep learn from audio. First, we extract Constant Q (CQT) features from audio, then we calculate Shared Harmonic Interval Profile (SHIP) features from multiple chord label reference annotations corresponding to the CQT frames. Finally, we train a deep neural network to associate a context window of CQT to SHIP features. From the deep learned shared harmonic interval profiles, we can create chord labels that match a particular *annotator vocabulary*, thereby providing an annotator with familiar, and personal chord labels.

We test our approach on a 20-song dataset with multiple reference annotations, created by annotators who use different chord label vocabularies. In an

experiment we compare training of our chord label personalization system on multiple reference annotations with training on a commonly used single reference annotation. In the first case we train a DNN on SHIPs derived from a dataset containing 20 popular songs annotated by five annotators with varying degrees of musical proficiency. In the second case, we train a DNN on the *Isophonics* single reference annotation. *Isophonics* is a peer-reviewed, and *de facto* standard training reference annotation used in numerous ACE systems.

We show that by taking into account annotator subjectivity, our system is able to personalize chord labels from multiple reference annotations. Comparable high accuracy scores for each annotator show that the model is able to learn a SHIP representation that is meaningful for all annotators, and from which chord labels can be accurately personalized for each annotator. Furthermore, our results show that personalization using a commonly used single reference annotation yields significantly worse results. From the results presented in this paper, we believe chord label personalization is the next step in the evolution of ACE systems.

An Approach for Hospital Planning with Multi-Agent Organizations (Extended Abstract)

John Bruntse Larsen and Jørgen Villadsen

DTU Compute, Technical University of Denmark, 2800 Kongens Lyngby, Denmark
{jobla,jovi}@dtu.dk

As a response to increasing demands and limited resources, the Danish public healthcare sectors are establishing emergency departments with the vision of offering acute treatment with the patient in focus [1]. With this vision arise challenges such as keeping waiting queues short and maintaining a staff work flow that balances acute and scheduled treatments. Existing work [2–6] shows that simulation can be used for decision support in planning hospital resources but provides little insight into the staff work flow. We proposed in our full paper [7] an approach for gaining such insight that applied agent-based simulation and research from the AI-communities [8].

We proposed using the AORTA framework for enabling organizational reasoning in agents [9,10] as an outset for providing this insight through agent-based simulation and formal logic [11,12]. To demonstrate the applicability of the approach, we showed how an informal description of acute patient treatment can be represented as a model in the AORTA framework.

In this extended abstract we show the process of creating the AORTA model. We use the following fragment of stages (1,2) and conventions (a,b,c) from the informal description in our full paper [7] as a working example in creating the model:

1. *Triage*: A nurse carries out the triage process on the patient.
2. *Diagnosis and Treatment*: A doctor performs a diagnosis and initial treatment on the patient.
 - a. Nurses fill out a triage form for the patient as part of the triage.
 - b. Doctors in the specialized departments take care of scheduled treatments.
 - c. Initial treatment of patients may require assistance from doctors from different specialized departments.

An important part of organizational reasoning is that humans generally act as expected, but are also able to do otherwise. AORTA gives agents the same flexibility by having agents maintain organizational knowledge separately from personal beliefs. In the reasoning cycle of the agent it then queries both its organizational knowledge and its personal beliefs. The organizational knowledge is represented as logic predicates in an *organizational metamodel*. Table 1 shows a subset of the symbols that can be used to define the metamodel and the remaining part of the paper describes the process of creating a metamodel for the above informal description of acute patient treatment. The role-predicates define roles and their main objectives. The stages in the informal description

Predicate	Informal meaning
<code>role(Role, Objs)</code>	<i>Role</i> is the name of a role and <i>Objs</i> is a set of main objectives of that role.
<code>cond(Role, Obj, Deadline, Cond)</code>	When the condition <i>Cond</i> holds, <i>Role</i> is obliged to complete <i>Obj</i> before the objective <i>Deadline</i> .

Table 1. Subset of predicates of the AORTA metamodel and their informal meaning.

<code>role(nurse, {triage(Patient)})</code>
<code>role(acute_doctor, {acute_treatment(Patient), treatment_plan(Patient)})</code>
<code>role(specialized_department, {scheduled_treatment(Department, Patient)})</code>
<code>role(specialized_doctor, {scheduled_treatment(Department, Patient)})</code>
<code>cond(nurse, fill_form(Patient, Nurse), triage(Patient), admission(Patient))</code>
<code>cond(acute_doctor, specialized_treatment(Patient, specialized_doctor), acute_treatment(Patient), specialistNecessary(Patient, specialized_doctor))</code>

Table 2. Metamodel based on the stages and conventions in the informal description.

mention: a *nurse* that carries out triage and an *acute doctor* that carries out acute treatment. Additionally, convention (c) mentions a *specialized department* that carries out scheduled treatments and *specialized doctors* that carries out scheduled treatments work in the department. The cond-predicates are deontic temporal statements that define conditions on how the agents should complete objectives when enacting a role. The conventions in the informal description mention: *nurses* should fill in the triage form before they finish the *triage* when a patient has been admitted, and that *acute doctors* should involve a specialist in the acute treatment if necessary. Table 2 shows the predicates of the metamodel for acute patient treatment.

We have shown how the informal description of the emergency department scenario can be represented in the AORTA framework for adding organizational reasoning to agents. Our work provides an outset for gaining insight into the staff work flow through agent-based simulation and logic. JaCaMo [13] offers another framework for implementing agent organizations that was used by the winning team in the Multi-Agent Programming Contest 2016. Future work includes investigating to what extent agent programming frameworks and process mining can facilitate agent-based simulation. Process mining is a method for discovering and extending process models based on event logs. In line with [14–16], we aim to use this method to extend the agent model and do quantitative evaluation based on logs of real world activity.

Acknowledgements

This work is part of the Industrial PhD project *Hospital Staff Planning with Multi-Agent Goals* between PDC A/S and Technical University of Denmark. We are grateful to Innovation Fund Denmark for funding and the governmental institute Region H, which manages the hospitals in the Danish capital region, for being a collaborator on the project. We would also like to thank Rijk Mercuur and Virginia Dignum for comments on a draft.

References

1. Hansen-Nord, M., Steensen, J.P., Holm, S.: Process driven patient tracks in FAM. *Scandinavian Journal of Trauma, Resuscitation and Emergency Medicine* **18(Suppl. 1)** (2010) pp. 27
2. Liu, Z., Rexachs, D., Luque, E., Epelde, F., Cabrera, E.: Simulating the micro-level behavior of emergency department for macro-level features prediction. In: *Winter Simulation Conference 2015*, IEEE Press (2016) pp. 171–182
3. Taboada, M., Cabrera, E., Iglesias, M.L., Epelde, F., Luque, E.: An agent-based decision support system for hospitals emergency departments. *Procedia Computer Science* **4** (2011) pp. 1870–1879
4. Pedersen, K.M., Petersen, N.C.: *Fremtidens Hospital*. Munksgaard (2014)
5. Siebers, P.O., Macal, C.M., Garnett, J., Buxton, D., Pidd, M.: Discrete-event simulation is dead, long live agent-based simulation! *Journal of Simulation* **4**(3) (2010) pp. 204–210
6. Hutzschenreuter, A.K.: *A Computational Approach to Patient Flow Logistics in Hospitals*. Eindhoven University of Technology (2010) PhD Thesis.
7. Larsen, J.B., Villadsen, J.: An approach for hospital planning with multi-agent organizations. In: *Rough Sets: International Joint Conference, IJCRS 2017, Part II*, Springer (2017) pp. 454–465
8. Weiss, G., ed.: *Multiagent Systems – 2nd Edition*. MIT Press (2013)
9. Jensen, A.S., Dignum, V.: AORTA: adding organizational reasoning to agents. *Proceedings of the 13th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2014)* **2**(3) (2014) pp. 1493–1494
10. Jensen, A.S., Dignum, V., Villadsen, J.: The AORTA architecture: Integrating organizational reasoning in Jason. *Lecture Notes in Computer Science* **8758**(3) (2014) pp. 127–145
11. Geuvers, H.: Proof assistants: History, ideas and future. *Sadhana* **34**(1) (2009) pp. 3–25
12. Nipkow, T., Paulson, L.C., Wenzel, M.: Isabelle/HOL — A Proof Assistant for Higher-Order Logic. Volume 2283 of *Lecture Notes in Computer Science*. Springer (2002)
13. Hübner, J.F., Boissier, O., Kitio, R., Ricci, A.: Instrumenting multi-agent organisations with organisational artifacts and agents. *Autonomous Agents and Multi-Agent Systems* **20**(3) (2010) pp. 369–400
14. Mans, R.S., Schonenberg, M.H., Song, M., van der Aalst, W.M.P., Bakker, P.J.M.: Application of process mining in healthcare – a case study in a dutch hospital. In: *Biomedical Engineering Systems and Technologies: International Joint Conference, BIOSTEC 2008 Funchal, Madeira, Portugal*, Springer (2009) pp. 425–438
15. Kirchner, K., Herzberg, N., Rogge-Solti, A., Weske, M.: Embedding conformance checking in a process intelligence system in hospital environments. In: *Process Support and Knowledge Representation in Health Care: BPM 2012 Joint Workshop, ProHealth 2012/KR4HC 2012, Tallinn, Estonia*. Volume 7738., Springer (2013) pp. 126–139
16. Rovani, M., Maggi, F.M., de Leoni, M., van der Aalst, W.M.P.: Declarative process mining in healthcare. *Expert Systems With Applications* **42**(23) (2015) pp. 9236–9251

Part V

Type C: Demonstrations

Hierarchical Reinforcement Learning for a Robotic Partially Observable Task

Denis Steckelmacher, H  l  ne Plisnier, Diederik M. Roijers, and Ann Now  

Vrije Universiteit Brussel, Brussels, Belgium
{dsteckel,hplisnie,droijers,anowe}@vub.ac.be
http://steckdenis.be/bnaic_demo.mp4

Abstract. Most real-world reinforcement learning problems have a hierarchical nature, and often exhibit some degree of partial observability. While hierarchy and partial observability are usually tackled separately, we illustrate on a complex robotic task that addressing both problems simultaneously is simpler and more efficient. We decompose our complex partially observable task into a set of sub-tasks, in a way that allows each sub-task to be solved by a memoryless option. Then, we implement Option-Observation Initiation Sets [4], that make the selection of any option conditional on the previously-executed option. Our agent successfully learns the task, achieves better results than a carefully crafted human policy, and does so much faster than an recurrent neural network over options [3].

1 Background

Options factor a complex task into simpler sub-tasks. The agent learns a top-level policy that repeatedly selects options, that in turn execute a sequence of actions before returning [5]. In Partially Observable MDPs (POMDPs), the agent does not fully observe the state of the environment. Remembering past observations therefore becomes necessary to behave optimally [2]. While most current approaches to POMDPs rely on recurrent neural networks [1], sometimes used on top of options [3], the experiment in this demonstration shows that Option-Observation Initiation Sets [4] allow a challenging robotic partially observable task to be learned, achieving better results than an expert policy, while providing intuitive explanations of the behavior of the robot.

2 Experiment

This experiment is inspired from a real-world industrial object gathering task, where a robot has to bring objects from two terminals to a central carrier belt, for further processing by other robots. A Khepera III robot has to gather objects from two terminals, separated by a wall, and to bring them to the root, as shown in Figure 1 (a). Objects have to be gathered one by one from a terminal until it becomes empty. When a terminal is emptied, the other one is automatically refilled. The robot therefore has to alternatively gather objects from both terminals. The root is colored in red and marked by a paper QR-code encoding 1. Each terminal has a screen displaying its color and a 1 QR-code

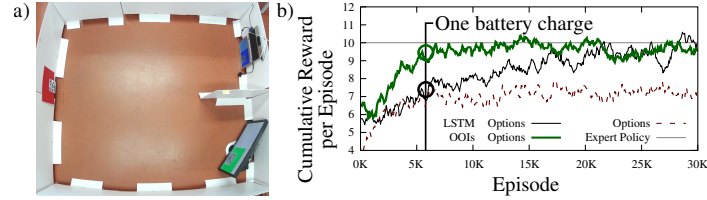


Fig. 1. a) Experimental setup. b) Cumulative reward per episode for different agents (*OOIs* + *Options* is ours). Our agent outperforms a human expert policy.

when full, 2 when empty. Because the camera mounted on the robot cannot read QR-codes from far away, the state of a terminal cannot be observed from the root, where the agent has to decide to which terminal it will go. This makes the environment partially observable. Fixed options allow the robot to move towards the largest red, green or blue blob in its field of view. The options terminate as soon as a QR-code close enough to be read. The robot has to learn a policy over options that solves the task.

During the demonstration, we will compare the policy learned by the robot with an expert policy, that empties a terminal before switching to the other one (see video¹). The learned policy obtains returns comparable with the expert policy, and can be learned by the robot in a single battery charge (see Figure 1, b).

Acknowledgments

The first author is “Aspirant” with the Science Foundation of Flanders (FWO, Belgium, 1129317N). The third author is “Postdoctoral Fellow” with the FWO (12J0617N).

References

1. Bram Bakker. Reinforcement learning with long short-term memory. In *Advances in Neural Information Processing Systems (NIPS)*, volume 14, 2001.
2. Long-Ji Lin and Tom M Mitchell. *Memory approaches to reinforcement learning in non-Markovian domains*. Carnegie-Mellon University. Department of Computer Science, 1992.
3. Mohan Sridharan, Jeremy Wyatt, and Richard Dearden. Planning to see: A hierarchical approach to planning visual actions on a robot using POMDPs. *Artificial Intelligence*, 174:704–725, 2010.
4. Denis Steckelmacher, Diederik M. Roijers, Anna Harutyunyan, Peter Vrancx, and Ann Nowé. Reinforcement learning in POMDPs with memoryless options and option-observation initiation sets. *CoRR*, abs/1708.06551, 2017.
5. Richard Sutton, Doina Precup, and Satinder Singh. Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning. *Artificial intelligence*, 112:181–211, 1999.

¹ http://steckdenis.be/bnaic_demo.mp4

MORL-Glue: A Benchmark Suite for Multi-Objective Reinforcement Learning

Peter Vamplew, Dean Webb, Luisa M. Zintgraf, Diederik M. Roijers, Richard Dazeley, Rustam Issabekov, and Evan Dekker

FLAG, Federation University, Ballarat, Australia
AI laboratory, Vrije Universiteit Brussel, Brussels, Belgium
`p.vamplew,d.webb,r.dazeley,r.issabekov,e.dekker@federation.edu.au & lzintgraf,droijers@ai.vub.ac.be`

Many — if not most — real-world decision problems involve multiple (possibly conflicting) objectives. If a model of the environment is not readily available, agents must interact with this environment, in order to learn what the best possible policies are. Unlike in single-objective problems, in such *multi-objective reinforcement learning (MORL)* [2] problems it is not immediately obvious what the best policy is, as different users may have different preferences with respect to the different objectives. Furthermore, even with only one user, it is typically very hard, if not impossible, for users to specify a *utility function* that reflects their preferences. Therefore, in MORL, we typically learn a set of possibly optimal policies, that reflect different trade-offs between the objectives. Specifically, agents aim to learn a *coverage set (CS)* with at least one optimal policy for every possible utility function that a user might have.

It has been argued [3] that MORL is underrepresented in the reinforcement learning (RL) literature. Perhaps one reason for this is that while many good benchmarks and benchmarking software exist for single-objective RL [4, 6], MORL research papers typically employ only one or two benchmarks that do not necessarily overlap, and those are often not freely available. This situation leads to only partial comparability and reproducibility. In this work we aim to mitigate this by proposing a benchmark suite that not only contains the most popular benchmarks for MORL used in the literature, but also proposes and includes new benchmarks that are generalised [6, 7]. Not only do these generalised benchmarks help to prevent method overfitting, they also provide researchers with an easy-to-use tool to investigate how their algorithms perform under different characteristics of their MORL problems (such as the size of state and action spaces, the amount of stochasticity in the transition function, etc.).

To create a good benchmark suite, we previously outlined the following criteria [5, 7]: 1) two or more objectives, 2) stochasticity in transition dynamics and/or rewards, 3) different Pareto front features such as concavities and discontinuities, 4) state dimensionality high enough to require the use of function approximation, 5) continuous state or action spaces, 6) partially-observable states, 7) a mixture of episodic and continuing tasks. In the proposed MORL-Glue benchmark suite, criteria 1, 2, and 3 are implemented via standard benchmarks, as well as the new generalised Deep Sea Treasure (G-DST) and random multiobjective Markov Decision Process (MOMDP) problems, and the Collecting Traveller problem (CTP)

[7]. The latter three problems are generalised, and can thus scale up with respect to the number of objectives, and the size of the state and action spaces (criterion 4). Furthermore, the CTP can be made continuous as well as partially observable (criteria 5 and 6). Furthermore, CTP and G-DST are episodic problems, while random MOMDPs are continuing tasks (criterion 7).

Our implementation builds on RL-Glue [4]. It runs a server to which separate agent, environment, and experiment processes connect via sockets. Therefore, these can be implemented in any (and different) languages, facilitating the reuse of standard benchmark implementations with new learning algorithms.

1 The Demo

The demo consists of training an agent on one of our generalised benchmarks, the generalised Deep Sea Treasure (G-DST) problem. We show how an algorithm, in this case Q-learning in combination with thresholding and lexicographic ordering over objectives [1], can be evaluated across different parameters of the problem. Specifically, we designed an experiment in which the size of the state-space is increased: at every instance, we determine the size of the state-space, and generate a random solution set (i.e., coverage set). We then show how many samples it takes before the agent can accurately estimate this coverage set, as a function of the size of this state-space. The demo contains a visualisation component, intended to show the behaviour of the agent at different times during learning. A video accompanying this demo can be found at <https://eportfolios.federation.edu.au/view/view.php?id=74593>, and the MORL-Glue software is available from <https://github.com/FedUni/MORL>.

Acknowledgements

Diederik M. Roijers is a postdoctoral fellow of the Research Foundation – Flanders (FWO). This research was in part supported by Innoviris – Brussels Institute for Research and Innovation.

References

1. Z. Gabor, Z. Kalmar, and C. Szepesvari. Multi-criteria reinforcement learning. In *ICML*, pages 197–205, July 24–27 1998.
2. D.M. Roijers, P. Vamplew, S. Whiteson, and R. Dazeley. A survey of multi-objective sequential decision-making. *JAIR*, 47:67–113, 2013.
3. D.M. Roijers, S. Whiteson, P. Vamplew, and R. Dazeley. Why multi-objective reinforcement learning? In *European Workshop on Reinforcement Learning*, 2015.
4. B. Tanner and A. White. RL-Glue: Language-independent software for reinforcement-learning experiments. *Journal of Machine Learning Research*, 10(Sep):2133–2136, 2009.
5. P. Vamplew, R. Dazeley, A. Berry, E. Dekker, and R. Issabekov. Empirical evaluation methods for multiobjective reinforcement learning algorithms. *Machine Learning*, 84(1-2):51–80, 2011.
6. S. Whiteson, B. Tanner, M.E. Taylor, and P. Stone. Generalized domains for empirical evaluations in reinforcement learning. In *ICML*, 2009.
7. L.M. Zintgraf, T.V. Kanters, D.M. Roijers, F.A. Oliehoek, and P. Beau. Quality assessment of MORL algorithms: A utility-based approach. In *Benelearn*, 2015.

RoboCup HQ: A new benchmark focusing on AI, HMI and Autonomous Agents

Tijn van der Zant¹
Lars Zwanepol Klinkmeijer¹

¹ SIM-CI, Zuidhollandlaan 7, The Hague, Netherlands
tijn.van.der.zant@sim-ci.com
lars.zwanepol@sim-ci.com

Abstract. The focus of the RoboCup Federation has expanded in the past two decades to include more AI and HRI. This paper describes a proposal for a new league which will be presented in December 2017. Typically a league is used as an international benchmark for scientific progress. This benchmark is called RoboCup HQ and runs in the cloud.

The benchmark consists of an online simulation environment and a virtual headquarters. In the simulation a near complete city, including infrastructure, is modelled in 3D. Similar to the current Robocup Rescue Simulation League, research groups can control agents such as emergency services. The control of the agents is both on the operational level and on the individual (driving) level. In addition Robocup HQ introduces possibilities for creating realistic population behavior using MAS. Last but not least a virtual HQ is introduced where a human controller has to make strategic and operational decisions. AI assisted Decision Support and Human Machine Interfacing need to be developed to assist those decisions.

Keywords: Benchmarking, disaster scenarios, AI driven HMI, autonomous agents.

1 Introduction to RoboCup

RoboCup started in 1997 as a response to the difficulties of benchmarking robots. Robots were operated in laboratories, but since all laboratories are different it is difficult to benchmark them. RoboCup started with soccer playing robots. Soccer added the complexity of a dynamic environment. It started with real world robots and robots in simulated environments. A few years later RoboCup Rescue was initiated, focusing on disaster scenarios such as a collapsed building and recently on the broken down nuclear reactor of Fukushima. In 2006 RoboCup@Home was initiated focusing on autonomous robots, natural human robot interaction and applicable technologies for domestic service robots. The idea was, and is, to create autonomous care robots and general purpose household robots. Also leagues focusing on logistics and warehousing were created by request from the industry. The simulation leagues were/are focusing on soccer and on disaster scenarios.

At the moment there are two rescue simulation leagues. One is focusing, akin the real world Rescue league, on a collapsed building scenario. The goal is to find victims

and call for human assistance. The other one is focusing on logistics of sending rescue workers and equipment. RoboCup HQ combines these league adding the complexity of an intuitive and adaptive interface for the controlling of emergency responses during an unfolding disaster.

In December 2017 SIM-CI will introduce the simulator and its ideas for the league to the RoboCup community at the Asia-Pacific RoboCup event (<http://www.robocup-ap2017.org/>) in Bangkok. During the event SIM-CI will give a workshop to discuss with interested parties on how to fill in the details of the competition and see what would be the most interesting path to follow.

2 RoboCup HQ benchmark details

RoboCup HQ provides an online simulation environment and a virtual headquarters. In the simulation a near complete city, including infrastructure, is modelled in 3D. Similar to the current Robocup Rescue Simulation League, research groups can control agents such as emergency services. The control of the agents is both on the operational level and on the individual (driving) level. In addition Robocup HQ introduces possibilities for creating realistic population behavior using MAS. Last but not least a virtual HQ is introduced where a human controller has to make strategic and operational decisions. AI assisted Decision Support and Human Machine Interfacing need to be developed to assist those decisions.

The Robocup HQ simulator provides a rich environment with several disaster scenarios. For example, flooding, earthquakes, cyberattacks and extreme weather.

The proposed challenges that can be developed include:

- Create realistic human crowds using MAS
- Create citizens with regular daily routines using MAS
- Create realistic traffic patterns using MAS
- Control swarms of drones for information gathering during a scenario
- Plan operations for emergency services with priority planning
- Control path planning for individual agents.

SIM-CI will provide different types of disaster scenarios such as flooding, earthquakes, cyberattacks and extreme weather. This creates a diverse set of challenges for the competitors. To create a lively city one of the first proposed challenges is to create realistic human agents. They should have normal daily routines but also extreme behaviors such as the possibility to plunder or flee a district or area. Potentially these human agents can add to the richness of the information by posting on ‘social media’ during the running of a simulation.

Taking near-future technology into consideration, it will be possible to control drone swarms and other autonomous robotic vehicles which act as sensors. This can be used, for example, to find dangerous situations and victims which require rescuing.

The HQ is the emergency management center where information is gathered and presented to a human controller. This controller has to make immediate decisions to get control of the emergency and mitigate its effects, for example to decide which place to evacuate. These decisions can have a large impact on the amount and type of victims. Autonomous emergency vehicles will have to adapt to these changing priorities.

Possible competition challenges are:

- Intuitive presentation of information (jury decision)
- Effective transformation of the data into information (jury decision)
- Intuitive interface to issue commands

3 Social importance

The long term goal is that this simulation helps to design more resilient cities. The HQ, where humans are interacting with the simulator, can be offered to local chiefs of police and other emergency response officers as a special training program. For this reason the HQ should be intuitive and easy to use, preferably also adapting itself to the users.

4 Continued development

At the RoboCup Asia-Pacific (<http://www.robocup-ap2017.org/>) the first workshop will be held for the interested teams. The details of the competition like rules, challenges and richness of the world will be worked out together with the participating Robocup Community. We aim to create a benchmark with a resiliency index and other metrics in order to give a score to a participating team. Other research areas are invited to assist in creating the resiliency index.

SIM-CI will continuously develop the simulator as it is based on the same technology as used for its commercial customers. We aim for ease of use, creating a low threshold to start using the simulator. This opens the competition up to more groups. Also the visuals are very important to SIM-CI which results in a more attractive event at the RoboCup. To lower the barriers even further 100k€ of compute time on the Google Cloud Platform is made available for participating teams during the first year. An example of the simulator can be found at: <https://vimeo.com/233472131>

The SimuLane Highway Traffic Simulator for Multi-Agent Reinforcement Learning

Manon Legrand*, Roxana Rădulescu, Diederik M. Roijers, and Ann Nowé

Vrije Universiteit Brussel, Artificial Intelligence Lab
manon.legrand@live.be, {rradules, droijers, ann.nowe}@ai.vub.ac.be

Abstract. SimuLane¹ is a highway traffic simulator modeling a multi-agent learning environment. Human drivers are simulated through a behavior-based model translating basic characteristics and desires of real-life drivers. SimuLane is designed in a fully parameterizable way and serves as a learning environment for self-driving cars; as such, a ratio of autonomous drivers – all using the same learning model – can be defined for either training or a simple simulation.

1 Simulator Overview

SimuLane features a fully discrete representation of straight highway lanes, i.e., each lane is represented as a list of cells by which cars pass. Each lane has a preferred speed, reflecting the idea that cars’ speed usually increases from right to left on highway lanes. Exits are subsets of consecutive cells in the exit lanes preceded by an indicator. The number of exits, the size of the exits and the distance between them are all parameters of the simulator.

Another important component of the simulator are the human drivers. They act according to a behavioral model which can be summarized by three “rules”: “*the driver must not crash*”, “*the driver must reach their desired speed*” and “*the driver must respect the lanes’ speeds*”. To simulate the *human* nature of these drivers, the fact that they make mistakes or non-logical choices, the *irrationality* is defined by the probability that the driver chooses an action randomly.

The *traffic density* is defined as the probability for each lane that a new driver arrives in that lane. The first cell of each lane is thus updated according to the traffic density at every step. The simulation step for the other highway constituents is divided into two sub-steps: the *observation* and the *update* phases. During the former, drivers in the simulator “observe” their environment and chooses an action while the actual state of the highway is updated during the latter. The observation phase introduces an arbitrary, pseudo-temporal order over the drivers’ actions; a driver choosing their action is aware of the action – the direction and the acceleration – of the drivers who precede them. Finally, if a crash occurs, it “stays” in the cell for a certain number of simulation steps. While it is present, the cell must be avoided by the drivers, otherwise it would cause another crash.

* This work was carried out for the first author’s master thesis studies at VUB [1].

¹ Demonstration video: <https://youtu.be/tHBzCNTKA6w>

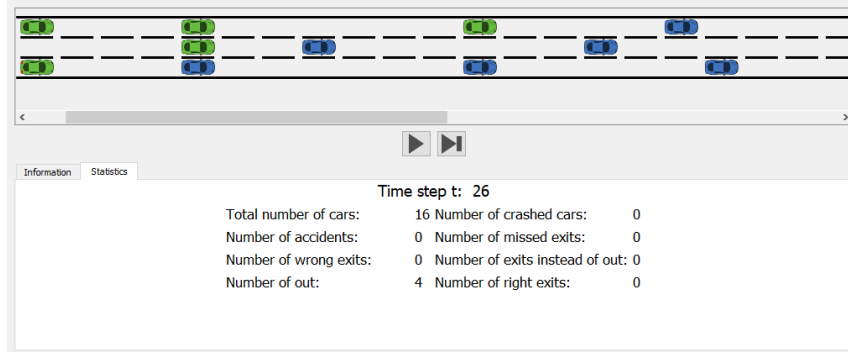


Fig. 1. Graphical interface of SimuLane where simulated human drivers are the blue cars, and the autonomous cars, the learning agents, are the green cars.

2 Learning Environment

The simulator aims to be used as the learning environment for a reinforcement learning problem with *autonomous cars* as agents [2]. SimuLane allows agents to retrieve their local state in the environment (limited by their field of view) and perform their selected actions. An action is a tuple $\langle direction, acceleration \rangle$; the direction can be left, forward or right, and the acceleration is bounded by the *maximum acceleration* of the car. The reward function is fully customizable for each possible outcome (e.g., crash, missed exit, correct exit). SimuLane can therefore also easily be extended for multi-objective RL [3, 4].

SimuLane offers two learning settings: single-agent and multi-agent. In the first case, there is only one autonomous car in the highway at a time, together with “human” drivers. In the second case, the *ratio* of autonomous cars is the probability that a new driver entering the highway is an autonomous car; in this multi-agent setting, all agents use the same learning model during learning (i.e., centralized training but decentralized execution).

SimuLane is implemented in Python, version 2.7 and can be used in combination with any Python-based (deep) reinforcement learning library.

Acknowledgements This work is supported by Flanders Innovation & Entrepreneurship (VLAIO), SBO project 140047: Stable Multi-agent LEarnIng for neTworks (SMILE-IT). Diederik Roijers is an FWO postdoctoral fellow (grant number 12J0617N).

References

1. M. Legrand. Deep Reinforcement Learning for Autonomous Vehicle Control among Human Drivers. Master dissertation, Vrije Universiteit Brussel, 2017.
2. M. Legrand, R. Rădulescu, D.M. Roijers, and A. Nowé. Neural network reuse in deep RL for autonomous vehicles among human drivers. In *BNAIC*, 2017.
3. H. Mossalam, Y.M. Assael, D.M. Roijers, and S. Whiteson. Multi-objective deep reinforcement learning. In *NIPS workshop on Deep RL*, 2016.
4. D.M. Roijers, P. Vamplew, S. Whiteson, and R. Dazeley. A survey of multi-objective sequential decision-making. *JAIR*, 48:67–113, 2013.

ECrowd: Enterprise Crowdsourcing for Training Cognitive Systems using the Workforce

Benjamin Timmermans¹ Zoltán Szilávik¹ Manfred Overmeen¹
Alessandro Bozzon²

¹*IBM Benelux Center for Advanced Studies, Amsterdam,* ²*TU Delft*

1 Crowdsourcing in an Enterprise Environment

Crowdsourcing has become a mainstream process for gathering ground truth data in order to build knowledge bases, and for the training, testing and evaluation of machine learning algorithms. In this process, small human annotation tasks are performed by people from all over the world in return for a small fee. The increased benefits of this crowdsourcing in terms of cost, time and annotation quality have been shown in previous work [5].

Although the benefits of using crowdsourcing in a corporate environment have been explored [6], its application in practice has been limited. The reason for this is twofold: employees have no incentive to perform tasks as they cannot get paid as reward, and crowdsourcing tasks can also not be sent to external annotators as companies often deal with confidential data. However, as artificial intelligence and, specifically, supervised machine learning algorithms become more common, so does the need for annotated data. In this paper we introduce a platform for enterprise crowdsourcing called ECrowd, that we a) aim to employ to get a better understanding on how enterprise mobile crowdsourcing (EMC) could be sustainably adopted in a traditional work environment, b) use for gathering training data for Cognitive Systems, and c) build on to create business applications. Our video demonstration of ECrowd is available at <https://vimeo.com/233505376> with password *ecrowd*.

2 ECrowd: Platform and Methodology

Our enterprise crowdsourcing application ECrowd is a platform in which mobile enterprise crowdsourcing applications and tasks can be instantiated and managed. Its back end is built with the Spring Framework to support a website front end for administrative purposes. It is designed to support the creation of custom EMC applications by allowing an administrator to have control over most of the aspects that compose a mobile application. The mobile front end is using Cordova, AngularJS and Ionic, which allows for the usage of standard web technologies and works as a container that interfaces between them and the native device capabilities while also supporting activity-awareness capabilities based on the Core Motion framework offered by the OS environment.

In order to understand crowdsourcing use in an enterprise environment, we have been using ECrowd for several experiments, building promising results as new features into subsequent versions of the platform. We have been looking into various topics relevant to enterprise crowdsourcing: comparing nudging strategies [2], gamification elements to investigate user engagement [1], user modeling (ongoing), and the privacy aspects of data handling in enterprise crowdsourcing [4].

To study engagement, we have been conducting experiments at several large Dutch organisations, with several hundred participants from various departments, for a duration of typically two months. Participants were asked to install appropriate versions of the application that a) were generated to match the design guidelines of the company, b) contained several tasks types to achieve a good level of task diversity, and c) corresponded in functionality to the research questions being addressed.

We also conducted experiments on probable causal relationship between different gamification elements and the quality and user engagement in enterprise crowdsourcing. We were interested in how different enterprise environments mediate the effect. We deployed qualitative research procedures to gain an understanding of the player types that exist in an enterprise environment and also discover a possible enterprise application for which crowdsourcing could be used. Along with the player types, we complement our findings with previous research done within IBM [3], to motivate and validate the selection of the gamification elements which will be used for our experimental study.

3 Results and Conclusion

The engagement study provided novel insights about: 1) the time slots and context when employees are more likely to interact with their mobile devices and perform crowd work; and 2) the type of crowd work that employees would be willing to perform during a working day. The results of our survey confirm the result of other exploratory studies performed in other environments and companies, but give additional insights on the types and duration of tasks preferred by employees. ECrowd enabled the implementation of notification strategies with varying temporal distribution and activity awareness. We found that timely notifications can foster participation and retention, that there were significant differences among treatment groups in terms of participation and retention. The outcome of the experiment was in contrast with the survey results, as activity awareness lead to lower participation and retention.

In the gamification experiments, we found that based on Bartles theory of player types, the qualitative exploratory analysis affirmed for the enterprise environment the suggestion of non mutually exclusive player type characteristics. More importantly, by combining our qualitative research results with those of previous studies on gamification in the enterprise, we were able to show the preference of employees in competitive and collaborative game dynamics. These results funneled our explanatory research, for which we deployed a gamified mobile crowdsourcing application which combines competitive and collaborative game mechanics. We used our experimental tool to verify its use in into two large multinational enterprises for an observational interval that lasted two months. Together with the engagement study this has shown that enterprise crowdsourcing is possible through mobile interaction and gamification.

Acknowledgements

ECrowd was made possible by our IBM graduate students Grigorios Afentoulidis, Sarah Bashirieh, Jenny Bern, Jordy Alblas, Misra Turp, Nikolay Polaynov and Robin de Jong, and by Anca Dumitrache and Oana Inel of the Vrije Universiteit, and Janna Kuijt of IBM.

References

- [1] Grigorios Afentoulidis. Gamification in enterprise crowdsourcing. *TU Delft, (Thesis)*, 2017.
- [2] Sarah Bashirieh, Sepideh Mesbah, Zoltán Szilávik, Robert-Jan Sips, Judith Redi, and Alessandro Bozzon. Nudge your workforce. a study on the effects of task notification strategies in enterprise mobile crowdsourcing. In *Proceedings of 25th International Conference on User Modelling, Adaptation and Personalisation. (UMAP 2017)*, Bratislava, Slovakia, July 2017. Springer.
- [3] Laurentiu Catalin Stanculescu, Alessandro Bozzon, Robert-Jan Sips, and Geert-Jan Houben. Work and play: An experiment in enterprise gamification. In *Proceedings of the 19th ACM Conference on Computer-Supported Cooperative Work & Social Computing*, pages 346–358. ACM, 2016.
- [4] Camiel Steenstra. A privacy aware infrastructure for an inclusive enterprise at ibm. 2016.
- [5] Benjamin Timmermans, Lora Aroyo, and Chris Welty. Crowdsourcing ground truth for question answering using crowdtruth. In *ACM Web Science 2015*. ACM, 2015.
- [6] Maja Vukovic and Claudio Bartolini. Towards a research agenda for enterprise crowdsourcing. *Leveraging applications of formal methods, verification, and validation*, pages 425–434, 2010.

Intelligent News Conversation with the Pepper Robot

Jonathan Gerbscheid, Thomas Groot, and Arnoud Visser

*Intelligent Robotics Lab
University of Amsterdam*

Abstract. The UvA@Home Team develops social behaviours for home-assistance robots. Beside the use of classical techniques for human robot interaction, such as speech recognition, image recognition and language processing, the team strives to explore new means of improving social interactions. To showcase how the inclusion of new fields can improve social behaviour a news-conversation agent will be demonstrated.

1 Introduction

Social behaviour is a complex phenomenon and reproducing it on a robot is quite a challenge. Human-like conversation is an important part of social behaviour and requires techniques from a wide range of fields. An especially difficult part of conversation and therefore language processing is the knowledge that is required to engage in it. Humans expect other humans to be aware of general events that are happening in the world. If we want robots to be able to converse like humans they must clearly also have some knowledge of current events [1]. It is to this end that the news conversation module was developed and implemented on the Pepper robot.

The news conversation module retrieves news articles from several sites and parses them to extract the general topics of the article. This knowledge is used to enable users to ask the robot to give them articles on certain topics. Additionally a preference profile of the user is made to in order to occasionally present the user with articles that they might find interesting.

2 RoboCup and SSPL

The RoboCup was originally founded in 1977 with the goal of developing fully autonomous humanoid soccer robots. However it has since branched out into many different forms of intelligent robotics. The goal of the @Home league is assisting humans in a home environment [2]. The UvA@Home¹ team participates in the Social Standard Platform League (SSPL) of the @Home League, which uses the Pepper robot and focuses on the social interaction aspect of home assistance, such as reminding users of appointments or take orders during a

¹ `uvahome.nl`

party [3]. This league is different compared to the other @Home leagues which put more emphasis on assisting in physical tasks, such as retrieving items or opening doors. In 2017 the UvA@Home team participated in the RoboCup for the first time and plans to return for the 2018 competition.

3 Demonstration

The system that will be demonstrated is able to inform a user of news articles with an opinionated undertone [4]. To start interaction, a person first stands in front of the robot, after which the system creates a user profile that is associated with a person by learning that persons face. After this the system will ask the user a series of questions regarding their news preference.

The system collects news from a variety of popular news websites (Reuters, NYT, BBC) and can return articles based on queries made by the user. During conversation, the person can give feedback on specific queries. The system can use this feedback to update the user profile in order to make the results of subsequent queries more relevant to the user.

The system will present its opinion on popular topics at random intervals to create a more natural conversation. Its opinion will be based on a consensus created by gathering posts from Twitter² that include the topic. The consensus is determined by doing sentiment analysis on the aggregated posts.

4 System summary

The goal of the system is to be a personal assistant that can inform a user about recent events that are relevant to the user. Conversations are made more natural by asking the user for feedback regarding the news articles found and by giving its opinions on popular topics. Because the interface is a robot, interacting with the system is straightforward as no interface is required. The system can be seen working on a Nao robot in the qualification video of the UvA@Home team for the Robocup 2017³.

References

1. Sloman, A.: Some requirements for human-like robots. In: *Creating Brain-Like Intelligence: From Basic Principles to Complex Intelligent Systems*, Berlin, Heidelberg, Springer Berlin Heidelberg (2009) 248–277
2. Stückler, J., et al.: Increasing flexibility of mobile manipulation and intuitive human-robot interaction in robocup @home. In: *Robot Soccer World Cup*, Springer (2013) 135–146
3. Visser, A.: A new robocup @home challenge. *Benelux A.I. Newsletter* **31**(1) (2017)
4. Gerbscheid, J., Groot, T., Wessels, J., Wever, R., Woerkom, W.V.: Personalized news conversations with the softbank pepper. project report, Universiteit van Amsterdam (March 2017)

² <https://twitter.com/>

³ <https://www.youtube.com/watch?v=-i8xgfzAFoQ>

Part VI

Type D: Thesis abstracts
Oral presentation

Modelling the Generation and Retrieval of Word Associations with Word Embeddings

A Case Study for a Guesser Agent in the Location Taboo Game

B.Sc. Thesis Abstract

Verna Dankers, Aysenur Bilgin, Raquel Fernández

Institute for Logic, Language and Computation, University of Amsterdam

1 Introduction

When words occur in close proximity frequently, an associative link is formed in the observer (Clark, 1970). Distributional Semantic Models (DSM) can be used for the artificial acquisition of associative links (Heath et al., 2013) and learn word representations from word co-occurrence patterns in a corpus.

In this thesis, a method for modelling word associations is developed by creating an Artificial Guesser Agent (AGA) for the Location Taboo Game (LTG).¹ In this word-guessing game, simple textual clues about a well-known target city are provided, and the AGA should guess the city. The AGA is evaluated through games that were successfully played by humans and should be able to mimic the associations that humans have with cities. The proposed AGA employs a semantic vector space that is created using context-predicting DSMs from a tailored corpus about travel destinations. The training of the vector space uses a novel annotation method to strengthen the associative link between the cities and their descriptions in the corpus. Four different game-playing strategies are implemented to retrieve guesses from the vector space, and several lists of candidate cities are compiled to only consider sufficiently relevant places.

2 Approach

Semantic Vector Space A tailored corpus was constructed from Wikipedia and Wikivoyage pages entitled with a country or city name and Wikivoyage outlinks to other Wikivoyage pages. A targeted annotation method is proposed to make the association between a geographical location and the content of its corresponding pages explicit for the DSM. For the pages entitled with a country or city name, the name is inserted in every sentence, for example: ‘*pizza traditionally eaten locally pasta **Verona** dishes feature widely restaurant menus*’.

The context-predicting DSMs employed are Continuous Bag of Words and Skip-Gram, combined with the Hierarchical Softmax and Negative Sampling algorithms (Mikolov et al., 2013a, 2013b). The similarity metrics used are Chebyshev, Correlation, Cosine and Tanimoto.

Candidate Cities Three lists of 500 candidate cities were constructed, according to the occurrence counts from the tailored corpus (list 1), the occurrence counts from a Google News dataset² (list 2) and the number of visits as listed on NomadList³ (list 3). A fourth list was created containing only the 120 cities present in example games to investigate the AGA’s best-case performance.

Game-Playing Strategy Four game-playing strategies were implemented: Using the cumulative similarity between vectors of clue words and vectors of cities (strategy 1). Eliminating inconsistent

¹ This study is partially supported by the Marie Curie Initial Training Network (ITN) ESSENCE, grant agreement no. 607062.

The LTG has been designed by ESSENCE: <https://www.essence-network.com/challenge/>.

² The vectors are available at: <https://code.google.com/archive/p/word2vec/>.

³ <https://nomadlist.com>

cities across game iterations (strategy 2). Narrowing down to a set of countries and choosing from the cities of those countries (strategy 3). Summing the vectors of all clue words to guess the nearest city in the vector space (strategy 4).

3 Experiments and Results

We have conducted several experiments to investigate the impact of hyper-parameter configuration for the DSMs and the AGA using 162 example games. Additionally, forty games were set aside for the final evaluation. Strategy 1 was the most appropriate game-playing strategy. The Tanimoto, Correlation and Cosine similarity metrics all resulted in very similar performance. Out of the first three lists of candidate cities, list 3 resulted in the highest performance. The targeted corpus annotation had a significant effect for all models (two-sided relative t -test, $p < 0.001$).

For the performance evaluation of the AGA three evaluation metrics were used: the game score, the accuracy and the faster-guessing performance (FGP). Skip-Gram Hierarchical Softmax (SGHS) was the best performing algorithm, for which the results are shown in Table 1, together with the results of the baseline architecture introduced by Adrian et al. (2016).

Table 1: The performance evaluation for SGHS and the baseline architecture. The game score is the number of submitted guesses, plus a penalty of 5 per unsuccessful game. The FGP represents the percentage of successful games for which the AGA submitted fewer guesses than the human.

Corpus	Algorithm	Game Score	Accuracy (%)	FGP (%)
Unannotated	SGHS	242	25.00	50.00
Annotated	SGHS	239	27.50	27.27
-	Baseline Architecture	290	5.00	0.00

4 Conclusion and Future Work

In this thesis, an AGA architecture is presented, which uses context-predicting DSMs to infer word associations from a tailored corpus and employs different game-playing strategies for the LTG. The architecture is an improvement compared to the baseline of Adrian et al. (2016) and can guess target cities with up to 27.50% accuracy. The presented method for the generation and retrieval of word associations could be generalised for different domains or adapted for different tasks. Targeted corpus annotation is proposed to amplify the strength of the associations implicitly present in the corpus and it significantly improves the performance in the LTG.

Regarding future work, multiple types of resources can be combined to improve the corpus content and the list of candidate cities. Furthermore, more complex architectures could be created, such as a multi-agent system or a system that can better interpret multi-word clues.

References

- Adrian, K., Bilgin, A. & Van Eecke, P. (2016). A semantic distance based architecture for a guesser agent in ESSENCES location taboo challenge. *DIVERSITY@ ECAI 2016*, 33–39.
- Clark, H. H. (1970). Word associations and linguistic theory. *New horizons in linguistics*, 1, 271–286.
- Heath, D., Norton, D., Ringger, E. & Ventura, D. (2013). Semantic models as a combination of free association norms and corpus-based correlations. In *Seventh international conference on semantic computing* (pp. 48–55). doi: [10.1109/ICSC.2013.18](https://doi.org/10.1109/ICSC.2013.18)
- Mikolov, T., Chen, K., Corrado, G. & Dean, J. (2013a). Efficient estimation of word representations in vector space. In *Proceedings of Workshop at the ICLR*.
- Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S. & Dean, J. (2013b). Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems* (pp. 3111–3119).

The Effect of Tutor Feedback in Language Acquisition Models

Jens Nevens

Promotor: Dr. Katrien Beuls

Advisor: Dr. Michael Spranger

Vrije Universiteit Brussel, Pleinlaan 2, 1050 Elsene

`jens.nevens@vub.ac.be`

`katrien.beuls@vub.ac.be`

Abstract

This work investigates the role of tutor feedback in agent-based models of lexicon acquisition where one agent (the tutor) is teaching another agent (the learner) a particular language. An important aspect of lexicon learning is referent resolution, i.e. understanding what object in the environment is referred to by a given word. We compare two dominant paradigms in this respect: *interactive learning* and *cross-situational learning* - which differ primarily in the role of social feedback, such as pointing or gaze, in solving the referent uncertainty problem. In the former, the learner not only observes objects and words, but also receives social, pragmatic feedback. This clearly restricts the number of referents of the words. In the latter, the learner only receives objects and words. Here, the referent uncertainty problem is solved either by storing a single hypothesised referent, until evidence for the contrary is presented, or by statistical analysis of the co-occurrences of objects and words. We opt for the latter approach as it seems to align well with empirical findings.

Almost all models in lexicon learning fall into either of these two categories. However, little work has been done to compare these two paradigms systematically, except for Belpaeme and Morse (2012), who show that cross-situational learning is slower in the continuous-meaning domain due to the absence of social feedback. Importantly, real life interactions between caregiver and child probably do not fall into either of the two categories but are really combinations of full, little or no social feedback. Consequently, learning algorithms must be able to deal with situations where there is feedback interleaved with interactions where there is no feedback. In this work, we propose a new *mixed* paradigm that combines the two paradigms. This new paradigm allows to test algorithms in experiments that combine no feedback and social feedback. We control the presence or absence of social feedback in consequent interactions.

To deal with this mixed feedback setting, we extend existing learning algorithms in a new way and show how they perform with respect to the KNN approach of Belpaeme and Morse (2012) and other prototype-based approaches. Our learning algorithms use prototypes to estimate the referent shown by the tutor during subsequent training interactions. The algorithms differ in the way their

internal prototypes are updated in the presence and absence of social feedback. Our prototypes are continuous-valued and multi-dimensional and can represent various sensorimotor spaces – in our case colours. After training, we measure the communicative abilities of the learner, both in language production and language understanding, by a number of testing interactions. The algorithms are evaluated first in a purely simulated environment where the objects are randomly generated. Afterwards, a grounded environment is used. This was created in other work by humanoid robots observing scenes with physical objects. These scenes emulate a more structured and realistic learning environment. We also modify the distribution of the object’s features (colours) in the grounded environment and investigate how the agents respond. In both environments, the agents are situated in contexts of multiple objects and use single words to refer to an entire object. Finally, we perform a study on a large number of parameters that all influence some aspect of language acquisition, e.g. context size, size of the tutor lexicon, total number of objects in the world and word production strategy of the tutor. We investigate how the effects of these parameters change with respect to social feedback and the proposed learning algorithms.

Our results suggest that the effect of social feedback is double. Not only does it allow the agent to learn faster, i.e. to reach the same level of success needing less interactions, the communicative abilities of the agent also become better with increasing presence of social feedback. Furthermore, we observe how the communicative success of the agents scales with respect to the social feedback in the novel mixed feedback setting. This is, however, dependent on the learning algorithm as some algorithms benefit from the presence of only some feedback more than others. Concerning the grounded world, our results paint a mixed picture. Performance in grounded worlds can be better than in simulated worlds, but this depends on the combination of algorithm and statistics of the environment. The agents are able to benefit from the additional structure present in the more realistic, grounded environments, but require at least some social feedback to do so. Finally, we deem it important not to underestimate the role of the tutor in language acquisition models and in language learning in general. In our experiments, we found that the way the tutor structures the world for the learner (i.e. the tutor’s word production strategy) can have a profound impact on the learner’s communicative success. This is especially true in the absence of social feedback.

Is Mirror Descent a Special Case of Exponential Weights?

Dirk van der Hoeven, supervisor: dr. Tim van Erven

Leiden University

Online Convex Optimization (OCO) is a sequential prediction setting that proceeds in rounds $t = 1, \dots, T$ in which a forecaster is to predict an unknown sequence of elements $\mathbf{w}_1, \dots, \mathbf{w}_T \in S$, where S is a convex set. In each round the forecaster suffers a convex loss $\ell_t(\mathbf{w}_t) : \mathbb{R}^d \rightarrow \mathbb{R}$, which accumulates over rounds. After T rounds the performance of the forecaster is measured by the regret $\mathcal{R}_T = \sum_{t=1}^T \ell_t(\mathbf{w}_t) - \min_{\mathbf{w}} \sum_{t=1}^T \ell_t(\mathbf{w})$, which compares the accumulated losses to an offline minimum. Under appropriate conditions it is possible to obtain a regret that is bounded by $O(\sqrt{T})$. We focus on three algorithms that obtain such a bound: Online Gradient Descent, Mirror Descent, and Exponential Weights. In literature the Online Gradient Descent and Exponential Weights are known as special cases of Mirror Descent. However, [1] observed that Online Gradient Descent can be seen as a special case of Exponential Weights. This raises the following question: are other mirror descent algorithms also special cases of Exponential Weights?

In the OCO setting the usual approach is to approximate the loss function with a first order approximation: $\hat{\ell}_t(\mathbf{w}_t) = \langle \mathbf{w}_t, \nabla \ell_t(\mathbf{w}_t) \rangle$, where $\nabla \ell_t(\mathbf{w}_t)$ is the gradient of ℓ_t evaluated at \mathbf{w}_t . We then run an instance of the Mirror Descent algorithm on $\hat{\ell}$, which predicts each \mathbf{w}_{t+1} as follows:

$$\mathbf{w}_{t+1} = \nabla F \left(\nabla F^*(\mathbf{w}_t) - \eta \nabla \ell_t(\mathbf{w}_t) \right), \quad (1)$$

where η is the learning rate, F^* is a Legendre function, and F is the convex conjugate of F^* , F . Note that ∇F and ∇F^* are inverses of each other. A special case of Mirror Descent is Online Gradient Descent, which is recovered for $F^*(\mathbf{w}) = \frac{1}{2} \|\mathbf{w}\|_2^2$.

Another special case of Mirror Descent is Exponential weights, which is usually run in a subsetting of OCO, Prediction With Expert Advice (PWEA). In the PWEA setting each round a set of K experts give a prediction. In each round the forecaster provides a probability distribution p_t over these experts and predicts with the mean over the expert predictions. The loss of the forecaster is then the expected loss of the experts: $\sum_{k=1}^K p_t(k) \ell_t^k$. Exponential Weights predicts the following probability distribution:

$$p_t(k) = \frac{\pi(k) \exp(-\eta \sum_{i=1}^{t-1} \ell_i^k)}{\sum_{k=1}^K \pi(k) \exp(-\eta \sum_{i=1}^{t-1} \ell_i^k)}, \quad (2)$$

where π is a prior distribution on the experts, usually taken to be uniform.

However, one can also play Exponential Weights with a non-uniform prior on a continuous set of experts, parametrized by \mathbf{z} . Let $\mathcal{E} = \{p(\mathbf{z}) = e^{\langle \mathbf{z}, \boldsymbol{\theta} \rangle - F(\boldsymbol{\theta})} K(\mathbf{z}) | \boldsymbol{\theta} \in \Theta\}$ be an exponential family with cumulant generating function F , sufficient statistic \mathbf{z} , and carrier $K(\mathbf{z})$. With expert loss function $\ell^{\mathbf{z}} = \langle \mathbf{z}, \nabla \ell_t(\mathbf{w}_t) \rangle$ and a prior from an exponential family we obtain the main result.

Theorem 1. *Let p_{t+1} be the Exponential Weights distribution at time $t+1$ with prior π , let the loss of expert \mathbf{z} be $\ell_t^{\mathbf{z}} = \langle \mathbf{z}, \nabla \ell_t(\mathbf{w}_t) \rangle$, let the forecasters loss be $\hat{\ell}_t = \mathbb{E}_{\mathbf{z} \sim p_{t+1}} [\langle \mathbf{z}, \nabla \ell_t(\mathbf{w}_t) \rangle]$, and let F be the cumulant generating function of \mathcal{E}_π . Let Mirror Descent be used with F^* , the convex conjugate of cumulant generating function F . Then the Mirror Descent algorithm is the mean of the Exponential Weights algorithm:*

$$\mathbb{E}_{\mathbf{z} \sim p_{t+1}} [\mathbf{z}] = \mathbf{w}_{t+1} = \nabla F \left(\nabla F^*(\mathbf{w}_t) - \eta \nabla \ell_t(\mathbf{w}_t) \right). \quad (3)$$

We provide a proof sketch. Let π be a member of an exponential family with natural parameter $\boldsymbol{\theta}_\pi$. For $\ell_i^{\mathbf{z}} = \langle \mathbf{z}, \nabla \ell_i(\mathbf{w}_i) \rangle$ the update step for the EW algorithm is:

$$\begin{aligned} p_{t+1}(\mathbf{z}) &= \frac{\pi(\mathbf{z}) \exp(-\eta \sum_{i=1}^t \langle \mathbf{z}, \nabla \ell_i(\mathbf{w}_i) \rangle)}{\int_{\mathbb{R}^d} \pi(\mathbf{z}) \exp(-\eta \sum_{i=1}^t \langle \mathbf{z}, \nabla \ell_i(\mathbf{w}_i) \rangle) d\mathbf{z}} \\ &= \frac{\exp(-F_\pi(\boldsymbol{\theta}_\pi) + \langle \boldsymbol{\theta}_\pi, \mathbf{z} \rangle - \eta \sum_{i=1}^t \langle \mathbf{z}, \nabla \ell_i(\mathbf{w}_i) \rangle K(\mathbf{z}))}{\exp(F_\pi(\boldsymbol{\theta}_\pi - \eta \sum_{i=1}^t \nabla \ell_i(\mathbf{w}_i)) - F_\pi(\boldsymbol{\theta}_\pi))} \\ &= \exp(-F_\pi(\boldsymbol{\theta}_{p_{t+1}}) + \langle \boldsymbol{\theta}_{p_{t+1}}, \mathbf{z} \rangle) K(\mathbf{z}). \end{aligned} \quad (4)$$

Hence, p_{t+1} is a member of the exponential family with cumulant generating function F and natural parameter $\boldsymbol{\theta}_{p_{t+1}} = \boldsymbol{\theta}_\pi - \eta \sum_{i=1}^t \nabla \ell_i(\mathbf{w}_i)$. Using convex conjugacy and a standard property of exponential families we find that the mean is equal to the Mirror Descent prediction:

$$\begin{aligned} \boldsymbol{\mu}_{p_{t+1}} &= \nabla F(\boldsymbol{\theta}_{p_{t+1}}) \\ &= \nabla F(\nabla F^*(\mathbf{w}_t) - \eta \nabla \ell_t(\mathbf{w}_t)) \end{aligned} \quad (5)$$

For a formal proof see Theorem 3 in [2]. Theorem 1 gives a large class of algorithms a new interpretation: the update step is updating a distribution and taking the mean of said distribution as the weights for the coming round. For instance, Online Gradient Descent is equal to Exponential weights with a Gaussian prior and predicting with the mean of the posterior. This new interpretation of Mirror Descent opens a route to find new algorithms in the OCO setting by exporting ideas from PWEA to OCO. Another possibility is sampling from the posterior, which is useful in the linear bandit optimization setting.

References

- [1] Koolen, W.: Gradient descent as Exponential Weights. Blog February 21: <http://blog.wouterkoolen.info/GDasEW/post.html/> (2016)
- [2] Van der Hoeven, D.: Is Mirror Descent a special case of Exponential Weights? MSC Thesis. Available from: <http://pub.math.leidenuniv.nl/~hoevendvander/> (2016)

Modelling Word Associations and Interactiveness for Describer Agents in Word-Guessing Games

A Case Study for the Location Taboo Game
B.Sc. Honours Thesis Abstract

Verna Dankers, Aysenur Bilgin, Raquel Fernández

Institute for Logic, Language and Computation, University of Amsterdam

1 Introduction

In word-guessing games, one player describes a stimulus and his partner should guess what the stimulus is. Producing associations that would allow the partner to guess the target correctly requires an associative mechanism and modelling a shared context (Clark, 1970).

The Location Taboo Game¹ (LTG) is a cooperative word-guessing game, in which a *describer* provides short textual clues about a target city and the *guesser* should guess this city. The clues should not contain terms from a list of taboo words. In this thesis, an architecture for an Artificial Describer Agent (ADA) is presented and evaluated in simulation with an artificial guesser, as well as in a study with human guessers. The describer’s objective is to elicit a correct guess by producing associations that are recognisable for the guesser. The ADA extracts word associations from a semantic vector space that has been created with a context-predicting distributional semantic model. To create *guess-independent* clues, several methods for extracting general associations for target cities are compared. Regarding the generation of *guess-dependent* clues, which add interactiveness to the game, a rule-based approach is proposed.

2 Approach

Guess-Independent Clues The first approach for clue generation assumes that the taboo words are strongly associated with the target city. For each taboo word the neighbour from the vector space that is closest to both the target city and the taboo word is selected as a clue (strategy 1). The second strategy builds upon this, as it uses neighbours from a vector space of Wordnet’s synonym sets in addition, created with the AutoExtend algorithm (Rothe & Schütze, 2015).

The second approach uses example games to infer associations for a new target city. Clues from 100 games were clustered into 10 groups, representing 10 clue categories, such as *food* or *history*. For every word within a cluster, the corresponding city was used in an analogy with the new target city to find a candidate clue – e.g. for the target city *Prague* the analogy with *Venice* and ‘*pasta*’ gives the candidate clue ‘*potato salad*’. This resulted in many candidate clues per cluster. Two metrics were used to choose one clue per cluster, resulting in two variants of this approach: the inverse document frequency (strategy 3) and the Cosine similarity (strategy 4).

For all strategies, the order of presentation is based on a clue’s Cosine similarity to the target.

Guess-Dependent Clues The geographical aspect of guesses is used to create the interactive game-playing behaviour. To simplify the notion of dependency, a guess-dependent clue is only based on the last guess given. The following guess-dependent clues can be given: ‘*different continent*’, ‘*different country*’, ‘*close*’ (within 100 kilometres of the target) and cardinal directions. Twenty percent of the describer agent’s responses were made guess-dependent, to approximate the percentage of guess-dependent clues found in example games.

¹ This study is partially supported by the Marie Curie Initial Training Network (ITN) ESSENCE, grant agreement no. 607062.

The LTG has been designed by ESSENCE: <https://www.essence-network.com/challenge/>.

3 Experiments and Results

Artificial Guesser The Artificial Guesser Agent architecture presented by Dankers (2017) was used to evaluate the ADA in simulation. The vectors employed were the Wikipedia and Wikivoyage (Wiki*) vectors tailored for the LTG (Dankers, 2017) and the pre-trained Google News vectors,² created with the context-predicting distributional semantic models presented by Mikolov et al. (2013). The Wiki* vectors were retrained to include unigrams, bigrams and trigrams. ADAs employing the four game-playing strategies were evaluated according to their accuracy (Table 1).

Human Guessers To evaluate the ADA with human guessers, 36 games about European target cities were made available in an experiment. There were 9 games per strategy. Three types of questionnaires about the participants’ geographical knowledge, familiarity with the target cities and ratings of the performance of the ADA were filled out by 9 participants. Overall, the ADAs could elicit a correct guess for 37.86% of all games played, and for 50.00% of the games in which the participants were familiar with the target city. On average, the use of natural language was valued with a 5.73 and the collaborative behaviour of the ADA was valued with a 5.90, on a scale from 1 to 10. For more detailed results per strategy, only the results for participants who indicated that they were familiar with the target city were taken into account (Table 1).

Table 1: The average performance per strategy. The accuracy is the percentage of games in which the target city was guessed correctly. The ratings express the usefulness of the clues according to the human guessers.

Strategy	Artificial Guesser	Human Guessers	
	Accuracy (%)	Accuracy (%)	Rating (1-10)
1	28.21	59.52	6.89
2	30.77	47.72	6.40
3	14.53	42.85	4.98
4	23.93	54.84	6.54

4 Conclusion and Future Work

In this thesis, methods for extracting word associations from a vector space have been applied to the creation of an ADA for the LTG. The ADA has been evaluated in simulation as well as in a study with human guessers. In the study, the proposed architectures could elicit a correct guess for 37.86% of all games, and for 50.00% of the games in which participants were familiar with the target city. A rule-based approach for modelling interactiveness has been proposed that allows the ADA to express relative geographical relations, depending on the last guess.

Regarding future work, dependent clues can be improved through modelling their helpfulness or adding different types. Secondly, the order in which the clues are presented could be improved. Thirdly, future work could include modelling individual word spaces that better capture an individual’s knowledge.

References

- Clark, H. H. (1970). Word associations and linguistic theory. *New horizons in linguistics*, 1, 271–286.
- Dankers, V. (2017). *Modelling the generation and retrieval of word associations with word embeddings* (Bachelor’s Thesis). University of Amsterdam.
- Mikolov, T., Chen, K., Corrado, G. & Dean, J. (2013). Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.
- Rothe, S. & Schütze, H. (2015). Autoextend: Extending word embeddings to embeddings for *synsets* and *lexemes*. *arXiv preprint arXiv:1507.01127*.

² The vectors are available at: <https://code.google.com/archive/p/word2vec/>.

Catch Them If You Can: Malicious Behavior Simulation in Deep Question Answering

Nikita Galinkin

Supervisors: Zoltán Szilávik¹, Lora Aroyo² and Benjamin Timmermans¹

¹ *IBM Benelux Center for Advanced Studies, Amsterdam*

² *Vrije Universiteit, Amsterdam*

1 Introduction

Recent advances in artificial intelligence and machine learning have allowed question answering systems to become much more prominent for retrieving information to handle day-to-day tasks. In the study, we investigate the impact of malicious user behavior in question answering systems specifically deployed in the cultural heritage domain. We need to be prepared for some users being ‘malicious’, trying to render a learning system useless by misusing it. To prepare, first we need to estimate the impact of malicious actions, and study ways to deal with this issue.

Recently, the problem of malicious attacks has been broadly studied in the machine learning community [3]. It has been shown that in some cases 20% of malicious data can lead to a ten-fold increase in classification errors [5]. The main trend in these studies is to investigate the worst-case scenario of the machine learning model with the assumption that the attacker has full knowledge of the system and its actions are optimal. This assumption rarely holds in a system used in real-life. In this study, we propose a user model to simulate users attacking the system to gain a better indication of what might happen.

2 SQALPEL: A Deep Art History Question Answering System

As a use case to study the impact of malicious feedback on question answering systems we use the SQALPEL system, developed during the like named project. The project is a collaboration between the IBM Benelux Center for Advanced Studies in Amsterdam, the Mauritshuis museum in Den Haag, and the Vrije Universiteit Amsterdam. The objective of the project is to make use of new techniques to answer and raise new questions about the subjects of a painting, relevant historical context and changing interpretations over the course of time in the art historical literature.

During the project, we have developed SQALPEL, a deep art history question answering system that is designed to answer questions about Rembrandt’s famous painting “The Anatomy Lesson of Dr. Nicolaes Tulp”. The painting, owned by the Mauritshuis, was chosen for this project by the museum as it has been widely studied in an art historical context over the course of time (see, e.g., [1]). The SQALPEL system is built on IBM Watson technology¹, relying on services such as the Natural Language Classifier, Retrieve and Rank, and Conversation, to engage the user by answering questions in conversational manner. The system relies on previously created question-answer pairs (i.e., via crowdsourcing) as well as on automatically extracted document passages retrievable by the system. End user input for training the system is taken into account in two ways: primarily, we employ a thumbs up/down functionality when users receive answers, but we also process never before seen questions after expert validation. This paper considers the former kind of user input.

¹<https://www.ibm.com/watson/products-services/>

3 Methodology for Malicious User Behavior

The data for this study contains 4037 question-answer pairs, consisting of 3797 unique questions and 554 unique answers. Of the answers, 131 were collected from interviews with museum visitors, the rest we have extracted from literature about the painting. Answers from both sources were enriched with questions through the crowdsourcing platform called CrowdFlower, where human workers were given the answer, and they had to come up with a diverse set of questions that match the answer within the task. Questions collected from the interviews are more frequently asked, thus, on average, there are four times more questions to each interview answer.

We simulate two weeks of operation with 200 users daily, each user simulated according to user models with various percentages of maliciousness expressed as providing the right or wrong feedback to answers given to questions with known answers in the knowledge base. The classifier we built was inspired by the question answering evaluation metric POURPRE that measures the co-occurrence of words between a given answer and an answer nugget. An answer nugget is defined as a string containing a fact for which the assessor could make a binary decision as to whether a response contained that nugget [4]. In the crowdsourcing task aiming to collect questions, users were asked to highlight the part of the answer that justifies the answer to the question. These annotations we use as the answer nugget.

4 Results and Conclusion

We found that with up until 20% of malicious feedback the system continues to learn, and system performance does not drop below the level of the initial system. With over 30% of the users being malicious, system performance will gradually decrease. With 60% of malicious feedback it drops by a factor of two after only two weeks. When a peak of low quality feedback appears, we investigated whether the timing of the peak makes a difference. The results indicate that a peak in malicious activity, in addition to negatively influencing system performance, creates a long term effect. The timing of the peak defines its influence on system performance. The later the peak occurs, the less effect it will have, as by now the system is expected to be more robust.

For somebody who plans to deploy a question answering system with limited technical staff in-house, the results of our study indicate that user behavior and misbehavior have to be watched closely in the first weeks of system operation, as the potentially negative effect on overall system performance can be severe. When the system is retrained with new data, it might indicate overall level of maliciousness. It can then be decided on when to restore the system to a previous state, discarding data with too high level of low quality input. For user-level filtering, we might consider employing various agreement-based metrics such as those, for example, offered by CrowdTruth [2].

References

- [1] Arnon Afek, Tal Friedman, Chen Kugel, Iris Barshack, and Doron J Lurie. Dr. tulip’s anatomy lesson by rembrandt: the third day hypothesis. 2009.
- [2] Lora Aroyo and Chris Welty. The three sides of crowdtruth. *Journal of Human Computation*, 1:31–34, 2014.
- [3] Marco Barreno, Peter L. Bartlett, Fuching Jack Chi, Anthony D. Joseph, Blaine Nelson, Benjamin I.P. Rubinstein, Udam Saini, and J. D. Tygar. Open problems in the security of learning. In *Proceedings of the 1st ACM Workshop on Workshop on AISec*, AISec ’08, pages 19–26, New York, NY, USA, 2008. ACM.
- [4] Jimmy J. Lin and Dina Demner-Fushman. Methods for automatically evaluating answers to complex questions. *Information Retrieval*, 9(5):565–587, 2006.
- [5] Huang Xiao, Battista Biggio, Gavin Brown, Giorgio Fumera, Claudia Eckert, and Fabio Roli. Is feature selection secure against training data poisoning? In *ICML - Volume 37*, ICML’15, pages 1689–1698. JMLR.org, 2015.

Part VII

Type D: Thesis abstracts
Poster presentation

‘Well, at least it tried’ The Role of Intentions and Outcomes in Ethically Evaluating Robot Actions¹

Daphne Lenders^a and Willem F.G. Haselager^a

^a Dpt. of Artificial Intelligence, Radboud University, Nijmegen, The Netherlands
d.lenders@student.ru.nl

Abstract. In order to make robots more trustworthy, it is important to find out which factors influence a human’s ethical evaluation of a robot. We found that intentions of a robot have a larger positive effect than the outcomes of its actions. Moreover, the influence of outcomes on ethical evaluations is larger when the action was perceived to be based on a good rather than bad intention.

Keywords: Human-robot interaction, Moral judgement, Trust

1 Introduction

Two important factors in morally evaluating the actions of an agent are the intentions behind the actions as well as their outcomes. Current research has shown that humans are especially guided by the intentions of actors, when ethically evaluating humans [1]. As far as we know, no research has however been done on the role of intentions and outcomes in the ethical evaluation of robot actions.

In this study we investigated how the perception of intentions and outcomes related to robot actions influence ethical evaluations of those actions. This was done by showing participants video clips of a robot appearing to have a good or bad intention followed by an action that leads to a good or bad outcome. To each video clip the participants had to give an ethical evaluation of the displayed behaviour of the robot.

Due to the theory of anthropomorphism, which says that humans tend to view inanimate agents like they do humans, it was expected that robots actions would be evaluated similarly to human actions [2]: Perceived good intentions and outcomes were expected to have a positive effect on ethical evaluations, while bad intentions and outcomes were predicted to affect these evaluations negatively. Furthermore intentions were predicted to have a bigger influence on ethical evaluations than outcomes. Finally we expected to find an interaction effect between intentions and outcomes.

Methods

In an online survey, 75 participants were shown videos of 20-50 seconds displaying a Pepper robot that appeared to have either a good or bad intention followed by an action

¹ The full thesis was submitted in fulfilment of the requirements for the degree of Bachelor of Science in Artificial Intelligence at the Radboud University in June 2017

that lead to a good or bad outcome. In a good intention-bad outcome scenario it was e.g. shown how Pepper tried to submit a paper for its owner, but failed because its tablet crashed unexpectedly through which the owner missed an important deadline.

After each video the participants had to answer three questions that together constitute a moral evaluation of the displayed robot.² The moral evaluations were used in a two-way repeated measures ANOVA, in order to analyze the influence of ‘intention’ and ‘outcome’ on the moral evaluation of the observed robot behaviour.³

2 Results

Just as expected intentions ($F(1, 74) = 350.541$, $p = .000$, $\eta^2 = .826$) and outcomes ($F(1, 74) = 160.93$, $p = .000$, $\eta^2 = .685$) have large significant effects on the ethical evaluation of actions. Good intentions and outcomes affect ethical evaluations positively, while bad intentions and outcomes affect these evaluations negatively.

Next to that a large interaction effect between intention and outcome was found ($F(1, 74) = 49.637$, $p = .000$, $\eta^2 = .401$). The ethical evaluations of actions based on perceived good intentions are more influenced by the outcome of an action than actions based on perceived bad intentions.

3 Conclusion

Good intentions and outcomes affect ethical evaluations positively, but the influence of intentions is stronger than the one of outcomes. This suggests that it may be worthwhile to investigate what makes humans perceive intentions of robots as good or bad. Finding an answer to this may be an essential step towards making robots more trustworthy.

From the results it can also be concluded that humans ethically evaluate robot-actions similar to human-actions, which can be interpreted as further evidence for anthropomorphism: Characteristics that are normally attributed to humans, such as intentions, can also affect the way people view inanimate agents, such as robots [1, 2].

Other studies however found differences in ethical evaluations of human and robot actions [3]. Thus research on ethical evaluation of robots is still in its beginning stage and an overall framework on this topic cannot be established yet.

References

1. Young, L., Saxe, R.: The neural basis of belief encoding and integration in moral judgement. *NeuroImage* 40(4), 1912-1920 (2007).
2. Fink, J.: Anthropomorphism and human likeness in the design of robots and human-robot interaction. *International Conference on Social Robotics*, 199-208 (2012).
3. Malle, B. F., Scheutz, M., Arnold, T., Voiklis, J., Cusimano, C.: Sacrifice one for the good of many?: People apply different moral norms to human and robot agents. *Proceedings of the tenth annual ACM/IEEE international conference on human-robot interaction*, 117-124 (2015).

² Videos and complete survey can be found online: goo.gl/E48acM; goo.gl/P9pP79

³ In this abstract we chose to focus on our main study only. Additional analyses and their results will not be discussed here

Gamification for Learning by Modelling in Interactive Learning Environments

David Stap¹, Bert Bredeweg¹, and Natasa Brouwer²

¹ Informatics Institute, Faculty of Science, University of Amsterdam

² Education Service Centre, Faculty of Science, University of Amsterdam
{D.Stap,B.Bredeweg,N.Brouwer-Zupancic}@uva.nl

Abstract. Learning analytics aims to optimise learning, typically by providing students meaningful insight in their own learning behaviour. Gamification deploys game mechanics to increase motivation and thereby boost the learning process. In our work, we use learning analytics to implement game mechanics that create a motivating learning experience. The educational context concerns students that engage in model-building to develop systems thinking expertise. Three mechanics have been implemented: badges, leaderboard and life. The gamification add-on was evaluated during high school physics classes. Data mining showed that gamification resulted in significantly higher self-reported scores on enjoyment but inferior student-created models. A strong correlation between delete-behaviour and correctness of the created models was also found.

1 Introduction

Gamification implements game mechanics in non-game contexts to induce motivation [2, 4]. Mixed results are found regarding the effectiveness of gamification for education [5, 6]. Learning Analytics (LA) uses machine learning to mine student data and provide feedback (typically via dashboards) to help students improve their learning behaviour [3].

Learning by constructing models is an effective way to learn, but requires perseverance. The aim of our research is to increase the motivation of students engaging in modelling tasks by implementing game mechanics that utilise real-time student behaviour data. Instead of showing LA results via dashboards, this information is used to drive the mechanics. The mechanics are implemented for DynaLearn [1], an instrument for *learning by modelling* used in secondary education that focusses solely on *conceptual* models (as opposed to numerical).

2 Implementation and Evaluation study

The game mechanics are implemented using a norm model, which specifies the desired assignment outcome. Three mechanics are implemented: badges, leaderboard and life. *Badges* are awarded to students if their behaviour matches certain desired behaviours (milestones). The application of badges was optimised to ensure a constant stream of positive feedback while doing an assignment. For the *leaderboard*, the score gets calculated and updated every time a student performs an action. A maximum score is obtained by minimising total number of actions, maximising components consistent with the norm model, and minimising mistakes ($\text{score} = (\sqrt{\text{penalize_steps}} * \text{completeness} * \text{correctness}) *$

100). This ensures that the score reflects the desired goal: a model built precisely according to the specifications of the assignment results in a high score (note that, $penalize_steps = min_actions_norm/total_actions$). The goal of the *life* mechanic is to have students deliberately think about their next action, as opposed to adding components without much thought. If an action is inconsistent with the norm model, the student loses a life. Upon losing all lives, the game is over and the student has to restart on a blank canvas.

Two groups participated in the evaluation: the control group ($n = 11$), who used the unaltered DynaLearn environment, and the treatment group ($n = 24$), who used DynaLearn enhanced with the gamification module. Evaluation was performed comparing two high school physics classes, where students completed a physics assignment in DynaLearn. The evaluation consisted of tracking student's modelling behaviour data using LA to identify differences between the groups, and a survey that measured the situational awareness and motivation.

3 Results and Conclusion

The data resulting from student behaviour was mined after assignment completion. Analysis showed that the treatment group created lower quality models compared to the control group. Also, the treatment group performed a significantly lower number of delete actions compared to the control group. The total number of actions was also lower (albeit not significant) for the treatment group. The standard deviation for the total number of actions was relatively high in the treatment group, which indicates an uneven activity distribution in the group.

A positive correlation between delete-behaviour and correctness of the student-created models was found for both groups. Students that perform a higher number of delete actions are more likely to create superior models. Analysing the survey questions showed that the treatment group scored higher on Q3 ('Ik zou nog zo een les willen volgen.'), indicating that they enjoyed the game mechanics.

Overall, the data suggest that the gamification resulted in more fun but inferior student-created models, and hence less learning.

References

1. B. Bredeweg, J. Liem, W. Beek, F. Linnebank, J. Gracia, E. Lozano, M. Wißner, R. Bühling, P. Salles, R. Noble, A. Zitek, P. Borisova, and D. Mioduser. Dynalearn—an intelligent learning environment for learning conceptual knowledge. *AI Magazine*, 34(4):46–65, 2013.
2. S. Deterding, D. Dixon, R. Khaled, and L. Nacke. From game design elements to gamefulness: defining gamification. In *15th International Academic MindTrek Conference*, pages 9–15. ACM, 2011.
3. R. Ferguson. Learning analytics: drivers, developments and challenges. *International Journal of Technology Enhanced Learning*, 4(5-6):304–317, 2012.
4. I. Glover. Play as you learn: gamification as a technique for motivating learners. In J. Errington, A. Couros, and V. Irvine, editors, *World Conference on Educational Multimedia, Hypermedia and Telecommunications*, pages 1999–2008. AACE, 2013.
5. J. Hamari, J. Koivisto, and H. Sarsa. Does gamification work?—a literature review of empirical studies on gamification. In *47th Hawaii International Conference on System Sciences*, pages 3025–3034. IEEE, 2014.
6. K. Seaborn and D.I. Fels. Gamification in theory and action: A survey. *International Journal of Human-Computer Studies*, 74:14–31, 2015.

Neural Network Reuse in Deep RL for Autonomous Vehicles among Human Drivers

Manon Legrand*, Roxana Rădulescu, Diederik M. Roijers, and Ann Nowé

Vrije Universiteit Brussel, Artificial Intelligence Lab
`manon.legrand@live.be, {rradules, droijers, ann.nowe}@ai.vub.ac.be`

In this thesis, we consider the problem of reinforcement learning for autonomous cars on a highway. This is inherently a multi-agent problem, in which agents have to learn to handle the dynamics of a system, other autonomous cars, and be able to account for some irrationality on the part of other drivers (typically human). We create a simple traffic simulator composed of straight lanes and simulated human drivers (following a rule-based behavioral model with some irrationality built into it), and design different neural networks as learning models for our self-driving cars. Using this simulator, we try to answer the question on how we can efficiently train agents in such a complex multi-agent setting. Our key insight is that we can reuse neural networks trained on a single-agent version of a multi-agent problem to speed up learning, leading to good performance.

1 Learning

We first create a traffic simulator. Our simulator is a discrete implementation of a highway of straight lanes – where a lane is a list of cells – and human drivers who act according to a pre-defined behavior. Autonomous cars – cars whose actions are determined by a learning model – can then be added in this environment.

We then design 5 neural network models that use different information about the environment or use similar information differently. In short, we have 3 feed-forward network models that use, respectively, the presence of the cars¹ for the current time step, the presence for the current and the previous time steps, and the presence of cars and their speeds for the current time step. Moreover, we create 2 convolutional neural networks: the first uses the presence and the speed of cars as 2-D matrices, and the second uses the presence of cars for the current and previous time steps as a 3-D matrix.

First, all models are trained with two hidden layers with a varying number of neurons in a single-agent setting. The training is done with deep Q -learning [2]: that is, the neural network serves as a function approximator for the Q -values and is trained with experience replay.

Afterwards, the same models are trained in a multi-agent setting. The first set of experiments consists in exactly the same setting as before but with more

* This work was carried out for the first author's master thesis studies, at VUB [1]. The other authors were the supervisors. Full work available at: http://ai.vub.ac.be/sites/default/files/thesis_legrand.pdf

¹ And other information about the learning agent itself that is used by all models

than one agent at the same time on the highway. We then repeat the same experiments but with a modified version of the models – models that include in the inputs whether the observed cars are autonomous. Finally, we take advantage of the fact that we have models already trained, in a single-agent environment, and use these trained models as the starting point for the multi-agent learning. This has a lot of advantages, as the trained network already contains most of the information needed to learn to behave optimally in the multi-agent setting. We show that such reuse is highly beneficial in the next section.

2 Results

In our experiments we distinguish between different outcomes: *goal* when the car reaches the end of the highway without crashing and *crash* otherwise. We show that the single-agent learning problem can be easily solved as most of the models/configurations achieved a good performance. However, further tests show that these trained models do not necessarily adapt well when the simulator’s parameters change (e.g. traffic density). More importantly, the single-agent trained models show poor performance in a multi-agent setting (Figure 1 left), i.e., while the autonomous car agents can learn to deal with the human behavior, they cannot respond correctly to other agents. Therefore, it is necessary to explicitly train in a multi-agent setting. Training from scratch in the multi-agent setting takes a very long time, and does not learn to perform adequately yet in the time we had available. However, when the multi-agent learning uses a model that was already trained in a single-agent setting as a starting point, the results are significantly better (Figure 1 right).

We therefore conclude that reusing neural networks trained on a single-agent version of a multi-agent problem can lead to significant speed-ups and good performance.

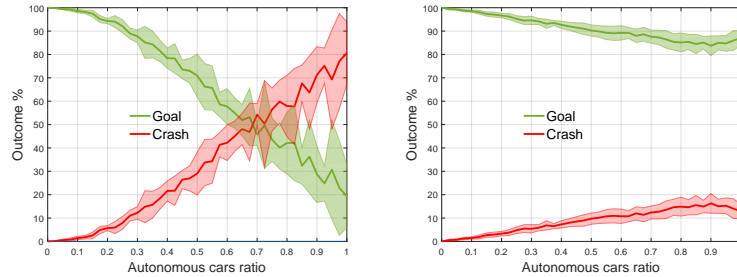


Fig. 1: Agent performance on multi-agent setting as a function of autonomous car ratio, trained on single-agent setting (left), and then re-trained on multi-agent setting (right)

References

1. Manon Legrand. Deep Reinforcement Learning for Autonomous Vehicle Control among Human Drivers. Master dissertation, Vrije Universiteit Brussel, 2017.
2. Volodymyr Mnih et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 2015.

An Agent-Based Model for Feasibility and Diffusion of Crowd Shipping^{*}

Max van de Westelaken and Yingqian Zhang

Eindhoven University of Technology

1 Introduction

In recent years, the sharing economy has revolutionized many different markets of different sizes. One relatively new example of the sharing economy is *crowd shipping*. In crowd shipping, packages are picked up and delivered by regular drivers using unused space in their vehicles. Unlike the big players (e.g., Uber and Airbnb), there is not much research being done on crowd shipping. In this thesis, an agent-based model was developed and simulated in order to gain insight into the feasibility and diffusion of crowd shipping based on results collected from a survey.

2 Method and Results

A model was created, aiming to research two topics: (1) the diffusion (spread) of crowd shipping, by looking at the diffusion of a new innovation as explained by Bass [1], and (2) the feasibility of crowd shipping, by looking at the percentage of packages delivered. A survey has been conducted among around 85 people. The following subjects are most relevant to the model: (1) the flexibility of the person for crowd shipping; and (2) the expected payment for a package. Questions on when to do crowdshipping and what kind of rewards were desired were asked, but a large majority (70%) replied “commute” and “money”, respectively.

The model was developed using RePast, an agent-based simulation platform. The model depicted two types of agents (Crowd Shippers and Regular Drivers) and the environment in which they moved. Both types of agents moved around on a grid, traveling between to their destination, while packages were dropped in the grid every few ticks (with every package having an origin and destination, as well as a reward associated with them). Only crowd shippers were able to pick up and deliver packages, drivers were not. However, drivers made a decision every 100 ticks to adopt crowd shipping or not, randomly chosen using the Bass method [1](see Equation 1), where $P(T)$ is the chance to adopt at time T , p reflects the fraction of adopters that already adopted the product, q/m represents the fraction of people the imitators comes in contact with, who have adopted the innovation, and finally, $Y(T)$ represents the number of previous adopters. In this way, adoption could be triggered in two ways: personal (through a bidirectional social network constructed at the beginning of the run) and mass media (through looking at the percentage of road users¹ who were crowd shippers).

$$P(T) = p + \frac{q}{m}Y(T) \quad (1)$$

The simulation starts with 5 crowd shippers and 195 drivers, i.e., 2.5% innovators. A crowd shipper generally moved between two points (home and work). Every tick, all packages in the model were screened². For every package, the *decision to pick up and deliver the package* is calculated by looking at the extra number of cells a crowd shipper had to move to go to the package, move to the package’s destination, and finally move to their own destination, as opposed to going to their own destination immediately. If this *extra distance* was smaller than the crowd shipper’s *flexibility* (and the *reward* high enough for the crowd shipper), the package was selected and picked up by the crowd shipper. Using the survey results, we generated the flexibility of an agent using a uniform distribution $\sim [5, 19]$, and the expected payment of an agent using a uniform distribution $\sim [4, 6]$, for a trip of at most 20 minutes (or 20 spaces on the grid).

Results. Figure 1a and 1b, the average adoption curve and average delivery chart are shown. These curves are based on 5 consecutive runs of 2000 ticks each. The figure shows that the rate

^{*} Originally submitted as Bachelor thesis in partial fulfillment of the requirements for the degree of Bachelor of Science at the Industrial Engineering & Innovation Sciences department at the Eindhoven University of Technology.

¹ It is assumed that all modeled road users are at least interested in adopting crowd shipping.

² Packages were removed after a certain amount of time to avoid clutter.

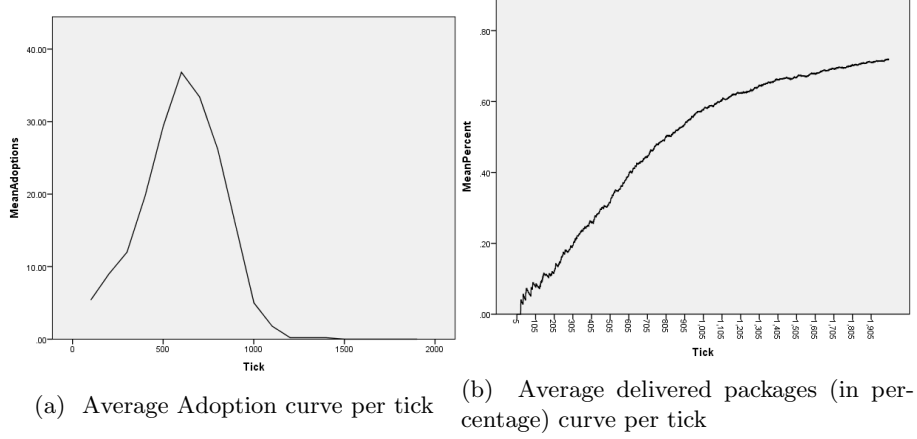


Fig. 1: Adoption and delivered percentage packages of 2000 ticks.

of adoption follows a bell-shaped curve, which is similar to the results found in the literature on different application domains. The runs from the model showed that deliveries increase over time, in accordance with there being more crowd shippers due to adoption. Larger sets of users showed a significantly higher percentage of packages delivered.

Finally, when compared to the standard set-up, increasing the flexibility range of crowd shippers showed the biggest improvement of percentage of packages delivered (see Figure 2). Similarly, lowering the flexibility range of crowd shippers showed a significant decrease in percentage of packages delivered. For each range in Figure 2 (5-10 for LOW, 10-15 for MID and 15-19 for HIGH), five runs were averaged.

The following factors are identified as the most important for feasibility of the model (and reality): flexibility (as seen in Figure 2), reward (lower rewards led to much lower effectiveness) and starting base (larger groups of starting users led to earlier conversion of potential crowd shippers, leading to an earlier rise in percentage of packages delivered). For entrepreneurs who would like to start a crowd shipping business, incentivizing users with a higher tolerance to drive further will be key to the successfulness of crowd shipping.

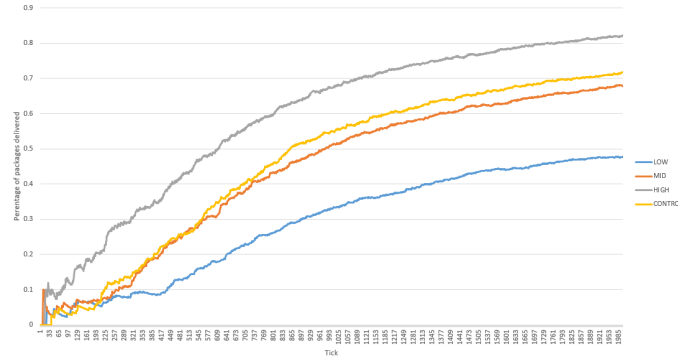


Fig. 2: Comparison of flexibilities

We have shown that there is certainly potential for crowd shipping if the right variables are met. In the future, we will work with crowd shipping companies and incorporate data collected from their platforms into the agent-based simulation model.

References

1. F. M. Bass. A New Product Growth for Model Consumer Durables. *Management Science*, 15(5):215–227, 1969.

Realtime Road User Detection and Classification with Single Pass Deep Learning

Robin Manhaeve¹

Promotors: Prof. dr. Luc De Raedt¹, Dr. ir. Kurt De Grave²

Supervisors: Dr. ir. Kurt De Grave², Dr. ir. Laura Antanas¹

¹ Dept. of Computer Science, K.U.Leuven, Belgium

`robin.manhaeve@cs.kuleuven.be`

² Flanders Make, Belgium

In autonomous vehicles (AV), computer vision supplements sensors such as radar and lidar because it excels at classifying road users into separate classes. Correct classification improves the prediction of their future movement and the risk of collision. However, in the real-time road environment, response time is as important as accuracy. A lot of highly accurate methods have been developed, but many have a large delay (detection runtime) [2], making them less suitable for AV. These techniques are slow because they are based on a two-step method (detection proposals and a classifier). Often, the classifier has to be evaluated many times, creating a large and variable delay. Lately, a new class of less accurate but faster methods has appeared: the single-pass methods. In this thesis, we discuss different positions in the trade-off between speed and accuracy. We also test the use of LWIR images for AV. These might be vital for road user detection during non-optimal lighting conditions, but existing research is limited.

The method used in this thesis is essentially the YOLO-method. [4] It performs detection in a single evaluation of the neural network, resulting in a big speed-up compared to two-phase methods. The original paper used a network based on their *Extraction* network. Because *Extraction* is not available in the Caffe framework (used in this thesis as it is more prevalent in research than the YOLO-framework), GoogLeNet [5] is used, of which *Extraction* is a simplification with similar performance. The effect of a ResNet-50-based [3] network on the accuracy and delay is also tested. The method is evaluated on the following datasets: Caltech Pedestrian Detection Benchmark [1], KITTI Vision Benchmark Suite [2] and a new dataset developed by Flanders Make. This new dataset consists of several camera feeds, radar, lidar, and IR images. Only the center front visual light (VL) camera and the Xenics LWIR (shown in Figure 1) were used in the experiments. This dataset is split into two parts. The first part contains only VL images (2,589 frames). The second part contains both VL and IR images (681 frames). In contrast to other datasets, this part was recorded during the evening. This allows the performance of VL detection to be evaluated under non-optimal lighting conditions. To perform detection on the IR images, several setups are compared: only VL images, only IR images, evaluating separate networks for VL and IR, and performing detection with a single network on the



Fig. 1. LWIR image from the Flanders Make dataset

combined 4-channel VL+IR image. The IR network was trained by first taking a network trained on KITTI and the first part of the Flanders Make data, then merging the RGB channels into a single channel (summing the filters), and finally fine-tuning on the LWIR data.

YOLO was first evaluated on the Caltech Pedestrian Dataset. In comparison to the most accurate result (F-DNN), this method is noticeably less accurate (.64 as opposed to .89 mAP on the *reasonable* set). It does display a lower delay: 40 ms as opposed to 300 ms (the difference being underestimated as the same hardware was not available). The Caltech dataset proved challenging for this method because most objects are small, a known weakness [4].

The method's accuracy (.42, .41 and .30 mAP) on cars, pedestrians and cyclists of moderate difficulty from the KITTI Vision Benchmark, failed to compete with the start of the art (SAIT: .90, .73 and .76 mAP). However, since the processing time of the methods are available, this benchmark gives a better overview on the Pareto-optimal methods. Here, Pareto-optimality is defined in the trade-off between delay and detection accuracy. YOLO showed to be close to the Pareto-front by being quite fast. A network based on ResNet-50 was also trained on this dataset. Although it had a small positive impact on the accuracy (1-4% mAP), it slowed the method down by 50%, pushing it away from the Pareto-front.

On the second part of the Flanders Make dataset, detection using only VL performed very poorly (.06 mAP on cars). Only using IR images yields superior results (0.59 mAP on cars). Combining the detections from VL and IR images (evaluating two separate networks) further improves the result for cars (0.64 mAP), however, other classes performed worse due to errors being introduced by detections in the VL images. Detection on both images in a single network did not perform well (0.16 mAP on cars).

Single-pass methods are less accurate than two-pass methods, but they are a lot faster. Considering the trade-off between speed and accuracy, single-pass methods prove more applicable for AV. The use of very deep networks in AV is not desirable, as their impact on the delay is too big. Finally, incorporating IR images is vital for AV vision during non-optimal lighting conditions.

References

1. Dollár, P., Wojek, C., Schiele, B., Perona, P.: Pedestrian detection: An evaluation of the state of the art. *PAMI* 34 (2012)
2. Geiger, A., Lenz, P., Stiller, C., Urtasun, R.: Vision meets robotics: The kitti dataset. *International Journal of Robotics Research (IJRR)* (2013)
3. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. pp. 770–778 (2016)
4. Redmon, J., Divvala, S., Girshick, R., Farhadi, A.: You only look once: Unified, real-time object detection. In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (June 2016)
5. Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., Rabinovich, A.: Going deeper with convolutions. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. pp. 1–9 (2015)

Balancing Imbalances

On using reinforcement learning to increase stability in smart electricity grids

Marten Schutten^{1,2}, Marco Wiering², Pamela MacDougall³

¹ Target Holding B.V.

² Institute of Artificial Intelligence and Cognitive Engineering, University of Groningen

³ TNO, Netherlands

Over the years interest in renewable energy sources is growing and the amount of electricity provided by such resources, such as photovoltaic (PV) or wind energy is increasing. The supply of renewable energy sources is highly dependent on environmental changes and therefore hard to predict and adjust. As a response, new methods have been proposed to shift control to the demand side of the electricity grid. Furthermore, it can be seen that there is a shift from a situation in which the supply of electricity is managed by a small group of very large suppliers to a larger group of smaller suppliers (e.g. wind farms or households with PV panels).

In order to maintain balance between supply and demand, any shortages or excesses of electricity are resolved by trading on the reserve market. Due to minimum production rates that are required to trade on this market, small suppliers can not participate in this trade. However, as this group becomes larger it would be a great addition to the reserve markets when it comes to maintaining stability in the grid. One solution to add these smaller parties to the electricity grid is to bundle groups of small prosumers into a cluster, controlled by an aggregator. The aggregator can then offer a certain range of power within which it is able to ramp the amount of consumption (or production) up or down.

In this thesis it is shown how reinforcement learning can be applied to successfully learn the boundary conditions within which an aggregator can safely ramp up or down its consumption. Furthermore, Neural Fitted CACLA (NFCACLA) is proposed as a new reinforcement learning algorithm, which is a neural fitted variant on the existing CACLA algorithm [1].

Two reinforcement learning agents were implemented in an internal simulation environment, developed by TNO (The Netherlands Organisation for Applied Scientific Research), used for simulating smart electricity grids. The Power-Matcher [2] was used to coordinate the supply and demand within the simulated grid through dynamic pricing.

The first agent was trained using the CACLA [1] algorithm, an Actor-Critic model that uses function approximators to approximate both the value and policy functions of the model to deal with a continuous state and action space. The second agent was trained with the newly introduced, NFCACLA algorithm: a model-free batch learning algorithm, inspired on the CACLA and NFQCA [3] algorithms.

The agents were trained and tested in a simulation containing 100 households equipped with a PV panel and either a heatpump (80 households) or a micro-CHP (20 households), which is a cogeneration device for generating both heat and power. Historical data was used to simulate weather conditions and imbalance demand for the month of March 2013. The results were compared with a baseline in which the agent offered a fixed percentage of its available flexibility.

Results of the experiments were both measured in terms of imbalances that were resolved and caused by trading, as well as the financial costs and gains that were made. Table 1 shows a summary of the results of the best performances for the baseline and both RL agents. The optimal performing baseline was when the agent offered 50% of its available flexibility.

Table 1. Results of RL agents versus baseline results

agent	resolved imb.	caused imb.	difference	gains	costs	profit
baseline (50%)	751.56 kW	507.15 kW	244.42 kW	€18.29	€5.97	€12.32
CACLA	734.21 kW	552.96 kW	181.25 kW	€18.04	€7.35	€10.69
NFCACLA	577.41 kW	383.68 kW	193.72 kW	€17.61	€4.71	€12.90

Over all results a number of observations can be made. The first observation is that the caused and resolved imbalances were generally lower when boundaries were set by the NFCACLA agent, even when larger portions of the available flexibility were offered. A second observation is that the amount of electricity was generally lower than the boundaries that were provided by the agents. This means that the actual trades had a smaller impact on the grid than the boundaries had allowed for. Finally, the NFCACLA agent is able to make a larger profit than the baseline, even though the amount of traded electricity is smaller.

In conclusion, this thesis shows that RL can be used for trading electricity on the reserve market. However, simpler solutions might prove more successful. One of the downsides in the scenario with which the simulations were performed, is that the balancing trades on the reserve market were rather small, thus having a small impact on the cluster itself as well. In order to test the usefulness of RL additional experiments should be done to test the resilience of the agents when larger amounts of imbalances are traded on the reserve market, such that the boundaries are more stressed.

References

1. Van Hasselt, H., Wiering, M.A.: Reinforcement learning in continuous action spaces. In: Approximate Dynamic Programming and Reinforcement Learning, ADPRL. IEEE International Symposium on. (2007) 272–279
2. Kok, K.: The powermatcher: Smart coordination for the smart electricity grid. TNO, The Netherlands (2013)
3. Hafner, R.: Dateneffiziente selbstlernende neuronale Regler. PhD thesis, PhD thesis, University of Osnabrueck (2009)

Should you Link(ed) Data?

Quality Assessment for Linking to Third-Party Data

Jesse Bakker¹, Wouter Beek¹, and Erwin Folmer²

¹ VU University Amsterdam, Netherlands

² University of Twente, Netherlands

1 Introduction

The key feature of Linked Data is, linking data. By assigning URIs to things, we can uniquely identify them out of the context of a single dataset. With this, we can in a way, ascent from the data silos, to a single knowledge graph. In this knowledge graph we have the ability to combine data from numerous databases. But this comes with a risk. Before associating your own data with veiled, third-party data, one should know whether it suits your quality needs. For this, the following research question was posed; *How can a quality assessment help justify the interlinking of Linked Data and foster trust?* A methodology is presented as the answer to the research question, see Figure 1. This Methodology constitutes three phases, namely a Preparation, an Assessment and an Interpretation phase. After which, an established opinion (as quality is subjective/context dependent) is formed based on an extensive quality measurement dataset. This dataset contains 1. measurements 2. metadata 3. URIs of erroneous "things".

The methodology has tested with real Linked Open Data. This uncovered both relevant statistics and otherwise secluded erroneous instances. By using generic SPARQL queries and python code, the metrics were easily measurable and could quickly be deployed. However, the standard metrics, in most use cases, had an unsatisfactory coverage. This can be solved with the custom metrics, of which the creation is costly in both time and (cross-domain) expertise.

During the preparation phase, two datasets are selected, one to be assessed and one as the context. Standard and Custom metrics are employed in order to cover as many quality facets as possible. The Assessor defines quality requirements, based on the context dataset, in SHACL, in order to automate parts of the interpretation.

A knowledge graph is iteratively constructed. When following the methodology, the assessor iteratively interprets partitions of the measurements. Prior to each interpretation step, the knowledge graph is automatically updated, such that all to-be-interpreted measurements are described in the knowledge graph. Moreover, the knowledge graph includes documentation on the metrics itself and concepts related to the notion of quality. For each interpretation step, the assessor utilises predefined SPARQL patterns to retrieve a set of measurements, related metrics and other concepts. In addition the assessor retrieves a validation report, generated with SHACL. These are foundation of the opinion on whether the datasets should be interlinked.

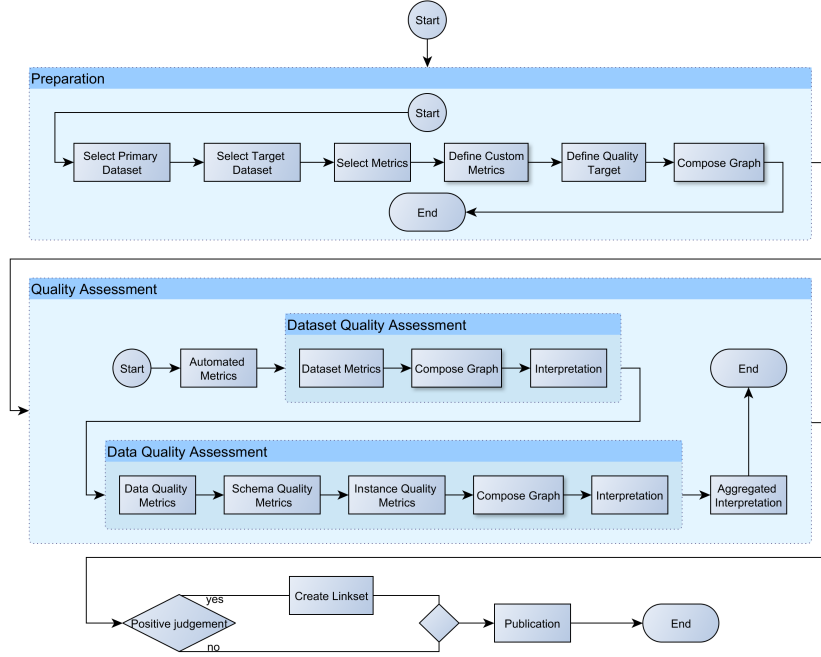


Fig. 1. Quality Assessment Methodology

SPARQL and Python are leveraged to compute the measurements. This is done semi-automatically, as metrics can sporadically require manual input. The measurement procedure is generic and can easily be reused, and extended for other use cases. Since resources have unique URIs, we can easily identify resources which negatively affect the resulting value of the metric, during the measurements and say something about them. This information is separately stored in the knowledge graph.

The knowledge graph is leveraged into an established judgement on whether one dataset should be interlinked with another, based on the quality of both the data and the dataset. Several Semantic Web standards are utilised in the process, such as SHACL, SPARQL, RDF(s), OWL and PROV-O. Regardless of whether the judgement favours the creation of a linkset (the set of triples relating the two datasets to each other), the knowledge graph should be published. This offers several benefits. First, the owner of the assessed dataset can use the knowledge graph to identify faulty resources and quality issues. Allowing the owner to amend them. Second, it provides a wealth of information to data consumers. Allowing them to inform themselves about possible uses, strengths and weaknesses of the data. Furthermore, even when no linkset resulted from the methodology, the knowledge graph can function as an admonition to inform data consumers on the risks of interlinking the two datasets.

Customer Profiling based on Electronic Payment Transaction Data

Michiel Van Lancker¹, Annemie Vorstermans¹, and Mathias Verbeke²

¹ Faculty of Engineering Technology, KU Leuven, Belgium

² EluciDATA Innovation Lab, Sirris, Belgium

Abstract. Customer profiling allows companies to profile a customer or group of customers based on their transaction behavior. In this study, transactions of electronic payment cards are considered. Contrary to classic market basket analysis, this transactional data does not contain information about the products bought, but only the amount of the payment, as well as a number of details on the respective shop. In order to create meaningful customer profiles based on this limited amount of information, four different techniques were compared. While the construction of customer profiles for groups of customers was possible, it proved difficult to thoroughly validate the profiles and methods used without extra reference data.

Keywords: Transactional data mining, Customer Profiling, Clustering

1 Introduction

In recent years, data mining of transactional data has become widely implemented in a variety of businesses. The practice of building customer profiles from transactional data allows companies to profile a customer or group of customers based on their characteristics or behavior. These profiles can subsequently be used for different purposes, e.g., to improve customer service or tailor marketing campaigns. Essential to capture the true nature of the shopping behavior of a customer, is the extraction of relevant features [1]. Contrary to classic market basket analysis, in which information is available on the products bought, in this study the problem of customer profiling is considered in which only a limited amount of details are available.

2 Method

To construct profiles that represent the shopping behavior of groups of customers, relevant customer features need to be extracted from the data. Because no information on the products bought is present in the data set, transaction characteristics were transformed into customer features with a frequency-based approach, resulting in the extraction of 26 relevant features from four transaction characteristics, namely the time, location, transaction amount and shop type of a transaction [2]. To improve the quality of the data, two major changes were made. Firstly, shops were categorized per shop type based on similarity of transaction pattern and presence of keywords in the name of the shop. Secondly, because transactions can be performed in both a work-related or a leisure-related context, a context-based geographical representation for the customers was developed. Based on the resulting features, customers were clustered using two different algorithms: K-means clustering and agglomerative clustering. For each of these techniques, two approaches were tried: A first approach clustered once based on all customer features, while the second approach combined clusterings based on disjoint sets of customer features into one final clustering, similar to bottom-up subspace clustering [3]. The resulting profile per cluster summarizes the behavior of a customer as the difference (in number of standard deviations) of the cluster mean customer from the population mean customer for each feature.

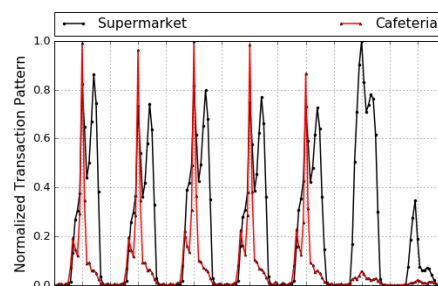


Fig. 1: Transaction patterns of shop categories show clear differences

3 Results

To analyze the results of different techniques, a comparison was made both between algorithms and between approaches. While K-means clustering slightly outperformed agglomerative clustering based on cluster quality, the difference was small and did not entail a better representation of the behavior of customer groups. Agglomerative clustering however offered multiple implementation advantages. Because of the deterministic hierarchical tree it constructs, the algorithm can be run once to build the tree and then generate multiple sets of clusters according to the needs of specific applications. Thus, agglomerative clustering can generate different levels of profiling without having to rerun the whole algorithm. K-means clustering does not offer this freedom. Furthermore, the effectiveness of K -estimation declines for data of higher dimensions and is thus less effective when clustering based on all customer features at once.

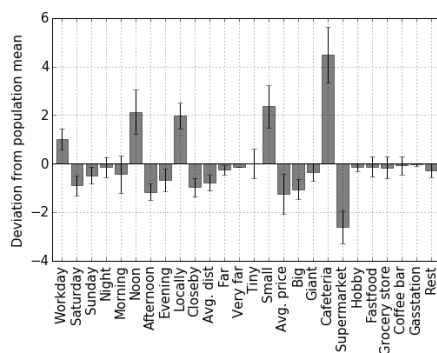


Fig. 2: The profile of a group of customers, showing the deviation from the population mean for all features, expressed in standard deviations

When comparing the results of clustering based on all customer features and clustering by combining clusterings based on disjoint subsets of features, multiple differences were observed. While the main advantage of the first approach is its simplicity, it is sub-optimal and generates a small number of clusters with rather generic profiles. Because the second approach is much more thorough, it is capable of finding more specific profiles. This is however paired with an explosion of the number of clusters: for K-means clustering 991 clusters were found, for agglomerative clustering 1322 clusters. It should be noted though that many clusters are very small and can be disregarded if needed. Overall, this approach can identify more specific customer behavior with only a slight extra effort compared to clustering based on all customer features. Additionally, this approach offers more freedom to the designer too: because clusterings from different subsets of features can be combined in multiple ways, different angles to customer profiling can be examined with only one run of the algorithms. The resulting customer profiles were evaluated in a qualitative way using domain knowledge. A quantitative analysis would be valuable to get deeper insights in the obtained results. This would however require an extensive customer study with manually labeled data which was out of the scope of this study.

4 Conclusion

Four different techniques for customer profiling based on transactional data from electronic payment cards were compared, in which only the transaction amount and a limited amount of information on the respective shop was available. Combined agglomerative clustering using subsets of customer features turned out to be the most interesting, because the qualitative analysis showed promising results and it offers a lot of freedom to the designer to tailor it to the needs of specific applications.

References

1. Jain, A. K.: Data clustering: 50 years beyond K-means. Pattern recognition letters, 31(8), 651-666 (2010).
2. Leonard, M., Wolfe, B.: Mining transactional and time series data. SUGI, 10-13 (2005).
3. Parsons, L., Haque, E., Liu, H.: Subspace clustering for high dimensional data: a review. ACM SIGKDD Explorations Newsletter, 6(1), 90-105 (2004).

Index of Authors

- Alvarez-Estevez, Diego, 161
Antanas, Laura, 421
Argentini, Andrea, 306
Aroyo, Lora, 410
- Bakker, Jesse, 425
Beckers, Sander, 350
Beek, Wouter, 425
Beuls, Katrien, 224, 403
Bex, Floris, 32, 172
Bilgin, Aysenur, 401, 408
Bontempi, Gianluca, 101
Borgne, Yann-Aël Le, 101
Bosse, Tibor, 61, 184, 376
Boulogne, Luuk, 326
Bountouridis, Dimitrios, 367
Bozzon, Alessandro, 396
Brandt, Armin, 76
Bransen, Jeroen, 381
Breda, Ward van, 213
Bredeweg, Bert, 415
Brinkhuis, Matthieu, 172
Broekens, Joost, 362
Brouwer, Natasa, 415
Brown, Dan, 367
Burguillo, Juan Carlos, 342
- Cock, Martine De, 268
Collins, Pieter, 116
Croft, W. Bruce, 372
- Dalpiaz, Fabiano, 281
Dankers, Verna, 401, 408
Dastani, Mehdi, 281
Dazeley, Richard, 389
Dehghani, Mostafa, 372
Dekker, Evan, 389
Dell'Anna, Davide, 281
- Dignum, Virginia, 357
Domingos, Elias Fernández, 342
Driessens, Kurt, 116
- Eecke, Paul Van, 224
Erven, Tim van, 405
- Fernández, Raquel, 401, 408
Folmer, Erwin, 425
Formolo, Daniel, 364
- Gaag, Linda van der, 32
Galinkin, Nikita, 410
Gerbscheid, Jonathan, 398
Gerritsen, Charlotte, 213
Gerven, Marcel van, 238
Goedschalk, Linford, 61
Grave, Kurt De, 421
Groot, Thomas, 398
- Haas, W. Bas de, 381
Harmelen, Frank Van, 360
Haselager, Willem F.G., 413
Henkemans, Olivier Blanson, 378
Hindriks, Koen V., 370, 378
Hoeven, Dirk van der, 405
Hogervorst, Jonathan, 317
Hoorens, Sébastien, 224
Houthuys, Lynn, 352
Hu, Qing, 360
Huang, Zhisheng, 360
Huijser, Stefan, 199
- Issabekov, Rustam, 389
- Jonker, Catholijn M., 357, 362, 370
- Kamps, Jaap, 372
Karevan, Zahra, 352

Klinkmeijer, Lars Zwanepol, 391
 Koeman, Vincent Jaco, 370
 Koops, Hendrik Vincent, 367, 381
 Kos, William, 172
 Krainski, Elias, 213
 Kwisthout, Johan, 346

 Lancker, Michiel Van, 427
 Larsen, John Bruntse, 383
 Legrand, Manon, 394, 417
 Lenaerts, Tom, 342
 Lenders, Daphne, 413
 Ligthart, Mike, 378
 Lubbe, Laura van der, 184

 Maathuis, Henry, 326
 MacDougall, Pamela, 423
 Manhaeve, Robin, 421
 McKelvey, T. Greg, 268
 Medeiros, Lenin, 376
 Mehrkanoon, Siamak, 355
 Mercur, Rijk, 357
 Merhej, Elie, 268
 Mihalák, Matúš, 291
 Moerland, Thomas M., 362
 Moret-Bonillo, Vicente, 161

 Neerincx, Mark, 378
 Nevens, Jens, 403
 Nowé, Ann, 387, 394, 417

 Okafor, Emmanuel, 317
 Otte, Marco, 61
 Otterlo, Martijn van, 145
 Overmeen, Manfred, 396
 Ozkohen, Paul, 145

 Pieters, Mathijs, 131
 Plisnier, Hélène, 387
 Polevoy, Gleb, 17, 46
 Prakken, Henry, 32, 348

 Rădulescu, Roxana, 394, 417
 Raedt, Luc De, 421
 Ramon, Jan, 306
 Reggiani, Claudio, 101
 Renaux, Jérôme, 306
 Renooij, Silja, 32
 Roijers, Diederik M., 387, 389, 394, 417
 Roos, Nico, 2
 Roschewitz, David, 116

 Schockaert, Steven, 268
 Schraagen, Marijn, 172
 Schulze-Bonhage, Andreas, 76
 Schutten, Marten, 423
 Seeliger, Katja, 238
 Severyn, Aliaksei, 372
 Smirnov, Evgueni, 291
 Stam, Marco, 213
 Stap, David, 415
 Steckelmacher, Denis, 387
 Sterk, Alef, 326
 Suykens, Johan A.K., 352, 355
 Szlávík, Zoltán, 396, 410

 Taatgen, Niels, 199
 Timmermans, Benjamin, 396, 410
 Troost, Marjolein, 238

 Vamplew, Peter, 389
 Veltkamp, Remco C., 367
 Veltman, Kim, 253
 Vennekens, Joost, 350
 Verbeke, Mathias, 427
 Verbrugge, Rineke, 253
 Villadsen, Jørgen, 383
 Visser, Arnoud, 398
 Visser, Jelle, 145
 Volk, Anja, 381
 Vorstermans, Annemie, 427
 Vugt, Marieke van, 76, 199

 Wal, C. Natalie Van Der, 364
 Webb, Dean, 389
 Weerd, Harmen de, 86, 253
 Weerdt, Mathijs de, 17, 46
 Westelaken, Max van de, 419
 Wiering, Frans, 367
 Wiering, Marco, 131, 145, 317, 326, 423
 Wieten, Remi, 32
 Wiggers, Bram, 86
 Wimmenauer, Florian, 291

 Yang, Jie, 360
 Ye, Qing Chuan, 374
 Yorke-Smith, Neil, 344

 Zamani, Hamed, 372
 Zant, Tijn van der, 391
 Zhang, Yingqian, 374, 419
 Zintgraf, Luisa M., 389

BNAIC

• 2 0 1 7 •

29th Benelux Conference on Artificial Intelligence
November 8-9, 2017, Groningen