Classification & Prediction Methods and their Application

Jan Stutzki

Dissertation
an der Fakultät für Mathematik, Informatik und Statistik
der Ludwig–Maximilians–Universität
München

vorgelegt von
Jan Stutzki
aus Aachen

München, den 21.07.2017

Classification & Prediction Methods and their Application

Jan Stutzki

Dissertation
an der Fakultät für Mathematik, Informatik und Statistik
der Ludwig–Maximilians–Universität
München

vorgelegt von
Jan Stutzki
aus Aachen

München, den 21.07.2017

Erstgutachter: Prof. Dr. Hans-Peter Kriegel
Zweitgutachter: Prof. Dr. Michael Gertz
Tag der mündlichen Prüfung: 24.11.2017

Eidesstattliche Versicherung

Hiermit erkläre ich, Jan Stutzki, gemäß § 8 Abs. 2 Pkt. 5 der Promotionsordnung vom 12.07.2011, an Eidesstatt, dass die vorliegende Dissertation von mir selbständig, ohne unerlaubte Beihilfe angefertigt ist.

München, 21.07.2017

Abstract

The contemporary management paradigm is focused on strategic management. A crucial component of strategic management is the definition of Key Performance Indicators (KPIs), and the ability to predict potential future developments. These two tasks are closely intertwined, as future developments can be guided using certain KPIs, which in turn can influence the future developments. On the other hand - does the future development require action which may lead to an adaption of new KPIs? This cycle of future development and guidance with the help of KPIs requires a close monitoring of the current state, and the anticipated state of the business venture as well as the relevant environment. The domain of economics has developed many tools which provide frameworks supporting the search for useful KPIs and the exploration of futures space. While the internal factors of an organization can be analyzed by management, the environment is less accessible and predictable.

In this thesis we propose approaches for environmental scanning and trend detection for a medium to long term time range. We identify three different data sources: macro indicators, web data and patent data, each covering a different time range. For short term developments, web data are useful, as any new trends are propagated almost instantly, and changes in any domain are reflected by changes in web content. With its accessibility and world wide coverage, the Web takes precedence over other sources of information. We describe a system that enables decision-making based on monitoring relevant topics in several dimensions; among which are country, time and intensity [133].

Long term developments are covered by macro indicators, e.g. child mortality or GDP. We survey various data sources and quantify their quality. We propose an architecture which unifies such indicators and allows forecasting and integration into more complex models [24]. As indicator data are provided as time series, we evaluate the performance of various inter- and extrapolation methods. Due to the importance of time series data, we do an excursus into time series classification, using deep learning techniques. As examples we use accelerometer data from human activities. Those are commonly used as there are sufficient sample data and published methods for evaluation available. We apply the evaluation of various network architectures, required window sizes and explore the benefit of a personalized activity recognition model.

With a typical time between first submission and publishing of 18 month, we use patent data to cover the medium time range. While the data is usually structured and dimensions like country of origin and certain dates are available, the challenge is the prediction of the

classification of a patent document. Based on the classification it can be decided whether a given application is relevant to the field at hand or not. We evaluate various state-of-the-art classification methods and propose two new methods. We use geo-spatial meta data included in patent application data to find clusters of classes. This information is combined with a text-based classifier to significantly increase the classification accuracy [134]. The second contribution uses a novel approach towards embeddings which allows us to perform sequence learning with recurrent neural networks on documents, outperforming the state of the art.

Zusammenfassung

Das zeitgenössische Management-Paradigma ist fokussiert auf Strategisches Management. Ein wesentlicher Bestandteil des Strategischen Managements ist das definieren von Key Performance Indikatoren (KPI) und die Möglichkeit, die zukünftige Entwicklung zu antizipieren. Diese beiden Aufgaben sind eng miteinander verflochten, da sich die weitere Entwicklung durch KPIs beeinflussen lässt. Andererseits kann es sein, dass die die zukünftige Entwicklung einen Einfluss hat, der eine Anpassung der KPIs notwendig macht. Aufgrund dieser Wechselwirkung ist es nötig, den Ist-Zustand und potentielle Entwicklungen genau zu verfolgen. Die Wirtschaftslehre hat viele Methoden entwickelt, mit denen das suchen nach sinnvollen KPIs und die Erkundung des Zukunftsraums unterstützt werden. Während die internen Faktoren einer Organisation direkt durch das Management analysiert werden können, ist das eine Analyse des Umfelds deutlich komplexer.

Diese Arbeit schlägt Ansätze zur Umfeld-Analyse und Trend-Erkennung vor. Dabei wir der Zeit-Horizont unterteilt in kurz-, mittel- und langfristig. Passend zu jedem Zeit-Horizont wird eine andere Quelle für Informationen verwendet: Makro-Indikatoren, Web-Daten und Patent-Daten. Web-Daten sind besonders geeignet um kurzfristige Trends schnell zu erkennen. Das liegt zum einen daran, dass das Web sich zu dem primären Medium für Informationen und zum Austausch von Nachrichten entwickelt hat. Zum anderen aber auch daran, dass die Eintrittshürden niedrig sind und somit jeder publizieren kann. Diese Arbeit beschreibt ein System, dass Entscheidungen unterstützen soll, indem es relevante Themen mit Hilfe des Webs in mehreren Dimension, wie z.B. Zeit und Ort überwachen kann [133].

Lang-anhaltende Entwicklungen werden in dieser Arbeit mit Makro-Indikatoren, wie z.B. Kindersterblichkeit oder Bruttosozialprodukt verfolgt. Es wird eine Bewertungsmethode für die Qualität von Indikatoren vorgestellt, die anhand von verschiedenen Indikatoren empirisch überprüft wird. Um verschiedene Quellen von Indikatoren, Methoden zur Zeitreihen-Verarbeitung und Modelle zu vereinen, wird eine modulare Architektur präsentiert [24]. Verschiedene Extrapolations-Verfahren von Zeitreihen werden mit Hilfe der Indikatoren evaluiert. Da Zeitreihen eine hohe Bedeutung zukommt und Deep Learning neue Methoden bereit stellt, wird untersucht wie sich klassisches Feature-Engineering gegen Feature-Learning auf Zeitreihen behauptet. In dieser Untersuchung werden Bewegungsdaten verwendet, da diese in großen Umfang und mit ausreichender Genauigkeit verfügbar sind.

Mit einer durchschnittlichen Zeit von der Anmeldung zum Publizieren von 18 Monaten,

eignen sich Patent-Daten für den mittelfristigen Zeit-Horizont. Patent-Daten sind strukturiert und Informationen wie Land, Anmeldedatum und viele mehr lassen sich leicht extrahieren. Das automatische klassifizieren von einer Patent-Anmeldung in ein bestehendes Klassifikations-System dagegen, ist immer noch eine Herausforderung. In dieser Arbeit werden zwei verschiedene Ansätze verfolgt um die Klassifikation solcher Dokumente zu verbessern: Meta-Daten, insbesondere die Adressen der Erfinder werden benutzt um lokale Patent-Cluster zu identifizieren und dadurch die Genauigkeit einer herkömmlichen Klassifikation zu verbessern [134]. In dem zweiten Ansatz wird eine semantische Methode verwendet, die die Struktur eines Patent-Dokuments in eine Sequenz abbildet die den Lesefluss eines Menschen durch ein solches Dokument nachahmt. Die Repräsentation eines Dokuments als Sequenz erlaubt das verwenden von Sequence-Learning mit Rekurrenten Neuronalen Netzen. Experimente zeigen, dass diese Methode genauer klassifiziert als der aktuelle Stand der Technik.

Contents

Chapter 1

Introduction

1.1 Motivation

With the technological revolution started by computers and continued by connecting them world wide through the Internet, businesses are facing new opportunities and issues. Digitalization of organizations and industrial installations lead to an enormous amount of digital data. Entrepreneurs identified the value of these vast collections of data for various purposes. One of the most common usages of analyzing this "Big Data" is probably targeted advertising. Google for example can track most of the web activity of a user either through Google Search, Google Mail or Google Analytics which is popular among webmasters. One of the goals of targeted advertising is to identify what products and services specific users are willing to spend money on by getting to know the user, possibly better than she knows herself. That this is possible, has been demonstrated during the US presidential election and the British Brexit vote 2017 by the company Cambridge Analytica which targeted Facebook users so that they vote for a particular candidate [6].

Separating the Facebook users into two camps for two candidates based on their preferences is a binary classification problem. One might argue, that in addition it would be worthwhile to identify the "undecided" Facebook users, as those might be swayed more easily and therefore represent a better investment for targeted advertising. That would transform the binary classification problem into a multi-class classification problem where each sample is only part of one class opposed to the multi-label classification task where one sample might be a member of an undetermined number of classes.

Cambridge Analytica uses Big Data to change audience behavior [2] and Google uses Big Data to show the most profitable ad to its users. As these corporations target individuals, a similar thorough analysis of behavior is possible for organizations. As digitalization is moving more and more processes to computers and networks, the accumulation of data e.g. in SAP allows a deeper analysis of the business for better insights into e.g. the strengths, weaknesses, opportunities and threats a business is currently facing or that might arise in the future. In the context of Strategic Management, Business Schools around the world have been developing frameworks and methods for organizations that support management

in deciding which insights are relevant by providing context, targets and ontologies. These frameworks commonly distinguish insights derived from the internal as opposed to the external environment. Due to the sensitivity of internal data, the information needed for insights into the internal environment is usually only available to an organization itself and can be highly domain specific.

In contrast, many of the external insights recommended by the methods of Strategic Management are based on publicly available data e.g. the demographic development of a country or the perception of a brand name. As Strategic Management is primarily concerned with developments in the future, the prediction of the future development or at least a timely analysis of the external environment is highly relevant.

The aim of this thesis is to research methods that provide quality information about certain aspects of the external environment for three overlapping time horizons, each being associated with a time range: short, medium and long. In the literature the future time horizons vary from a few months to more than 15 years [25, 37]. On the other hand, the influential study of Lawrence and Lorsch [93] concludes, that the future time horizon depends on the organizational unit which is using the horizons. Different future time horizons require different methods, as the assumption is, that methods which use the most up-to-date information provide rather volatile results. This is because they suffer from a lot of noise where short-lived topics and trends arise and fade away that are not relevant to the planning of an organizational unit. Studies of a popular micro-blogging service confirm that [126]. Concluding: for larger future time horizons, a larger past time horizon is beneficial as the most recent information are less relevant due to the issue of filtering signal from noise which increases with the maturity of a source and associated cost of publishing.

1.2 Short-Term

As this work aims at supporting business decisions made in various organizational units, the short term horizon is defined as a range from a few days to a few months, possibly a year. With this definition of short term, popular social networks like Facebook or micro-blogging services e.g. Twitter do not represent a reasonable data source. Trends and topics in these networks shift with a too great time-resolution and are prone to bias as the user base is not representative [126]. Web pages on the other hand are less volatile and provide a greater amount of information for further analysis. The proposed method that primarily focuses on the short term environment scanning, gathers web pages relevant to a predefined topic in fixed intervals to detect trends over time and derives metrics that can be used for further analysis (see Section 4.1). Topics are defined by the user in the form of keywords. Besides the unstructured text contained in a web page, additional insights are gathered from the meta data e.g. how often a page is updated, how do search engines rank a page and what languages, respectively countries dominate a given topic. The method is published in [133].

1.3 Medium-Term

Due to the overlap of the time horizons, the medium time span is specified to a range of about 12 month to approximately 3 years. A reputable source for information corresponding to that time horizon is found in patents. A patent application contains a new innovation made by inventors and is usually publicly available 18 month after the application is submitted to a patent office though exceptions can be made for an earlier publishing date. With publishing, the patent is pending. Since several years now the time from application to a granted patent is steadily decreasing [15] and, depending on the field, varies between 24 to 30 month on average [15]. The cost associated with a patent application ensures that only serious claims are made. Digitalization at the United States Patent and Trademark Office (USPTO) in cooperation with Google results in publicly available patent applications and grants reaching back as far as to the year 1920.

Monitoring patents of competitors and patents that are relevant to the business interests is already widely practiced [48]. To enhance the utility of patent analysis, a reduction in the lag between publishing and granting can lead to a competitive edge. One of the issues that hinders a reduction of that lag is the classification of a patent. As most patent offices categorize patents into an ontology of topics, with each patent being member of possibly multiple topics, the classification is used to determine the relevance of a patent to a company's interest. This thesis proposes the use of geospatial data contained in a patent application to enhance the performance of automated patent classification in an effort to reduce the lag (see Section 5.4.1). The method is published and peer reviewed in [134]. This work evaluates new approaches in Natural Language Processing (NLP) made possible by methods of Deep Learning and evaluates their potential in regards of enhanced classification performance using the patent text only. The semantic approach is extended by a hierarchical context-aware representation that allows the creation of sequences that mimic the way a human would parse a document (see Section 5.5). Compared to the state of the art, significant improvement of classification performance is achieved.

1.4 Long-Term

Many of the previously discussed methods are still useful for long term analysis, but for long term planning, another set of information is more relevant. Ease of doing business, stability of a countries political system and grade of development are key aspects e.g. for the expansion into other countries. These factors are captured by macro indicators. These macro indicators provide a coarse time resolution ranging from quarterly to yearly and are provided for most countries in the world allowing for an easy comparison. Leading indicators which predict future development are based on these macro indicators e.g. Goldstone Adverse Regime Change Onset [69]. The combination of models that create leading indicators together with time series extrapolation allows decision makers to identify probable future scenarios. A platform for visualization, integration of models and data sources that is based on micro-services is presented in this thesis (see Section 6). Time series process-

ing with recently developed methods from the field of Deep Learning is evaluated on the example of Human Activity Recognition. Though unrelated to the field of Strategic Management, this type of application for the evaluation is used as the amount of ground truth available is sufficient for supervised training of deep neural networks. Related work for approaches using traditional feature engineering and new Deep Learning approaches are compared in various dimensions (see Section 7.1).

1.5 Thesis Overview

The following Chapter introduces the concepts of Strategic Management and some of the tools that are used as a prerequisite to identify tasks which require extensive data analysis with either classification or prediction methods. Due to the nature of the data that are processed, an extensive array of methods applied during preprocessing and the actual computation exist. A brief introduction on the preliminaries for natural language processing (NLP), text mining and classification is presented in Chapter 3. The following chapters discuss the proposed methods categorized into the three time horizons: short-term in Chapter 4), medium-term in Chapter 5 and long-term in Chapter 6). Evaluation of Deep Learning and feature engineering are evaluated in Chapter 7. The last Chapter 8 concludes the results of the research and discusses an outlook for further research.

1.6 Contributions

Publications incorporated in this thesis have been primarily developed with researchers from the Database Systems Group of the Ludwig-Maximilians-Universität München (LMU). The doctoral adviser, Prof. Dr. Hans-Peter Kriegel supervised all publications, with the exception of [133].

Chapters 5 and 7 are the results of joint efforts so that a more detailed explanation of the contributions are discussed in this section.

1.6.1 Macro-Indicators

Chapter 6 is the work of the author of this thesis. The integration and evaluation of various data-sources as much as the implementation of a proof-of-concept was done by Jan Stutzki. Experiments to determine the data quality and to evaluate the performance of various time-series extrapolation methods have also been conducted by Jan Stutzki.

1.6.2 Multilingual trend detection in the web

The publication [133] is the exclusive work of the same author of this thesis. The author of this thesis contributed the initial idea to use web data, the idea and development of various metrics for trend detection and conducted the experiments. The segmentation of the trend analysis by country and language is also a contribution by the author. Implementation of

a proof-of-concept was done by the author. Related work and writing of the publication was also the sole responsibility of the author.

1.6.3 Geodata supported classification of patent applications

This publication [134] was a cooperation between Matthias Schubert and Jan Stutzki. The author of this thesis contributed the initial idea to use location-specific class-distributions to enhance the overall classification result in combination with a text-classifier. Experiments and writing was contributed by the author of this thesis. The other author contributed experienced writing and guidance throughout the development- and research-process.

1.6.4 Fixed Hierarchy Vectors

The work on Fixed Hierarchy Vectors in Section 5.5 was a cooperation between Marawan Shalaby, Matthias Schubert, Hans-Peter Kriegel, Stephan Günnemann and the author of this thesis. The idea of segmenting text into semantic elements for which separate context models are trained, the majority of the experiments and large parts of the writing are contributions by the author of this thesis. The evaluation of various text-representations, preprocessing of the data and implementation was a joint effort of Jan Stutzki,Marawan Shalaby and Matthias Schubert with a majority of the work contributed by the author of the thesis. Related work and writing of a proposed publication has been done by Jan Stutzki, Marawan Shalaby and Matthias Schubert. Supervision was provided by Hans-Peter Kriegel, Stephan Günnemann and Matthias Schubert.

1.6.5 Human Activity Recognition

In Chapter 7 the author compares feature-learning with feature-engineering-approaches by using state-of-the-art methods from both ends of the spectrum. Other insights involve the sensor-position for optimal Human Activity Recognition (HAR), generic models versus specific models on per individual basis and the evaluation of various parameters e.g. window size. The author of this thesis conducted the experiments, implemented the references, researched the related work and developed a competitive neural network architecture. Matthias Schubert provided experienced writing and guidance throughout the development process.

Chapter 2

Strategic Management

2.1 Introduction

This Chapter bridges the gap between research and application and discusses the context in which the proposed methods can be used. The leading management paradigm for organizations that is taught by business schools, is referred to as Strategic Management. One of the key-elements of Strategic Management is an activity called environmental scanning. During the activity of environmental scanning, many parameters are analyzed that might influence the current and future development of an organization. The methods proposed in the following Chapters serve to enhance the scanning process. To underline the importance of Strategic Management and to provide a orientation to the reader in which context of Strategic Management the developed methods are to be placed a overview of Strategic Management and some popular frameworks is given.

Recent failures of well established organizations to not only recognize change in their business environment but also to react to it in a timely are warnings that emphasize the importance of environmental scanning, an activity suggested by Strategic Management. In various cases, trends which are not recognized or anticipated lead to the failure of established companies. Popular examples may be the problems the print media is struggling with as funding is directed to channels which offer a closer contact with consumers or the issues energy producers are facing as they are challenged by smaller producers using wind and solar energy.

Both examples have in common, that the challenges did not appear in an instance. Another similarity is, that the organizations or to be more precisely, the management, leading these organizations hit by the challenges seemed unprepared and unaware. A common mode for management to operate on is following a paradigm called Strategic Management. We present tools which can support the strategic process by providing additional information so that developments can be detected and anticipated. Our approaches use quantitative methods and aim for Three different time-ranges.

The term “Strategy” historically originates from the military. In Greek it was understood as the “art of troop leading”. A later interpretation by the Prussian general-major

Carl von Clausewitz describes strategy more abstract as "the usage of skirmishes for the purposes of war". In the military context, strategy can be understood as a holistic approach to define goals and means under consideration of resources and anticipated future developments. Carl von Clausewitz compares the trade of war with the trade of conducting business where problems of a similar nature arise. Henry Mintzberger discusses the definition of strategy in the business context detail in his article "Five Ps For Strategy" [108] analyzing why the use of the term can be confusing and solving the issue by stating that there can be multiple definitions which hold at the same time. Because his conclusive approach to define Strategy opens up multiple perspectives on Strategic Management we go into it in detail in the following Section 2.2.

The goals of Strategic Management can be derived from definitions given by Scholars in this field: In 1962 the American professor of business history at Harvard Business School, Alfred Chandler, defines strategy as "the determination of the basic long-term goals of an enterprise, and the adoption of courses of action and the allocation of resources necessary for carrying out these goals." [42]. With this definition Chandler sets the time dimension for a strategy as "long-term" without giving a more specific definition of the time frame.

At the end of the 1960s the application and exploration of strategy became a scientific discipline, at first in the United States of America (USA) and later branched out worldwide. Names for the new science were e.g. "Business Policy" and "Long Range Planning". The name emphasizes that a long time horizon is considered. As of today, the scientific field is mostly known as Strategic Management. The concept that a strategy determines the shape of an enterprise, as deducted from the definition of strategy by Chandler soon was up for discussion, opening up further research questions.

In summary it can be said, that Strategic Management is concerned with long term planning of goals and resource allocation to reach those goals. Resources are usually monetary assets, intellectual property but can be extended to more abstract elements like brand name and recognition. One important aspect is that the structure of an organization can be interpreted as a shape-able resource which effects are extensively studied. While Chandler suggests that the structure of an organization directly follows the goals it tried to archive, we assume that the issue is not binary but recognize a strong influence of the targets onto the structure. Because of their overall importance, goals should be defined using the SMART criteria which is an acronym for Specific, Measurable, Assignable, Realistic and Time-related. To be able to define goals according to SMART, extensive research into many dimensions must be done. Various dimensions were identified by researchers and put to use in planning methods.

To determine the goals, the environment is analyzed. Three different types of environment are distinguished: micro-, meso- and macro-environment. While all environments affect the organization, the abstraction-level and time-horizons of observations vary. For this thesis we define them as follows.

Definition 1 (Micro-Environment)**.** *The Micro-Environment is concerned with the organizational level, restricting it to internal factors.*

Definition 2 (Meso-Environment)**.** *Meso-Environment describes the transactional level*

and refers to market forces, e.g. competitors, customers, suppliers.

Definition 3 (Macro-Environment)**.** *The term Marco-Environment summarizes great shifts in demographic, political and technological development.*

Due to the sensitive nature of internal information of organizations, all methods proposed in later Chapters are analyzing aspects of either the meso- and/or the macro-environment.

The following Sections introduce some of the most common methods and frameworks. Their scope varies as a few only suggest a way how to find and explore dimensions relevant for the formulation of goals, while others propose complete feedback loops of the goals onto the structure of an organization and back onto the goals. For each method we will describe the scope and the dimensions it explores.

2.2 Tools and Approaches of SM

2.2.1 SWOT Analysis

The origins of the SWOT analysis are unclear. The first appearance dates back to the 1960s [22]. The acronym stands for Strength, Weaknesses, Opportunities and Threats. These fields of analysis are separated into internal (Strengths and Weaknesses) and external (Opportunities and Threats) factors of an organization. When transforming the environment into this categorization, micro- and meso-environment would be dominated by internal factors while the macro-environment consists of mainly external factors. The aim of the SWOT analysis is to identify factors which are important to archive an objective. In regards of defining a goal, SWOT can be used in an iterative process, where a goal is defined, analyzed accordingly and the result evaluated in weather the goal is achievable.

The SWOT analysis is applicable to a multitude of management issues as it provides merely an framework on how to evaluate a position, goal or project and does not directly propose any allocation of resources. The exact definition of the terms Strengths, Weaknesses, Opportunities and Threats can be adapted to the needs of the subject of inquiry.

As SWOT is a generic analysis framework, the type of dimensions which are of interest change with the subject. Nevertheless, typical dimensions in the context of Strategic Management are resources, experience and activities as internal factors as much as economy, funding, legislation, environment and future trends and their development as external factors.

2.2.2 PEST

To support decision makers to use the generic SWOT method efficiently for strategic management, suggestions are made on how to structure the forces which can affect an organization. Speaking in the terms of SWOT, the the PEST framework focuses on the external

factors or mapped to the various environments, provides an structured approach to the analysis of the macro-environment.

The acronym stands for Political, Economic, Social and Technological factors which indicates which dimensions are considered. With SWOT being only a mode of analysis, each of these areas can be subjected to a SWOT analysis. There are a multitude of extensions and variations of the PEST framework. For PESTLE, legal and environmental dimensions were added. STEEPLE added ethics and demographic factors. They all have in common that they propose a way on how to structure the analysis of the external macro-environment in regards of a specific target variable or a goal. A dimension not explicitly named is the time frame. This must be derived from the goal which is analyzed.

2.2.3 Porters Five Forces

In 2008 Michael Porter, Professor at Harvard Business School, criticized the use of SWOT for strategic management for being based on the assumption that "every case is different and that the relevant considerations are company specific" [23]. In an attempt to create a more structured approach to the issue of business strategy development he created the five forces analysis. While Porter himself considers the five forces to be part of the micro environment, later definitions of market environment places the five forces in the micro- and meso-environment which is a clear distinction to PEST and its variations. Accordingly, Porter sees his method best applied at the "line-of-business industry level". This is another distinction from the more generic PEST framework.

The five forces Porter identifies are: threat of new entrants, threat of substitutes, bargaining power of customers, bargaining power of suppliers and industry rivalry. For each item in the list Porter branches of several potential factors which can be considered in an analysis. Contrary to SWOT, there is no clear distinction between internal and external factors on any level of the five forces tree structure, but factors similar to those suggested by SWOT are connected with lower branches of the five forces.

In conclusion it can be said, that the scope of the five forces is at line-of-business industry level and the dimensions at the most abstract level are described by the five forces. The method is mostly used to evaluate the current position of an organization in order to get a better understanding where an organization starts when it defines new goals.

2.2.4 Five Ps

Earlier than Porter, Henry Mintzberger in 1987 creates five definitions for the term strategy in the field of strategic management [108]. In his view strategy can be defined as a plan, ploy, pattern, position and perspective which are referred to as the five Ps. He makes the case that one definition alone is not sufficient to describe what strategy means to an organization and determines that the various terms he recognizes as definitions are interdependent. For example as strategy is representing a plan it is at the same time representing a pattern to realize the plan.

By providing only multiple definitions and pointing out how these are interrelated, Mintzberger does not so much provide a framework as rather a set of aspects to consider when formulating a strategy and translating it into actions. The scope of the five Ps includes all areas in which strategic management is applied and does not fit into the internal/external factor classification created by SWOT nor in the micro-, meso-, macro-environment distinction used by Porter or PEST even though he makes use of both these distinctions to elaborate each of the five definitions.

Similar to Porters five forces, each of the five Ps has some dimensions attached which are not explicitly listed by Mintzberger. Mintzberger stresses the importance of future development for strategic management but does not give a suggestion of what time frame is sensible. In the same section he also explains that strategy could be potentially about anything which makes the five Ps more generic than Porters five forces.

2.2.5 Balanced Scorecards

Balanced Scorecards first appeared in 1992 in a publication by Robers Kaplan and David Norton [83]. The general idea of the Balanced Scorecards is the definition of financial and non-financial key performance indicators (KPIs). For the KPIs to be sensible, the choice of the KPIs must be made so that they are actually measurable, have a reference value available and it is possible to influence the value (similar to SMART). According to the authors, these KIPs should be partially of financial nature e.g. Return on Investment (ROI) or sales growth. The other part of the KPIs should be non-financial e.h. yield or life cycle of a new product towards maturity.

The first iteration of Balanced Scorecards suggests four "perspectives" for which KPIs should be defined: Financial, Customer, Internal business processes and Learning and growth. Critics of this iteration of the Balanced Scorecard argue that the "perspectives" limit the use of this method to specific industries because of the possible KPIs derived from the perspectives. As a reaction to this criticism various different instances of Balanced Scorecards with different perspectives are created.

The first generation of Balanced Scorecards is targeted towards commercial organizations with a very specific structure. The dimensions are loosely defined by the perspectives and specified by the KPIs derived from them. While the strategy of an organization can influence the KPIs the framework is rather rigid.

To address the issues of the first generation of Balanced Scorecards, the second generation connects the four perspectives closely with the strategy of an organization [84]. Instead of selecting from a set of common KPIs, the KPIs of each perspective are now derived from the strategy. While the dimensions are still connected to the perspectives, the development of KPIs in connection with the strategy provides a tool to clarify and possibly explore goals. The scope also increases. While the perspectives are still limiting, the free choice of KPIs extends the number of organizational units this method can be applied to.

In the third generation of Balanced Scorecards [94] the perspectives are moved to other instruments which are required by the Scorecards and this version offers a tool for strategy

development. A "top-down" approach is applied to define the KPIs. A vision statement is created in which the organization is described at a specific time in the future covering the required perspectives. This document is then used to derive a strategy and identify strategic goals. Because the method is generic it can be used in any type of organization.

2.3 Environmental Scanning

In the previous Section we discussed established methods used in Strategic Management. Though the approaches vary, they all require information about the future or about the current state of things. Different definitions of Environmental Scanning exists (see [49,60,109]), though they all agree that the acquisition and use of information about the external environment is part of the process. This thesis is following that definition, therefore focusing on the meso- and macro-environment.

A prominent analysis to evaluate the meso- and macro-environment is the commercial, technology-focused Gartner Hypecycle (see Figure 2.1). The main purpose is to indicate to organizations the maturity of various technologies. Depending on the maturity-level a organization might decide to invest into the technology at a given risk-affinity. Trends are differentiated into 5 phases: "Innovation Trigger", "Peak of Inflated Expectations", "Trough of Disillusionment", "Slope of Enlightenment" and "Plateau of Productivity". The Gartner Hype Cycle is based on the assumption, that every new technology goes trough these phases. More detailed information about the methodology used is not available.

Though primarily focused on Foresight, Popper [40] presents multiple methods the Foresight Diamond (see Figure 2.2) that can be used in the environmental scanning process. The author assigns the properties creativity, interaction, evidence and expertise to each method and arranges them accordingly.

The application and selection of these methods is left to the organization.

2.4 Relation to proposed Methods

Following the practice of Strategic Marketing, methods for environmental scanning can be distinguished by the dimensions time-horizon and type of environment that is analyzed. Popper uses the dimensions creativity, interaction, expertise and evidence as additional features of methods.

Using the Foresight Diamond (see Figure 2.2) as guideline, quantitative methods are a natural fit for computer aided decision support systems. Evidence based systems that follow a rationale and use objective information to generate insights benefit from deterministic computer systems. Creativity, interaction and expertise can be captured by software, though the need of interaction with users remains. Subjects touched in this thesis fall under the foresight methods: Patent Analysis, Indicators/Time Series Analysis (TSA) and Extrapolation. External factors are analyzed in the meso- and macro-environment. Short-, medium- and long-term developments are to be detected.

Figure 2.1: Gartner Hypecycle 2016; Source: Gartner (August 2016)

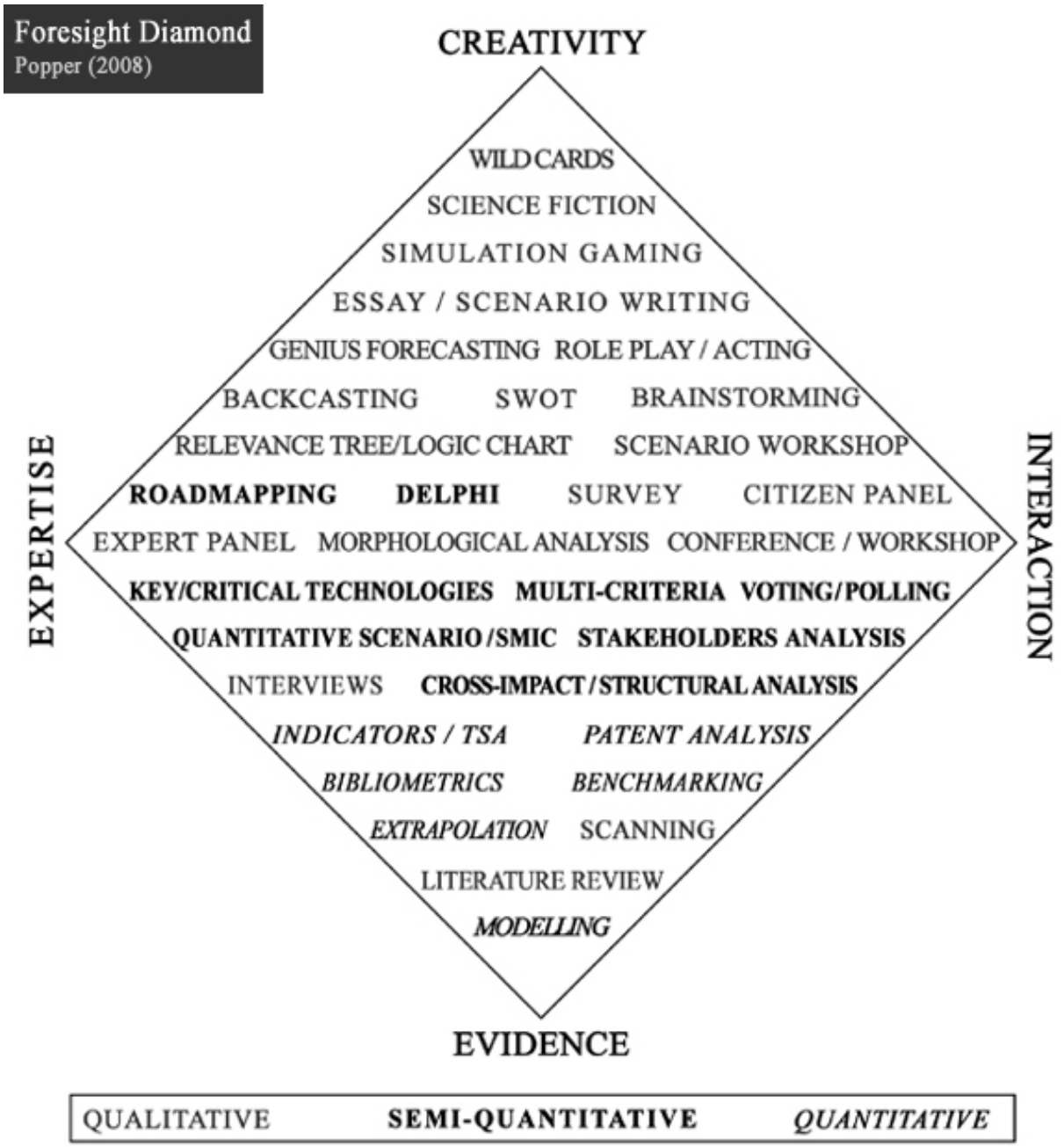

Figure 2.2: The Forecast Diamond; Source: The Forecast Diamond (Popper 2008)

In this thesis various methods are proposed to enhance the effectiveness and efficiency of Environmental Scanning. The selection of tools used by Strategic Management to determine is used to exemplify that there is a need for Environmental Scanning. From the forces described by Porter to the KPIs that can be defined by Balanced Scorecards, the methods benefit from a tighter feedback loop made possible by a automated Environmental Scanning method. To ensure that the quality of the data is not affected, the automatism is expected to offer meaningful metrics and suggestions instead of absolute decisions. Thus a decision-support-systems are proposed in the following Chapters.

Chapter 3

Preliminaries

3.1 Data Properties

Properties of the data affect which method is applicable and determines their performance. Some of these properties are: dimensionality, class distribution, amount of available observations, scales (e.g. discrete or continuous) and number of classes per item.

3.1.1 Curse of Dimensionality

The dimensionality affects the performance of various methods as the "curse of dimensionality" results in a sparse distribution of observations in a high dimensional space. The rational behind the curse of dimensionality is the rapid growth of observations needed to avoid sparsity. As an example we assume observations with d binary features. The amount of possible combinations is 2^d. So to avoid sparsity in a $d+1$-space, the number of observations has to increase at exponential rate. Sparsity in the data space is an undesired property as models derived from that space might not generalize due to insufficient observations. Another downside of high dimensional data is that methods, that rely on a distance measure e.g. k-Nearest Neighbor (see Section 3.2.1) are impeded. Observations become more similar to each other as with an increase of d, each dimension becomes less descriptive of an observation. One coping mechanism with the curse of dimensionality is feature reduction. Feature reduction can be achieved by feature selection or feature extraction. Applying domain knowledge, mutual information, correlation coefficients and recursive feature elimination are methods to reduce the number of features. With feature extraction, the available features are transformed into a lower dimensional representation. Popular methods include the Principal Component Analysis (PCA) [106], Linear Discriminant Analysis (LDA) [106] and approaches using NNs in combination with the previous methods or as stand-alone solution e.g. autoencoder [78]. For visualization of high-dimensional data, t-distributed Stochastic Neighbor Embedding (t-SNE) [99] is a popular non-linear method that projects observations into 2- and 3-dimensional space though higher-dimensional representations are possible.

3.1.2 Class Distribution

For classification tasks, the class distribution of the observations can bias the result of a classification methods towards a certain label. As an example the Support Vector Machine (SVM) soft-margin classification (see Section 3.2.2) with class A consisting of many objects and class B consisting of only a few representatives is used. Given a fixed penalty, a larger percentage of class B are allowed to violate the soft-margin decision boundary than class A which leads to a bias in the model towards class A. Common precautions to avoid this bias include weighting of classes, generating balanced training sets and adding synthetic observations for training which does require a model for the given class.

3.1.3 Dataset Size

The amount of available information is of relevance for most supervised methods. Most of the Deep Learning methods require a comparatively (e.g. to Random Forest) large amount of labeled data which is a direct result of the number of trainable parameters. In supervised learning, the amount of available training-data is further reduced due to the need of a test- and validation-dataset. A model is fitted to the training data set and the performance is evaluated on the test dataset. Later Chapters discuss the performance metrics and which to use for what type of data (see Section 7.3). In instances where the test-dataset is used to affect the training, e.g. stopping of training of the performance metric stops improving, a third dataset is required. The validation dataset is independent of the training process and used to validate that a model does sufficiently generalize. A requirement for a split of the data into train-, test- and possibly validation-set is that the contained observations in each set are still representative of the dataset as a whole. The representativeness is affected by the class balance and the respective observations. Stratification is used to assert that each set contains approximately the same percentage of observations for each class as the complete dataset. The probability that primarily unrepresentative observations are selected for either training or test decreases with the amount of available data. For datasets with a small number of observations cross-validation can be used to mitigate the effects of unrepresentative data [75]. k-fold cross validation uses k train- and test-datasets where all the train- and test-datasets are disjunct. Cross-validation is also used to handle overfitting [75].

3.1.4 Over- and Underfitting

The notion of overfitting describes a model that is more complex than necessary as a simpler model would fit equally well [75]. A symptom of an overfitted model is that it is not as generalizable as random noise contained in the observations of the dataset is incorporated into the model (see Figure 3.1). Regularization-methods for NNs include drop-out, early stopping and L1/L2 regularization (see Section 3.2.4). For most methods, the selection of hyper parameters plays an important role in avoiding overfitting e.g. the penalty parameter in SVMs (see Section 3.2.2). The concept of underfitting, on the other

hand, describes a model that is too simple to capture the aspects of the data (see Figure 3.1).

Typical scales are nominal, ordinal, interval, ratio and continuous. Depending on the scale, some methods might not be applicable to the data. Linear regression [127] on nominal data will not produce a usable result as the mapping of the nominal elements to labels is arbitrary. Limitations of the underlying data e.g. percentage which can not exceed 100% or scores pose additional limitations on the selection of a prediction method.

3.1.5 Number of Classes

The classification problem can be categorized in three different types. The simplest classification problem is the binary classification. Observations have exactly one of two possible labels. The SVM for example is a binary classifier. Problems with more than two labels are referred to as multi-class classification which is a generalization of the binary classification problem. Each observation of the dataset is member of exactly one of the k classes e.g. Human Activity Recognition (see Section 7). There are strategies available to extend the capabilities of binary classifiers to that problem (see Section 3.2.2). In multi-label problems each observation can be a member of several classes at the same time e.g. patent classification (see Section 5). Depending on the nature of the classification problem the performance metric must be adapted (see Section 7.3).

3.2 Classification and Regression

In this Section we discuss generic methods for processing specific types of data into suitable representations to gain insights. Also common classification and prediction methods are discussed as many state-of-the-art methods rely on them. The classification task is formalized as follows: a classifier c returns for an observation d a list $c(d) = p_d = (p_{k_1}, ..., p_{k_i})$ containing the label membership probabilities of d with $p_{k_i} \in [0, 1]$ and $k \in K$ where K represents a set of all available labels. The regression task consists of a regressor r and predicted variables Y and features X so that $Y = r(X)$. Classification and regression methods provide the functions c, respectively r.

3.2.1 k-Nearest Neighbor

To make use of k-Nearest Neighbor (kNN), a distance metric e.g. Euclidean or absolute distance must be defined on the vector space. Using that distance metric, kNN determines the k nearest neighbor to an item and uses the label of those neighbors to determine the label of the item in question. For the case $k = 1$ the nearest neighbor would determine the predicted label. For $k > 1$ the predicted label would be the most frequent label in the neighbors k.

An extension of kNN is weighted kNN [77] where the weight of an observation increases, respectively decreases with the distance between the observation and the item. This ex-

tension allow the application of kNN for regression tasks. By selecting the k neighbors, the method can predict the average value of an item.

3.2.2 Support Vector Machine

The Support Vector Machine (SVM) [52] is a machine learning model used in supervised learning. During training with labeled data this method learns a hyperplane. That can either be used for regression tasks as Support Vector Regression (SVR) or for classification. Due to the hyperplane which is also called decision boundary the SVM is a binary classifier. The side an item is located relative to the hyperplane determines the predicted class label. Initially, SVM was used as a linear classifier. But by applying the kernel trick, SVMs are able to classify non-linear data by projecting the data to a higher dimensional space where it is linear separable [20].

The Soft-margin is introduced in order to apply SVM to non-linear problems. This allows labels to be on the wrong side of the decision boundary, but a penalty is applied. The goal of the optimization is to minimize the penalty in order to find a locally optimal decision boundary.

In order to apply the SVM to a multi-class and multi-label problem with N various classes, the one-vs-rest (sometimes called one-vs-all) and one-vs-one strategies are available. Both strategies train multiple instances of SVMs. The one-vs-rest strategy requires n independently trained SVMs, one binary classifier for each class with the classes being $A \in N$ and $\forall x \notin A$. The one-vs-one strategy requires $\frac{N(N-1)}{2}$ classifiers as for each different pair of labels a new classifier is trained. Because one-vs-one accompanies a greater computational complexity, one-vs-rest is mostly used in the literature.

The SVM is used in many state-of-the arts methods and this thesis is using the SVM in several methods as part of an ensemble approach or as baseline classifier.

3.2.3 Random Forest

Random Forest is an ensemble method consisting of multiple decision trees. The algorithm uses tree bagging to reduce the variance of the model [54]. Random forest extends this concept by selecting a random subset of features at each candidate split in the learning process. This measure prevents overfitting [57].

3.2.4 Recurrent Neural Network

Some of the methods introduced in this work are based on classifying sequences of vectors. To classify sequential inputs, we rely on recurrent neural networks (RNN). NNs consisting of input and output layers and a variable number of hidden layers are well known to be universal approximators, able to approximate any given function with arbitrary precision given a sufficient number of hidden layer neurons. RNNs provide an even richer dynamic representation of functions, by including weighted self loops within their hidden layers, allowing them to operate on sequences of data instances of arbitrary lengths. Theoretically,

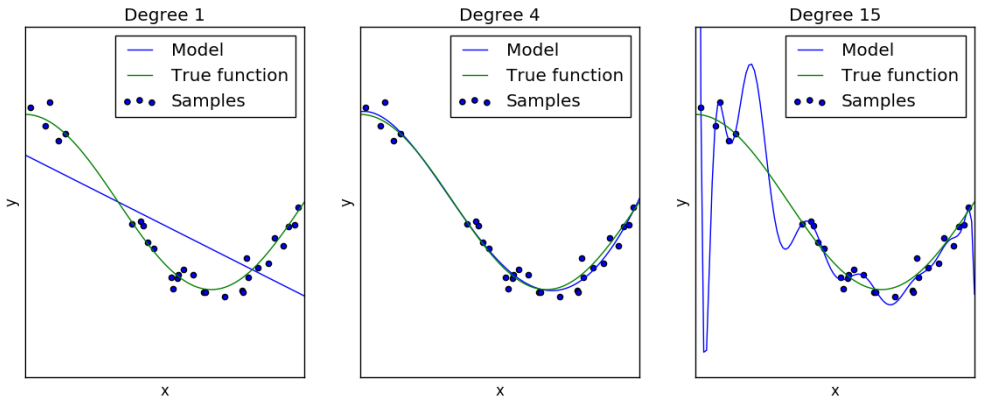

Figure 3.1: example for under- and overfitting

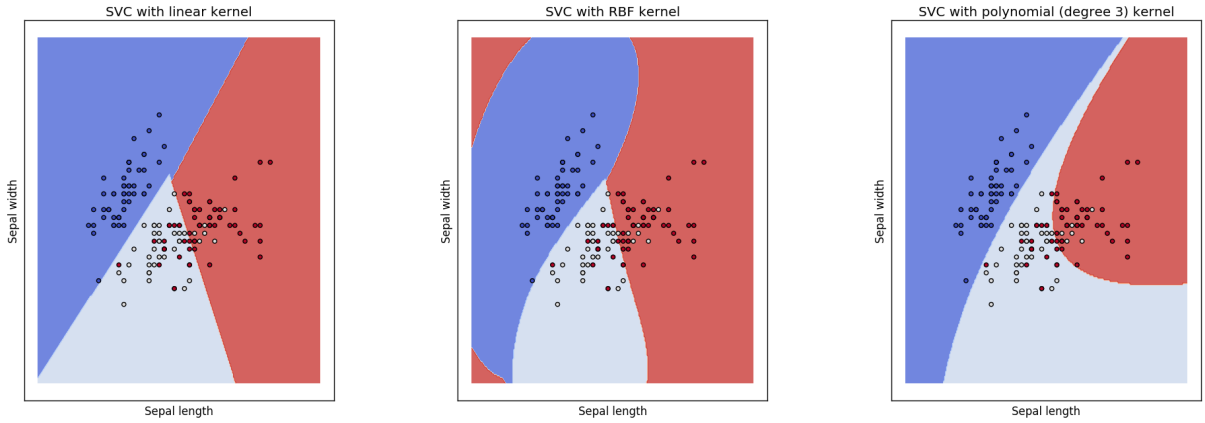

Figure 3.2: visualization of decision boundary for various SVM kernels

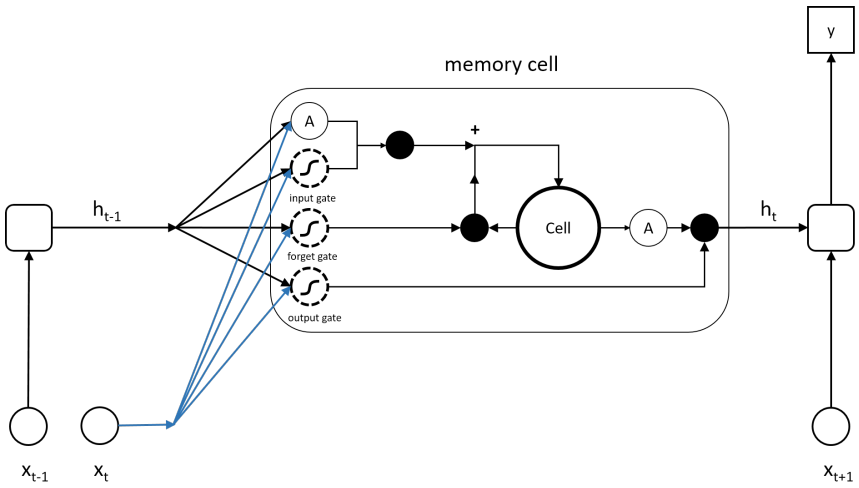

Figure 3.3: The LSTM architecture for the input x_t in step t and the output h_t.

RNNs are able to approximate any sequence to sequence mapping $h : (x_1, x_2, .., x_T) \mapsto (y_1, y_2, .., y_U)$ where $U \leq T$ [71]. In practice however, many RNN architectures face major difficulties in training over long sequences of time steps due to the vanishing/exploding gradient problem [30] caused by the weights on the recurrent connections. Long Short Term Memory (LSTM) models [79] were proposed to fix this issue. The sequential text classifier being proposed in Section 5.5 is based on LSTM neurons as is the best performing method for Human Activity Recognition (HAR) (see Section 7).

Being parameter rich, NNs are prone to overfitting. In order to produce generalizable models, various methods against overfitting can be employed during training. Drop-out is applied layer-wise and removes elements from the input based on a probability. This creates additional, unique training data and has shown to lead to more robust models [131]. Another approach to regularization is early stopping. This method monitors the training- and validation-loss as the loss indicates the fit to the data. A patience-parameter determines how often the training-loss is allowed to decrease while the validation-loss increases as that would indicate overfitting to the training-data [41].Normalization of the data with L1 and L2 norm is another popular approach towards regularization [41].

3.2.5 Long Short Term Memory

A second very important technique for our text classification scheme is the use of LSTMs. For classifying FHVs, it is extremely important that the employed classification architecture does not have a fixed time horizon but can keep any relevant information previously seen while parsing the document description. Though we cannot give an in-depth description on how the used LSTM [79] architecture works, we shortly explain how this architecture provides the required properties. An overview of the LSTM architecture can be found in Figure 3.3.

In contrast to regular RNNs, LSTMs replace the hidden neurons with a more complex

structure called a memory cell containing a hidden state and 3 sigmoid multiplicative gates called the input, output and forget gates. These gates dictate respectively whether the cell should accept input from the previous time steps, whether it should output its current state to the next timestep, and whether it should keep its current state. This setup leads to an RNN that is able to operate on very long time series with negligible attenuation of the gradient. For our use case, this means that relevant local context information can be stored while parsing the FHVs if it is relevant to determine the correct classes for a document.

3.3 Text Processing

Classifier and regressors only work with vectorized data. In order to process other types of information, e.g. audio, images, sensor data or text, a transformation function from the original representation into a appropriate representation is necessary. This thesis is focusing in the use of textual data and its representations. The first step in processing text is the creation of a dictionary D containing a mapping of each token t, that a tokenizer (see Section 3.3.1) extracted from a collection of documents $d \in G$, to an unique abstract representation t_r, e.g. an integer. To reduce the dimensionality of the dictionary, various methods can be employed: minimum frequency of a token, maximum frequency of a token, minimum number of documents a token is in. Methods like stemming [98], ignoring large and lower case, removing special characters (e.g. numbers, question marks, dots, exclamation-marks, etc...) and stop-word-removal [129] decrease the dimensionality of the dictionary further. The growth of a dictionary does not scale linear with the number of tokens observed. Heaps' law can be used to estimate the size of a dictionary (see Formula 3.3) resulting from a collection G containing n tokens. K and β are empirically determined. For the English language $10 < K < 100$ and $0.4 \leq \beta \leq 0.7$ [123] so the increase is approximately $\sqrt{n}$

$$|D| = Kn^{\beta}$$

In general two different paradigms exist: Bag of Words (BOW) and semantic. BOW treats the tokens independently from each other so that no context is available to a token. The context would be neighboring tokens in the text and their order. The disregard of context makes it impossible for BOW approaches to discriminate between synonyms. BOW is utilized to derive features from the text e.g. term-frequency (see Equation 3.1), document-frequency (see Equation 3.2) and rank. Semantic methods on the other hand regard the context of a token so that semantically similar tokens are embedded at similar points in the vector space. E.g. Word2vec (see Section 3.3.5) is considered a semantic approach as it is learning the vector-representation for a token either by predicting a context from a given token or by predicting a token from a given context.

3.3.1 Tokenization

Traditional tokenization of a western text is performed on the word boundaries which are usually denoted by whitespace. Asian languages require other approaches to tokenization, most relying on a dictionary [138]. n-grams [36], where $n \in \mathbb{Z}$ and $n > 0$ are used to capture phrases of speech and often co-occurring terms e.g. "Donald Trump" if the length of the phrase is n. For $n = 1$, each term represents a token. The 1-grams of the sentence "A quick brown fox jumps over the lazy dog" would result in the dictionary "A", "quick", "brown", "fox", "jumps", "over", "the", "lazy", "dog". 3-grams of the same sentence result in "A quick brown", "quick brown fox", "brown fox jumps", "fox jumps over", "jumps over the", "over the lazy", "the lazy dog". n-grams can be used in combinations. Dictionaries created with uni-, bi- and occasionally tri-grams can often be found in the literature [21, 28, 36]. Combinations of more than uni-, bi- and tri-grams are less popular, as the curse of dimensionality applies to n-grams and the utility of n-grams declines with increasing n due to sparsity.

3.3.2 TF*IDF

The most common way to represent text documents for retrieval and classification are Bag Of Words (BOW) approaches where a document $d \in D$ is described as a term frequency vector of the contained words [100] which disregards any ordering of the tokens. The most prominent BOW measure [26] is Term Frequency multiplied by Inverse Document Frequency (TF*IDF). There are various ways to calculate TF and IDF. The most basic form of TF is counting the term t in a document d. As this introduces a bias for longer documents, normalization is recommended. The Equation 3.1 uses the length of the document for normalization. Other approaches are logarithmic scaling and using the most frequent term of a document to reduce the bias.

$$\mathrm{tf}(t, d) = \frac{f_{t,d}}{\sum t', d} \tag{3.1}$$

$$\mathrm{idf}(t, D) = \log \frac{N}{|\{d \in D : t \in d\}|} \tag{3.2}$$

The intuition behind the inverse document frequency is, that terms which occur often and in most if not all documents $d \in D$ of a corpus are not very descriptive thus Equation 3.2 diminishes the weight of such tokens. Similar to Equation 3.1 various variations of Equation 3.2 exist that strive to optimize how the descriptives of a token in relation to D is calculated.

$$\mathrm{tfidf}(t, d, D) = \mathrm{tf}(t, d) * \mathrm{idf}(t, D) \tag{3.3}$$

TF*IDF (see Equation 3.3) returns a score indicating the relevance of d relative to t. In information retrieval it is common to use a fixed number of terms, referred to as a dictionary, which are derived from the corpus. Common methods to limit the size of

the corpus are a required minimum term frequency and a maximum term frequency as much as a minimum and maximum document frequency. The vectorized representation of document d using TF*IDF has the length of the dictionary, as for each element of the dictionary the TF*IDF score in regards to the document is calculated. As the dictionary is usually in the hundred-thousands of tokens, the resulting vector space is high dimensional.

3.3.3 BM25

A more refined TF*IDF-like method is the Okapi Best Match 25 (BM25) [120] scoring function which also accepts a query consisting of multiple terms $Q = t_1, ..., t_n$ instead of single tokens. This enables BM25 to directly measure the similarity of whole documents, where TF*IDF requires an additional distance metric. Similar to TF*IDF, variations of BM25 exist which address problems in document length normalization and term distribution. One of the most commonly used instances can be seen in Equation 3.4.

$$\text{score}(D, Q) = \sum_{i=1}^{n} \text{idf}(t_i) \cdot \frac{f(t_i, D) \cdot (k_1 + 1)}{f(t_i, D) + k_1 \cdot \left(1 - b + b \cdot \frac{|D|}{avg(D:|d| \in D)}\right)} \tag{3.4}$$

In later Chapters we use BM25 as a baseline for classification performance as experiments show that it outperforms TF*IDF independent of distance measure. Because of the impressive performance, various database products e.g. Lucene recently switched to BM25 as the default similarity measure [1].

3.3.4 Latent Dirichlet Allocation

A drawback of weighted term frequency vectors is that documents about the same topics might not be recognized as similar if the used terms are not overlapping enough. Thus, topic-models are introduced, mapping words to a latent representation space describing possible topics of which the documents in the corpus are composed of to a certain degree. Currently, the most prominent topic-model approach is Latent Dirichlet allocation (LDA) [38]. LDA is a generative model that uses sampling techniques to iteratively refine a predefined, fixed number of topics so that the final model would produce a similar token distribution to a training document given identical topic distributions. Different extensions to LDA allow online processing [21] and hierarchical topics [116]. Though topic models describe the connection between words and topics, they do not consider the local context in which a term is used.

3.3.5 Word2vec and Extensions

Bengio et al. [29] propose a novel approach considering local context and the order in which terms occur in a text. In this approach, a neural network learns a low-dimensional token representation and predicts the next token within a window around each term based on the learned representation. This work is built upon by Mikolov et. al [107], who

introduces word2vec where the network architecture is simplified by only learning the token representation as a lower dimensional vector which is also referred to as an embedding. Word2vec comprises two approaches (see Figure 3.4): the Skip-gram Model where the context is predicted based on the token and the Continuous Bag of Words Model (CBOW) where the current token is predicted based on its surrounding. While skip-grams provide better results in semantic tasks, the CBOW architecture is faster in training and slightly more performant in syntactic benchmarks. Stochastic Gradient Descent (SGD) and Back-propagation are used for training. Additionally, word2vec introduces negative sampling as an extension of noise contrastive estimation which avoids computing the normalization factors of the softmax function in the output layer. Thus, negative sampling greatly speeds up training of word2vec models.

Le et al. [95] extend word2vec by introducing PV. PV add an additional vector space representing entire paragraphs. The vector space is learned jointly with the token vectors of the tokens within the paragraphs. Similar to CBOW and Skip-gram for word2vec, PV includes two strategies: PV Distributed Memory (PV-DM) uses a sliding window of tokens on the target paragraph and learns embeddings representing the paragraph as well as the tokens. For a given sequence of tokens the paragraph representation is averaged or concatenated to the token vectors and then used to predict the middle word in the window. The error gradient of this prediction is then used to train both embeddings.

The second strategy, PV Distributed Bag of words (PV-DBOW) uses the token in the paragraph as input and tries to predict the words in the current window. In [61] an extension to PV is proposed which builds on the structure within a documents. The method learns a single model from training instances from all structural context levels. The resulting vectors for each context level are then concatenated to represent the document. Evaluation on a binary classification task showed degrading performance with each added level of detail and a significant increase in training time. Though this method is similar to the Fixed Hierarchical Vectors (FHVs) being proposed in this work, FHVs use different models for different parts of the document and enable a sequential classification method leading for increased performance when appending additional levels.

Our method FHV (see Section 5.5) strongly relies on the PV model due to its ability to learn embeddings based on any type of content. Therefore, we will give a short summary on computing the PV distributed memory (PV-DM) model as described in [95]. The goal of PV-DM is to learn a mapping $\Phi(P)$ for an input paragraph P to a q-dimensional vector space describing P's contents. Formally, $P = (t_1, \ldots, t_m)$ is a sequence of m tokens representing either words, shingles or other textual primitives. The mapping is based on two linear transformations D and W. Whereas D maps the paragraph id pid to the target space, W maps the tokens in the local context of each token in P to the target space. In particular, the local context for token $t_i \in P$ is represented by its local context $t_{i,context} = \{t_{i-w}, ..., t_{i-2}, t_{i-1}, t_{i+1}, t_{i+2}, ..., t_{i+w}\}$ surrounding t_i with the window size w. Since every token $t_i \in P$ and the given paragraph id provide a q-dimensional output vector, all output vectors are either concatenated or averaged into a single q-dimensional vector. To learn both matrices, PV train a two layer architecture which is depicted in Figure 3.5. The idea of training is to maximize the average log likelihood to predict a

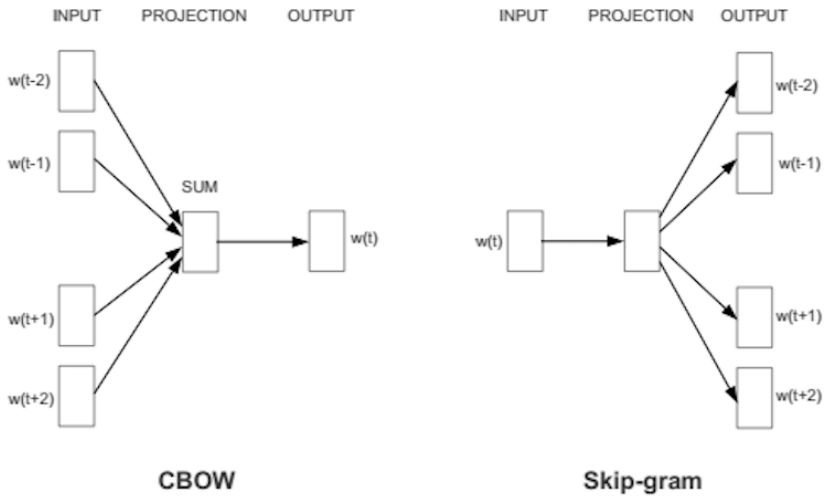

Figure 3.4: An overview of the word2vec-methods [107]; Source: [107]

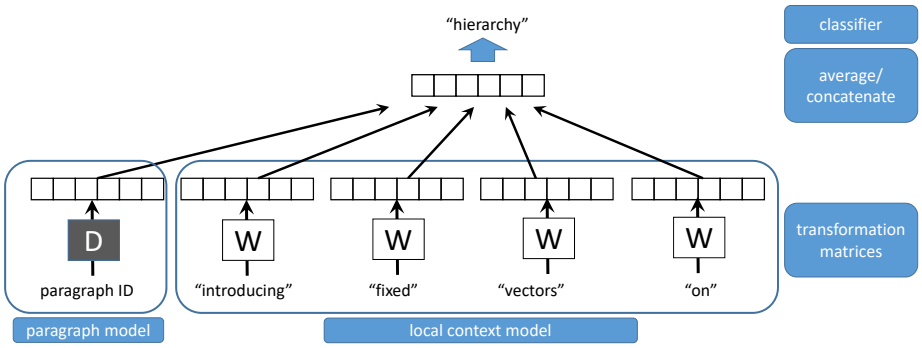

Figure 3.5: An overview of the employed PV-DM model [95]. The paragraph matrix D is trained to provide the global context for each token window representing the local context.

token t_i based on its context $t_{i,context}$ and the paragraph id pid it occurs in:

$$\frac{1}{m}\sum_{i=w}^{m-w}\log p(t_i|t_{i,context},pid) \tag{3.5}$$

Now, $p(t_i|t_{i,context},pid)$ is trained as a softmax classifier which is based on a linear prediction functions:

$$p(t_i|t_{i,context},pid)=\frac{e^{y_{t_i}}}{\sum_j e^{y_j}} \tag{3.6}$$

where $y_{t_i}=b+U\Phi(t_{i,context},pid))$. Let us note that Φ can be applied to $t_{i,context}$ as well as the complete paragraph P because adding the tokens of the complete paragraph only adds additional vectors for the average function aggregating the final result. Training is performed with Stochastic Gradient Descent (SGD). As in word2vec, negative sampling is applied to speed up softmaxing. For mapping a novel paragraph, unknown paragraph ids have to be integrated into the paragraph matrix D. Thus, new columns have to be added to D and afterwards, D is optimized again using gradient descent while freezing all other parameters h, U, and W.

Chapter 4

Short-Term - Web Data

The work of this Chapter is published as [133].

4.1 Introduction

every day more information is made available than a human being can possibly process as a result of Digitalization and Big Data. At the same time monitoring of relevant topics gains priority as the speed of information dissemination and technological development are increasing. As a result, feedback loops that formerly took several days, must be sped up to take only a fraction of that time now e.g. in the field of journalism [85]. Automated systems that are able to emphasize aspects of topics based on quantitative methods help to reduce the information load while maintaining flexibility.

The World Wide Web (the Web) is the number one information source of our time. Almost all of the knowledge of todays world is available online in the form of Hypertext Markup Language (HTML) documents. Current news gets published almost in real time [85]. Current events and anything people are interested in is mirrored in the content of popular and relevant on-line documents. For a long time now the Web is in an transition from a tool used by scientists to an everyday commodity used by anyone [137]. While most of the web pages are still written in English as the de facto lingua franca of science, the part which uses other languages, is growing in absolute and relative size [7].

With the global community expressing its interests and worries online (see Figure 4.1) it seems appropriate to tap that potential and develop a system that is able to perform automatic environmental scanning. To achieve this, it is required to have an reliable detection mechanism for activities and trends in the environment of a given topic. This mechanism must also account for different languages and be able to distinguish trends as signals from occurring background chatter or noise.

This work focuses on activity and trend detection on Web data. The developed system enables the collection and analysis of potentially relevant data in a scalable and robust way. Several metrics for automated activity and trend detection are proposed in this work. These systems rely on third-party services to perform the relevance rating of documents

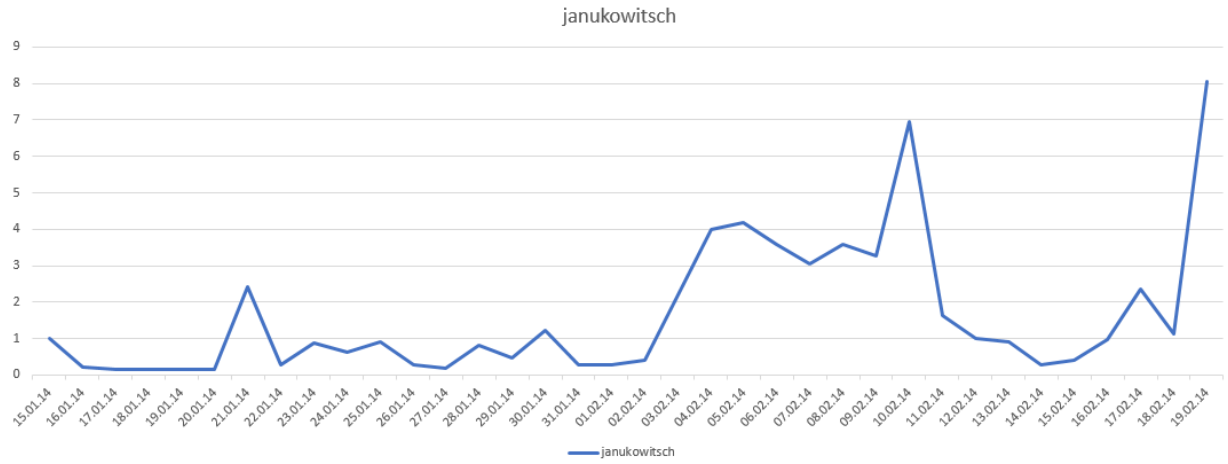

Figure 4.1: Relative term frequency of "janukowitsch" at the brink of the Ukraine unrest; Wiktor Janukowitsch was president of Ukraine at that time

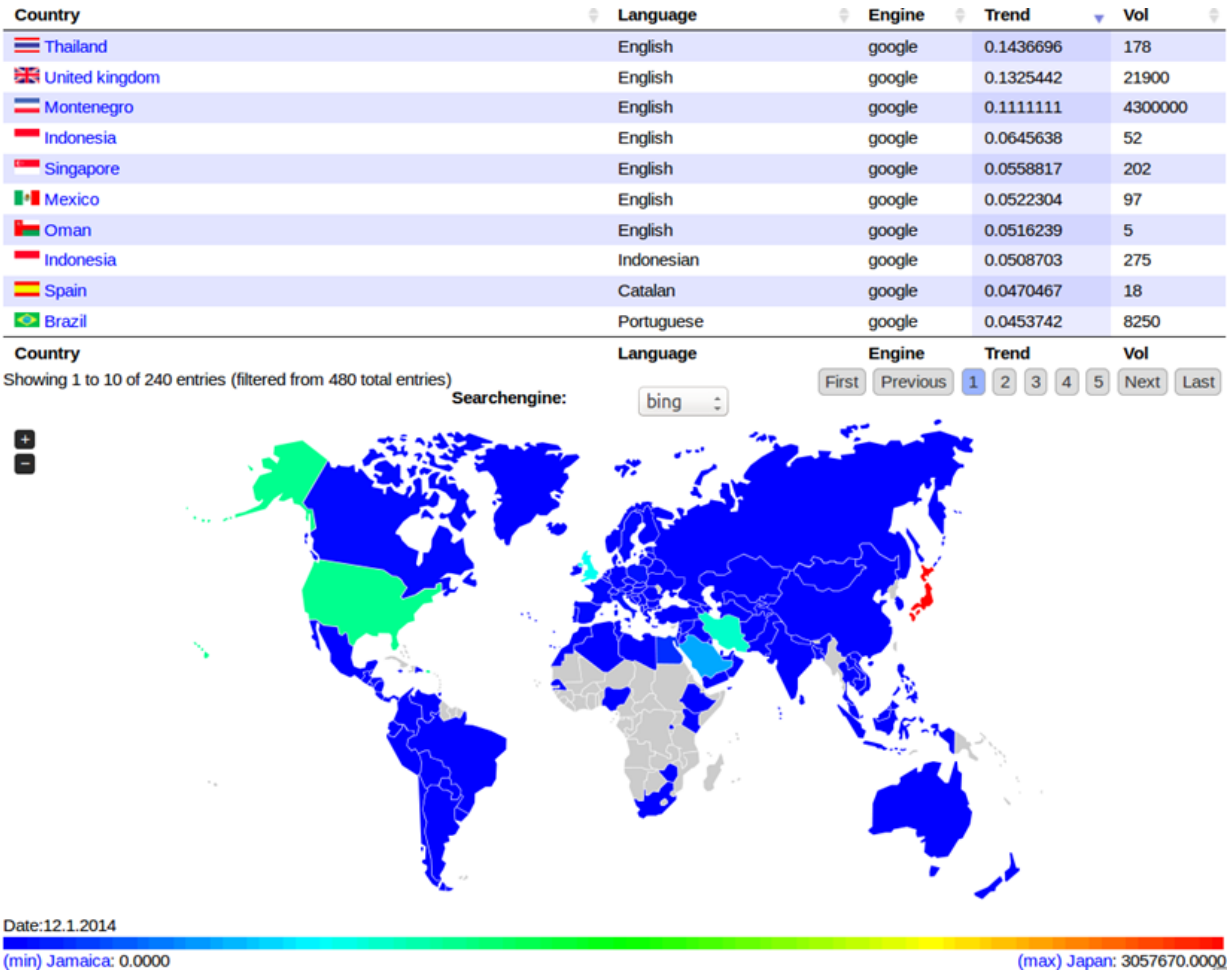

Figure 4.2: Screen shot of the prototype

regarding a topic. These methods are evaluated in regard of their ability to detect trends and check their performance for different settings e.g. specificity of a topic. To evaluate the methods and as a proof of the concept we developed a prototype (see Figure 4.2). The web based application is easy to use and provides quick access to the results of the proposed methods.

The discussion of this Chapter is structured as follows. Related work is presented in Section 4.2. Section 4.3 introduces the definitions used throughout this Chapter and presents the concept for data acquisition, data representations and metrics for activity- and trend-detection and localization. The empirical evaluation of the detection metrics and data acquisition is covered in Section 4.4. Section 4.5 contains the conclusion and gives an outlook for further research.

4.2 Related Work

The existence of many commercial systems for trend detection emphasize the importance of such an application and indicate that there is demand for automatic trend detection and monitoring in open source intelligence (OSINT) e.g. Web data. Though the methods used are seldom detailed, they are omitted from further inspection with the exception of Google Trends.

Google Trends reflects the frequency of search phrases over time and can distinguish the geographic location where a search query originates. The absolute amount of search queries is hidden. Although it is possible to derive a momentum and the intensity of a trend, a comparison between trends is misleading. Due to the high frequency of updates of Google Trends, it is used for "now-casting" as in predicting the current state. Other institutions that provide similar numbers might have a coarser time resolution [46]. The methods Google Trends is using for calculating the number of queries and to localize the origin of a query is not published. Missing trends for less common keywords indicate that a minimum volume of queries is necessary. This property reduces the utility of Google Trends for applications where niche-topics are the subject of monitoring and analysis.

Trend-detection on textual data is often associated with social-media services e.g. Facebook and Twitter. The Application Programming Interface (API) of Twitter allows easy access to tweets including hashtags and meta-data. In addition to the raw text, hashtags add semantic meaning and data like location, language and username are available. This makes twitter data an attractive research subject not only for trend detection, but also for user obfuscation and identity linkage [128].

Authors of [104] detect "bursty keywords" characterized by sudden increase of term-frequency and use co-occurrence to establish the token-composition of a trend. These trends are analyzed in depth for data sources and geographic locations. Entity detection is also applied.

The focus of [126] is the detection of "significant topics". The authors treat the detection similar to outlier detection and suggest a metric similar to the z-score to determine if a topic is significant. The authors use exponentially weighted moving average and variance

over time-series in combination with a bias-term β. The bias-term is used to reduce noise and to stabilize the results. A focus of that paper is the efficient processing of large quantities of data. Co-occurrence analysis is used to increase the number of tokens associated with a trend.

To grasp the zeitgeist of society, authors of [31] use TF*IDF and relative normalized term frequency to identify trending topics on uni- to tri-grams.

Though the use of Twitter data is popular given the amount of free, structured data and the real-time aspect, a bias towards the target group of Twitter is undeniable. Another issue with Twitter data besides being representative is the pervasiveness of bots. Bots are computer programs that automatically tweet. The work of [50] analyzes the composition of the Twitter user-base categorized into humans, cyborgs and bots. Cyborgs are either human-assisted bots or bot-assisted humans. The conclusion is, that of the 500,000 accounts 48.7% are classified as human.

Another popular source for trend detection is Facebook. Though the data is less accessible, Facebook has about 2 billion active users per month, compared to Twitters 328 million. Public posts are retrieved and Post Topic Identification is applied [56]. A modified version of TF*IDF is proposed to identify post topics. Trending topics are divided into three categories: popular topics, disruptive events and daily routines depending on the properties of the speed and scale of information distribution. The authors of that publication emphasize the effort it takes to retrieve sufficient data from Facebook.

Contrary to the data from social networks, Web data is less structured and accessible. While the common Web document is written in HTML, there is no consensus on the semantic structure (e.g. headline, abstract, etc.) and syntactic errors are common as only 4.1% of all tested web pages validate [3] against the standard. Another issue is the vast amount of data. The web is estimated at 4.47 billion pages [13] which must be skimmed for relevant documents. That amount of information exceeds the analysis capabilities of any non-specialized organization. Literature utilizing web data is mainly associated with information retrieval at scale and relevance of documents [100].

4.3 Method

4.3.1 Preliminaries

Definition 4 (Token)**.** *A token t is a textual primitive e.g. a word or a n-gram.*

A general course, prevailing tendency or a drift of something is referred to as a "trend". Apart from the colloquial use of the term "trend", a trend a has the property of a direction, e.g. increasing or decreasing and is associated with a specific momentum. Furthermore, a trend refers to a specific subject, or token t. The momentum indicates the amount of acceleration of a trend. In the context of this work, trends are represented as signed numbers, where the sign $sgn(a)$ indicates the direction of a trend and the number $|a|$ the momentum. To estimate a trend, observations over time represented as time-series are

used. In this work we assume that the time passed between two consecutive observations remains constant.

Definition 5 (Trend)**.** *Let the count of a token t in a corpus of documents $d \in C$ be $c_D(t)$. The relative term frequency at time i is defined as*

$$f_{D_0}(t, D_i) = \frac{c_{D_i}(t)}{c_{D_0}(t)} \tag{4.1}$$

for $c_{D_0}(t) > 0$. A time-series consisting of i time-steps of the relative token frequency is $S(t) = (f_0, .., f_i)$. The trend $a = g(S)$ is the incline of a line fitted with least square linear regression to S.

The notion of a topic is used to describe tokens which are used to determine the relevance of a document regarding the interests of a user.

Definition 6 (Topic)**.** *A topic T is defined as a set of tokens $t \in T$. If $U \subset T$ we refer to U to be more generic than T and T being more specific than U.*

Activity is more generic than a trend, as direction is omitted. The intensity of activity is relative to previous states.

Definition 7 (Activity)**.** *Activity describes the delta between points in time relative to a greater context.*

Due to the definition of trend, a small variation of term-frequency on a low-count token leads to a strong trend. To adjust for that behavior the notion of reach is introduced. The reach indicates the dissemination of a token in a given corpus.

Definition 8 (Reach)**.** *The reach r of a token t is defined as the ratio of documents d out of a corpus D that contain the token t.*

$$r_D(t) = \frac{|d \in D : t \in d|}{|D|} \tag{4.2}$$

4.3.2 Concept

The proposed concept for data retrieval and analysis consists of six steps:

1. defining topic
2. translating topic
3. selection country and language combinations
4. querying search engines for documents
5. downloading the documents

6. analyzing the documents and meta-data

Step 2-6 are repeated in user defined intervals until it is stopped by the user.

The task of presenting the results of the documents is considered an extra function, independent of the data collection and analysis process. The idea behind the concept is that experts define the general area in which to look for trends. An expert might be tasked to find new trends and/or centers of activity and developments for a certain topic. Without any further knowledge it may be hard to know where to start.
The expert defines a few topic specific search terms in his or her own language and uses a web interface to transfer them to a web based tool. The tool can automatically translate the search terms in the most used languages and allows the user to adapt the automatic translation in case of mistakes [113].
In the next step the user selects which language should be considered for which country. This saves a lot of resources if not all combinations are required. By default for each country the official languages (if available) and English will be preselected as almost 70% of the Web is in English [100] and English therefore can be assumed as an universally used language independent of the country.
Using the county and language combinations together with the translated search terms the system starts to query one or more search engines for URLs of relevant documents. Dedicated web search engines are specialized in ranking web data according to relevance in regards to the interest of a user expressed by keywords. The ranking methods are not published though it is rumored that Google for example uses approximately 200 different criteria [5]. As Google and Bing do punish scraping [105] we use the official APIs which for Google is Custom Search Engine (CSE) and Bing Azure Websearch. Scraping retrieves the resulting URLs from the official HTML result pages of the search engine by simulating a user and a browser via software. Web search engines disallow the use of scraping because software does not click on advertisements or leaves personal information so there is no source for revenue for the service provider. The punishment for scraping is often done by excluding the client from the service which would be detrimental to our goal of retrieving URLs in fixed intervals. One run of downloading documents is referred to as one crawl and the component doing the downloading the crawler.
The fixed interval between crawls is termed the crawl interval. The shorter the crawl interval is, the less time the data has to change. For very static topics a long search interval might be sufficient while for active topics like news a daily search interval is already at risk of loosing developments. For the analysis of the results of each crawl it is important that the interval between crawls is equidistant.
The crawler will then start to download the documents indicated by the URLs from the search engines. Not all documents are available to the crawler (see [135]) and not every document is actually usable for trend or activity detection e.g. because of the file type. For this reason the crawler has to be exceptionally robust. Besides the text contained in the web documents, additional information is available. The data from the search engines includes a list of URLs and an estimate of overall matches to the query. The results of

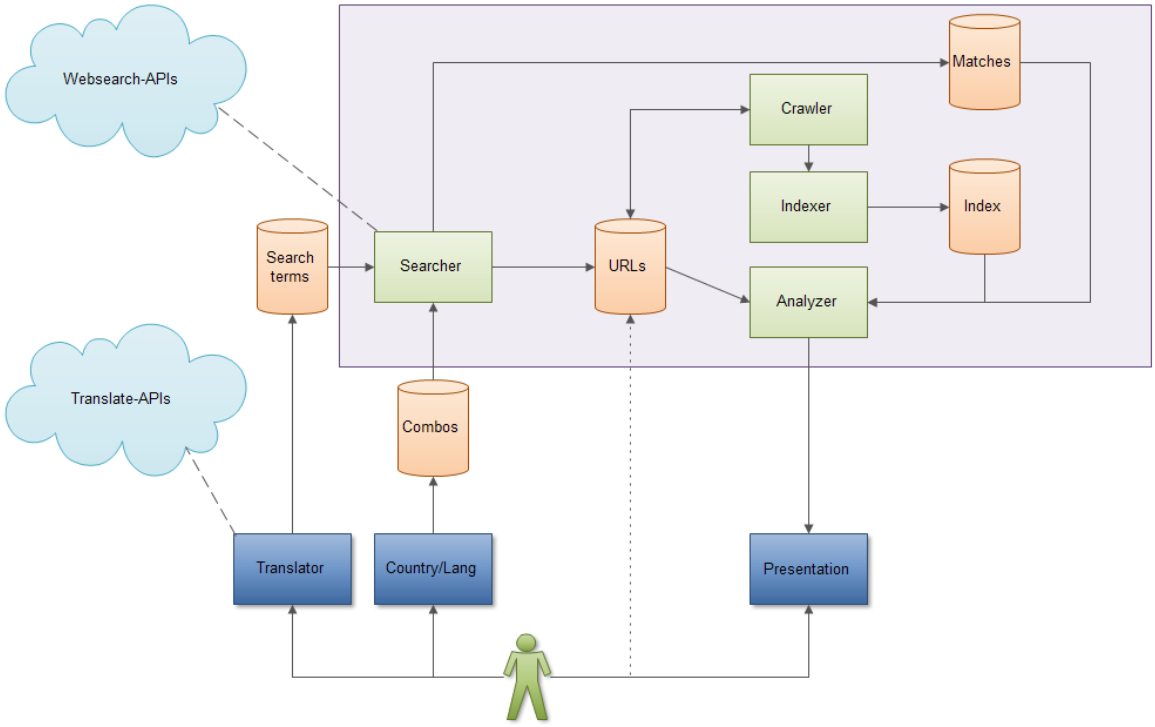

Figure 4.3: System structure

multiple crawls and the time when each crawl was done is also present. From the crawls we can deduct several data structures we can use for further analysis. The system is shown in Figure 4.3. The whole violet box is operating unsupervised and is responsible to perform the crawls in predefined intervals.

4.3.3 Crawling the Web

The requirements for the crawler are robustness, speed and correctness. For the crawler to be robust, a software architecture is needed, which is able to check itself and recognize when a critical state is reached (see Section 4.1). The best case would be that the software does not reach a state in which it crashes or hangs for an undetermined amount of time. Unfortunately does the Web consist of unforeseeable data and transmissions. The effort required to cover all eventualities exceeds the gains. As a reasonable trade off between effort and gain we use an architecture which continually checks itself. Certain thresholds and required return parameters managed by a master process ensures that subordinate processes are constantly monitored and in case of misbehavior get shut down and replaced by a new instance.
We define the web as a directed graph $G = (U, E)$ with $u, v \in U$ being the nodes and $(v, w) \in E$ being an edge. A node represents an URL and an edge represents a hyper link from one URL to another. A crawl path P is defined as an ordered series along nodes visited by the crawler $P : (u|u \in U)$. We define the depth of a path P as $d_P = |P|-1$. The initial seed list of URLs U is populated at runtime by new links found in parsed documents which are tagged with an incremented path depth. The stop-criteria for the crawler is if the seed list is either empty, or all documents in the seed list have a depth greater than the value defined by the user.
Each connection to access an URL is handled by an individual process. A timeout is put on the duration of an connection so that exceptionally slow servers or systems which discard any incoming IP packages don't block the crawl process. Some pages restrict the access for crawlers. A set or rules can be defined by a web page in a document called "robots.txt". The crawler tries to retrieve this document and parses it so that it can follow the rules. When the document is available and the rules restrict the crawler from access or parsing fails, it is assumed that the page does not allow crawlers access to the URL so it is discarded. If access is granted or no "robots.txt" present, the HTTP status code is checked weather it is the expected 200 (OK, The request was fulfilled).
An issue can arise if the graph contains a circle, which is a given crawl path which translates to multiple visits of the same URL. This is a waste of resources at our end and at the end of the URL. In addition to that it raises the issue of weighting results, where URLs which are included more than once can influence any analysis of the data later on, as any contribution by that site will be weighted. We address this problem in three different ways: first, the user is required to define a maximum path depth i which stops the crawler when $d_P \geq i$. The second protection is a shared list of all crawling processes of visited URLs. While the previous two methods prevent the download of a duplicate, there are pages which explicitly target crawlers. These pages, called spider traps, use dynamically

generated URLs to disable the second protection, to let a crawler remain on the site which provides usually the same content over and over. The maximum path depth i limits the damage to i instance of the same content in the database. For enhanced protection we use a checksum over the parsed text content of a page. We use the checksum over the parsed text as some pages tend to include a time stamp or other dynamically generated content to create the perception of a new page. While these methods give not guarantee against duplicates, a measurable increase URL diversity is observed.
Other issues arise from the content an URL provides. The crawler aims to scrape text from HTML documents, but a lot of URLs point to different formates r.g. PDF or ZIP files. To assure that only resources providing text are retrieved, the Multipurpose Internet Mail Extensions (MIME) type is checked and the Content-Type if transmitted in the HTTP header. If these types do not match text content from that URL is discarded and the URL is marked as unfit for current and future analysis. If the type suggests that text is transmitted or the type is not set, a HTML parser is used to extract all text nodes from the document. If the extraction fails, it is assumed that either the type did not match the submitted content or that an encoding issue occurred.
In a multilingual setting the correct character encoding of an resource is crucial for further processing e.g. tokenization and to the subsequent construction of an index. The HTML standard strongly suggests the declaration of the used character encoding. In the cases where it is not set, we assume that UTF-8 is used.
The data collected by the crawler is processed by the Indexer which creates dictionaries with additional information from the data. These index structures are the primary data source for some of the metrics described in Section 4.3. The indexer creates a new dictionary from the tokenized texts extracted from the documents. The dictionary contains information about the absolute term frequency, the documents it occurs in and how often it occurs in each document. The index is created for each country-language combination independently and a new index is created each time the crawler is started so that a time series of index structures is created.

4.3.4 Methods

We analyzed several of the following methods and evaluated them in regards to their ability for activity measurement and trend detection:

- Relative Term Frequency
- Estimated Matches
- Page Updates
- New Sources

All methods have in common that results can only be seen relative to previous results. Informally we define trend as a directed activity. An activity is a change between t_0 and t_1

which is detectable within the data we collect. Dynamically created content and information retrieval technology used by search engines creates also activity. So our methods have to filter the changes in the data for noise and for changes actually resulting from human behavior. With this trend definition everything done by humans would qualify as a trend. The direction of an activity can be deducted by an increase or decrease of an inspected parameter. While there is no problem with seeing everything as a trend we use reach (see definition 8) as a property of a trend which allows the user to filter for trends which occur on a given percentage of documents. One objective unit for a trend we propose is a term as terms pose a standard used by most documents.
Activity is more generally the deviation between two measurements. This definition allows us to utilize all activity detection methods to some degree for trend detection as a trend is a directed activity. On the other hand activity measurement methods do not necessarily allow us to detect trends as the direction (e.g. rising/falling) component might be missing or carries no value. We use the activity to measure if a topic is still active e.g. there is still research done or has become static.

4.3.4.1 Relative Term Frequency

This method requires a time frame as an additional input as terms might not be present in the documents crawled at a given time. While the absence of a term could be used as an indicator it saves memory and improves the performance when terms are ignored that are not present from t_0 to t_i.
The relative frequency can be used as an indicator for activity as any change in the relative frequency can be tracked back to change in the underlying text. A high activity is expressed by huge changes in the overall relative term frequency. To get good results it is necessary to look at several, if possible, all terms.
In combination with least square linear regression a long term trend can be derived from the data and enable the user to focus on a few exceptional instances of terms for further inspection. Because this method needs to tokenize a text in order to extract terms, problems arise with languages which do not have an explicit word delimiter (e.g. Chinese and Japanese). For other languages this method exceeds expectations and will be part of future research to improve the trend filter and make exceptional trends more obvious. Tokenizers is an area of active research [65].
We can use this method for the dimensions: language, country, time and text corpus. Depending on the flexibility of the time frame and the inspected dimensions the computational complexity of the method is rather high as the tokenizer needs preprocessed input where the documents are sanitized and cleaned of any markup. While this can be pre-calculated if the time frame is fixed, non fixed time frames need to be calculated on demand. This can be supported by index structures which are generated after a crawl is finished, but for this to be feasible the dimensions and aggregation level needs to be defined in advance.
Visualization is done on a per term basis. For each term a graph is generated for the inspected time frame. The terms are presented as a sortable list. The list is sortable by

term, reach and incline of the trend line.

4.3.4.2 Estimated Matches

The number of "Estimated Matches" is acquired during a crawl. It is the number of estimated results as returned by a search engine as part of the reply to a specific query. As all queries are country and language specific we can assign the number of estimated matches to a country and a language.
Depending on the target variable, the results of one crawl may be sufficient. Questions like "where in the world are the most matching documents located" can be answered. Because all languages of a country are treated separately we aggregate the number of estimated documents for each country over all corresponding languages.
If a trend is to be detected, at least two crawls are required to retrieve two data points which then can be used to estimate a trend.

4.3.4.3 Page Updates

This method requires at least the data of two crawls to work. The method extracts all URLs which are part of the crawls c_{t_0} and c_{t_1}. As the documents were retrieved by the crawler and are stored locally at t_0 and at t_1 we can compare these two versions of the document. If they are dissimilar we assume that an update of either the content or the document structure was made. As the absolute number of updated documents varies as the index of the search engines changes and the ranking of the documents get reevaluated we use the relative update rate of all documents which are part of the intersection of c_{t_0} and c_{t_1}.
Activity is detected easily as an updated document indicates that either by a user, web master or at least by an automated system the effort was undertaken to change content. Counting any change as activity is a generalization we chose on purpose as it drastically improves speed of analysis compared to other approaches and avoids the problem of quantifying the degree "change" of documents. The downside is that things like an automatically generated time stamp on a page would increase the activity rating besides no activity taking place. While this could be fixed with a constant pool of documents where the effects of a rogue document would be canceled out over time we decided to take the risk because we value the relevance rating of the search engines over the occasional misinterpretation of activity.

4.3.4.4 New Sources

A basic requirement is the results of two crawls c_{t_0} and c_{t_1}. By comparing the URLs provided by the search engines we look for sources which were previously not part of the crawl. The assumption is that if a search engine adds or removes URLs to the result set of a specific query something must have happened to change the relevance rating of this particular or following URLs. Therefore we assume that some kind of activity has happened

which results in the changed URLs. We use the search results from third parties as big search engines are more capable to analyze a large part of the Web, determine relevance and have a tested infrastructure.

4.4 Experimental Evaluation

During our research we also considered synonyms and stemming to be relevant for trend detection as we develop some methods which are text based. We did not regard synonyms as they are heavily context dependent and the index structure currently used provides no efficient support. This might change with a positional index but the challenge to extract or collapse the right synonyms for the supported languages remains. Also there are few sources for comprehensive data about multilingual synonyms. Experiments with stemming indicate, that too much relevant information is lost (e.g. gender for job descriptions in German) so that a trend might get lost among the other terms which share the same stemmed form while only slightly reducing the dictionary size of the inspected languages. Experiments with a German dictionary and various sources lead to an approximate reduction of the dictionary by only 15%. With the same reasoning we decided against the usage of character folding. While it is certainly reasonable to use it in a information retrieval context the conclusion is that its effects are detrimental to trend detection.

The dictionary growth is used as to determine the number of documents to crawl. The number of documents after which the dictionary size of an index stops to increase super linear in size is considered sufficient. This is because with linear growth, diminishing returns are to be expected by additional documents. Heaps law [59] states, that the discovery of new tokens becomes less like with increasing corpus size. Figure 4.4 shows, that the rate of diminishing returns is dependent on the topic. We observe that the rate for the topic "semantic web" decreases faster than e.g. "geoengineering". By adding more terms to a topic, a topic becomes more specific. We can also observe that the more precise definition of the topic has nearly no effect on the diminishing rate. In regards of the trend detection task we interpret the increase of the dictionary size as an indicator for novelty added by including one more document to the analysis. From Figure 4.4 we conclude that, while the topic at hand and the average length of an document is of relevance, for the inspected cases at hand 100 documents are considered enough to achieve an linear increase in new terms to the dictionary.

For each method we evaluate what data is required and if it is available. Furthermore we consider the dimensions inspected by the methods and a fitting way of visualization and estimate how complex a method is to run on current standard hardware (Intel i5, 8GB RAM). Because the method targets documents in several languages, semantic approaches are disregarded, as they are usually language specific. Instead the methods used focus on

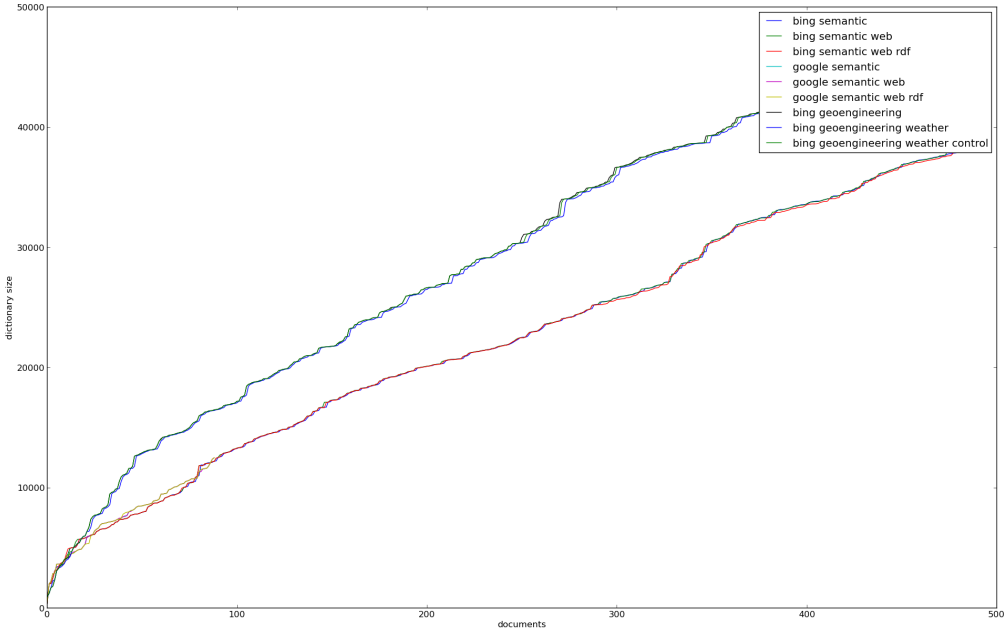

Figure 4.4: dictionary size growth

statistical analysis of the text corpus at hand.

4.4.1 Crawling the Web

In Table 4.1 we can see that the most common error is an unusable file type which is unfortunately only discovered after the header of an URL is sent. The second most common reason for skipping a resource is that the "robots.txt" does not allow access for crawlers. More than 2% of the errors are caused by unreachable hosts which includes timeouts. A timeout of 45 seconds is used.

4.4.2 Relative Term Frequency

Experimental evaluation of the relative term frequency indicates, that it is a reliable metric to detect emerging keywords (see Figure 4.1). For trend detection this method is recommendable as each relative term frequency time line can be understood as a trend indicator. Using relative term frequency allows us to work without stop lists as often used terms show only slight deviations in frequency over time. Paired with other metrics, e.g. reach, it indicates which terms are currently rising over proportionately in frequency which indicates a signal that is created by human behavior in contrast to the noise of content-generating

Table 4.1: frequency of particular errors

error code	abs. freq.	freq. (%)
no error	3932827	84.81
content not parsable	11942	0.00
unexpected HTTP status code	1	0.00
connection failed	112404	2.42
document too large (>15 MB)	17858	0.04
robots.txt not parsable	35468	0.08
blocked by robots.txt	120337	2.60
unparsable ULR scheme	48438	1.04
unusable file type	357676	7.71

systems or information-retrieval systems.

4.4.3 Estimated Matches

The estimated number of matches for a given query provided by the search engines varies (see Figure 4.5). Especially Bing offers unreliable results. In order to give an informed trend many more than two data points are required. Google performs slightly more stable but is still behind expectations. The search engines offer no explanation on why the estimates vary. It can be assumed that different versions of the index are used for the estimate.
For activity detection it is required to have at least two observations. While the number of estimated results from one observation can be used to derive a general feeling for the importance of the matter in question we need two observations to get an idea in which direction possible activity is heading. For large numbers Google and Bing provide very volatile results (50%-70% change in a matter of hours) which obviously does not reflect the actual state of the Web. The estimated matches proved to be more stable for a smaller result set. Therefore we suggest this method in regards to activity measurement only for topics which yield a small number (a few hundreds to thousands) of results.
As far as trend detection is concerned, a recommendation against using this method must be made as the number of results varies too much and a fluctuation might induce a trend which is not actually there though it seems to stabilize if there are enough measurements done. With this method we consequently regard the dimensions time, country and language. Search engine could be added as additional dimension. For visualization a world map and a slider for the time dimension would be sufficient. The time series can be displayed as graphs. The basis data needed for this method is a byproduct of each query. As almost no processing is taking place this method is easily implemented and has low hardware requirements.
More data is required and further research has to be done to give estimates how much data is necessary for a good trend detection and activity measurement.

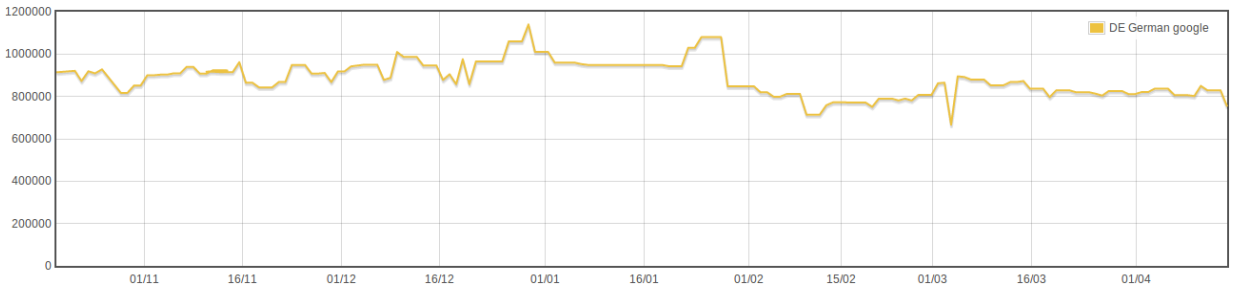

Figure 4.5: Estimated Matches by Google for “news” in German

4.4.4 Page Updates

In regards to trend detection this method is of limited use. A direction can be deducted by the increase or decrease of the activity over time. But currently it is not possible to extract more specific information about the underlying forces of a trend from this method. First experiments with this method have shown that it provides rather stable results and is opposed to e.g. Estimated Matches not prone to unreliable search engine data. A comparison of two topics showed that “daily news” has a 95% update rate while the more stable “geoengineering” only has a 25% update rate aggregated over the whole topic. This method also works independently from the number of documents. The recommended amount of documents for this method is still subject of ongoing research.
The dimensions inspected by this method are time, country, language and search engine. Various aggregations e.g. all languages of a country are possible. Changes of the results are also subject of ongoing research by different ways of aggregation.
Visualization is done by a chart and a world map. The computational complexity depends on the way differences between document versions are detected and treated. The approach via hashing keeps the computational complexity and memory requirements low, though a simple time-stamp embedded in a document will yield a hash-mismatch and is interpreted as a page update. A direct comparison of the HTML structure or extracted text-nodes could provide more concise results. In the field of Search Engine Optimization (SEO) methods are developed to trick systems into thinking that a new version of a document is published as more recent content is rated higher by most popular search engines. This development increases the cost of implementing a reliable method to distinguish between a real page update and a minor change to a point, where it is hardly feasible for the task of trend detection.

4.4.5 New Sources

Naturally this method is primarily developed with activity measurement in mind. We can show that a topic like “geoengineering” behaves differently from “news” but due to the way modern search engines handle their index structure this method suffers from the same symptoms as the Estimated Matches method. Search engine provider usually operate with several index structures in parallel. A “current” index is used to answer queries while

the next index is built in the background. Adaptations of the current index are done by lists which contain deleted URLs [100]. Currently it is impossible for us to distinguish between actual activity and the switching to a new index at the side of the search engine provider. A high influx of new URLs might suggest a new index structure and in the case of successive large changes activity might be deducted the possibility of the reply to the search queries coming from different data centers or index structures cannot be canceled out. Further research has to be conducted to see how this method performs in the long run and how we can detect which index version is delivering the URLs so we can compare changes to the same index.
For many observation this method can be used for trend detection. In the current state the results are too imprecise and fuzzy to detect a direction of the activity.
This method is also working on query basis which allows us to inspect the dimensions time, country, language and search engine. Similar to Page Updates aggregation can be used to get more general information e.g. about the development in a particular country.
Only a chart is used for visualization, as the relationship between available documents and index rebuilding at the premise of the third parties, and the effect on the result set is not yet clear so a country-by-country comparison does not to seem feasible yet.

4.5 Conclusions & Outlook

The previous Sections outlined a new approach to activity monitoring and trend detection in the Web. The new approach consists of a concept to acquire data and a number of methods for activity and trend detection. The concept for data acquisition and analysis is outlined and a fully implemented prototype is used to prove the feasibility. Experimental evaluation shows, how methods can be applied to the data and which conclusions can be drawn from each method using examples and gathered data (see example “geoengineering” in Figure 4.2).
Activity measurement and trend detection is tightly linked. To develop better methods for trend detection, is fundamental to have the ability to detect activity, so that different aspects of activity can be researched and analyzed in depth. In order to improve the activity measurement methods, there is the need to collect more data and research the relationships of the various dimensions that affect the data, e.g. the search engine used.
The developed methods can be used to analyze the performance of various search engines in regards to index variance and refresh rate. Also we need to evaluate the stability of the results of the search engine providers so that a qualified suggestion can be made, which service provider is a best fit for a given method.
In regards to trend detection the research of the average live span of trends in various topics can yield a further characteristic of a topic. Also a field of our research is the clustering of trend terms via correlation analysis and spatial analysis based on the downloaded documents.
Further technologies which we would like to include in future work are positional indexes. Positional indexes enables the detection of terms which consist of more than one token

(e.g. New York) by searching for combinations of terms which appear close to each other (n-grams). The proposed text-based methods for activity and trend detection are unaffected by a positional index as an abstraction layer can be built to keep the input format unchanged.
The increase of complexity of the analysis is linear with the number of observations. Though focused on short-term trend detection, this property make medium- to long-term analysis of the data feasible, as the time-frame of the analysis is for the analyst to select.

Chapter 5

Medium-Term - Patent Classification

Patents are a valuable asset of organizations that allows controlling the competition due to exclusiveness and by preventing other organizations from market entry. A prominent example of a field where patents play an important role as a strategic tool is the mobile-phone business. Changes in the patent-situation affect the meso-environment. Therefore patent-analysis is an important part of environmental scanning. Popper categorizes patent-analysis as a quantitative and evidence-based method [40]. Several arguments can be made why patent-data is not ideal to detect short-term shifts in the environment. There is the fact, that an application for a patent is done after R&D is complete, and that can take an unspecific amount of time. Patents also require a non-trivial investment, so it is reasonable to assume that activity, indicated by patent-documents, can be regarded as "serious business". Another issue is the amount of time the patent-application-process takes. The process at the United States Patent and Trademark Office (USPTO) typically takes at least 18 month from application until publishing; and the patent process is not ending at that point. Considering that this process starts after Research and Development (R&D) of uncertain length, we assume that the lag of patent data is at least two years, resulting in the metrics derived from patent data as lagging indicators. Although we will discuss ways of reducing the lag down to the publishing date. Activity in certain fields in a patent class reflects investment into those areas, in form of the cost of R&D, the legal process and usually substantial fees. Due to the volatility of these investments we do not consider patent data fit for long term indicators.

Despite the greater time-lag than e.g. web data, there are multiple benefits of patent data analysis for trend detection and environmental scanning, which other data sources do not provide in this combination:

- Patents are relevant. Due to the cost associated with a patent, the applicant considers his invention to be relevant enough so that she sees the protection of her intellectual property as warranted. A patent gets granted if the issuing authority considers the invention as novel and non-obvious.

- a patent document is structured. Contrary to web data, there are standards which describe how a patent is structured. This allows a more thorough exploration of the

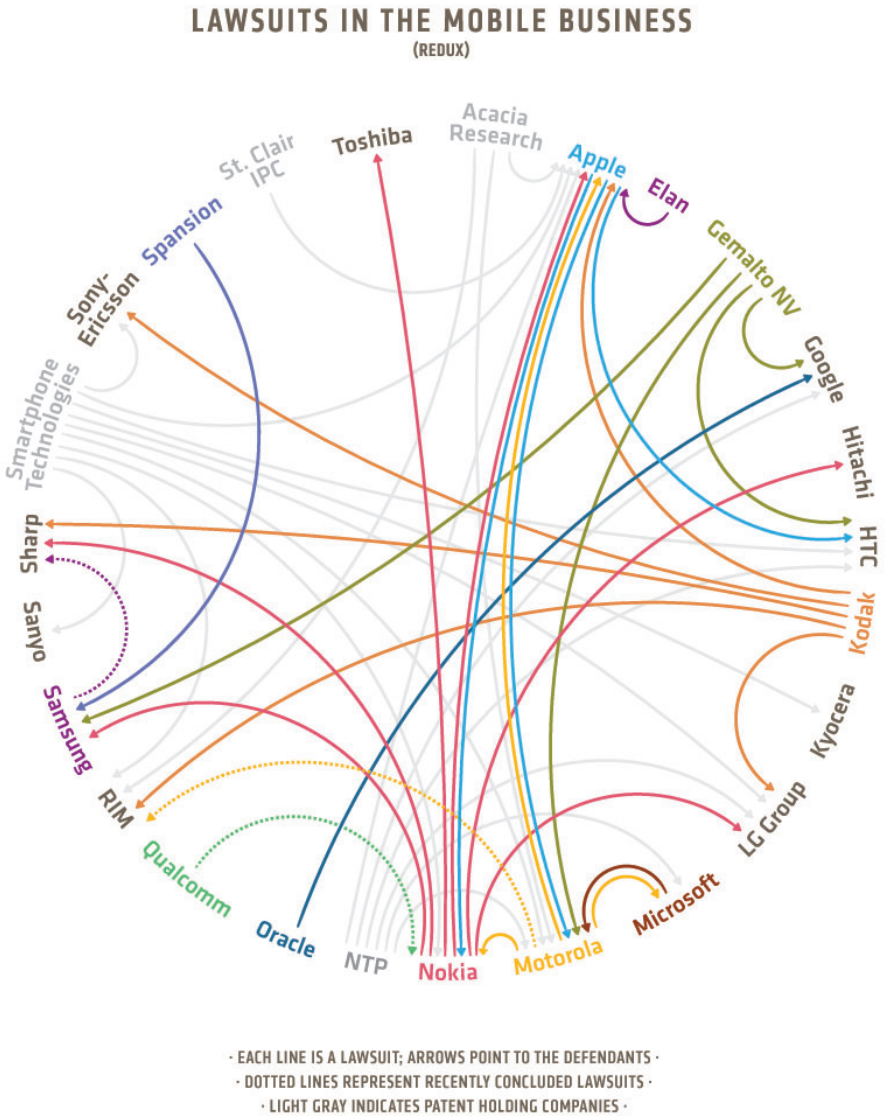

Figure 5.1: Patent Lawsuits in the mobile business; Source: [8]

actual content of a document. Another benefit of the structure is that meta data is thoroughly described. Fields like application date and inventors address can provide additional insights.

- Patent data has a history. Since the first patent in the US was filed on 31st of July in 1790, the USPTO is managing information about new inventions from any field. This provides sufficient ground truth data e.g. for methods that rely on supervised training.

- Patent data is pre-classified by area for which a patent is relevant. If the area of interest is known, this classification allows to focus on a small subset of patent data. As a patent can be member of multiple classes and most classification system are organized as a hierarchy, branching out to other classes which share relevant documents and switching between hierarchy-levels can be used to find other areas of interest and new applications.

- Patent data is available from all over the world. Many countries around the world employ a patent system. In a globalized world a local protection of an invention is often not enough so global efforts lead to a common classification system and an alignment of patent processes which are accepted by most industrial nations world wide. Other than the obvious benefits for patent applicants and holders, the benefits of an harmonization from the point of view of a data analyst is that the number of available documents increases without the uncertainty introduced by a mapping between different patent offices and systems.

In order to maximize the usefulness of patent data, reducing the lag is crucial. With the publishing of a patent application 18 month after R&D a minimum lag exists. At that point, there is no fixed categorization of the document available as this is decided by the patent office when the patent is granted. By automating the classification process with sufficient precision we can:

- support patent examiners in patent offices

- reduce the lag of patent data

- search for related and similar work and prior art

In later sections it is described in depth why the categorization is so important and metrics are defined to measure performance. This thesis proposes two new approaches to enhance the automatic classification of patent documents.

Naturally, Strategic Management considers the dimensions “when” and “where” to be of importance. Patent data contains these dimensions in multiple ways. Location is available via the issuing authority, the inventors address, the attorneys address, the companies address etc... The time dimension can be covered by using the application date,

publication date, issuing date and various others. All of these data-fields are mandatory parts of a patent-application and patent-grant.

The subsequent Sections are structured as follows: the Introduction 5.1 discusses the significance of patents and the interest of different stakeholders. The general structure and various fields contained in a patent application and patent grant are explained in detail. A description of the patent process as it is practiced by the USPTO and an introduction to patent classification systems to understand following issues is also part of that Section. As the focus is on the issue of patent classification, the Section 5.2 outlines the state-of-the-art in that field and touches on the issue of various datasets.

The motivation for the meta-data driven method GeoDAT is in Section 5.4.1. Work that is specifically related to that approach is discussed in Section 5.4.2. In Section 5.4.3 the method for patent classification is discussed in detail. Section 5.4.4 presents the experimental setup, dataset and results of the empirical evaluation of GeoDAT. The last Section 5.4.5 provides a summary and gives directions for future work on the GeoDAT-method.

The GeoDAT method is a joint work with Matthias Schubert and published as [134].

Section 5.5 motivates the text-based approach: Fixed Hierarchical Vectors (FHV). The following Section 5.5.2 explains the method in detail with Section 5.5.3 presenting experimental results. In the final Section 5.5.4 we discuss future work on the FHV-approach.

The development and evaluation of FHV is the result of a cooperation with Marawan Shalaby, Matthias Schubert, Hans-Peter Kriegel and Stephan Günnemann and is going to be published.

After presenting the methods we use the last Section 5.6 for a summary and outlooks on future work.

5.1 Introduction

As most scientists can attest to, the development of a new process, method or any innovation in general is usually not a simple, straight forward task. The involvement of several people and material that drives cost without guarantee of any usable result. It is not unheard of that competing companies use methods that are not considered legal in order to reduce their own cost and risk at the expense of the competitor. While this is an extreme example, it is common practice in the industry to seek inspiration from products made by rivaling companies in order to get insights into the applied processes and to potentially save on the own R&D budget. Without any further protection, innovations and inventions would rarely make it to market as the break even for the initial investments changes. The number of units sold before a competitor offers a similar product decides if the investment was worth it. So either the investment is very low or the innovation is of such a nature that there is little risk of replication. Both cases are increasingly rare.

To protect the investment of an innovation after release to the market, patents evolved. The basic idea of a patent is that it grants exclusive rights to the patent holder to use the idea, method or any other concept without having to fear that it is copied by another party. This protection shifts the ratio for the break even of the initial investment away from the

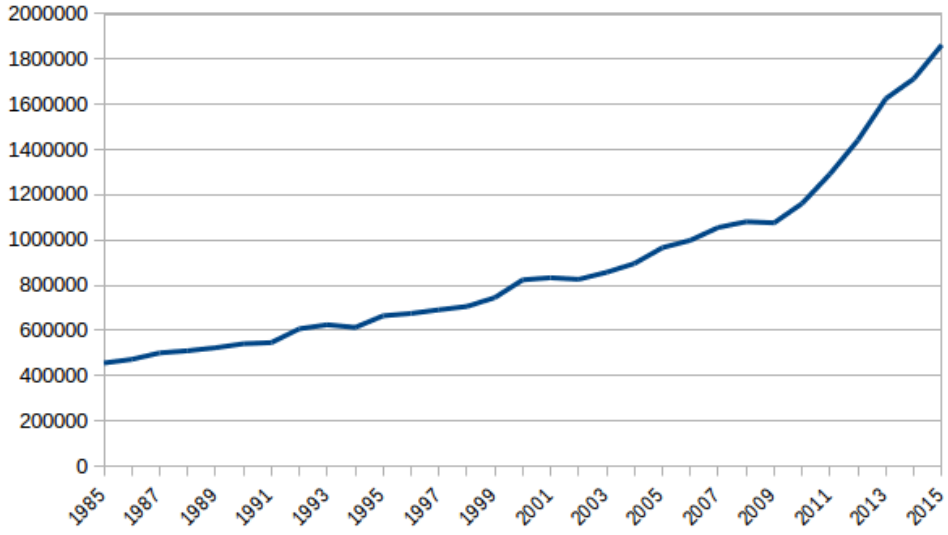

Figure 5.2: Patents granted world wide; Source: Worldbank 2017

products sold until it is successfully copied by other market members to the duration of a patent grant. While patents can and do expire, the possible viable investment into R&D is increased as there is a guaranteed, larger window of opportunity.

Another benefit of patents comes from the fact that they have to be published to have effect. Without publishing, it might happen that several parties develop the same innovation independently and use them in their products. If one party decides to patent this particular idea and it is not published, the other parties are violating the intellectual property unknowingly. Because of publication of patents, this issue should never arise and parties interested in a particular field can inform themselves if an investment into a concept would be warranted or if the problem is already solved and patented by another party. In theory, this leads to a more efficient allocation of R&D funds for all market participants, as patents can be traded and licensed.

Even though the patent system is certainly not without inefficiencies, it is by now a globally accepted method to protect intellectual property. While the concept of patents is accepted world wide, the implementations vary in level of protection and circumstances where a patent offers protection. As globalization leads to an increase in trade activity across borders, the World Intellectual Property Organization (WIPO) pushed for an harmonization of various patent systems through the Strasbourg Agreement in 1971 which results in the International Patent Classification (IPC). As of now, this system is used by over 100 countries. While there are arguably 194 sovereign states right now, all of the big (by Gross Domestic Product (GDP) and citizens) countries have joined the system so that there is sufficient coverage to secure investments. Figure 5.2 shows that there is a nearly continuous increase in patents granted world wide that underlines the acceptance of the concept of patents.

5.1.1 Stakeholder

A stakeholder is an entity with a direct or indirect interest regarding a subject. The subject of a patent or many patents has as most obvious stakeholder the inventors and the company the inventors are working for. In the case of larger companies, the inventors usually receive a bonus payment for a granted patent and yield all rights to their employer. So the employing company is another stakeholder. Patents play multiple roles for companies. As initially intended, they assure the company exclusive rights to use the patented innovation. Another aspect is that the number of patents owned by a company can affect directly its valuation by investors. Recently there are cases where patents are used strategically to block competition. While this behavior is not intended by the concept of patents it certainly evolved to be another application of patents. Last but not least, companies use patents as a bargain in cooperations (e.g. Intel and AMD) or just trade licenses for additional revenue.

On a larger scale, patent data is used for the following activities: performance analysis, trend detection and monitoring. An example for performance analysis is that the number of patent applications per country is often used as a way to compare different national economies or to evaluate increased productivity (e.g. WIPO). Indicators based on patent applications are part of the system based on macro-indicators discussed in earlier Sections. Trend scouts can use larger collections of patent data to identify particular active areas of research leveraging the structure of patent classification systems and identify the participants of that field. Organizations monitoring patent data for developments relevant to them. Relevant would be any patent application published by a direct competitor or any patent application touching on the subject of business.

The most important stakeholders might be those who primarily work with patents. This includes patent attorneys, patent examiner and patent offices. The interest of these stakeholders are not necessarily aligned. While a patent attorney is usually working on behalf of an inventor, he tries to maximize the protection of a patent. There are various strategies employed but among them is the intent to keep the description of the actual innovation as generic as possible so that a maximum amount of patent classes can be attached to the patent (e.g. a certain type of hook might be used for fishing but also to attach instruments to lab equipment). This is in conflict with the interests of the examiners. They have to decide if the patent application contains enough novelty and is non-trivial so that a patent grant is warranted. Unspecific wording and vague descriptions increases the difficulty for the examiners to categorize a patent application correctly into a patent classification system. At the same time, the applicant has to search for related work and prior art as not to start the costly patent process without a possible expectation of a patent grant. In later Sections we will show, that the convoluted style of writing used in patents does challenge automated state-of-the-art information retrieval technologies and we propose methods to increase their performance.

5.1.2 Introduction to Patent Classification Systems

The modern patent system was invented around the 19th century, well before computers were available. To enable a patent clerk to check for related work and prior art, some kind of system is necessary so that she does not have to scan all available documents. Another aspect is the type of protection a patent provides. A method developed for agriculture should only be protected in the area of agricultural application. This limitation of patent protection offers additional help for the search of related work. Patent Classification Systems combine both aspects: a system for simple information retrieval and for limiting protection of a patent to certain fields.

Thus a Patent Classification System contains an overview of multiple fields. In the systems we are considering in this work these fields are organized in a hierarchy to reduce lookup time compared to a flat list and to allow various stages of protection depending of the hierarchy level that is referenced on a patent grant.

In this work, we focus on classifying patents into the Cooperative Patent Classification (CPC) system and the International Patent Classification (IPC) system. Let us note that the same methods can be applied to any other patent classification system being currently in use, e.g. the United States Patent Classification (USPC). The CPC system is developed mainly by the European Patent Office and is now widely adopted by various patent offices around the world. It has its roots in the European classification (ECLA) system which is similar to IPC system.

The IPC is a hierarchical system of up to six layers. The top level called Section is denoted by a letter from A-H, followed by two numbers signifying the class, followed by a letter indicating the subclass. A space separates the sub class from the group number which is followed by a slash separating a main- or sub-group from the group number. An example is A01B 1/00 where with A being the section, 01 being the class, B being the subclass and 1/00 identifying the group and main- or sub-group. One patent can be a member of multiple categories.

Similar to IPC, the CPC is a hierarchical system in which the top layer consists of nine instead of the eight Sections of ICP, indicated by one of the letters A-H or Y (see Table 5.1), followed by Class which is indicated by two digits, followed by Subclass (one letter). Lower layers are called Group (two digits) and Main group (two digits). A granted patent can be assigned to multiple classes at the leaves of various branches of the CPC hierarchy. An issue with classification systems is that they don't anticipate future development. The process of reclassification is used to remedy that problem. Reclassification is applied retrospectively to patents when the classification system containing the patents is extended or restructured. In the CPC classification system the Y section is specifically introduced for emerging technologies to allow a faster response of the patent classification system to new technologies so that reclassification can be reduced.

Since 2013 the USPTO started classifying some patent grants according to the CPC and is increasing the volume as Figure 5.3 shows. While more IPC classified data is available, we explicitly choose to use CPC for the research of geo assisted classification because of its future potential and the widely adoption.

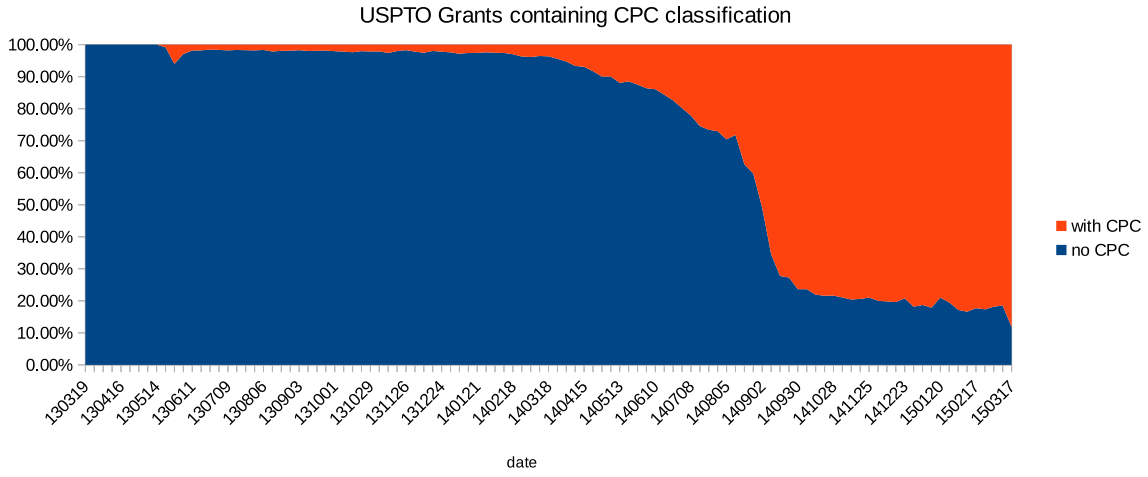

Figure 5.3: progression of USPTO in CPC classified grants

5.1.3 Patent Process as practiced by the USPTO

To receive a patent grant inventors have to file an application at a patent office. The patent application consists of the text elements title, abstract, description and claims. The claims specify the scope of the protection and can be changed during the patent process. As the patent application publication provides the first publicly available text for a patent, we map each classified patent grant to its respective application to get as close to the initial submission text as possible using the document id.

Further elements of the published structured document patent application are: inventors names and addresses, applicant name and address (which is oftentimes a company name), document id, publication date, kind of application, assignees and assignees addresses, related documents e.g. other patents and a domestic and CPC classification. Implicitly the publication date is also included in the application document. In general, the publication date is 18 month after the filing date. After the publication date the document is valid prior art. A complete list of all fields included in an current application publication document can be found here [114].

After the publication of the patent application the applicant can decide to receive a patent grant. The issue date of the grant sets the time from which on infringement can be charged. The patent grant document contains the final claims, classification and related documents and references as meta data. Figure 5.4 shows a flowchart provided by the USPTO that visualizes the whole process from invention to a patent grant.

5.1.4 Patent Structure

In this Section we describe the data provided by the USPTO as eXtensible Markup Language (XML) files. Together with the data, the USPTO publishes a Document Type Definition (DTD) file which can be used to validate the XML structure as it defines the

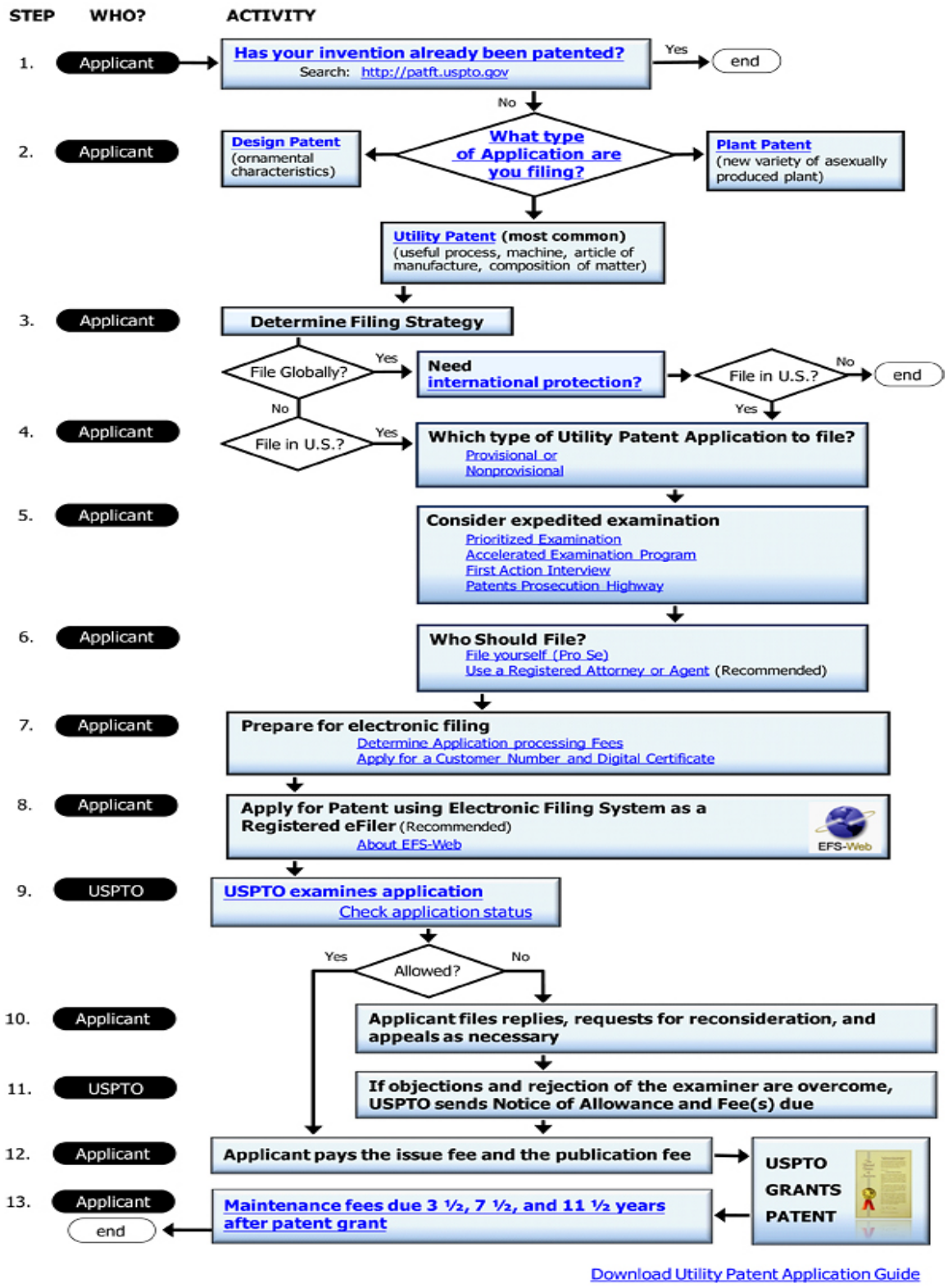

Figure 5.4: Flowchart of the USPTO patent process as provided by the USPTO

XML tags used and what properties and child tags are allowed. Additionally a document that explains the fields referenced in the DTD is published. In this Section we use the latest version (v4.3) dated December 2012 of that document as reference and focus on the information contained in patent grants. While updated DTDs exist (as of writing v4.5 is the most current), the fundamental documentation did not receive an overhaul.

The USPTO changed the publishing format of patent data several times. Current data adheres largely to "Red Book" standards, while data published from 1976 to 2001 adheres to "Green Book" standards which does not use XML and contains different fields, though classification information according to the corresponding version of IPC is contained in both standards.

A large part of the DTD is used for classification systems. As the USPTO is currently employing three different classification systems (CPC, ICP and USPC) each field must be associated with exactly one of those systems and with the correct version of the system. Other information is concerned with the inventor, examiner, attorney and associated organizations. Further parts cover citations of other patents as these are commonly used to enhance the classification of a new invention and to determine the value of a patent. The bulk of a patent document is unstructured text consisting of the fields: abstract, claims and description which cover the invention and the novelty. In general, a patent document answers the questions: what, who, where and how does this relate to prior art?

For the CPC the USPTO publishes one main CPC classification with each hierarchy level separated into specific tags and allows optionally the addition of further CPC classifications. The version of the CPC used is a required part of the classification data. Part of the main CPC classification is the classifier which in all cases is human for now. The main CPC classification "serves as the basis for the assignment of tasks and determines the competence of the examining section" [102] and in most cases is already contained in the patent application. The IPC and USPC is treated similar. Another part of the classification information is: which organization created the classification. As patent protection can be world wide, some organizations share patents and their classification.

References and citation of prior art is an important part of a patent document. Each reference to another patent consists of the country, document number, year and classification of the reference in a structured form. In addition to that a field tracks who added a particular reference to the patent being either the applicant, the examiner or a third party. The references are the result of an extensive prior art search and produce considerable overhead. A high quality of citations can be used for research, patent valuation, information retrieval and is used to secure the intellectual property.

Bibliographic information is another important aspect of a patent grant. Organizations and persons involved in the patent process are attached to the grant. Inventors, applicants and agents are included by name and address in the patent grant. The same goes for the assignees. Assignees are mainly the employees of the inventors and their address can be the headquarter or any other location. Inventors are usually recorded by their home address, allowing a geographic reference to where the innovation actually took place. The primary and assistant examiner are only referred by name. In addition to names and addresses, document-ids and various dates are also part of the grant. The application id and date is

Table 5.1: CPC Sections and number of documents in CPC dataset

section	description	documents
A	Human Necessities	95,417
B	Operations and Transport	86,111
C	Chemistry and Metallurgy	86,739
D	Textiles	3,301
E	Fixed Constructions	10,545
F	Mechanical Engineering	41,230
G	Physics	132,574
H	Electricity	224,042
Y	Emerg. Cross-Sectional Tech.	18,012

as much included as the publication date and the grant id. With the application id it is possible to link to the original application.

Published patent application contain similar information, though they might be subject to changes. The classification information and any citations and references are not part of the structured bibliographic information yet.

Besides the bibliographic data, a patent grant contains a title, abstract, description and claims. These fields are required for any patent except design patents and usually represent the largest part of a patent document. The title consists only of a few words and the abstract is a short summary of the patent. The description on the other hand can go into great detail about the novelty and any images and formulas are typically located here. While semantic elements like headings and paragraphs are allowed, the typical description is rather unstructured compared to the bibliographic data and thus hard to interpret by machines. The enumerated claims section lists the novelties and inventions that are to be protected by the grant. While the enumeration gives some structure to the element, the use of free text creates the same challenges as the description. Additional difficulty is created by patent attorneys who deliberately use vague and general language in the free text sections to extend the protection of a grant. As this type of language is not ICP/CPC class specific, patents from totally different classes could exhibit a similar language which proves a challenge for BOW methods.

5.2 Related Work for Patent Classification

This Section surveys the state-of-art of scientific literature for automatic patent classification. Following the process shown in Figure 5.4, at least two elements can be identified, which would benefit from an accurate automatic patent classification: the first activity in step one “Has your invention already been patented?” as information retrieval task can benefit from a similarity measure between patent documents. Second, the 9th activity “USPTO examines application” where classes are assigned and another prior art search is performed among other things. Benzineb et al. [33] provide an overview of the cur-

rent state of the art in the field of automatic patent classification, stating that SVMs and NNs represent the current best practice solutions. As vectorization of the objects is not mentioned, following [26] it is assumed that the BOW method TF*IDF is used.

A major task of information retrieval in patent data is finding prior art for a new patent application (e.g. [92]). The literature concerned with retrieval can be distinguished by approaches which create their own class system and those which follow CPC or ICP.

Creating a class system independent from CPC or ICP is reasonable if the only performance-metric is the similarity of documents. Research in this direction is mostly based on document clustering. The work about retrieval methods which follow CPC or IPC is mostly describing various query languages, filter methods or efficient data structures to search a given corpus.

Due to the formerly limited access to patent data and the small size of the patent community the literature on classification into systems like CPC and IPC is limited. Due to many different data sets with different formats, languages and information it is difficult to compare or reproduce results. According to [64] the Japanese Patent Office reaches a precision of 90% when preclassifying patents into 38 different classes. Unfortunately, there is hardly any literature available on the methods being used to achieve this result. The same paper claims that for 90% of the documents it can predict the main IPC class correctly within top 3 suggestions using [88], a variant of Winnow. Winnow is a binary classification algorithm representing a simple neural network similar to the perceptron algorithm but is using a multiplicative scheme for updating the weights instead of additive.

A comparison of the classification methods Balanced Winnow [97] and linear SVMs [36, 52] draws the conclusion that SVMs consistently outperform Balanced Winnow. Balanced Winnow is doubling the number of training examples compared to Winnow by adding the inverse of each element to the training set. For this reason Balanced Winnow uses two hyperplanes which allow a better performance at multilable problems. For further reading about SVMs see [125]. Though SVMs have longer training times, they have a clear advantage in learning more robust models for uneven class distributions in the training data. Given the increased performance of modern computer systems, the disadvantage of longer training times becomes more and more irrelevant. Additionally, [36] proposes to use Logarithmic Term Count (LTC) instead of TF*IDF to reduce the effect of large differences in term frequency. The best F-Score in [36] is 0.7597 when setting the soft margin to $C = 0.25$ and the cost factor to $J = 10$.

The authors of [53] focus on the benefits of Mutual Information to classify patents into the United States Patent Classification (USPC) System. They follow a BOW approach utilizing a linear soft margin SVM and apply Mutual Information to the text. The parameters of the SVM are not disclosed and the performance measurement used is Break-Even Point (BEP). The BEP reaches an average of 0.735. The minimum number of documents in a class used for training is documented at 506. The cut off of the dictionary varies between 1,000 and 10,000. 19 different classes were classified.

Automated patent classification into the IPC is also examined in [62]. Naive Bayes and SVM are compared with SVMs providing similar or superior results using a customized success criterion. The dictionary of the linear SVM is limited to 20,000 terms and 500

documents per class. The focus is classification on IPC class and IPC subclass level. Referring to [92], it is investigated which part of a patent provides the best results for classification. Results show that the more text is available the better the SVM performs. Other methods might do well with the title and [92] suggests to triple its weight. Results show that the title has only minor impact on the SVM classification. The authors also note that indexing phrases seem to enhance search results in patent data sets but are detrimental to patent classification.

Beside full-text approaches, methods utilizing the meta data are also explored. One method uses the addresses of involved parties to generate a spatial probability distribution of patent classes [134]. Though this method improves classification performance, it should be used in combination with other classifiers.

In practice, automated patent classification is still too unreliable to replace experts at the patent offices. In the XML schema, the USPTO is using to publish its data, the tag $classification-data-source$ indicates how a classification was obtained. As mentioned earlier, currently the value of the tag is always set to "H" indicating "Human-Generated".

5.3 Dataset Usage

There are various patent data sets available. One crucial property is the capability to use them online or offline. For our purpose, we consider databases which can be used offline as beneficial to our mode of operation as we have many very general queries and request bulks of data for certain time frames. To classify patent data, we need classified grants and their corresponding unclassified applications. The patent grants may contain the same text as the application but there is a chance that it was changed in the process of granting. So to operate as close as possible to real world data, we need a grant dataset and an application dataset. Many of the commercially available patent databases offer additional features like annotations, translations, normalization and are preprocessed in ways that benefit primarily the search for prior art. Among the more prominent patent datasets are the PATSTAT database of the European Patent Office (EPO), Espacenet of the EPO, the Derwent World Patents Index from Thomson Reuters and PatFT of the USPTO. There are many more services provided by different patent offices and commercial service providers. Most of them are only available via web interfaces or can be accessed via rate limited APIs. While there exist offline versions of PATSTAT and Derwent World Patents Index they are not available for free and their price indicates the great amount of work which went into post-processing, translation and normalization. The commercial option usually consists of a form of subscription to provide current information.

In June 2010 the USPTO announced a joint effort with Google to provide bulk patent and trademark data to the public. Google is providing their technical capabilities and the USPTO provides Google with the data. This cooperation is still active and updates of patent data are published often. Available for download are the granted patents and the patent applications as full text. According to Google the data from the USPTO includes data from the EPO and World Intellectual Property Organization (WIPO). The

data is available as download in zip files for offline processing or search-able via the page *patents.google.com*. As the data is free of charge and contains all the information of patent documents, it is attractive to perform research with it.

For the methods proposed in this thesis focus on two different aspects of the patent-data, and two different datasets are created.

5.3.1 CPC-Dataset

Figure 5.3 already indicates, that while there are less documents with CPC classification than with IPC, the usage of CPC-classified documents is a guarantee to use patents granted recently. This has the benefit that the meta-data has a higher standard of quality. Preliminary experiments with older grants indicated that the meta-data have a high rate of errors (e.g. wrong and/or missing dates). Due to the focus on meta-data of the method developed (see Section 5.4.1) and the relevance of the new CPC system the decision to use CPC-only documents is reasonable. A positive side-effect of that decision is that the available data only covers a time range of a few years so that the effects of uprising and decline of industrial centers (e.g. Milwaukee [4]) can be neglected. The relatively short time-range also increases the probability that the meta-data, e.g. any addresses, are still up-to-date.

Due to the topicality of the data, it can be assumed, that the application to a grant is not too far in the past, also provides high quality data. This is exploited in the creation of the dataset by requiring the application to each CPC classified grant. That measure ensures that the original text from the application is used by any text-classifier. A major distinction of the CPC from the ICP is the introduction of the Y section. This section is used to cover emerging technologies for which no appropriate class is available yet.

The final dataset consists of the text from patent applications for which a matching and CPC classified patent grant could be found. The time-range is between 5th March 2013 to 17th March 2015 which results in 136,870 documents. Figures 5.5, 5.6 and 5.7 show that the classes are unevenly distributed which must be accounted for by classification methods and evaluation metrics.

Many classes on all levels of the CPC hierarchy overlap. When filtering for classes with more than 550 documents, 2 cases remain where an overlap of more than 50% of the documents on class-level (see Figure 5.9) and 6 on subclass-level (see Figure 5.10) exist. Even on section-level we observe an overlap of 43.6% between the sections Y and H (see Figure 5.8). This indicates that the co-occurrence of classes has the potential as a feature to increase classification accuracy. Though that is not part of this work.

5.3.2 IPC-Dataset

The goals that are to be achieved with the IPC-dataset differ from those of the CPC-dataset. Deep Learning is to be applied and evaluated. Due to the many parameters of that method, a lot of training data is required. Additionally, the quality of the meta-data is of low priority as long as the classification-information is correct.

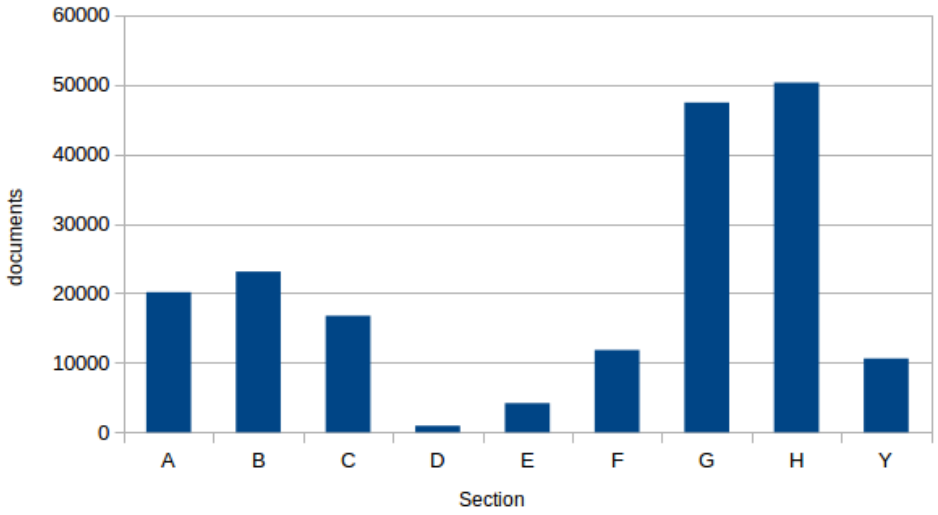

Figure 5.5: Section distribution in the CPC dataset

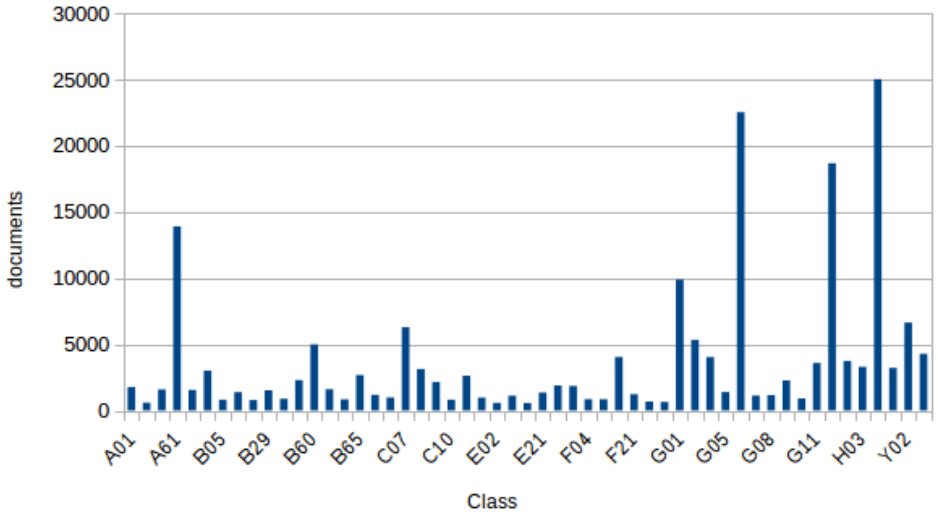

Figure 5.6: Class distribution in the CPC dataset

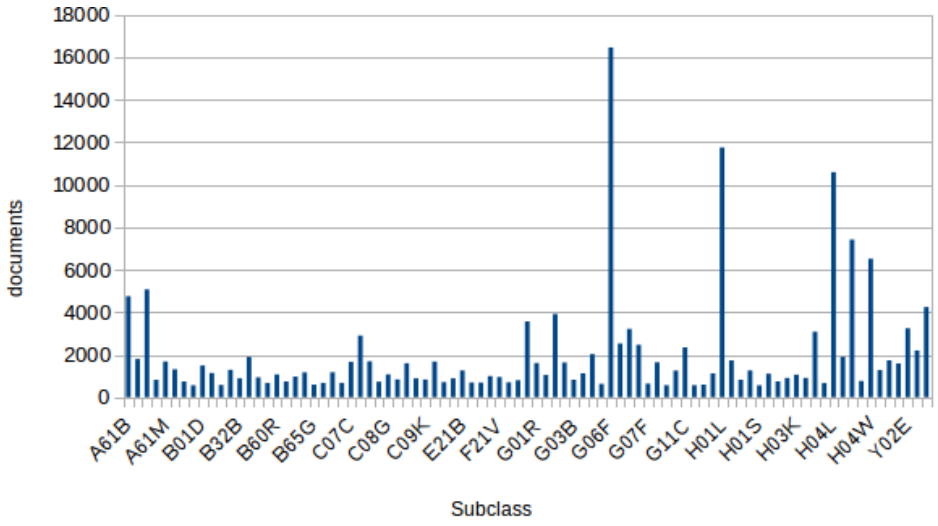

Figure 5.7: Subclass distribution in the CPC dataset

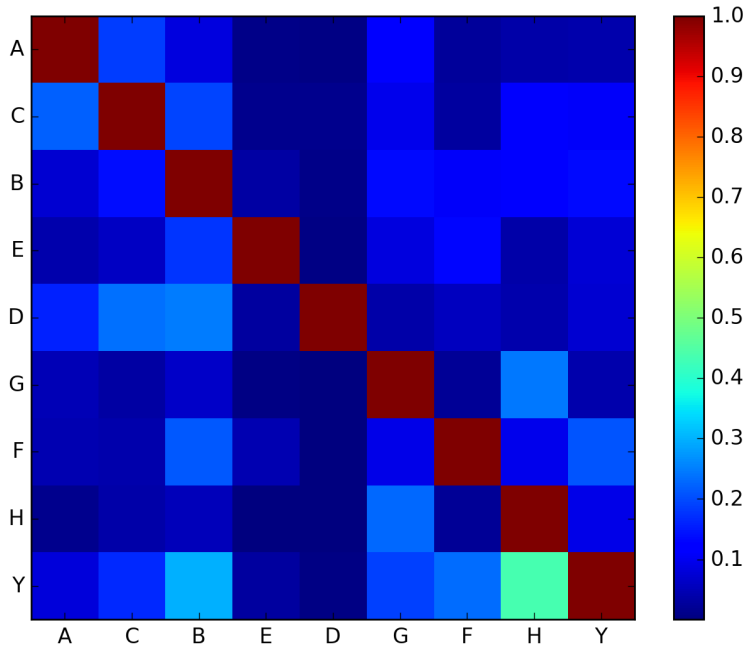

Figure 5.8: Overlap of sections in the CPC dataset

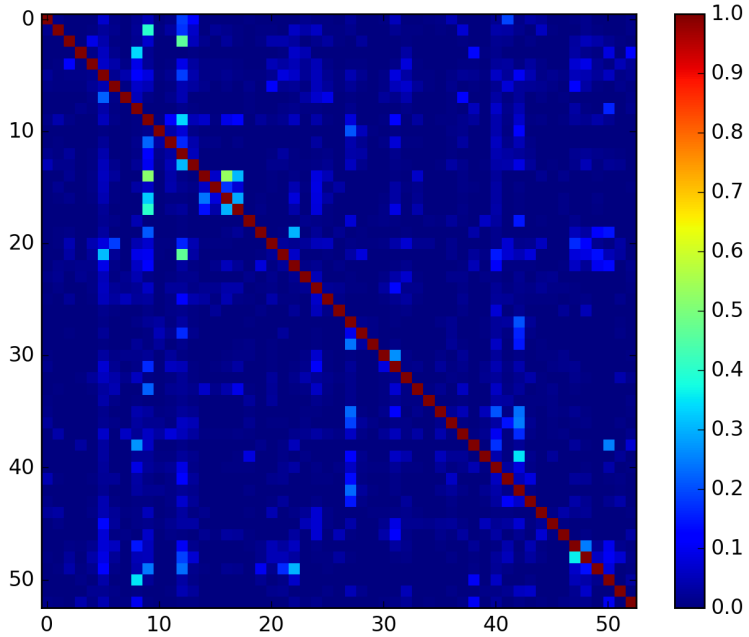

Figure 5.9: Overlap of classes in the CPC dataset; class-labels are replaced for readability

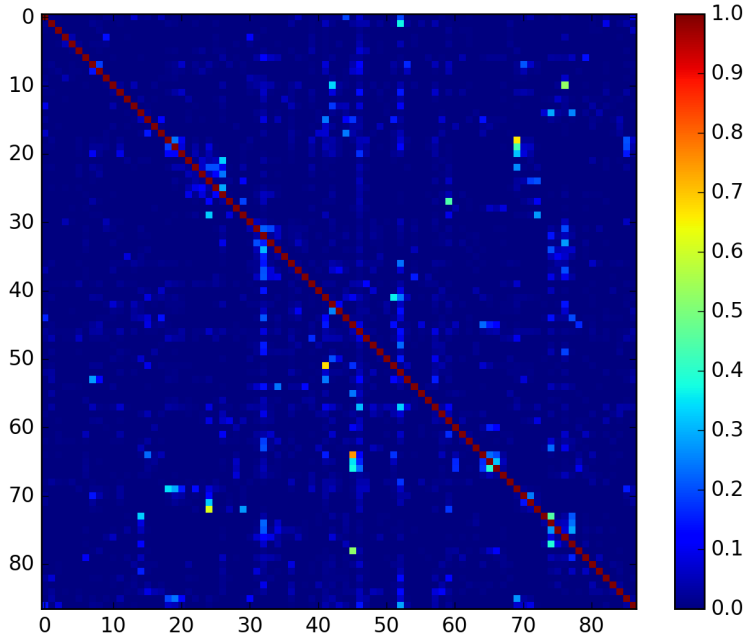

Figure 5.10: Overlap of subclasses in the CPC dataset; class-labels are replaced for readability

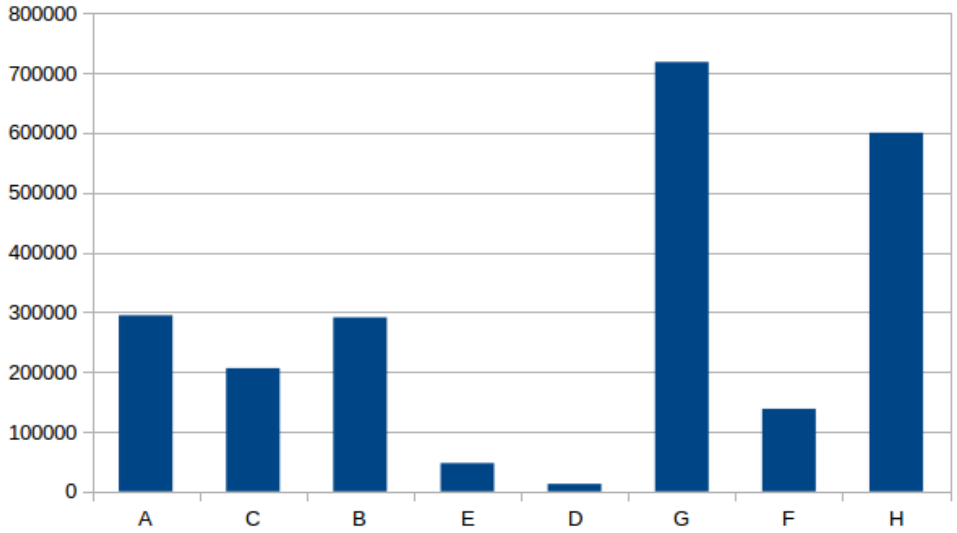

Figure 5.11: Section distribution in the IPC dataset

With the CPC-classification the requirement of a large dataset is not sustainable. Using the older but similar IPC-classification (see Section 5.1.2) the available data increases manifold. A side effect of the larger dataset is the increase in class-numbers on the levels below sections.

The dataset contains about 2,000,000 patent grants from 2006 to 2015. Documents of all 8 sections of the IPC are found. On the class level 387 different categories are available and 1840 different subclasses are contained. The uncompressed size of the corpus is about 200 GB. Figures 5.11, 5.12 and 5.13 indicate that the class distribution is not even. Overlap between classes can also be observers as Figures 5.14, 5.15 and 5.16 show. When considering a threshold of at least 40% overlap, 3 cases on the class-level can be observed and 28 cases on the subclass-level.

The CPC-dataset is created by matching the application-document to the grant document. This step is omitted in the creation of the IPC-dataset in order to get as many documents as possible. Experiments with the CPC-dataset show, that modifications of the text-fields between the application and the grant are rare (approx. 10%).

5.4 GeoDAT

5.4.1 Introduction

In the following Sections, a geo-enriched method for patent classification is proposed, that exploits the fact that companies working in the same field of technology are often clustered in certain geo-spatial regions. For example, many companies working in the field of space travel are located close to the NASA facilities at Houston, Texas. Thus, given the address of an inventor, a good estimation about the technological fields of the invention can be

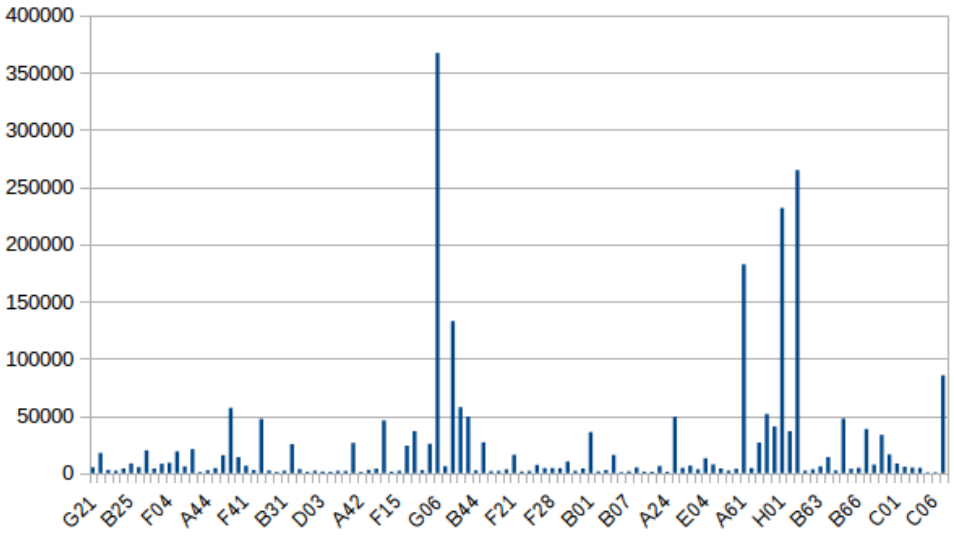

Figure 5.12: Class distribution in the IPC dataset

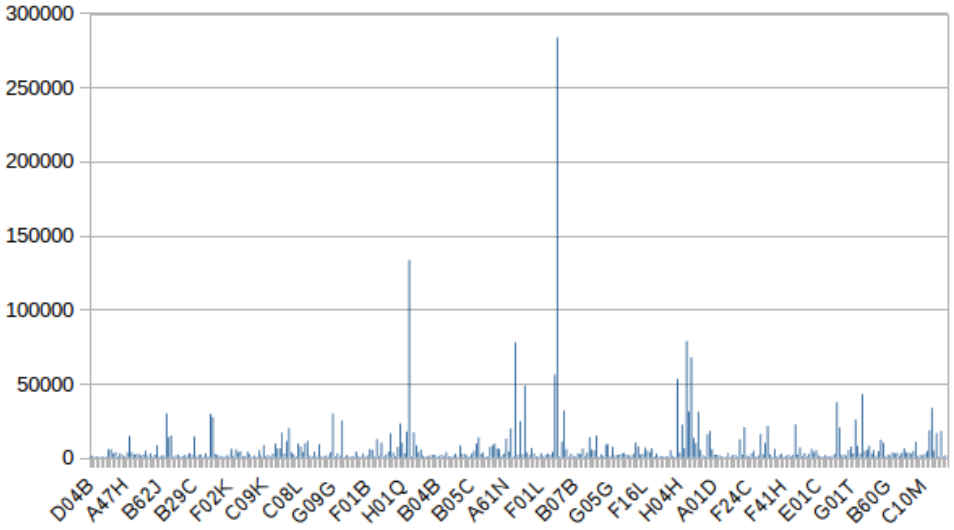

Figure 5.13: Subclass distribution in the IPC dataset

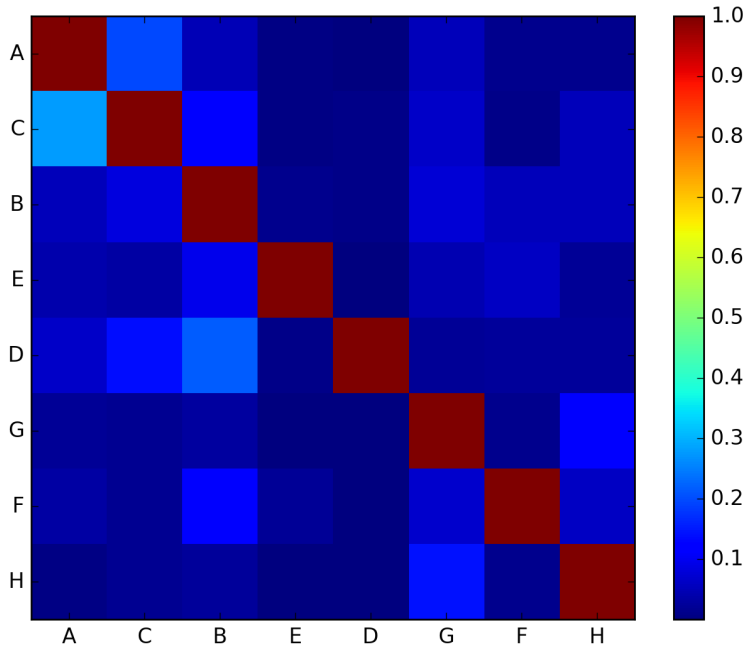

Figure 5.14: Overlap of sections in the IPC dataset

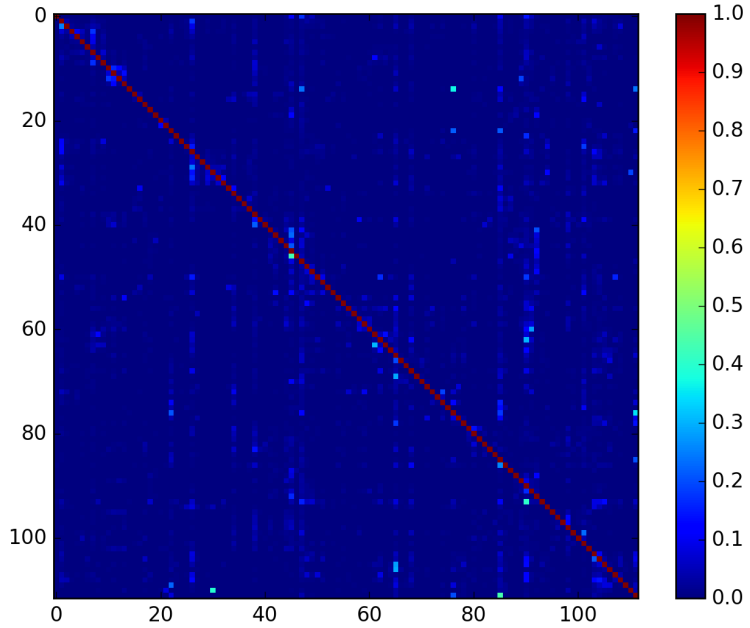

Figure 5.15: Overlap of classes in the IPC dataset; class-labels are replaced for readability

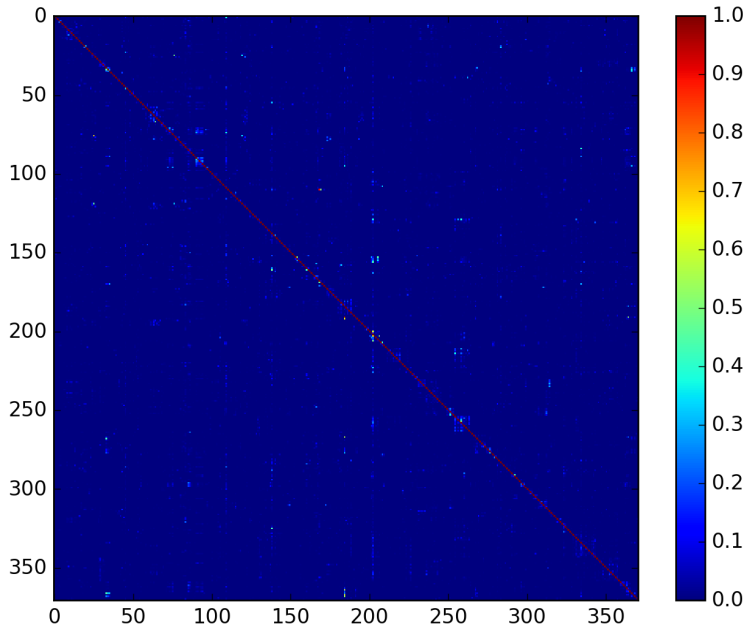

Figure 5.16: Overlap of subclasses in the IPC dataset; class-labels are replaced for readability

made. Using this type of background knowledge often provides valuable information, especially if the inventors on a patent are located in an area hosting certain types of industries. Therefore, the approach combines an established text classification method with a spatial predictor to improve classification. The spatial predictor derives an additional representation of the patent, by analyzing the categories of patents being written by spatially close inventors. The text and spatial representation are then mapped using a combined classifier to generate a common prediction of all CPC classes a patent could belong to as a ordered list with descending class probabilities.

That method is published [134] as a joint effort together with Matthias Schubert.

5.4.2 Related Work

Different perspectives can be used for related work. In the following Section the distinction between geographic aspects and bibliographic aspects is made.

The assumption, that certain industries concentrate geographically is explored by Porter [117]. He defines Industrial Clusters as "concentration of interconnected companies and institutions in a particular field". Various studies analyze industrial clusters (see [39, 43], though they are often very location-specific and the definition of the geographic size varies. All studies agree, that the size of a Industrial Cluster is "local", "metropolitan region" or "regional" without an exact definition. Industrial Clusters are seen as beneficial for innovations as the proximity of the actors stimulates interaction and exchange of ideas.

To measure the effectiveness of a Cluster, the work of [19] shows that patents are "a

fairly reliable measure of innovative activity". The patent-classes are not evaluated in this work. The authors note, that knowledge-spillover between Cluster participants is difficult to track and that the effects of the presence of a university is underrepresented. The authors of [111] empirically analyze a Cluster project in Japan and come to the conclusion, that the number of patents in a Cluster does not increase if there is no university in the Cluster. The distribution of patent-classes within the cluster is also not part of that work.

Though the Cluster-output can be measured with patents, the literature does not conclude whether the patent-class-distribution does reflect the focus of the Clusters, or not.

Using bibliographic aspects of the data is a widely used approach to enhance patent-classification. Co-citation is utilized by [90] in order to create an alternative patent-classification-system as the authors criticize IPC for not being analysis- and research-friendly. Similarity of patent documents is measured by co-citations and the resulting classification system is empirically evaluated by domain experts on the example of the semiconductor industry. The methods, the authors refer to it as patent-co-citation analysis (PCA), creates consistent classes though it struggles with non-classification and multi-class classification.

Enhancing a text-classifier with a meta-data-approach is researched by the authors of [119]. A k-Nearest Neighbor classifier is used to predict the class based on fields in the meta-data. The targeted classification systems do not include the IPC. Inventor/assignee-combinations are evaluated and, depending on the targeted classification system, significantly enhance the classification performance. Another work that enhances the results form a text-based information retrieval method (BM25) propose the authors of [63]. The top-N results of the text-based method are enhanced by co-citation analysis using either PageRank or a topic-sensitive citation method. The co-citations are represented as graph. The results of the text-based method and the co-citation system are combined by a static influence factor of the co-citation result. Due to the focus on information retrieval, the classification aspect is not considered.

The combination of the geographic and bibliographic aspects can be used to enhance patent classification in a existing classification system. This will be the focus of the following sections and has not yet been covered by the existing literature.

5.4.3 Method

This Section introduces an extension to text-based patent-classification methods utilizing meta-data. A document is separated into two representations. The first representation is a classical text representation. To map the text vectors to a vector of patent classes a one-versus-rest SVM classifier is trained. Additionally, a spatial predictor is created, mapping the addresses of the inventors to a second distribution over the patent classes. Finally, a neural network is trained to combine the results on both prediction methods and return a list of class membership probabilities. The complete classifier is depicted in Figure 5.17.

Each classifier C returns for any document d a list $C(d) = p_d = (p_{c_1}, ..., p_{c_i})$ containing the class membership probabilities with $p_{c_i} \in [0, 1]$ and $c \in K$ where K represents a set of all classes available e.g. all observed CPC sections. A class probability distribution is

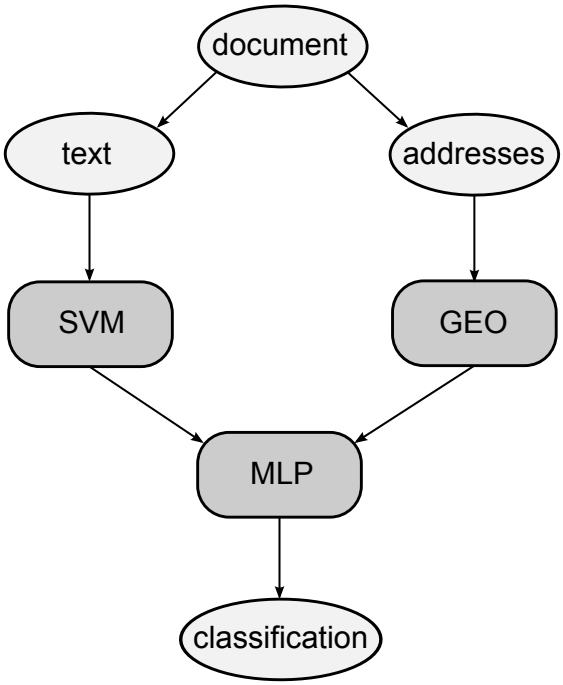

Figure 5.17: layout of classifier

referred to as p_d.

5.4.3.1 Text-Based Classification

For preprocessing text, the text parts of the description, title, abstract and claims are extracted. No weighting of any of these parts is performed. English stop word removal and stemming using the Porter stemmer are applied. The tokenized text using uni- and bi-grams of each document was vectorized using TF*IDF vectorization [115]. The vectorized text provides the input for the text classifiers. The dimensionality of the vector space is limited to the 10,000 most frequent terms. For text classification, a linear SVM C_{SVM} with a one-vs-rest multi-class strategy and parameter selection according to [36] is employed.

5.4.3.2 Spatial Prediction

To predict class probabilities from geo-data, the home addresses (street, zip code, city, state, country) of the inventors named in the patent grant is extracted and mapped to geographic longitude and latitude. The usage of the inventors home address over the applicants addresses is sensible as it provides more addresses and it mitigates the problem of nation- or worldwide cooperations which always use the same site to apply for patents. Note that one patent grant and application can have multiple inventors.

For training, a mapping of the classes of each patent grant to the addresses of its inventors is performed. This way, it is possible to aggregate the class distributions for each named address. The result is a set of locations L_T where each location l_i has its

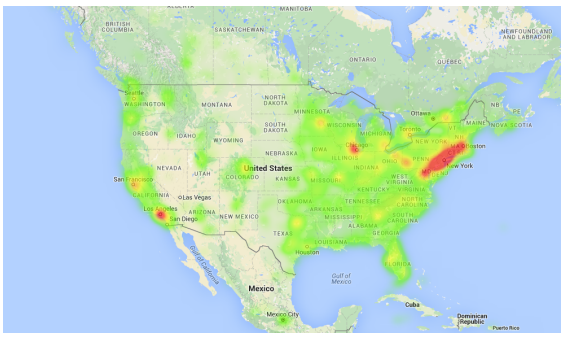

Figure 5.18: Locations of geo-classifiable documents with $\sigma = 2$

own distribution of patent classes p_l. To classify a new patent application d, resolving the inventors addresses $L_d = \{l_1, ..., l_i\}$ to geo locations and search for the k nearest known locations $L_{kNN}(L_d) \in L_T$ is done. A Kd-Tree [32] filled with the geo coordinates of all available locations of the training set B_T for the k-NN queries is used for enhanced performance.

Given $L_{kNN}(L_d)$, p_l for each $l_i \in L_{kNN}(L_d)$ is retrieved and used to predict the class probability distribution p_d. To consider that the distance between an inventor's address and one of the nearest neighbors influences the prediction accuracy, a decay term $s(\sigma, l_i, l_d) = e^{-dist(l_i,l_d)\cdot\sigma}$ with σ being a constant with $\sigma \geq 0$ referred to as the decay is introduced. In this Section the function $dist(a, b)$ is the distance between the geo coordinates a and b in kilometers calculated using the Haversine Formula.

The class probability distribution p_d for document d based on the locations L_d of the inventors' home addresses using a constant decay of σ is:

$$C_{GEO}(d) = p_d(L_d, \sigma, k) = \sum_{d=0}^{|L_d|-1} \left(\sum_{i=0}^{k-1} \frac{p_l \cdot e^{-dist(l_i,l_d)\cdot\sigma}}{k} \right) \cdot \frac{1}{|L_d|}$$

The intuition behind the two parameters σ and k is as follows: σ is used to limit the influence of a neighboring document onto the classification with respect to the distance between the inventors addresses. The cardinality $k = |L_{kNN}|$ controls the radius r of the region in which the nearest neighbors are evaluated and adjusts the region to density differences. The necessity for this can be observed on the east coast of the US (see Figure 5.18). Figure 5.18 shows each patent for $\sigma = 2$ and $k = 4$ for which a class probability distribution can be calculated, with red signifying a high density of patents and green a low density.

5.4.3.3 Combination Network

Finally, to combine the results of C_{GEO} and C_{SVM} each output is normalized and feed into a 4 layer neural network C_{merger} which is built using Keras [47]. C_{merger} returns as C_{GEO} and C_{SVM} a list of class membership probabilities. $|K|$ is the cardinality of the class set K. The network consists of $|K|\cdot 2$ input nodes and $|K|$ output nodes with two hidden

layers with 1200 nodes. As activation function PReLU is used and initialize the weights he-normal [76]. The learning algorithm is stochastic gradient descent with momentum. Early stopping is used to prevent over-fitting. To select the hyper parameters, best practices are followed as proposed by [27]. For every k/σ-combination an individual neural network should be trained to achieve optimal results.

5.4.4 Experimental Evaluation

In the experiments, four approaches to patent classification are compared. The baseline is represented by a random classifier C_{RND} which models the class distribution in the respective dataset. To present the state-of-the-art, the text-classifier C_{SVM} without geo-enrichment is used. To demonstrate the predictive power of the inventors address, C_{GEO} without any additional method is evaluated. Finally, the combination of both representations C_{merger} using a state-of-the-art neural net is tested.

The experiments are conducted with the CPC dataset. The dataset used and its properties are detailed in Section 5.3.

The dataset is split into training B_T and test dataset B_V with a ratio of 0.5. To consider a class, a minimum of 550 labeled documents is demanded of which at least half of them are put into the B_T. This selection results on the CPC section level in 9 classes and on the CPC classes level in 53 classes. In the CPC subclass level 87 classes satisfy the requirement of at least 550 documents per class. Further selection of documents reduced the available number so that no experiments on this and lower levels of the hierarchy are performed. In B_T 23,307 resolved locations provide a class distribution which is used to initialize C_{GEO}.

5.4.4.1 Success Criteria

As the classification of patents is a multi-label problem, a different metric than the accuracy is required, as this criterion tends to be very harsh on multi-label data where a mismatch of one of multiple possible classes is counted as a complete miss. Two measurements are used: coverage error m_{ce} and three forms of a custom success criterion m_n.

Coverage error m_{ce} describes the average of labels which have to be inspected by the algorithm in a sorted list of class probabilities to find all true class memberships of a document. The perfect score would be the average number of class memberships a_{ce}. The worst score would be the number of available classes $|K|$. In case that two classes have the same probability, coverage error is conservative and picks the last one. In regards of decision support this metric is important as the actual classes of a document should appear closer to the top.

The custom criterion m_n measures how many of the correct classes of a document are contained in the top n likely classes identified by the classifier and calculates the average over the entire classified set. This measurement reflects the actual use case where the automated system is used to support the manual classification. The best score would be 1.0 and the worst 0.0. By looking at the top 3, 5 and 10, a bias is introduced to favor documents which are member of more than 3, 5 or 10 classes. In our dataset the average

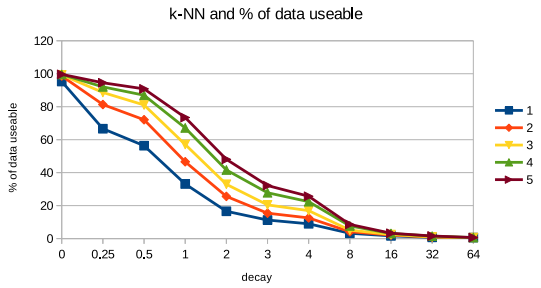

Figure 5.19: usable data vs. decay (σ)

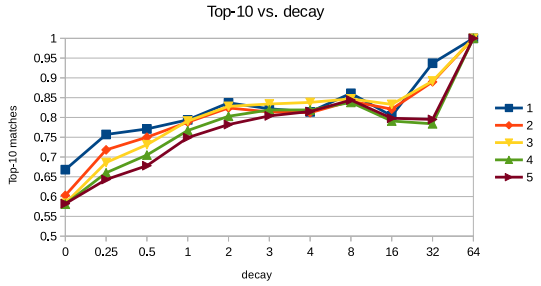

Figure 5.20: Top 10 matches vs. decay (σ)

Table 5.2: Results for CPC class level C_{merger} and improvement over C_{SVM}

k	a_{ce}	$\lvert B_{V_U}\rvert$	m_{ce}	m_{10}	m_5	m_3	σ	$i_{m_{ce}}$	$i_{m_{10}}$	i_{m_5}	i_{m_3}
1	1.48	16.57%	2.103	0.990	0.977	0.957	2.0	26.39%	1.12%	2.73%	5.28%
2	1.48	25.58%	1.992	0.992	0.981	0.963	2.0	35.39%	1.43%	3.37%	6.17%
3	1.48	32.91%	2.056	0.991	0.978	0.959	2.0	31.52%	1.33%	3.16%	5.97%
4	1.48	41.60%	2.001	0.992	0.980	0.963	2.0	34.53%	1.43%	3.38%	6.29%
5	1.48	48.05%	1.971	0.992	0.981	0.964	2.0	**36.23%**	1.43%	3.48%	6.40%

number of CPC section memberships is 1.3880 with a maximum of 6 CPC sections. The average number of CPC class memberships is 1.4836 with a maximum of 12 and the average number for CPC subclasses is 1.6962 with a maximum of 15 in the dataset. As this method is limiting the number of predicted classes with the metric m_n, documents with more than n labels will never receive a prefect score as this method is cutting off some true labels. This affects a few documents in the CPC class level (311 affected) and CPC subclass level (1415 affected) and influences the scoring as the calculated score might be slightly worse than the performance actually is. As a consequence, the best score of 1.0 can not be reached for class and subclass levels.

5.4.4.2 Results

Table 5.2 shows that the results with C_{merger} outperforms C_{SVM} w.r.t. m_{ce} by up to 36% for $\sigma = 2$. While in Table 5.4 C_{GEO} beats C_{RND} in m_n in all cases, a value of $\sigma \geq 0.5$ is necessary. Below this threshold the results are worse than C_{RND} which might be caused by the sparsity of the known locations and the insufficient discounting of distance between inventors address and observed class distributions in the environment.

Experiments show how the parameters k and σ influence the results of the geo classifier. As it can be seen in Table 5.4 the increase of k for a constant σ the number of documents to which the geo classification can be applied increases. This is explained by more neighbors increase the sum of probabilities and mitigate the influence of the decay. The experiments also show that an increase in k does not necessarily improve the classification results.

As shown in Table 5.5 the performance of C_{SVM} and C_{RND} is almost constant on the subsets B_{V_U} of the validation dataset B_V for which C_{GEO} could be applied. It can be conclude that the subsets do not contain a collection of particular hard to classify documents. The reason why m_{ce} for the geo classifier is extraordinary high (see Table 5.4) is that in many cases classes which are part of a document d_c are not observed in the environment of d_c. As the coverage error quantifies after how many visited labels all correct labels are found, these documents receive a score of $|K|$. The decrease of the coverage error with increasing decay indicates that the relationship between classes of d_c and the location the inventor is living is strongly dependent of the particular location.

For the parameter k, it can be observed that the percentage of the dataset which is classifiable with geo data B_{V_U} increases with increasing k. This effect is due to the fact that with an increasing number of neighbors the overall class probability distribution increases faster than the decay leads to a convergence close to zero. It can be concluded that this effect favors items where the environment has a consistent class probability distribution and additional neighbors are at a similar or not significantly further distance from the $k-1$ environment. While the geo classifier works best in regards of the m_n metrics when only using one neighbor, the method is in this case, only relevant to 16.57% of the data. It is also worth mentioning that m_{ce} becomes smaller with an increase of k as more, and particularly, the correct classes become available.

While k has an impact on B_{V_U}, parameter σ has even more so. As is shown in Figure 5.19 for $\sigma = 2$ B_{V_U} for all tested k is well below 50% (see Table 5.4 for exact numbers). Figure 5.20 shows that the improvement in the metric m_{10} declines for $\sigma > 2$ while B_{V_U} drops for all $k > 1$ with $k = 1$ already being as low as 11.24%. Experiments show that while a smaller σ can improve the metrics m_n, m_{ce} is suffering. At the same time a larger σ (e.g. $\sigma = 3$) leads to $B_{V_U} < 33\%$.

5.4.5 Conclusions & Outlook

This work shows, that the results for an automatic patent application classification system can be improved by incorporating the home address of the inventors. Especially the important metric coverage error can be hugely decreased. For a decision support system

Table 5.3: Results for CPC class level C_{RND}

k	a_{ce}	$\lvert B_{V_U}\rvert$	m_{ce}	m_{10}	m_5	m_3	σ
1	1.48	16.57%	13.986	0.635	0.516	0.391	2.0
2	1.48	25.58%	14.029	0.635	0.515	0.388	2.0
3	1.48	32.91%	14.088	0.632	0.512	0.385	2.0
4	1.48	41.60%	14.072	0.632	0.514	0.385	2.0
5	1.48	48.05%	14.038	0.642	0.515	0.385	2.0

Table 5.4: Results for CPC class level C_{GEO}

k	a_{ce}	$\lvert B_{V_U}\rvert$	m_{ce}	m_{10}	m_5	m_3	σ
1	1.48	16.57%	36.758	0.837	0.632	0.496	2.0
2	1.48	25.58%	36.321	0.824	0.627	0.438	2.0
3	1.48	32.91%	36.230	0.828	0.643	0.464	2.0
4	1.48	41.60%	34.655	0.803	0.611	0.441	2.0
5	1.48	48.05%	33.236	0.782	0.591	0.426	2.0

where the order of the class list indicates the membership probability the decrease of m_{ce} by up to 36% is a vast improvement. Another result is that this method can describe the influence of an environment on the application classification by using the parameters σ and k. An evaluation of these parameters showed how they interact with each other and which combination yields good results.

The text-based classifier matched the performance of the state of the art, which relied on IPC, on the CPC system. Our combined approach outperformed the text-based classifier. We would like to test this method on a larger datasets using the IPC system. New challenges would include that the applications and grants are available over a longer time frame and cyclic patterns are observed by other studies. So the selection of a reasonable time frame and possibly dealing with a shift in terminology are tasks which have to be solved for the text-based classification. Another issue is the reclassification of documents. The classification systems like CPC and IPC are steadily expanded to match new fields of research. The availability of reclassification data is uncertain.

Considering geo data the automatic identification of centers for certain technologies and their spatial extension poses an interesting question. The usage of co-authors in patent grants to link those centers seems promising. Furthermore, an investigation the radius r in which k-NNs influence the patent application classification should be done in more detail, with regards to technological fields and population density in particular.

Table 5.5: Results for CPC class level C_{SVM}

k	a_{ce}	$\lvert B_{V_U}\rvert$	m_{ce}	m_{10}	m_5	m_3	σ
1	1.48	16.57%	2.658	0.979	0.951	0.909	2.0
2	1.48	25.58%	2.697	0.978	0.949	0.907	2.0
3	1.48	32.91%	2.704	0.978	0.948	0.905	2.0
4	1.48	41.60%	2.692	0.978	0.948	0.906	2.0
5	1.48	48.05%	2.685	0.978	0.948	0.906	2.0

5.5 Fixed Hierarchy Vectors

5.5.1 Introduction

Recently, topic models such as Latent Dirichelet Allocation(LDA) (see Section 3.3.4) and word embeddings such as word2vec (see Section 3.3.5) or Paragraph Vectors (PV) (see Section 3.3.5) demonstrated the capability to map documents into a feature space representing document content independently from the actually used words. Even though these methods might be tuned to any application, they do not exploit document structure beyond a certain degree. In many document collections, however, all documents are required to have a unique structure. For example, scientific articles appearing at a particular journal or conference usually have to include an abstract, keywords, main sections and a bibliography.

Another very important collection of documents following such a fixed hierarchy are patents. Like publications, patents describe novel methods, technical advances and genuine ideas. However, unlike publications patents are not formulated to efficiently transfer knowledge but to provide as broad as possible protection of the contained intellectual property. This often causes large problems when employing standard text processing methods for searching and categorizing patent data. To facilitate patent research, patent offices categorize patents into huge taxonomies like the Cooperative Patent Classification (CPC) or the International Patent Classification (IPC). Manually assigning new patent applications to the correct set of classes within these taxonomies is a time-consuming and expensive process. Thus, automatic patent classification offers the opportunity to speed up the process and relieve experts.

In the following Sections, a novel method for text classification is presented. The focus is on document collections requiring a fixed structure. Experiments demonstrate its advantages on the challenging use-case of patent classification. From a technical point of view, the new method is based on two ideas. The first is to learn word embeddings which are specialized on particular parts of a document. Given these specialized embeddings, a document is considered as a hierarchy where parent nodes summarize further partitioning of the document. In particular, the root of the hierarchy describes the complete document, whereas the next levels describe the fixed semantic structure of the document. This representations is denoted as Fixed Hierarchy Vectors (FHV) because the partitioning is currently based on the fixed structure in the analyzed collection. The second idea behind

the method is that understanding a text is a sequential process and paragraphs often can only be interpreted in the right way when knowing the previous content of a document. Thus, sequentializing FHVs by a depth-first-traversal and classify the document using a Recurrent Neural Network (RNN) (see Section 3.2.4) mimics the reading process of a human. The experiments on a large patent corpus demonstrate that FHVs provide a better picture of the document content and that sequential classification yields better performance than vector-based approaches. In summary, the contributions of this work are:

- A novel document representation for text corpora with a fixed structure.
- A classifier mimicking the sequential process of understanding a document.
- An evaluation based on the challenging task of patent classification.

The rest of the Sections are structured as follows: The following Section explains the purpose of FHV. Section 5.5 specifies the problem setting, describes our novel hierarchical document representation and explains the sequential classification method based on this representation. The Neural Network (NN) architectures, employed by the method presented here, are described in Section 5.5.2. In Section 5.5.3, we show that FHV outperform other approaches like PV and bag of word approaches (BOW) on a large patent corpus. Section summarizes the contributions of this work and discuss directions for future work in Section 5.5.4.

That method is to be published as a joint effort together with Marawan Shalaby, Matthias Schubert, Hans-Peter Kriegel and Stephan Günnemann.

5.5.2 Method

5.5.2.1 Problem Setting

Given a set of documents $D = \{d_1, \ldots, d_n\}$ where d_i is a structured document. The structure of a document is given as a tuple $d_i = (p_1, \ldots, p_l)$ for each $d_i \in D$. For example, in the use case of patent classification each patent is represented as a tuple of three parts $(abstract, description, claims)$. Each part p_i is described as text and later on will be considered as a sequence of tokens $(t_1, \ldots, t_m)$ such as words, shingles or other textual primitives. In this setting, each document $d_i \in D$ can belong to multiple classes $c_j \in C$ where C denotes the set of all classes. $class(d) = \{c_j, .., c_k\}$ denotes the set of all classes document d belongs to. Thus, the problem at hand is a multi-label classification problem. In this use-case, the first three levels of the International Patent Classification (IPC) are examined. The task is to train a function $f : D \rightarrow 2^C$ to map document d to the correct set of classes $class(d)$.

The proposed solution to this task is based on two steps. First, unsupervised representation learning is employed to map each document to a hierarchical representation by traversing through the hierarchical representation and creating a sequence from the representation. For the classification-step RNNs are used with the sequences as input to predict the classes the document belongs to.

5.5.2.2 Fixed Hierarchy Vectors

This Section describes the transformation of a text document into a representation which is more suitable to capture its contents. As mentioned above, a document d_i is represented as an ordered tuple $(p_1, \ldots, p_l)$ of partitions. Each partition p_i consists of a sequence of tokens $(t_1, \ldots, t_m)$. A token can either be a term or a sequence of characters, e.g a shingle. A key task of learning a good text representation is to consider the right context to interpret each token. Whereas word2vec showed that the local context of surrounding words is important, other approaches, e.g. PV, showed that this very local context is often not enough to completely capture the meaning of each token. Thus, this new approach is following the idea of combining text representations for various levels of locality from surrounding tokens up to the complete document. In the following, each document d_i is represented as a context hierarchy $H(d_i)$. Formally, $H(d_i)$ is a tree where each node $N_{i,j}$ describes a part of the document, i.e. the token subsequence $(t_i, \ldots, t_j)$. The children of $N_{i,j}$ represent a complete and disjunctive partitioning of $(t_i, \ldots, t_j)$. In other words, for node $N_{i,j}$ having l children nodes there are $l+1$ border indices $s_0, \ldots, s_l$ where $s_0 = i$, $s_l = j$ and $\forall k \in \{0, \ldots, l-1\} : s_k < s_{k+1}$. Thus, the ith child node corresponds to the subsequence $(t_{s_i}, \ldots, t_{s_{i+1}})$ of token sequence of the father node $(t_i, \ldots, t_j)$. Therefore, each level of $H(d_i)$ contains all tokens in d_i, but lower levels split the token sequence of d_i into more partitions.

$H(d_i)$ is generated as follows: The root level represents the complete document d_i. The next level is denoted as semantic level because d_i is split along the fixed semantic partitioning $(p_1, \ldots, p_l)$, characteristic to the document collection. Finally, on the third level, each partition p_i is split into a fixed number of chunks k_i. Chunking is done by considering the complete number of tokens m in partition p_i and evenly split p_i after $\lceil \frac{m}{k_i} \rceil$tokens. Note that each partition might have a varying number of chunks which is selected based on the average length of the partition. For example, the description part of a patent will be split into more chunks than the abstract part which naturally contains far less tokens.

After defining the context hierarchy, each node in the hierarchy is described by a meaningful embedding vector. To take full advantage of the variety of contexts, a feature transformation is needed that is taking the given level of context into account. Since paragraphs in PV [95] are not strictly defined but can be applied to any partitioning of text like document, sentence, chunk or section, PV perfectly complements the document hierarchy proposed above. Therefore, it is feasible to employ the paragraph vector (PV) model to learn an individual feature transformation for the nodes in the context hierarchy $H(d_i)$. In particular, a separate model is trained for the root level and each node on the semantic level. For the chunk level, a single model is trained for all the children of each node on the semantic level. For example, one model is trained over all chunks in the description part. Thus, the number of trained PV models is $1 + 2 \cdot l$, i.e. one for the root, l for the semantic partitions and l for the chunk level. In the patent-classification use-case, 7 PV models are trained, given that there are 3 semantic partitions. Experimental evaluation shows, that PV-DM outperforms PV-DBOW in most of the cases. Due to these results, the PV-DM

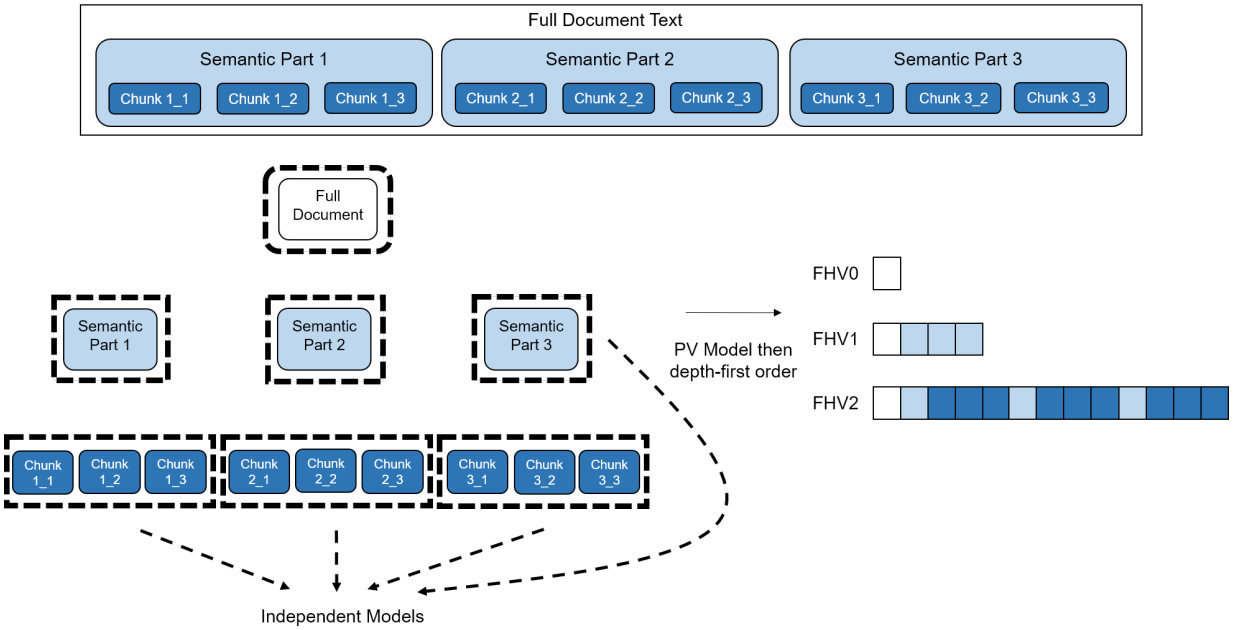

Figure 5.21: An complete overview of representing documents as FHV. On the top is the context hierarchy on each document including the full document, the semantic parts and the chunks. The bottom depicts how the context hierarchy is used to train PV models and the three levels of sequential representations that can be derived. Whereas FHV_2 includes the full hierarchy, FHV_0 and FHV_1 stop at root level and, respectively semantic level.

model is employed for learning FHVs. Within the PV-DM model, intermediate results are combined by taking the average due to its better performance and faster runtimes compared to concantenation. More details on PV-DM are provided in Section 3.3.5.

For inference on a new document $\hat{d}$, all partitions of $\hat{d}$ are transformed into the context hierarchy $H(\hat{d})$. Afterwards, the PV models for each node of the hierarchy are applied to map each node to the q-dimensional embedding vector where q corresponds to the dimensionality of the latent space in all the PV models. Thus, deriving a single vector for the each node in the context hierarchy $H(\hat{d})$. An overview on the complete document representation is depicted in Figure 5.21.

5.5.2.3 Classification based on FHVs

The simplest way to do document classification based on FHVs is to flatten the hierarchy by concatenating the vectors representing its nodes. Let us note that this is only possible because the number of nodes in the context hierarchy is exactly the same for each document. This naive approach does not provide the best results for text classification as presented in Section 5.5.3.

Understanding a complicated text document for a human is often a timely process. To capture the contents of a complex document as in our use case of patents, people often read

multiple times over the same document. Whereas the first read has the purpose of getting a general overview, subsequent reads aim to get a deeper understanding of the details. The reason for this behavior is that some of the details within the document cannot be judged correctly without knowing the complete context. Another aspect, which is natural to understanding text is sequential parsing. Usually a document is read from the beginning and often sections within a document cannot be understood correctly, without knowing the contents of previous sections. For example, in a research paper, the method description usually depends on nomenclature and definitions being described earlier in the document. To build a classification scheme which mimics both effects, the proposed method traverses the FHV hierarchy in a depth-first traversal and thus, generates a sequence of q-dimensional embedding vectors. The complete sequence has a length of $1 + l + \sum_{i=0}^{l} k_i$ vectors. The sequence starts with a vector describing the complete document at root level and thus, represents a quick overview on everything contained. Due to the depth-first-traversal, the next vector represents the context of first item of the semantic elements contained in the documents. At the level below all semantic structures, are the chunks which represent the content at a greater detail as the context is narrowed down to the number of tokens contained in a chunk.

To classify the resulting sequence of embedding vectors, it is necessary to employ a method which is capable of keeping important context information presented earlier in the sequence for an arbitrary amount of time. Thus, all sequential classification schemes using a fixed-order Markov property are not suitable to exploit FHVs. A method providing the wanted property are Long Short Term Memory (LSTM) RNNs [79]. LSTMs are known for their capability to keep relevant context information even over the cause of long input sequences. LSTMs are discussed in detail in Section 3.2.4. Evaluation of various network architectures show that a single LSTM layer is a good trade-off between prediction quality and model complexity. The evaluated architectures as well as hyper-parameter tuning of the LSTM classifiers is described in Section 5.5.3.

5.5.3 Experimental Evaluation

5.5.3.1 Dataset

In contrast to the previously discussed GeoDAT, the method introduced in this Sections focuses on text. Supervised training is used to learn embeddings. Due to the many parameters that are adjustable during training of these embeddings, the amount of data required is significant. Mikolov et al. evaluate the effect of a larger corpus on the accuracy of a Semantic-Syntactic Word Relationship test set [107] that shows that the quality increases significantly with the number of training words for a sufficiently high dimensionality of the embedding. For that reason the dataset containing only CPC classified documents does not contain sufficient instances to train the embeddings for high dimensional data (see Section 5.3.1). The dataset used is based on the IPC classification and does only consider patent grants so that no matching between application and grant reduces the available amount of data. The dataset used for evaluation is covered in detail in Section 5.3.2.

5.5.3.2 Success Criteria

To evaluate the various classifiers, the standard $F1_{micro}$ and $F1_{macro}$ measures are commonly used in text classification tasks. Micro-averaging sums up the classification decisions of all instances (whether an instance is a true positive, false negative, etc..) regardless of class, then computes the $F1$ score based on this aggregate sum. This helps to give a general picture of how effective the classifier is in general at identifying the correct labels for documents. Macro-averaging on the other hand computes the $F1$ score independently for every class, then averages the $F1$ scores for all classes to arrive at the $F1_{macro}$ score. This provides a more complete picture of how the classifier fares when there is an unbalanced distribution of the number of documents for each class whereas classes with a large number of documents can dominate smaller classes in $F1_{micro}$. Two additional measures geared towards multi-label classification are used: Coverage Error and Top 3 percentage. Coverage error counts the elements in a ranked list of labels that have to be covered on average in order to account for all the true positive labels. It is calculated as follows:

$$\text{Coverage Error} = \frac{1}{N}\sum_{i=1}^{N} \max_{y_j \in Y[i]} rank(y_j) - 1 \tag{5.1}$$

While Top 3 calculates the percentage of correct labels that are within the top 3 scores.

5.5.3.3 Experiment Setup

The highly efficient multi-threaded Gensim [118] Python implementation is used to generate the PVs. The training of the PV models is executed on a 32-core machine with 380GB of RAM using multiple concurrent pipelines. Creating the paragraph vectors for all different nodes in the hierarchy takes approximately 2 days to complete. For MLPs and RNN, the Keras learning library [47] is used and experiments are performed on a machine with 64 GB RAM and a Nvidia Titan X for tensor operations.

5.5.3.4 Classification Results

For a concise evaluation of the classification performance of the proposed method, various combinations of classifiers and vector representations are compared. This is to evaluate the performance of either the use of embeddings in the patent-classification task, then to determine the effect of the FHV and finally to get insights if the sequence-representation of documents contains additional information that are accessible. Each of these research-questions is compared to a baseline. To determine the use of embeddings and FHV, a comparison to X^2 and LDA representation is made. The following classifiers are used to measure the suitability of the representations: Linear Support Vector Classifier (SVM), Multi-Layer Perceptrons (MLP) and RNNs. SVM uses a One-Vs-Rest strategy for training a linear kernel. The direct comparison of MLP and RNN is used to exemplify that additional information is contained in the sequence-representation and can be exploited.

Table 5.6: Section Classification Results

Pipeline	Coverage Error	F1 Micro	F1 Macro	Top 3
SVM & X^2	1.696	0.734	0.671	0.924
SVM & LDA	1.875	0.617	0.401	0.896
SVM & FHV_0	1.733	0.663	0.569	0.923
MLP & X^2	1.425	0.791	0.704	0.972
RNN & X^2	1.429	0.786	0.693	0.973
MLP & LDA	1.462	0.776	0.689	0.967
RNN & LDA	1.482	0.763	0.672	0.965
MLP & FHV_0	1.414	0.793	0.736	0.975
RNN & FHV_0	1.410	0.798	0.747	0.975
MLP & FHV_1	1.380	0.814	0.766	0.978
RNN & FHV_1	1.368	0.821	0.776	0.980
MLP & FHV_2	1.374	0.818	0.772	0.979
RNN & FHV_2	**1.355**	**0.828**	**0.787**	**0.982**

The parameters for SVM are chosen using a smaller validation-set. Parameters and architectures for MLP and RNN are the result of our hyper parameter evaluation which is described in Section 5.5.3.5 in more detail.

To evaluate the document representations, we employ BM25 [100] (see Section 3.3.3) on uni-grams and bi-grams, LDA, PV and FHV. BM25 is selected as experiments on a smaller training/validation set proved it to be superior to other BOW representations. Feature selection with X^2 is performed on the BM25 vector space to retain the top 10,000 features. A parameter evaluation of LDA leads to a dimensionality of 1000 topics. With MLP and RNN outperforming the SVM on PV, LDA and X^2 by a wide margin, no further experiments are conducted with the other representations using a SVM. For the method FHV, the semantic hierarchy levels are distinguished into $l \in [0, 1, 2]$ of the input data. FHV_l in the classification experiments is composed as follows:

- FHV_0: operates on the root level of $H(d)$. This method corresponds to the standard approach to using PVs.
- FHV_1: uses only the root and the semantic level of $H(d)$ (e.g. abstract, description, claims in the patent-classification example).
- FHV_2: uses three levels in $H(d)$. In total, 30 chunk are added in comparison to FHV_1: 3 chunks for the abstract, 23 chunks for the description and 4 chunks for the claims.

All the FHV experiments are conducted on a vector space with 200 dimensions and the embeddings are trained for 8 epochs with a dictionary size of about 400,000 terms. The number of terms is determined by imposing a minimum word count threshold of 100

Table 5.7: Class Classification Results

Pipeline	Coverage Error	F1 Micro	F1 Macro	Top 3
SVM & X^2	6.359	0.630	0.178	0.735
SVM & LDA	48.591	0.417	0.032	0.611
SVM & FHV_0	5.074	0.542	0.130	0.665
MLP & X^2	2.866	0.682	0.206	0.861
RNN & X^2	2.927	0.665	0.176	0.859
MLP & LDA	3.440	0.640	0.155	0.827
RNN & LDA	3.544	0.616	0.154	0.816
MLP & FHV_0	2.791	0.663	0.208	0.857
RNN & FHV_0	2.698	0.684	0.225	0.866
MLP & FHV_1	2.535	0.703	0.240	0.877
RNN & FHV_1	2.438	0.712	**0.255**	0.885
MLP & FHV_2	2.501	0.710	0.245	0.882
RNN & FHV_2	**2.410**	**0.721**	0.251	**0.889**

occurrences. RMSprop is used as the learning algorithm and a batch size of 4096 with the exception of experiments with FHV_2 data. Due to memory constraints, a decrease of the batch size to 2048 is necessary for FHV_2.

For regularization of the neural networks, early stopping with a patience of 15 epochs and a minimum required decrease of $1e-5$ in validation loss is applied. Dropout is used throughout the network with $p = 0.5$. The same methods and values are also used for the RNNs.

Tables 5.6, 5.7 and 5.8 show the results of the experiments for the IPC section, IPC class and IPC subclass levels of the patent classification. We can see a consistent advantage of RNNs over MLP when it comes to FHV representations and a consistent increase in performance for both models as more hierarchy-levels are added to FHV.

For the SVM classifier, FHV_0 lags behind X^2 in $F1_{micro}$, $F1_{macro}$ and Top 3, but that can be explained by the fact that the lower dimensional representation employed by FHV is not as amenable to linear separation as the high dimensional bag of words format, and may require a non-linear discriminator. LDA is another representation that benefits greatly from the non-linearities of MLP and RNN even though it performs worse than either X^2 or FHV.

An experiment is conducted to test whether the use of independent models for the varying nodes in the tree as we have explain in Section 5.5.2 is justified as the time- and space-complexity of the method is greatly increased by that approach. In that experiment all the nodes in the tree are added to one model and monolithic model is trained using all the documents and their subdivisions. As Table 5.9 shows, the results of this experiment reaffirmed that the use of independent models for independent contexts is necessary to achieve good results as the results of the monolithic model lags far behind those of the independent models.

Table 5.8: Subclass Classification Results

Pipeline	Coverage Error	F1 Micro	F1 Macro	Top 3
SVM & X^2	16.322	0.507	0.130	0.488
SVM & LDA	165.311	0.266	0.014	0.161
SVM & FHV_0	15.784	0.423	0.093	0.325
MLP & X^2	7.096	0.567	0.168	0.753
RNN & X^2	7.381	0.515	0.107	0.741
MLP & LDA	9.510	0.506	0.102	0.696
RNN & LDA	9.588	0.473	0.103	0.689
MLP & FHV_0	6.818	0.526	0.167	0.737
RNN & FHV_0	6.531	0.559	0.186	0.749
MLP & FHV_1	5.860	0.583	0.201	0.768
RNN & FHV_1	5.677	0.605	**0.223**	0.782
MLP & FHV_2	5.728	0.590	0.206	0.775
RNN & FHV_2	**5.482**	**0.612**	0.215	**0.789**

Table 5.9: One Model Results

Pipeline	Coverage Error	F1 Micro	F1 Macro	Top 3
Sections RNN & FHV_0	1.585	0.697	0.605	0.950
Sections RNN & FHV_1	1.465	0.771	0.709	0.968
Classes RNN & FHV_0	3.550	0.558	0.119	0.807
Classes RNN & FHV_1	3.055	0.636	0.172	0.837
Subclasses RNN & FHV_0	10.567	0.399	0.059	0.667
Subclasses RNN & FHV_1	8.036	0.502	0.120	0.714

5.5.3.5 Hyper Parameter Evaluation

For MLPs, the parameters are selected by applying a random search [34] procedure on a validation set over a grid of parameters for layer size (100 to 2000), number of hidden layers (1, 2 or 3) and hidden layer activation functions (relu, sigmoid and tanh). The reported numbers are for the best performing combination of each experiment. For RNNs, which require much longer to train, a grid search over a more constrained set of parameters is performed. tanh is used as the activation function and grid search only performed on the hidden layer size where 1000 neurons are determined to be the best performing across the experiments.

In the previous Section, the neural networks consist only of a single layer, however stacking of layers may lead to performance gains. Table 5.10 shows the results of stacking for the same model for IPC-sections and it clearly shows an advantage for adding a second RNN layer. Adding a third layer however leads to additional model complexity and runtime duration, but no significant performance improvements.

The effect of running PV training for different epochs $e \in [1, 2, 3, 4, 5, 6, 7, 8]$ and the

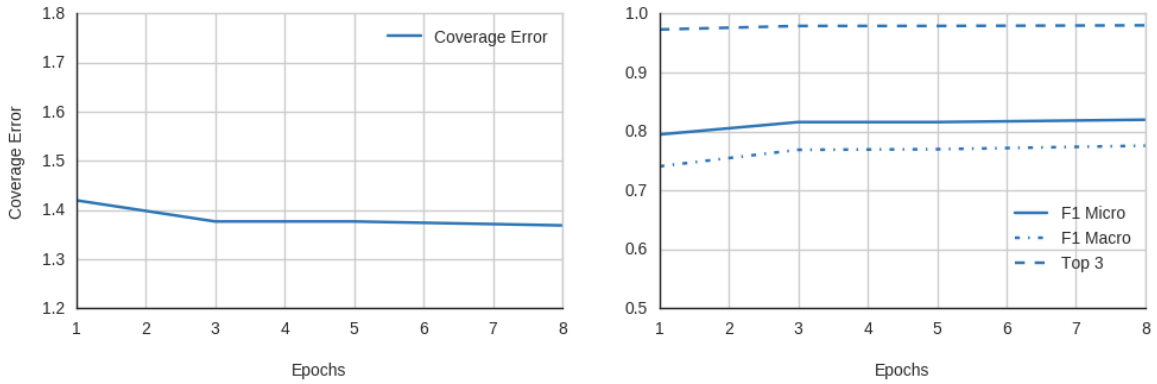

Figure 5.22: Progress over epochs for sections on the same RNN model for FHV_1.

Table 5.10: RNN Stacking for sections

Model	Coverage Error	F1 Micro	F1 Macro	Top 3
1-layer RNN & FHV_0	1.410	0.798	0.747	0.975
2-layer RNN & FHV_0	1.390	0.808	0.761	0.978
3-layer RNN & FHV_0	1.391	0.808	0.761	0.978
1-layer RNN & FHV_1	1.368	0.821	0.776	0.980
2-layer RNN & FHV_1	1.362	0.823	0.780	0.981
3-layer RNN & FHV_1	1.360	0.824	0.781	0.981

effect they have on the performance of our model is also evaluated. As Figure 5.22 shows, running paragraph vectors for around 3 epochs is enough to produce sufficiently good results. After that, diminishing returns for additional epochs can be observed. Though this is dependent on the number of training instances in the dataset as tests with a smaller dataset take more epochs to reach a point of diminishing returns.

Finally, experiments with adding a convolutional layer in front of the LSTM layer as a way of finding more descriptive features are conducted. However, it is discovered that adding the convolutional layer does not lead to any significant increases in performance. Adding a max pooling layer after the convolutional layer always leads to worse performance.

5.5.4 Conclusions & Outlook

In the previous Sections, a new approach to classify documents is presented, with an inherent structure in multi-label settings. The method uses the inherent, semantic structure of a document to derive a hierarchical description of the document. Whereas each level in this hierarchy represents a different level of summarization, the children of each node represent a sequential partitioning of the parent node. For each node in the hierarchy, a paragraph

vector model is trained, mapping the corresponding text in the document to a lower dimensional representation vector. To classify a document, experiments show that the propose a depth-first traversal through the hierarchy is superior to a breadth-first-representation. The resulting representation is mirroring a coarse to fine as well as a sequential process of text understanding. The final classification is done by an RNN consisting of LSTM neurons which allows for remembering already seen contextual information. The results indicate that this new method can beat state-of-the-art approaches in the challenging use case of patent classification. Furthermore, the experiments indicate that the sequential processing using LSTM shows a better performance than general MLP on a concatenation of the complete sequence.

Another aspect the experimental evaluation shows, is that the performance gain from the semantic level provided by abstract, description and claims is significantly larger than the performance gain reached by adding chunks to the sequence representation. From this observation two possible directions for future work can be derived: chunking drastically increases the computational complexity with relatively little to gain. So there might be a more efficient way to perform chunking and to estimate a good number of chunks per semantic item. The other direction would abstain from the use of chunks and utilize solely semantic items of a document. Which is strongly connected to the next conclusion.

One of the limitations of the proposed approach is that it requires a fixed hierarchy for all documents in the corpus, which is probably not the case for all document corpora. A future direction for this work is to figure out if imposing a synthetic hierarchy for a document (e.g. by using chunking starting from the first level in the tree as opposed to just in the leaf nodes) would lead to improved results as well. Another research-question would be the comparison of documents with a dissimilar semantic structure. The number of PV-models and their mapping to semantic elements would have to be restructured for that. Additionally, this approach can be extended to other types of document datasets. Furthermore, experiments can examine whether this approach is suitable for other data types which humans also process in a sequential way and may have to study several times until understanding is reached like video data.

5.6 Conclusions & Outlook

Automatic patent-classification remains a difficult problem. With the digitalization of the USPTO and the free access to all patent grants and application documents the entry barrier for research in that area is low enough to attract a wider audience of researchers. While progress is made, especially coming from the the information-retrieval field concerning the identification of prior-art, the classification of a document into various classes of a predefined hierarchy remains challenging. One of the reasons is that the hierarchy is artificial in the sense, that it does not represent the actual classes that would occur in the corpus using a distance measure. This is partly a result of the fast technological progress in comparison to the bureaucratic process that is necessary to adapt the classification system. Another reason is, that the difficulty of the classification problem can be increased

by descending another level in the patent classification hierarchy. Last but not least, methods that rely on text-analysis suffer from the fact, that patent attorneys aim for a generic expression of the innovation in order to gain protection as broad as possible.

Related work, state-of-the-art methods and the experience gathered during the development of the methods proposed in this Chapter indicate, that for a full-fledged automatic patent classification system an ensemble-approach is recommendable. This thesis proposes methods that can enhance two different aspects of the patent-classification: meta-data usage and text-analysis. Experiments show, that resolving the inventors addresses to geo-coordinates allows the identification of Industrial Clusters that can be described by a individual patent-class distribution. Factoring this distribution in with the results of another classifier leads to a better classification performance.

By using FHV, the text-analysis of a patent for classification can be enhanced significantly. The experimental results indicate, that this semantic approach is superior to the BOW methods that describe the patent documents or elements of a patent document with context-free statistics. Another benefit is that a sequence-representation can be derived from FHV that mimics the parsing of a document by a human and models different context-levels chained together in a semantic meaningful way. State-of-the-art sequence-learning is able to exploit this representation which results in a further increase of classification performance.

A integrated patent-classification system that combines the GeoDAT approach together with FHV is interesting future work. This ensemble-approach could be extended by the integration of a co-citation systems that use graph-representations. Also a unsupervised method that is not bound by an predefined patent classification system could prove to be useful as the similarity of patent documents might not be correctly captured by the classification system. The specialization of a patent attorney and a patent assignee can provide additional information to refine the classification.

With sufficient classification accuracy, a analysis and monitoring of changes and trends in certain elements of a patent-classification system with an adaption of the methods proposed in Section 4 could be used to gain further insights.

Chapter 6

Long-Term - Macro-Indicators

The previous Chapters propose methods to utilize data for the short- and medium-term. In Chapter 4 time-series of different metrics over a dynamic corpus of documents is used to differentiate between signal and noise. Chapter 5 proposes methods to increase the utility of patent documents for analysis-purposes. Considering a framework like PEST (see Section 2.2.2), the proposed method for short-term analysis does cover all of its aspects. While it is argued, that the short-term method can be extended for medium-term-use, scaling-issues and noisy data make it unpractical for long-term analysis and monitoring. Patent-classification covers the technological aspect of PEST in the medium-term range and is a promising candidate for long-term analysis due to the lower noise to signal ratio. Nevertheless, for a long-term coverage of the factors politics, economics, social and technology, a different approach, relying on a different data-sources is required. This Chapter discusses a method that uses macro-indicators and models found in the literature to provide insights.

6.1 Introduction

An indicator is a statistic value of a target variable (e.g. Gross Domestic Product (GDP)), usually consisting of various observations over time. General directions of the target variable can be derived form these indicators. Popular indicators for economic development are Gross Domestic Product, Consumer Price Index, unemployment figures and the price of oil. Besides the economic indicators, other fields provide their own observations e.g. the World Happiness Index [18], Democracy-Dictatorship Index [44] and the Word Development Indicators [17].

Different organizations publish indicators in varying intervals. While the relevance of a particular indicator towards a target variable can only be determined by an domain expert, the integration of various such data-sources into a single directory can support an analyst. Reducing the the time that is spent searching for a source, automatic updates of data, increased visibility of indicators and automatic evaluation of data-quality are benefits of an integrated approach. This Chapter proposes a system, that provides these benefits by

following a modular approach for every step in the processing pipeline.

With integration of the data from multiple sources, the creation of individual models relying on those services becomes feasible. Regular updates allow monitoring of certain target variables. By applying extrapolation methods to the time-series, an educated guess can be made about future developments. This work evaluates the quality of most of the 10.000 indicators provided by the World Bank and introduces a new quality score.

While domain-experts can select relevant indicators, the selection of an extrapolation-method for a given indicator is a different task. This work proposes the use of the method BestFit which automatically selects the extrapolation-method with the least diversion from past values to predict future observations.

The indicators used in this work are often referred to as macro-indicators. Similar to the distinction in the environments defined for in Strategic Management, the macro refers to the granularity of the observed object. In the case the indicators are observations are made about state-actors, and are typically updated yearly.

6.2 Related Work

Merriam Webster defines an indicator as "any of a group of statistical values (such as level of employment) that, taken together, gives an indication of the health of the economy" [9] or more general, "one that indicates". In the context of this work, an indicator is understood as a collection of time-series of observations. Each time-series is associated with a country. The observation is in relation to a target-variable, e.g. level of unemployment.

Indicators can have the property of leading, lagging and coincident. Generally speaking a lagging indicator follows an event while an coincident indicator reacts at the same time as an event is happening. Leading indicators predict an event or condition. A popular example is the unemployment rate as an indicator for economic performance.

Due to the importance of the level of unemployment, the authors of this paper [80] develop a new leading indicator for unemployment in Germany. The new indicator is validated using existing models and covers the time-horizons of 1, 2, 3 and 6 month. A survey is used to create the new indicator. This indicates the comparatively high costs that are associated with the creation of an indicator compared to environmental scanning using Web data. As the geographical scope of this Chapter is world wide, indicators are required to provide data for as much countries as possible.

Leading indicators are a popular field of research, especially in the field of economics. The determination of a leading indicator requires domain knowledge and the ability to track an indicator. An analysis of the behavior of established economic leading indicators regarding the 2001 economic crisis [132] showed that not all of them were able to predict the crisis. The author concludes, that inconsistent behavior of singular indicators regarding past crisis require the constant observation of multiple indicators combined to generate a warning signal.

Meta-indicators model the target-variable by proxy. An example is patent data that correlates with R&D expenditure. The authors of this publication [73] suggest the use of

patent data as it is more readily available.

Institutes like the Pew Research Center [11], the Organization for Economic Co-operation and Development (OECD) [10], Word Bank [16] and United Nations Statistics Division [14] collect data and present the information in the form of multiple indicators on-line. Pew Research Center conducted over 450,000 interviews in 64 countries to collect the information [11]. The indicators can be found on the web-presence of these organizations or purchased. The data-types and formatting varies from organization to organization. In addition to these organizations, many institutes publish only one indicator, e.g. Polity IV Project [12].

Larger data provider, like the World Bank, offer visualization and browsing of the data on-line. For a deeper analysis, download is necessary and other sources can not be included in the online-tools. A thorough search of the related literature yielded no indication that the integration of multiple indicator-sources has already been done.

The literature about methods to asses the quality of an indicator is usually concerned with the predictive power or the correct representation of the target-variable. In the context of this work though, the problem arises that the integration of multiple sources yields an overwhelming number of indicators. Because of that fact a measurement is needed to determine, to quickly determine the coverage of an indicator regarding the time- and country-dimensions.

With time-series extrapolation, a coincident indicator can become a leading indicator. Thus, extrapolation methods on time-series are an active field of research. The authors of survey [103] compare the performance of Support Vector Regression (SVR), wavelet transform, neural networks, kNN and ARIMA among others. Due to the nature of a survey, the different methods are tested on different datasets against different baselines. For that reason no recommendation for an extrapolation method is made.

In publication [124], the use of SVMs from the field of machine learning for forecasting is analyzed. The authors conclude, that SVR is a viable method for time-series extrapolation, and the various kernels and hyper-parameters of SVR allow sufficient flexibility to create good models.

Autoregressive Integrated Moving Average (ARIMA) models are commonly used to model time-series. In [81] the authors propose a method to automate the process of selecting the order of an ARIMA model. The proposed method uses a penalized method that is based on in-sample data so that no additional validation data is needed. An iterative process is proposed, where the best model out of multiple is selected by minimizing Akaike's Information Criterion (AIC). The first iteration is used to determine whether the time-series is cyclic or not. In the next iteration, four different configurations are tested. Depending on the AIC, up to 13 more variations of the best previous model are evaluated.

6.3 Method

This work has multiple objectives:

- create a system that allows analysis and visualization of indicators and the use of models relying on them
- integration of indicators from various sources
- the efficient identification of usable data
- an easy-to-use option for time-series extrapolation

. The following Sections discuss the approach to each of these objectives in detail.

6.3.1 Indicators and Models

Definition 9 (Indicator)**.** *Indicators in this work refer to a set of time-series associated with countries measuring a target variable. The dimensions are time, country and observations. The observations are equidistant in the time-dimension. Missing observations are possible.*

The analyst who is using the system must consider any other properties of the indicator, e.g. if it is a lagging or leading indicator, the scale and what is the target variable. Such considerations are not included in the proposed system.

In the context of this Chapter, models combine various indicators to provide observations for a new target variable. The result is an indicator according to the definition in 9. This requires models to preserve the country- and time-dimension while altering the observations. Due to the dependency of models on indicators and the same data type, the presentation- and interaction-interface to the analyst can be the same. Also any processing applied to an indicator can be applied to a model.

The interface of the analyst and the processing of the data is separated into parts that can interact. This reduces complexity as the components of each part are decoupled and allows other interfaces to utilize the processing-unit.

Different data-sources are provided in various formats, e.g. CSV, Excel-files, XML and many more. Integrators are proposed to parse the data-source and make the information available to the processing-unit. Besides the parser for the data, an integrator can provide regular updates of the data and, depending on the data-source, can provide data on demand, without having to store everything locally.

6.3.2 Quality Score

As there are a multitude of indicators which are not raised every year or only for a selected, few countries, this score allows a user to quickly estimate if this indicator I could be potentially valuable. An indicator I consists of $|I|$ time series $X = \{x_{t_1}, .., x_{t_n}\}$ where each time series contains the data for one country. The score is normalized between 0.0 to 1.0 with 0.0 signifying no values are available and 1.0 signifying that all values are available.

To calculate the score, three criteria are considered. The first criteria incorporates how many countries are available in a given indicator ($score_{countries}$). While the number of the

countries varies over time, $i_{countries}$ is the sum of all former (starting 1960) and current countries is used, as it is not known which time frame is of particular interest for the analysis so that a country configuration could be adjusted. Each criteria is associated with a weight w that allows to emphasize the corresponding score.

$$score_{countries} = |I|/i_{countries} \tag{6.1}$$

Another criteria is how many data is contained in the various time series of each country ($score_{completeness}$). As experiments show, it is a common occurrence for an indicator to be raised only once for many countries which would yield a good score with $score_{countries}$ but be of little value if development over time is of interest.

$$score_{completeness} = \sum_{X \in I} |X|/(t_c - t_0) \tag{6.2}$$

The third criteria $score_{gaps}$ measures if there are gaps in the time series. The counting of a gap starts only after the first observation is made.

$$score_{gaps} = \frac{\sum_{X \in I} [\forall x_i \in X : x_{i-1} \in X \vee i = min(i)]}{|I|} \tag{6.3}$$

For the final score $score_{quality}$ the criteria are weighted and summarized.

$$score_{quality}(I) = w_{countries} * score_{countries} + w_{completeness} * score_{completeness} + w_{gaps} * score_{gaps} \tag{6.4}$$

6.3.3 BestFit

Various methods allow the extrapolation of time-series. Each of these methods has its strengths and weaknesses and creates a different model. To determine which method produces the best fit for a given time-series, BestFit is proposed. The BestFit-method is based on the assumption, that the method creating the best model of a time-series in the past will produce the best model for the present. BestFit applied to an indicator, selects on a per-country level the method, that generates the model where the predictions fit the actual values the best. For each regressor $c \in C$ and each time-series $X \in I$ the prediction of k values is performed. Each regressor c is trained with the time-series $X_{train} = \{x_{t_1}, ..., x_{n-k}\}$ and validated on $X_{valid} = \{x_{n-k}, ..., x_n\}$. The metric used to determine the performance of the predictions $X_{predict} = \{p_{n-k}, ..., p_n\}$ is the average Manhattan distance between predictions and observations (see Equation 6.5). The method with the minimum distance between $X_{predict}$ and X_{valid} is used to extrapolate the time-series by another k observations.

$$\frac{\sum_{i=n-k}^{n} |x_i - p_i|}{n} \tag{6.5}$$

6.4 Experimental Evaluation

For experiments and evaluation, a web-based system is created. The programming language used for the processing-unit is Python, as the popular scikit-learn [115] provides support for multiple extrapolation methods. Directory-Services are provided by a MySQL database. The front-end is implemented in Javascript and relies in HTML for structure. Evaluation of the quality score and BestFit is performed on indicators provided by the World Bank Data Catalog [16]. Due to filtering for duplicates and the omission of indicators that do not fit the expected yearly publication frequency, the number of indicators usable for evaluation is reduced from about 10.000 to approximately 8544.

As a prove of concept for models, the "Global Model for Forecasting Political Instability" according to [69] is implemented with three different target-variables: Civil War Onset / Adverse Regime Change Onset, Civil War Onset and Adverse Regime Change Onset. The global model [70], created by the State Failure Task Force is also implemented. These models rely on indicators e.g. child mortality or the Polity dataset.

6.4.1 Indicators and Models

Integrators for the CSV file-format, in particular for the Polity IV dataset [12] and the Political Violence Dataset are implemented. The Data Catalog of the World Bank [16] is also connected via an integrator that uses JSON for access and downloads requested indicators on demand.

The proposed tool to process and present indicators and models consists of two distinct components. One is for visualization and user interaction which runs in a user's web browser (see Section 6.4.1.1). This component is referred to as the client component or just client. The second component, the processing-unit is called the server (see Section 6.4.1.2) and it performs the data processing, storage, caching and keeping a directory of all available modules. The communication between the components relies on asynchronous JavaScript and XML (Ajax) but uses Javascript Object Notation (JSON) instead of XML.

6.4.1.1 Client

The client offers the following windows: "Map", "Graph", "Indicators", "Models", "Methods", "Infos", "Datatable" and "Calculate Formula". Figure 6.1 shows a screen-shot of the client. The most prominent object of the client is the window "Map" which displays a interactive world map showing most of the sovereign states $i_{countries}$ and a slider. Depending on the selected indicator the color of a country changes on the spectrum between red and blue with blue being the lowest value and red being the highest. Because of the multitude of indicators no semantic meaning is associated with the colors. The colors are only calculated on a year to year basis, so there is no comparison possible between years based on the color. The color grey is reserved for "no data available". The slider allows the user to skip through the years of an indicator starting in the year $t_0 = 1960$ until the current year t_c plus 15 years t_f. The minimal and maximal value of each year and

the corresponding country is displayed left and right of the slider. The exact value of a country at a given year is displayed when the mouse hovers over the country.

In the "Indicators" window the user can browse the available data sources which provide indicator data. These are organized into three select elements: "Topic", "Source" and "indicator". "Topic" and "Source" are mutually exchangeable. After each source is the number of available indicators in parenthesis. After each indicator in the drop down menu is a quality score $score_{quality}$ for the indicator and the indicator id I_{id}. The quality score is only available for indicators previously used and gets updated each time the data is refreshed by the server.

After selection of an indicator the "Infos", "Datatable" and "Map" windows are updated. "Info" displays a description of the data source, the indicator and if selected, the inter- and extrapolation method. The "Graph" window is now usable. By clicking one or more countries, the development of the selected indicator over time is plotted. Clicking on the country again will remove it from the graph. The "Datatable" window is displaying the values of the year selected by the slider in the "Maps" window and the previous year indicating increasing and decreasing values by the colors red and green. The semantic of the colors does not follow the semantic of an indicator. E.g. an increase in "child mortality" is displayed green.

The window "Models" is on the same level as "Indicators". It allows the selection of predefined models from the server and is discussed in the Section 6.4.1.2. The extrapolation selection can be used to extend model time series into the future. In the window "Methods" a user can choose how a time series should be processed. To evaluate simple models the window "Calculate Formula" allows the input of simple formulas operating on any indicators. The indicators are referred to by their I_{ID} and can be used like normal variables. Results are displayed in the same windows as indicators. To provide as much flexibility to the analyst the data loaded into the client can also be exported into a CSV file. Given a custom integrator (see Section 6.4.1.2), the data can be re-imported into the system.

6.4.1.2 Server

With the client being the interface to the user and for visualization of data, the server mainly fulfills three tasks: directory service, data management and computation. The directory service contains lists of modules and sources, the data management keeps previously downloaded information and updates it dynamically as much as uses caching to reduce the load caused by computations.

As the data in the directory service is highly structured, all information is stored in a relational database. The resources in the database include lists of extrapolation methods, interpolation methods, indicator sources, indicator topics, indicators, models, country codes, "integrators" and the mapping of indicators on topics and sources. The extra- and interpolation methods are capsuled in modules which can be loaded on demand and applied to any time series. The directory contains a description of each method and the information for the server on how to load the modules. The list of models follows the same structure. The indicator directory contains all the meta data associated with

an indicator. Namely the source, the topic, the quality, a description and the integrator. Also the date of last update of the indicator together with information about the scale and minimal and maximal value can be stored here.

Figure 6.2 depicts the data-sources (green), directories (yellow) and the integrators/exporters (mauve).

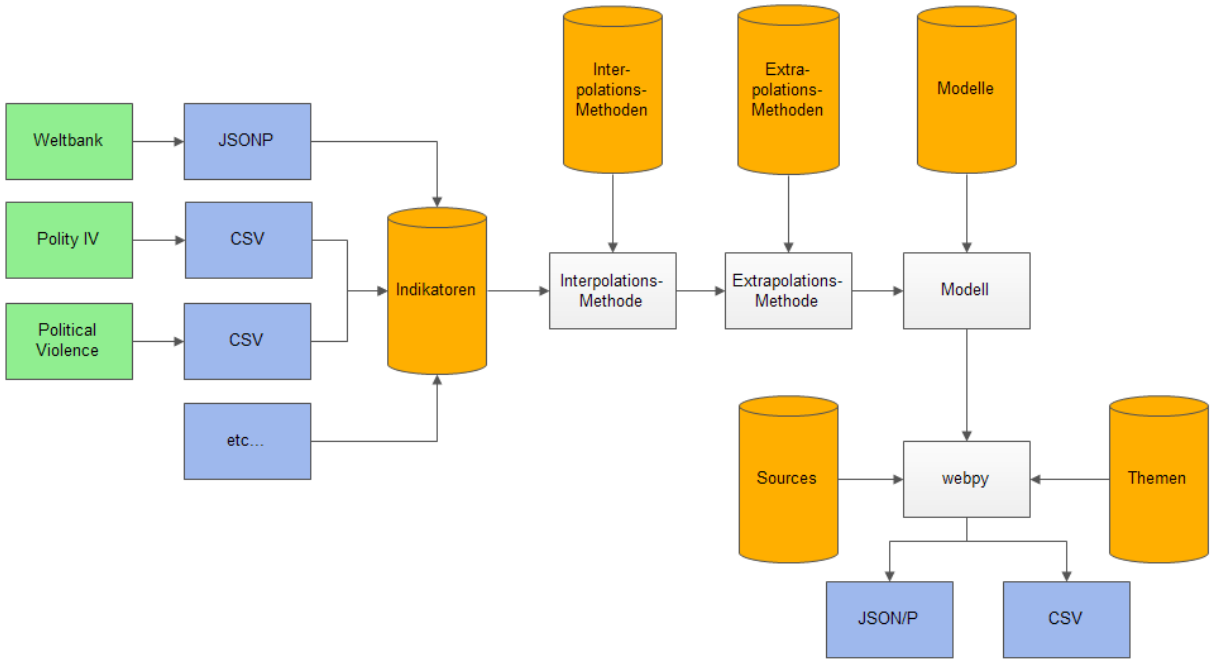

Figure 6.2: architecture of the server component showing examples of integrators and exporters

One aspect of the data management is that most of the data is not stored on the server. When it is possible only the available data catalog of a source is imported into the directory service. This reduces the memory requirement and the stress on a data source. To allow the server to retrieve requested indicators on demand integrators are used. An integrator is a module that describes the data format of a source and translates it into the format used on the server. Application Programming Interface (API) keys are stored in an integrator. The concept of integrators allows the dynamic inclusion of any time series that provide data about a country. Another aspect of the data management is caching. This refers to keeping the output of the integrators as much as keeping the results of extra- and interpolated indicators as well as models.

The computations performed on the server are inter- and extrapolation of time series, evaluating models and suggesting extra- and interpolation-methods. The models are treated similar to indicators with the distinction that not just a single indicator is required but usually several from various sources. As the model is a module on the server, more

methods to manipulate the data are available. Besides that, models are treated as an indicator after the initial stage. As a result, all inter- and extrapolation methods are available to the results of models as to indicators.

6.4.2 Quality Score

The weights are chosen to emphasize the world wide coverage with $w_{countries} = 0.5$, $w_{completeness} = 0.25$ and $w_{gap} = 0.25$. Time-series-completeness and the avoidance of gaps is weighted as equally important.

At the time of the experiments, 212 countries are considered. Due to the vanishing of various countries since 1960 (e.g. Soviet Union) and the inception of new countries (e.g. Djibouti) the actual score can be greater than 1.0, depending when the indicator was set up. The quality of approximately 8544 indicators from the World Bank is evaluated with the proposed scoring. Figure 6.3 indicates, that more than 25% of the indicators do not provide sufficient data. Just 29% reach a score higher than 0.5.

The following Figures allow conclusions why the score of many indicators is so low. Not many indicators cover all countries as Figure 6.4 illustrates. A major issue with the data is completeness (see Figure 6.5). A complete time-series from 1960 till 2013 is expected for a perfect score of 1.0. Only 23 indicators out of 6438 satisfy the criteria completely. Two are exceeding expectations by already containing extrapolations which is not accounted for.

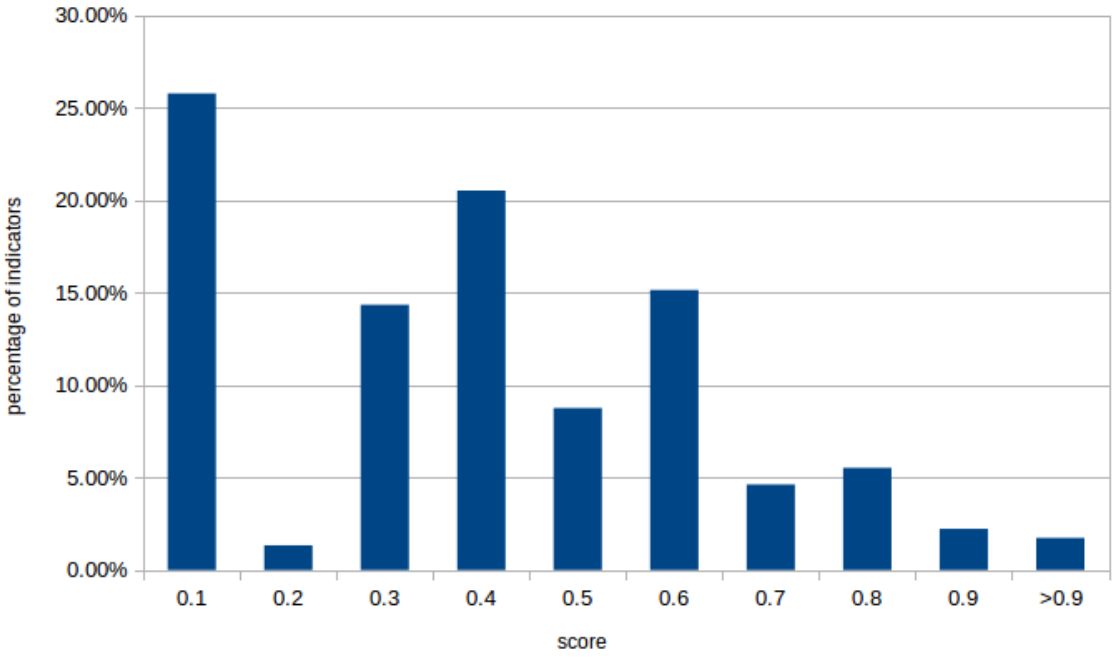

Figure 6.3: Distribution of $score_{quality}$ of the evaluated indicators

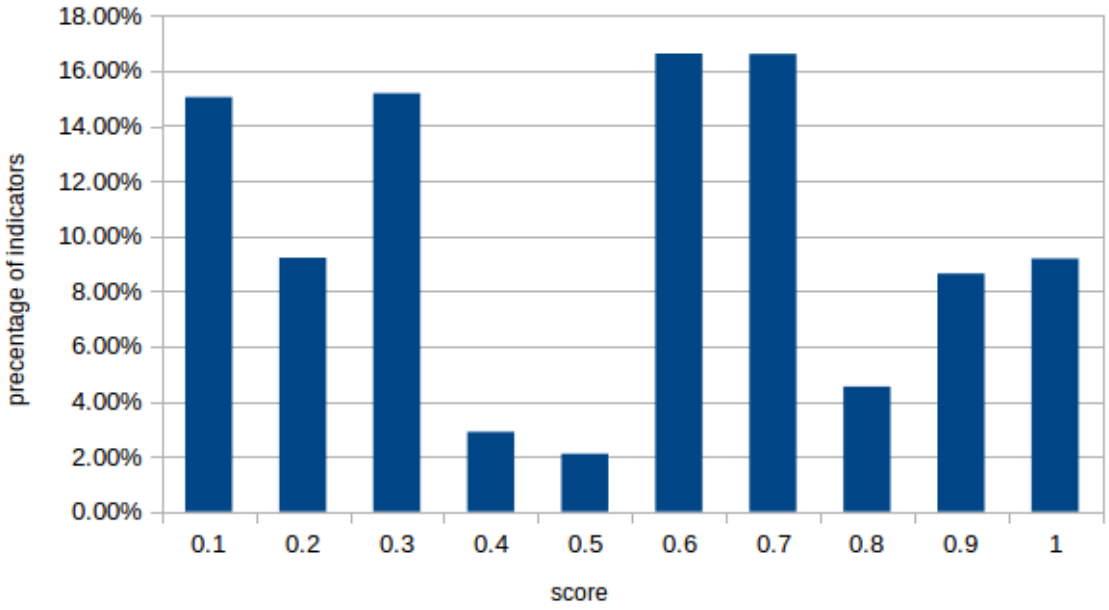

Figure 6.4: Distribution of $score_{countries}$ among the evaluated indicators

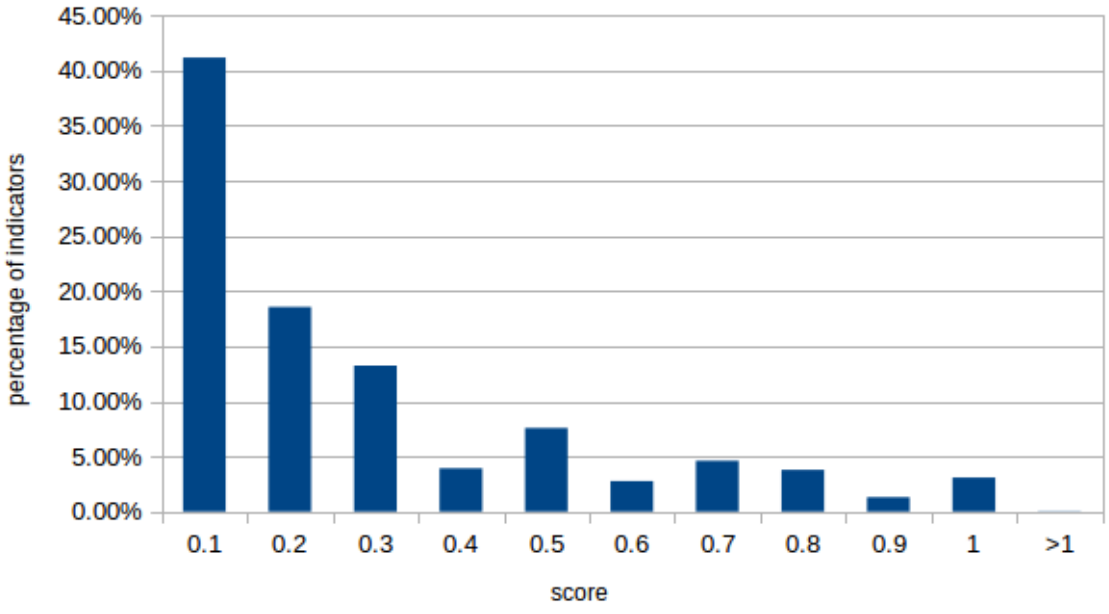

Figure 6.5: Distribution of $score_{completeness}$ among the evaluated indicators

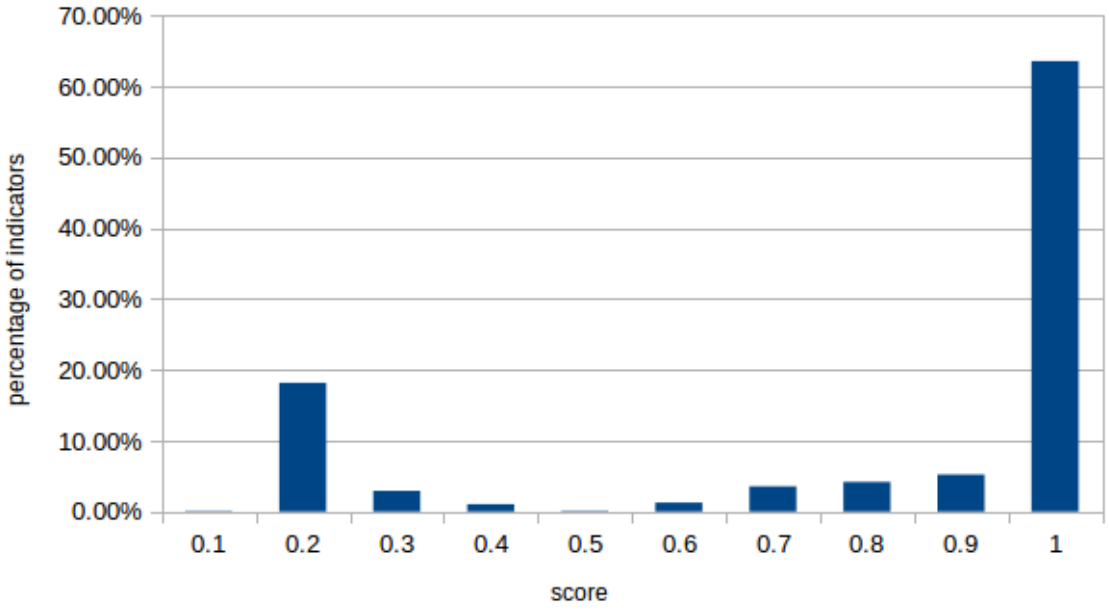

Figure 6.6: Distribution of $score_{gap}$ among the evaluated indicators

Experiments to evaluate $score_{gap}$ are only performed on time-series with at least two observations. This reduces the amount of available indicators to approximately 5279. The results are very encouraging (see Figure 6.6) and indicate, that when a time-series interrupted the duration of the gap is minimal as 64% of the inspected indicators score 0.9 or higher.

6.4.3 BestFit

In this Section, the performance of various regressors $c \in C$ is evaluated. As the method BestFit is applied on a per-country level to an indicator and each $c \in C$ is tested, the time-series of every country in an indicator is tested with 6 different regressors. 356 randomly chosen indicators provide sufficient data for the empirical analysis. The value $k = 15$ for a 15 year prediction is used.

The regressors in C and their parameters used in the evaluation of the BestFit-method are:

- SVR - linear kernel with $C = 1$, $\epsilon = 0.0001$
- SVR - polynomial kernel with $C = 0.00001$, $\gamma = 0.1$, degree=4, $\epsilon = 0.0001$
- SVR - radial basis function (rbf) kernel with $C = 0.0001$, $\gamma = 0.1$, $\epsilon = 0.0001$
- auto Autoregressive Integrated Moving Average (ARIMA) [81]
- linear regression

- MLP 30 hidden neurons with tanh as activation function

Experiments show (see Table 6.1), that in most cases the auto ARIMA produces predictions that differ the least from the ground-truth. It is not possible to quantify the accuracy of the prediction, as many of the indicators used are of categorical nature or use percentage. An effect of this is, that being off by the Manhattan distance of 50 when predicting the future population has less of an impact than being off by the same value when percentage is used. Table 6.1 indicates, that it is necessary to evaluate more than one method and that a large percentage of the indicators can be extrapolated by the simple linear regression.

Further experiments must be conducted to make a statement about the extrapolation quality, as the dataset used is too diverse regarding scales, so that no comparison is possible of the quality-criteria used in this work beyond the indicators. Another issue is the applicability of a regressor. While linear regression does not require a lot of data, the MLP need interpolation of the time-series so that no missing values exist in the training data. This also raises the question if there exist better neural network architectures to process time-series.

Table 6.1: best-performing prediction methods

c	% of time-series
auto ARIMA	43.39
SVR linear	17.54
linear regression	14.53
SVR polynomial	7.74
SVR rbf	7.23
MLP	3.17

6.5 Conclusions & Outlook

This Chapter introduces an extensible tool for aggregation, visualization and analysis of indicators. To gauge the quality of indicators, a tailored metric is proposed. With time-series extrapolation being a common use-case, BestFit can automatically select the extrapolation-method that historically produces the best results for a given time-series, so that no expert-knowledge is needed.

For the representation and visualization of indicators and models, a web based tool is proposed that accesses a server that is also available to other tools. Presenting the dimensions of time and country in the user interface is done with a world-map and graphs. Table-views provide detailed information and an export-function into a CSV file allows the analyst to process the data locally. Directory services on the server keep track of methods for time-series processing, models and indicators. With the usage of integrators, new sources for indicators can be easily made accessible and use the benefits of caching,

updates and downloads on demand. To make the indicators and models more accessible, a text-bases search in addition to the categories provided can support a user. Also the semantically correct coloring of indicator-values would be beneficial, though a labeling of all indicators is required and not every source provides that. An extension of the operators that can be used on indicators would enhance the capabilities of an analyst to create her own models. Storing these models and sharing them with other users of the tool is also future work. The same goes for setting thresholds on indicators and models for automatic notifications and automatically generated reports about the current state.

A new measurement for the quality of a macro-indicator is proposed. The metric consists of three different parts that can be weighted according to individual needs and is normalized. Experiments with indicators collected by the World Bank show, that few of these indicators satisfy all of the three criteria. Thus the score is useful in identifying indicators that actually containing sufficient time-series for sophisticated models and for a more accurate extrapolation.

For time-series extrapolation, BestFit is proposed. This method applies a set of regressors to a shortened time-series and selects the regressor that produces the best fit for extrapolation. The Manhattan distance is used to evaluate the performance of these regressors. Different time-series within an indicator might react differently to a extrapolation method, so the fine granularity is sensible. Experiments show, that there is a great variance in best-ranked regressor, even within an indicator. Due to the properties of the dataset used, a qualitative comparison of accuracy of predictions is not possible. Categorical and percentage scales are treated the same as continuous scales. This makes a direct comparison between two indicators not feasible. For future work, an annotated dataset, that includes the scale of an indicator could change that. Also, different distance measures can be evaluated as the current setup does not weight the predictions over time. The result is, that predictions that are not to far in the future weight the same as predictions very far in the future. A weighting scheme could resemble the preference of the analyst. The use of Deep Learning for indicator extrapolation is another interesting field that needs to be explored.

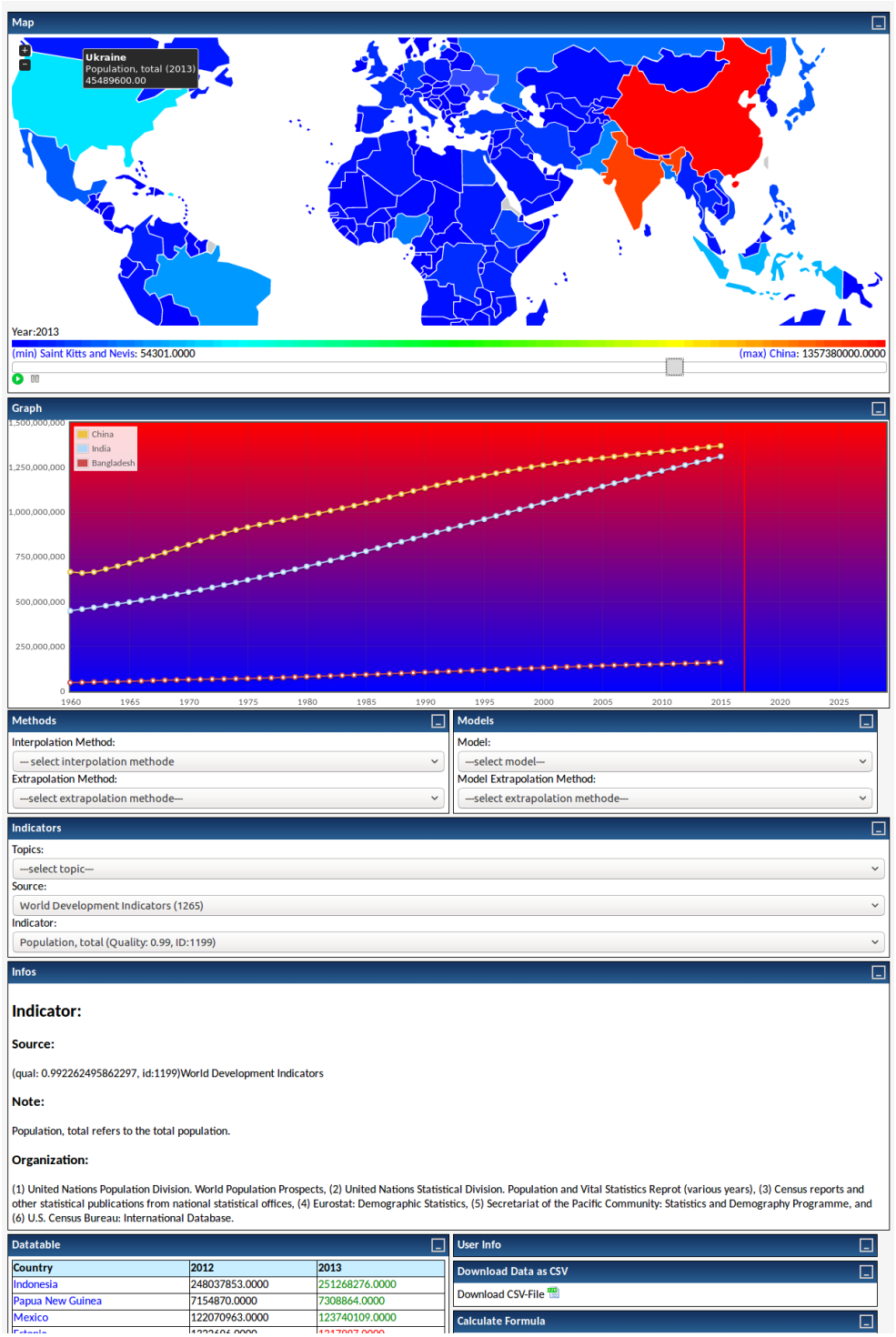

Figure 6.1: Screen shot of the client showing the indicator “population total”

Chapter 7

Human Activity Recognition

The previous chapters proposed methods that can be used to perform environmental scanning, an activity necessary to collect input for Strategic Management methods. At first glance, this Chapter is discussing something completely different: Human Activity Recognition (HAR) from accelerometer data. To put the subject of this Chapter into the context of the previously presented work, the application must be abstracted from the methods used. Chapters 4 and 6 use time-series to represent developments of metrics (e.g. trends, volume of documents, GDP) over time. In Chapter 4, time-series are presented "as is" so that an analyst can gain further insights from the data. In Chapter 6 simple time-series processing is performed to interpolate missing values and to extrapolate into the future, though the presentation to the analyst remains as time-series. Chapter 5 discusses the classification-problem of documents into a predefined system in detail. The predefined system allows a swift categorization of documents and narrows down the potential candidates that must be analyzed in more detail.

As a summary, combining the observations from the previous chapters: a classification of the provided time-series data into predefined categories can help an analyst to distinct time-series that are relevant to his subject from those that are not relevant.

Similar to the field of NLP, current developments of Deep Learning techniques change the field of time-series classification (see Section 7.2 for more details). The performance of approaches using feature-engineering to extract, often domain-specific, feature-vectors from time-series is challenged by feature-learning approaches. Feature-learning uses filters in combination with neural networks to learn the relevance of a multitude of features to the given problem.

The research-question is: how do feature-learning approaches compare to the traditional feature-engineering. This question can be dissected into the following questions:

- How much is feature-learning dependent on a specific dataset?
- How much is feature-engineering dependent on a specific dataset?
- How do feature-learning approaches and feature-engineering approaches compare in performance

- what are the strengths and weaknesses of each approach

For a concise evaluation of these questions, a large dataset with solid ground-truth data and related-work is required. With 216 countries world wide, the use of that dataset for supervised learning approaches is limited. While the text data, especially the trend detection information can be used to generate a huge amount of time dependent information, the nature of Natural Language Processing (NLP) tasks makes the generation of a ground truth difficult if not outright impossible. To circumvent the issue of combining a dataset with fuzzy ground truth with a novel supervised time series processing method, datasets are used which provide a high quality ground truth and have been evaluated with multiple approaches. The emphasis on high quality ground truth stems from the fact that especially in the supervised machine learning environment great progress is made by methods that utilize various architectures of neural networks with multiple hidden layers, enhanced optimizers and new activation function types as much as new neuron types.

While this Chapter focuses on the multi-class task of HAR, the transfer of the methods towards the time-series generated by methods discussed in earlier Chapters is straight forward. The generating of ground-truth from that data would be a task for the analyst as it is certainly task-dependent.

This work is to be published and is created in a joint effort with Matthias Schubert and the supervision of Hans-Peter Kriegel.

7.1 Introduction

Smart phones and other mobile devices have become permanent companions of our daily lives. Recently, these devices are often complemented with further smart devices such as smart watches and fitness trackers. All of these devices contain various type of sensors such as GPS localization, temperature sensors, heart rate detectors, cameras and last but not least accelerometers. The combination of these sensors allows to collect valuable information about daily activities which are tightly connected to various health questions. Questions like, do we perform a sufficient amount of movement or how much energy do we burn during the day, can help us to stay fit and healthy. Furthermore, monitoring daily activities and being able to pinpoint the type of activity is often required to assist the treatment and recovery of various illnesses and maladies. For example, a person having a high heart rate during running is considered normal. However, an increased heart without any demanding form of activity might reveal a critical condition. The task of classifying activities is also referred to as Human Activity Recognition (HAR).

Though various sensors offer useful information on this questions, accelerometer data are the most suited information source when it comes to distinguishing various types of activity and movement. An important advantage of this data source is the relative robustness and the rich data allowing to distinguish multiple activities. Accelerometers work indoors as well as outside and, as will be shown in this work, can be used to distinguish a large variety of activities like running, walking, cycling, etc.. Another benefit of accelerometers is the low cost and their subsequently widespread deployment in smart devices.

Technically, an accelerometer records a three dimensional data stream with a fixed frequency. This data stream can be instantly transferred to a smart phone or even a remote server to provide interactive analysis. If no up-link is available the data is stored on the device and can than be analyzed off-line. Let us note that especially for patient monitoring it is usually important that the employed activity detection method works on the continuous data stream in real-time.

To implement such a method the incoming data stream is partitioned by a windowing approach and a feature extraction method transforms each temporal window into a set of descriptive features. The performance of the classification method strongly depends on the right set of features, sensor position and data set.Thus, various previous works [91] introduce suitable feature spaces for activity detection and then, use standard classifiers such as Support Vector Machines (SVMs) and random forests to predict the current activity. A major drawback of the feature-engineering approach is that the method strongly depends on the suitability of the employed feature space. More recent approaches still use the windowing but rely on feature-learning and deep neural networks (DNNs) for the recognition task. These new approaches claim the benefit of using raw data and thus requiring little preprocessing.

These feature-learning or representation-learning approaches outperform the classic feature-engineering on various benchmarks. But the literature describing the networks and their training often lacks a broad evaluation and sometimes documentation of the many hyper parameters so that reproducibility is limited. Among the interesting hyper-parameters are not only the architectural elements of the neural network e.g. number of layers, type of layers, activation-function and initialization method, but also the choice of the windowing, regularization and the normalization parameters that lead to the published result. In this work, these parameters are evaluated and documented. Another aspect of this work is the identification of a sensor positions which yields reliable results and a closer look at the activity classes. Confusion matrices are used to determine classes that are particularly challenging for the classifiers. To ensure the generality of the created HAR models the train- and test-datasets are separated by individuals thus allowing the evaluation of the model on persons whose movements it has never observed before.

In general, there are various ways to train neural networks on temporal data, though all use windowing of the input. The first is using a feed forward architecture, neglecting the temporal dimension of a sequence. The second method to cover time series is to use recurrent architectures like LSTM [79] or GRU [45]. Optional to both approaches is the use of convolutional layers which boosts the performance of various computer vision tasks, as features derived from the original data are learned from the convolutions rather than engineered; thus often understood as feature-learning, though strictly speaking, convolutions are not necessary for that. Recently, the combination of these paradigms has been shown to learn better models. The following Sections examine the question whether a combined architecture yields better results than a network architecture that is purely based on one of the paradigms. Furthermore, the performance of methods integrating feature-learning and the classic approach based on feature engineering is evaluated. Two different datasets are used, to gauge the dataset-dependency of the paradigms and methods.

To conclude, the contributions of this Chapter are:

- A hyper parameter evaluation of various network architectures.
- A feature learning based approach to activity detection based on several neural network architectures.
- An experimental evaluation of two different datasets comprising X persons and Y different activities.
- An evaluation of state-of-the-art reference methods following the feature-engineering and feature-learning paradigms.

The rest of this Chapter is structured as follows: Section 7.2 discusses related work in the area of HAR. In Section 7.3, the problem is formalized and in Section 7.4, several possible network architectures are described for building a suitable prediction model. Section 7.4 also describes the used data sets, the different experimental settings and compares the feature-learning methods to state-of-the-art approaches. The Chapter concludes with a summary and an outlook to future research directions.

7.2 Related Work

This Section discusses current approaches to HAR, and more general approaches that generate feature-spaces from time-series. Time-series can be characterized by frequency and amplitude. The frequency describes the oscillations during a defined time-interval, usually a second. 1 Hertz (Hz) is one oscillation per second. Amplitude, or intensity, describes the energy of the oscillation. Methods like Fourier-transformation or Wavelet-transformation can be used to decompose a time-series into frequency-ranges with their associated intensities. These create rich feature spaces.

For a successful classification of time-series, a similarity-measure is required. Time-series are assumed to consist of equidistant observations without missing values. A naive comparison of the time-series A and B with the length N by using $a \in A, b \in B$: $\sum_{i=0}^{N} |a_i - b_i|$ requires that the time-dimension in both time-series align perfectly. For time-series where an offset between observations is possible, methods like Dynamic Time Warping (DTW) [130] find an optimal match, by non-linear transformation of the time-dimension. The distance as calculated by DTW and the path that corresponds to the warping can be used as feature space for classification. The works of [82] addresses the issue of phase-difference in time-series as DTW does not account for that. Increasing classification performance by providing a better similarity-metric is one of the state goals of this work.

In the field of HAR, two different approaches are to be distinguished: the classic approach using heavy preprocessing of the data and feature-engineering where various metrics are derived from the data to create features which allow a distinction of the activities and the more recent approach using feature-learning in combination with Deep Learning where

neural networks with more than one hidden layer operate on raw or sparingly preprocessed data. The classic methods differ in features, classifier and parameters e.g. window size. Methods like DTW and Fourier transformation are likely to be used by these approaches. Methods using feature-learning distinguish themselves mainly by the type of network architecture, the input dimensions and the training-mode used which is particular relevant to neural networks. Common to all approaches is the use of sliding windows of a input time-series. The window length is often regarded as an important parameter, as the assumption is that activities with a cyclic pattern need to fit into a single window to allow recognition.

The comparison of the performance of published methods is difficult as there are not only different metrics used documenting the quality of a method, but also different datasets. One publication often uses a single, specific dataset with a unique set of activities. While the classes of the datasets are mostly comprised of locomotion, some datasets regard transitional states (e.g. from walking to lying) as an additional class, which is reasonable assuming large time frames of observation. E.g. windows that can wholly contain such an action. Since its inception in the year 2010, the Opportunity Challenge Dataset (OPP) [121] is established as quasi standard in the scientific community for the evaluation of HAR methods. The properties of the datasets used in this work, in particular OPP, are discussed in depth in Section 7.4.1.

[91] provides an excellent overview of the field of HAR based on accelerometer data, separated into on-line- and off-line-methods and is reporting performance of each method as accuracy. Unfortunately, the required window length of each method is not part of this survey, so we elaborate on the most accurate off-line activity recognition system. The authors demonstrate the importance of the dataset and the influence of the performance measure of a method by citing [87] who observed 95.6% accuracy in the lab versus 66% accuracy on real live data.

15 activity classes are distinguished in this work [86]. 8 of them are transition states and 7 activity classes according to our definition. The data is obtained from 6 individuals, wearing the sensor on the chest. Artificial neural networks (NN) are used as classifier. A window size of 3.2 seconds is used. 97.9% accuracy is reached by using an hierarchical approach, limiting the actual classification task to 4 classes at max on the lowest hierarchy. The features used are: autoregressive coefficients, signal-magnitude area and tilt angle in the second stage of classification and mean, standard deviation, spectral entropy, and correlation in the first stage. The placement of the sensor allows to draw conclusions from the tilt angle and reduces noise from minor movements. Type and training methods of the NN are not described in the paper nor which measures are applied against overfitting.

Using a single accelerometer, the method in [121] reaches an accuracy of up to 99%, distinguishing 8 activities and using a window size of 5.12 seconds. Classification accuracy degrades to 73% in scenarios, where the training- and validation-data is not from the same person. Feature extraction generates 12 features per window: mean, standard deviation, energy and correlation for each axis. At the best accuracy, the training is performed on the same persons as the validation. Classifiers evaluated are different trees, Naive Bayes, kNN and SVM. The best performance is archived using plurality voting.

The data for this paper [89] is collected by smart phone sensors worn in the front pants leg pocket by the authors. The dataset consists of 6 classes. From 10 second windows without overlap 43 features are generated with average, standard deviation, average absolute difference, average resultant acceleration, time between peaks and binned distribution. A multilayer perceptron (MLP) reaches 91.7% accuracy though the training method is not described in detail.

This recent publication [110] considers 4 locomotion classes: stand, walk, sit and lie. The dataset is created by the authors of the paper and only accelerometers are used. Performance is reported by accuracy numbers with a maximum accuracy of 91.98%. The data of a sliding window with the length of 4.27 seconds and with 50% overlap is used to create 12 features. Mean, energy, correlation between axis and frequency domain entropy was used. In this work, random forest (see Section 3.2.3) as classifier achieved the best results. We compare to this method in our experimental evaluation using the $F1_{weighted}$ score.

The dataset TUM that is used in the experiments is collected by researchers from the Technical University Munich School of Education in cooperation with Sendsor GmbH for Actinulin. Actinulin is developed for medical purposes [35]. Naive Bayes is used as a classifier. Applying the unmodified method described in [35] yields 38% accuracy on that specific dataset.

Another aspect to consider besides the pure classification accuracy is the identification of individuals based on their gait. Work on that topic [55, 101] suggest that it is possible to distinguish people based on the data collected with accelerometers. The implications for activity classification are that a model is at risk of fitting onto features inherent to the individuals instead of generic characteristics. To avoid this issue, the test- and validation-dataset are separated by individuals so that no characteristics from the individuals movements can affect the classification performance. In general, this approach to generating the datasets decreases the performance.

Recurrent Neural Networks (RNN) show excellent performance when applied to NLP tasks, e.g. speech recognition [72]. The similarities of the speech-recognition-challenge to the HAR task and the excellent performance of RNNS motivates several works to apply RNNs to HAR.

The authors of this work [112] propose the use of deep RNNs for HAR naming their particular architecture DeepConvLSTM. They evaluate various hyper parameter of RNNs on the OPP dataset and on the lesser known Skoda dataset. Performance is reported by $F1_{weighted}$ score with a peak performance of 0.9157 on OPP using 113 features including other sensors besides accelerometers and distinguishing 18 gestures. Four locomotion-classes are classified with an $F1_{weighted}$ score of 0.864 on the same feature set. Using 15 channels only from accelerometers, the authors report an $F1_{weighted}$ score of 0.689 which we will use as another reference in our experimental evaluation. This reported performance is reached with a relaxation of the early stopping criterion. Linear interpolation is used to fill missing sensor data; channel wise normalization of the data to [0, 1] is done. A sliding window with a length of 24 samples (0.8 seconds/window) is applied to segment the data.

In this paper [58], the focus is on efficiency of the classification process, as mobile devices

do not yet offer the same computing power as stationary systems. The paper distinguishes three configurations: C1 only one individual is considered for training, test and validation, C2: The data from all individuals is mixed for training, test and validation and C3 where an individual is not used in training and validation but for testing. As performance measure the $F1_{weighted}$ score with $\beta = 1$ is used. The OPP dataset, PAMAP2 and a custom dataset are used for experimental evaluation. The proposed BLSTM-RNN architecture reaches a $F1_{weighted}$ score of 0.78 on C2 and C3. For C1 the score is 0.74 as little data is available.

Recently [74] did an evaluation of various neural net architectures on accelerometer data. The authors use the popular datasets from the PAMAP2 dataset, Daphnet Gait Dataset (DG) and the OPP dataset which we also use in our evaluation. As performance metric the authors use the $F1_{mean}$ and $F1_{weighted}$ score to compare their results to the state of the art. Evaluated network architectures include deep neural networks (DNN) utilizing PReLU neurons, convolutional neural networks (CNN) and three types of recurrent neural networks (RNN) using variations of the LSTM neuron. The authors observe significant variance between training runs using the absolute delta from the median score. In the neural network architecture comparison, the best architecture depends on the dataset. On PAMAP2 the CNN outperforms all other architectures while on the DG and OPP dataset RNNs dominate.

To conclude, though many of the approaches above already achieve impressive accuracies, experiments are often restricted to individual datasets. The required window-size is often rather long, i.e. several seconds. Furthermore, the datasets for training and testing are containing samples from all monitored persons, limiting the applicability for commercial applications.

7.3 Method

This Section describes the HAR task and presents different network architectures which are evaluated.

The objective is to create a generic classifier for a time-series x, containing accelerometer data up to the moment t of the length $w = |x|$ and the dimensionality 3 into one of $K = \{k_1, ..., k_i\}$ classes where each class represents an activity. With datasets where there multiple activities are contained in the same time-series (e.g. OPP), the class of x is determined by the class of the last measurement. The classifier $C(x_t)$ returns the probabilities for each class $C(x_t) = max(p_{x_t}) = max((p_{k_1}, ..., p_{k_i}))$. The class with the highest probability is assumed the class of x.

As the length of the time-series w describes how many observations are available for an classification, it is beneficial for a fast classification to keep w as small as possible. Together with methods relying on engineered features, various NN architectures M are evaluated in their ability to approximate the classification function C by adapting the weights associated with each neuron. The output of the kth neuron is defined as: $y_k = \phi(\sum_{j=0}^{m} w_{kj} x_j)$ with w being the weights and x being the inputs. ϕ is the activation function. As activation function Rectified Linear Units (ReLU) [68] with $\phi(x) = max(0, x)$ are used as this

function is similar to biologic behavior and delivers better overall performance compared to activation functions like sigmoid or tanh [139]. Because of the randomization of RReLU and the α-parameter of leaky PReLU, we choose ReLU.The special requirements in dealing with sequences are considered by RNNs. These have the ability to take the dimensions of sequences into account, by using an additional, recurrent connection inside a layer. State-of-the-art neurons include the Long Short Term Memory Cell (LSTM) [79] and the Gated Recurrent Unit (GRU) [45]. We evaluate both units as there is no clear evidence in the literature [51] which recurrent unit is better.

Convolutions are used to generate feature maps from usually multiple elements of the input data by an activation function. The convolutional kernels, often referred to as kernels, can be visualized [140] which can help to understand what kind of features the network has learned. We examine if convolutions can increase the classification accuracy using tanh as activation function and a window of 4 samples. Convolutions can be applied in various dimensions. As time-series are the focus of this work, 1-dimensional convolutions over the time-dimension are evaluated. As other works have already shown [112], 2-dimensional convolutions, covering the channel- and the time-dimension provided an increase in classification performance.

Meaningful combinations of layer types are restricted. GRU and LSTM are competitors so no combinations of these neuron types are considered. Convolutions generate features which are trainable, so it is used as input layer. The use of a non-recurrent layer after an recurrent one is suggested by [122] so this specific configuration is explored. A non-recurrent layer before a recurrent layer deprives the recurrent layer of the time-dimension, so this configuration holds no benefits.

Because a supervised learning method is used, the loss-function l, optimizer o and back-propagation are used to reach a local minimum at which the weights contained in the NN represent a model consisting of learned features for the classification task. Mini batches decrease the duration of training and can lead to a more generalizable model as the gradient of descent is calculated over multiple training examples simultaneously. As optimizer o, we use RMSprop [136], the loss-function is categorical cross-entropy. The classification is done on a batch of the size j denoted as $B_t = (x_{t-(j*o)}, ..., x_t)$. B_t is created by a sliding window with the length w and the offset o over the data. For NNs we keep the offset to 50% of the window size, which is consistent with the literature.

To determine the network architecture M, we examine the following dimensions: number of layers $|L|$, neuron type per layer $|c_{l_i}|$ and number of neurons per layer $d_l = c_{l_i}$ representing the layers output dimensionality in regards of the reached accuracy performance. We omit the last layer in our evaluation as it is a softmax layer of cardinality $|K|$ and remains fixed. This layer creates a categorical probability distribution.

Best practice, theory and empirical evidence suggest that deep architectures with a larger number of layers perform better than shallow architectures with a high number of neurons [27]. To avoid exponential complexity with the various combinations, we assume that the evaluated dimensions are independent of each other. While this assumption does not hold true (e.g. a layer of LSTM neurons requires less neurons to perform as good as a sigmoid neuron, because more parameters are available per LSTM neuron), we argue

that the relation of the evaluation w.r.t. the dimension in question holds true, even if the parameters of the other dimensions change.

This allows to evaluate e.g. the influence of $|c_{l_i}|$ independently of $|L|$. For the following experiments, training-time per epoch and the number of epochs are disregarded. With early stopping [66, 67], the moment is determined when a neural net starts to overfit on the training data and the training is stopped. Disregarding further optimization of hyper parameters like learning rate or momentum, this guarantees that the currently evaluated architecture reaches its maximum learning capacity within a predefined number of epochs (6) during which the validation loss is allowed to increase.

$$\text{accuracy}(y, \hat{y}) = \frac{1}{n_{\text{samples}}} \sum_{i=0}^{n_{\text{samples}}-1} 1(\hat{y}_i = y_i) \tag{7.1}$$

$$\text{precision} = \frac{1}{\sum_{l \in L} |\hat{y}_l|} \sum_{l \in L} |\hat{y}_l| \, P(y_l, \hat{y}_l) \tag{7.2}$$

$$\text{recall} = \frac{1}{\sum_{l \in L} |\hat{y}_l|} \sum_{l \in L} |\hat{y}_l| \, R(y_l, \hat{y}_l) \tag{7.3}$$

$$F1_{weighted} = \frac{2}{|K|} * \frac{precision * recall}{precision + recall} \tag{7.4}$$

Multiple metrics are used for performance-evaluation. The normalized accuracy $a \in [0, 1]$ (see Equation 7.1) as the most commonly used metric in the literature where the best achievable score is 1.0 and the worst 0.0. Using the accuracy can be misleading though, as class imbalance is not captured. A more refined metric is $F1_{weighted}$ (see Equation 7.4), which does take label imbalance into account. It is assumed that precision (see Equation 7.2) and recall (see Equation 7.3) are equally important and use $\beta = 1$ which results in Equation 7.4.

7.4 Experiments

This Section, and in particular Section 7.4.1 discusses in detail the two datasets used in the experiments. Table 7.1 provides a overview of the experiments performed and lists the main-features of each dataset. Experiments to determine a good location for the sensor (see Section 7.18) are only performed on the TUM dataset.

While cross-validation is employed, each neural network is trained only once. Because of this, variance due to random weight initialization and weight noise is unknown.

The basic configuration of a network architecture used for the TUM dataset consists of 512 LSTM-neurons in one layer, a windows size $w_{TUM} = 64$ 0.64 seconds, overlap of 32 samples (50%) and batch size $|B_t| = 256$. The initial learning rate is set to 10^{-5}. If not declare otherwise the data obtained from the sensor worn on the right wrist is used.

Table 7.1: dataset properties and performed experiments

experiment / dataset	TUM	OPP
individuals	min. 88	3
activity classes	8	4
accelerometers used	1	12
accelerometer sampling rate	100 Hz	30 Hz
folds cross-validation	4	3
NN architecture evaluation	yes	yes
window size evaluation	yes	yes
accelerometer position evaluation	yes	no
using multiple sensors simultaneously	no	yes

Similar parameters are used for the OPP dataset. 512 LSTM-neurons and a window length of 16 samples (relating to 0.53 seconds) and an overlap of 8 samples (50%) are the baseline parameters and correspond to those frequently used in the literature.

For each architecture, the mean $a_{average}$, median a_{median}, minimum a_{min}, maximum a_{max} and standard deviation a_{std} for each performance measure over the folds is evaluated.

7.4.1 Datasets

As mentioned earlier, two different datasets are used to evaluate the performance of the methods: TUM and OPP. All datasets contain continuous, labeled time series during which activities $k \in K$ are performed by various individuals. In both datasets each person is simultaneously wearing multiple accelerometers at different positions $P = \{p_1, ..., p_i\}$. While the focus of this work includes the identification of the best position to place the accelerometer, different combinations of sensors are not considered. To fit a model specific to a location p, a dataset D_p is created, containing only data from the sensors worn at p and models are trained independently for each position. As it is shown that a walking individual can reliably be identified by accelerometer data [55, 101], the data $D_p = (D_{p_t}, D_{p_v})$ is separated into the training-dataset D_{p_t} and D_{p_v} validation-dataset so that the data from an individual partaking in an experiment is disjunct in the sets. This avoids that a model is using individual-specific cues for classification.

To model the possibility that k_x can change with every instance of x, the sequences in D are randomly shuffled for training and validation.

The used dataset TUM contains accelerometer data from 88 to 143 children depending on activity. The participants are girls and boys around the age of 11 and 14 years old. The eight performed activities $k \in K_{TUM}$ are: cycling, lying, running, sitting, stairs down, stairs up, standing and walking. Each activity is a few minutes long and presented as a single file per person. For TUM each test subject is wearing 4 sensors. The positions $p \in P$ are the wrist of the right hand (H), on the left ankle (L), on the right ankle (R) and in the pocket (T). The sensors measure acceleration in 3 axis (x, y, z) up to 2g. Intensities above

Table 7.2: *TUM* activity class distribution

activity / sensor	H	L	R	T
cycling	21%	23%	22%	22%
lying	9%	10%	10%	10%
running	19%	18%	19%	18%
sitting	9%	9%	9%	9%
stairs down	5%	3%	3%	4%
stairs up	6%	4%	4%	5%
standing	10%	10%	10%	10%
walking	21%	23%	23%	22%

Table 7.3: *OPP* activity class distribution

activity	
null	16%
standing	41%
walking	23%
sitting	17%
lying	3%

2g are cut off. The sensor orientation remains the same between test-subjects. Because of gravity, 1g downwards is constantly present. The sampling rate of the sensors in *TUM* is 100 Hz.

The distribution of the activities in *TUM* per sensor is described in Table 7.2.

The Opportunity dataset [121] differs from the *TUM* dataset in multiple aspects. First, it is using 30 Hz sensors instead of 100 Hz. Second, not only accelerometers are included in the dataset but also Inertial Measuring Units (IMU) and 3D localization information. The 4 participants (Subject 1-4) in the *OPP* dataset wear 12 different accelerometers, the locations are shown in Figure 7.1. Of the 4 individuals, S4 is selected by the authors of the dataset to assess the robustness to noise so the data contains added rotational noise which makes it unfit for an objective comparison e.g. during cross-validation. Because of the separation of the training and test-set by individuals, we perform a 3 fold cross-validation on the *OPP* dataset, instead of the 4 fold cross-validation on *TUM*. While the number of participants is low, the amount of data is much greater than in the *TUM* dataset as the recordings cover a longer timespan. Also, the data is not segmented into various files, but each sample from the sensors is labeled individually. For our purposes the information of the non-accelerometer sensors and the IMUs is omitted. Another important difference is the completeness of the data. Where the *TUM* dataset has no missing signals, the *OPP* dataset shows lost measurements for all sensors at one time or the other which we consider when selecting the features for the classification. The most heavily affected accelerometers are LH and RH (see Figure 7.1) with LH missing 76 minutes of data and RH missing 158

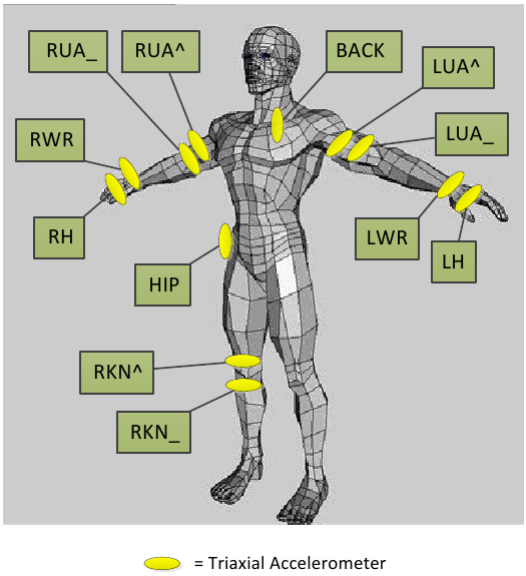

Figure 7.1: accelerometer placement for OPP dataset

Table 7.4: $F1_{weighted}$ for various $|c_{l_1}|$ on TUM dataset

$\|c_{l_1}\|$	mean	median	std	max	min	gain (%)
128	0.591	0.599	0.019	0.607	0.560	0
256	0.598	0.601	0.026	0.629	0.562	1.18
512	0.619	0.628	0.019	0.634	0.587	4.74
1024	0.628	0.641	0.027	0.650	0.581	6.26
2048	**0.637**	0.641	0.042	0.690	0.577	7.78

minutes of data. To mitigate the effects of the lost measurements, linear interpolation is applied. The OPP dataset distinguishes the classes into four locomotions (lie, stand, walk sit see Table 7.3 for class distribution) and 16 gestures. Optionally locomotion and gestures contain a *null* class that is the label for samples that to not belong to any other category.

7.4.2 Number of Neurons

Networks consisting of a single layer of LSTM neurons with the following numbers of neurons: $|c_{l_1}| \in (128, 256, 512, 1024, 2048)$ are tested on both datasets. The results of these experiments are shown in Tables 7.4 and 7.5.

It can be observed, that for both datasets the best results are reached with the highest number of neurons. While the TUM dataset exhibits a steady increase of performance with an increasing number of neurons, the OPP dataset, as described in Tables 7.5, requires a deeper analysis, as the performance initially decreases with more neurons. A general

Table 7.5: $F1_{weighted}$ for various $|c_{l_1}|$ on OPP dataset

$\|c_{l_1}\|$	mean	median	std	max	min	gain (%)
128	0.730	0.787	0.082	0.788	0.615	0
256	0.673	0.800	0.186	0.808	0.410	-7.81
512	0.703	0.815	0.159	0.816	0.478	-3.70
1024	0.786	0.845	0.111	0.882	0.630	7.67
2048	**0.836**	0.886	0.122	0.954	0.668	14.52

property of the OPP dataset is, that the locomotion of one of the three individuals is very different from the rest. Thus the standard deviation on the TUM dataset is larger than on the OPP dataset despite having significantly less individuals than TUM. Also the disparity in the min and max $F1_{weighted}$ can be explained in that way that the one of the three models in the cross-validation performed significantly worse than the rest. The limited capacity of the 128 neuron model might create a good general model, while higher capacity models were able to approximate 2 of the 3 individuals better but perform worse on the outlier individual. The random elements (e.g. weight initialization, etc.) also play a role. With the fourfold of neurons, this effect can not be observed and allows a model with a better min and max $F1_{weighted}$ performance.

Confusion matrices and learning curves of the OPP-dataset evaluation are available in Section A.1. The information presented there, refers to the best fold which shows a steady increase with the number of neurons. Figure 7.6 in Section 7.4.6 contains a confusion matrix of the worst fold (individual 0) for the default M.

7.4.3 Stacking

This experiment analyses how the performance of various neuron types varies if the number of layers is increased. All unrelated parameters are kept constant. So for the OPP dataset 512 neurons per layer are used and the windows size is fixed at 16 samples with 50% overlap. For the TUM dataset we also use 512 neurons per layer and 64 samples per window with 50% overlap. Examined are GRU, LSTM and simple ReLU neurons with a depth of up to 4 layers. The gain is calculated relative to the results of the same neuron type with one layer.

The experiment shows, that stacking does increase the performance, most visible on the OPP dataset, where regardless of neuron used the increase exceeds 10% going from one layer to four layers. In conclusion, in most cases more layers lead to a better classification performance. A notable exception is the LSTM on the TUM-dataset. Two LSTM-layers seem to be sufficient to model the data. More layers add complexity to the model that increases the time it takes for training and makes it more susceptible for overfitting. That might be the reason for the decreased performance. The variation might also come down to the initialization of the weights and the random nature of the learning-mechanism.

Table 7.6: $F1_{weighted}$ for various depth of A on OPP

neuron	depth	mean	median	std	max	min	gain (%)
ReLU	1	0.627	0.717	0.136	0.728	0.435	0
ReLU	2	0.664	0.739	0.130	0.772	0.481	5.90
ReLU	3	0.655	0.733	0.132	0.762	0.470	4.47
ReLU	4	0.708	0.770	0.098	0.785	0.569	12.92
GRU	1	0.647	0.740	0.143	0.756	0.445	0
GRU	2	0.701	0.812	0.158	0.812	0.478	8.25
GRU	3	0.726	0.822	0.169	0.868	0.489	12.21
GRU	4	0.768	0.885	0.180	0.906	0.513	18.70
LSTM	1	0.703	0.815	0.159	0.816	0.478	0
LSTM	2	0.746	0.858	0.192	0.905	0.475	6.12
LSTM	3	0.807	0.850	0.104	0.907	0.663	14.79
LSTM	4	**0.841**	0.907	0.121	0.944	0.671	19.63

7.4.4 Window Size

In the literature on HAR various window sizes are used. The consensus on feature-engineering approaches is, that sufficient samples are needed so that a full cycle of cyclic activities (e.g. walking) can be recognized. As a result, window-sizes vary between 1 to 6 seconds. DeepConvLSTM uses a window size of roughly 0.5 seconds. Due to this wide range, the following experiments evaluate time intervals from 0.13 to 2.13 seconds on the OPP dataset and 0.32 seconds to 2.56 seconds on the TUM dataset.

In this work the window size w is defined as the number of observations contained in the time-series used for classification. The number of observations that are obtained in a fixed time interval is dependent on the sensor used. For TUM the frequency is 100 samples per second and for OPP 30 observations per second. As a result of this, the window-size for both datasets is different when covering the same time interval.

For the OPP dataset $w \in 4, 8, 16, 32, 64$ is evaluated. Due to the higher frequency of the sensors used to create the TUM dataset, $w \in 32, 64, 128, 256$ is tested.

Confusion matrices, learning rate of the neural networks and additional metrics are presented in Section A.3 in detail. Details about the performance-evaluation of the feature-engineering approach with varying window-size can be found in Section A.2.

The experiments show, that larger windows do not necessarily lead to a better performance with feature-learning. One reason is that the whole sequence is used for the classification-task. Of particular interest is the result of OPP as observations of the previous activity are contained in a sequence that already has the label of the next activity as the label of the last observation determines the label. Assuming that activity transitions are something that occurs infrequently, smaller windows decrease the probability of a window containing a large amount of the previous activity, and it reduces the number of windows that contain observations with mixed labels.

Table 7.7: $F1_{weighted}$ for various depth of A on TUM

neuron	depth	mean	median	std	max	min	gain (%)
ReLU	1	0.550	0.547	0.018	0.577	0.528	0
ReLU	2	0.569	0.570	0.016	0.591	0.546	3.45
ReLU	3	0.577	0.581	0.016	0.596	0.552	4.91
ReLU	4	0.579	0.578	0.015	0.601	0.560	5.27
GRU	1	0.597	0.600	0.028	0.633	0.557	0
GRU	2	0.615	0.624	0.030	0.647	0.566	3.02
GRU	3	0.618	0.621	0.027	0.645	0.583	3.52
GRU	4	0.629	0.635	0.023	0.655	0.591	5.36
LSTM	1	0.619	0.628	0.019	0.634	0.587	0
LSTM	2	**0.634**	0.646	0.030	0.659	0.585	2.42
LSTM	3	0.627	0.643	0.036	0.657	0.566	1.29
LSTM	4	0.632	0.641	0.039	0.676	0.569	2.10

Table 7.8: $F1_{weighted}$ for various window sizes on OPP

w	seconds	mean	median	std	max	min
4	0.13	0.713	0.753	0.105	0.818	0.569
8	0.27	**0.721**	0.784	0.101	0.801	0.578
16	0.53	0.703	**0.815**	0.159	0.816	0.478
32	1.07	0.684	0.810	0.183	0.816	0.426
64	2.13	0.720	0.752	0.056	0.766	0.641

7.4.5 Mixed Network Architecture

In the experiments for the network architecture we evaluate various combinations of layer types $L = \{l, g, d, c\}$. LSTM is denoted as l, GRU denoted as g, ReLU denoted as d and convolutional layer denoted as c. The convolution works on a window of 4 samples in the time-dimension and is succeeded by a max-pooling layer with a stride of 2. The architectures $A = (l_1, ..., l_i)$ with $l \in L$ are abbreviated as a string where the left-most string is the input layer and the right most the layer just before a softmax layer which is used to retrieve the final classification result.

The applicable layers are initialized he-normal [96] to improve convergence speed. Early stopping and drop out is used to prevent overfitting [66,67]. The drop-out probability is 1% and the criteria for early stopping is the validation loss. If the validation loss is increasing for 6 consecutive times, it is assumed that further training is only benefiting the fit to the training data and the training is considered finished.

The results in the Tables 7.10 and 7.11 indicate, that 1D convolutions coupled with max-pooling do not enhance the capability of the architecture to find features relevant for classification but diminish the performance. Similar to the observations of [51], we see that the RNNs outperform the feed forward net using ReLUs. LSTM has a slight advantage

Table 7.9: $F1_{weighted}$ for various window sizes on TUM

w	seconds	mean	median	std	max	min
32	0.32	0.602	0.606	0.022	0.628	0.568
64	0.64	0.619	**0.628**	0.019	0.634	0.587
128	1.28	0.607	0.608	0.017	0.629	0.582
256	2.56	**0.620**	0.625	0.013	0.634	0.599

Table 7.10: $F1_{weighted}$ for various network architectures on OPP

M	mean	median	std	max	min
d	0.627	0.717	0.136	0.728	0.435
g	0.647	0.740	0.143	0.756	0.445
l	**0.703**	**0.815**	0.159	0.816	0.478
cg	0.465	0.466	0.048	0.524	0.406
cl	0.456	0.456	0.040	0.506	0.407
cld	0.484	0.493	0.053	0.543	0.415
clld	0.552	0.595	0.086	0.629	0.433
clll	0.623	0.670	0.074	0.681	0.518

over GRUs.

The *clld*-architecture proposed in [122] performs worse than a single layered RNN though the additional hyper-parameters introduced by the convolution (e.g. stride, kernel size and border behavior) would warrant a separate evaluation. Related work shows, that the use of 2D convolutions that cross channel-boundaries, can increase the classification performance.

The network architectures behave similar on both datasets. The relative ranking of architectures does not change by any metric.

7.4.6 Comparison of References

Two different reference-methods are used. One representing the traditional feature-engineering approach, and one using feature-learning and RNNs for classification. As described in Section 7.2 the feature-set and classifier proposed in [110] are used as the reference for the traditional state-of-the-art method. The same parameters with a windows size of $w_{OPP} = 16$ samples and an overlapp of 50% remain unchanged from the publication. Random forest is used as a classifier, as it does outperform all other evaluated methods. The performance achieved by this reference-method can be seen in Tables 7.12 and 7.13. Further results from the window size evaluation with the classic methods are presented in A.2.

Because the classic reference methods rely on engineered features the training time is negligible. All features are already predefined and the calculation of those does not challenge modern computers. The neural network in comparison does not have any previously generated expert knowledge of the data or any features so the training time is significant.

Table 7.11: $F1_{weighted}$ for various network architectures on TUM

M	mean	median	std	max	min
d	0.550	0.547	0.018	0.577	0.528
g	0.597	0.600	0.028	0.633	0.557
l	**0.619**	**0.628**	0.019	0.634	0.587
cg	0.540	0.540	0.025	0.575	0.504
cl	0.572	0.577	0.017	0.588	0.547
cld	0.581	0.581	0.010	0.594	0.566
clld	0.602	0.606	0.017	0.621	0.575
clll	0.612	0.610	0.023	0.643	0.587

For a fair comparison of training-time, the time needed to find the features had to be added on top of the training time of the run-time.

The reference for the state-of-the-art neural networks is provided by [112]. The source code is available on-line together with the trained weights and the preprocessing methods used. Due to the focus on accelerometers, the network-architecture must be adapted to the changed dimensionality and the network retrained. This is done by keeping the architecture exactly the same and by modifying only the input-dimensionality. To allow for a fair comparison, early stopping is introduced with the same patience of 6 as used in all other hyper parameter evaluations.

The results of the isolated hyper-parameter evaluations are used to create a network architecture, that combines the best performing instances of the following hyper-parameters: position of the sensor, number of layers, number of neurons, and type of layer. The classification-performance of this architecture is shown in Tables 7.16 and 7.17. Figure 7.3 shows the architecture used for OPP in detail. The results are ambiguous. While on the OPP dataset the NN is able to outperform the feature-engineering approach, it does lag behind on the TUM dataset. Compared to DeepConvLSTM, the new architecture clearly outperforms it on the TUM dataset. DeepConvLSTM on the other hand, performs significantly better on the OPP dataset.

Table 7.12: feature engineering approach on TUM dataset with $w = 64$

	mean	median	std	max	min
$F1_{micro}$	0.662	0.665	0.025	0.689	0.620
$F1_{macro}$	0.508	0.518	0.022	0.531	0.464
$F1_{weighted}$	0.645	0.647	0.020	0.666	0.608

Figure 7.4 indicates, that the classes “stairs up” and “stairs down” are particularly hard to discern. They also get mixed up with cycling. “lying” and “sitting” are also challenging. This might be due to the sensor being in a similar position. Using multiple sensors at once could address that issue but that is not in the scope of that work. Cyclic motions e.g. cycling, walking and running are well classified.

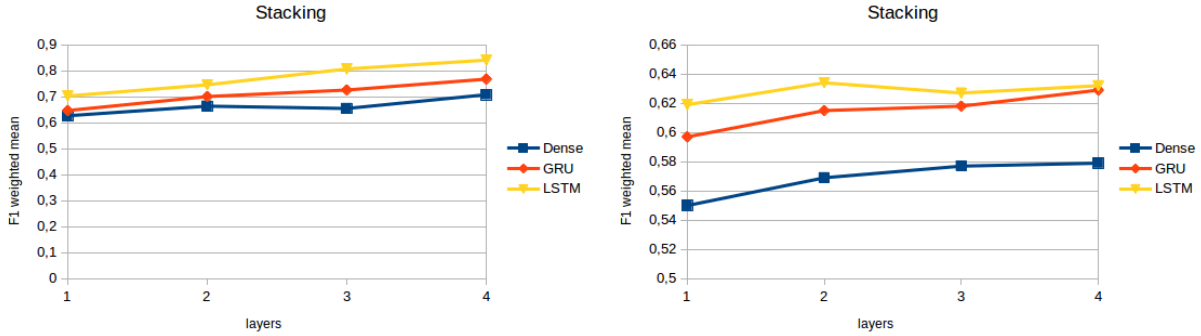

Figure 7.2: stacking OPP on the left, TUM on the right

Table 7.13: feature engineering approach on OPP dataset with $w = 16$

	mean	median	std	max	min
$F1_{micro}$	0.813	0.954	0.202	0.965	0.511
$F1_{macro}$	0.819	0.947	0.187	0.960	0.530
$F1_{weighted}$	0.802	0.954	0.218	0.964	0.468

Table 7.14: DeepConvLSTM architecture as described in [112] on TUM dataset with $w = 64$ and accelerometers only

	mean	median	std	max	min
$F1_{micro}$	0.623	0.631	0.030	0.651	0.580
$F1_{macro}$	0.469	0.478	0.027	0.497	0.424
$F1_{weighted}$	0.600	0.603	0.024	0.626	0.567

Table 7.15: DeepConvLSTM architecture as described in [112] on OPP dataset with $w = 16$ and accelerometers only

	mean	median	std	max	min
$F1_{micro}$	0.903	0.953	0.073	0.955	0.800
$F1_{macro}$	0.898	0.948	0.079	0.960	0.786
$F1_{weighted}$	0.897	0.953	0.081	0.955	0.783

Table 7.16: best performing architecture on TUM dataset with 4 LSTM layers and 2048 neurons/layer

	mean	median	std	max	min
$F1_{micro}$	0.656	0.665	0.055	0.716	0.578
$F1_{macro}$	0.511	0.533	0.065	0.575	0.404
$F1_{weighted}$	0.640	0.656	0.058	0.699	0.550

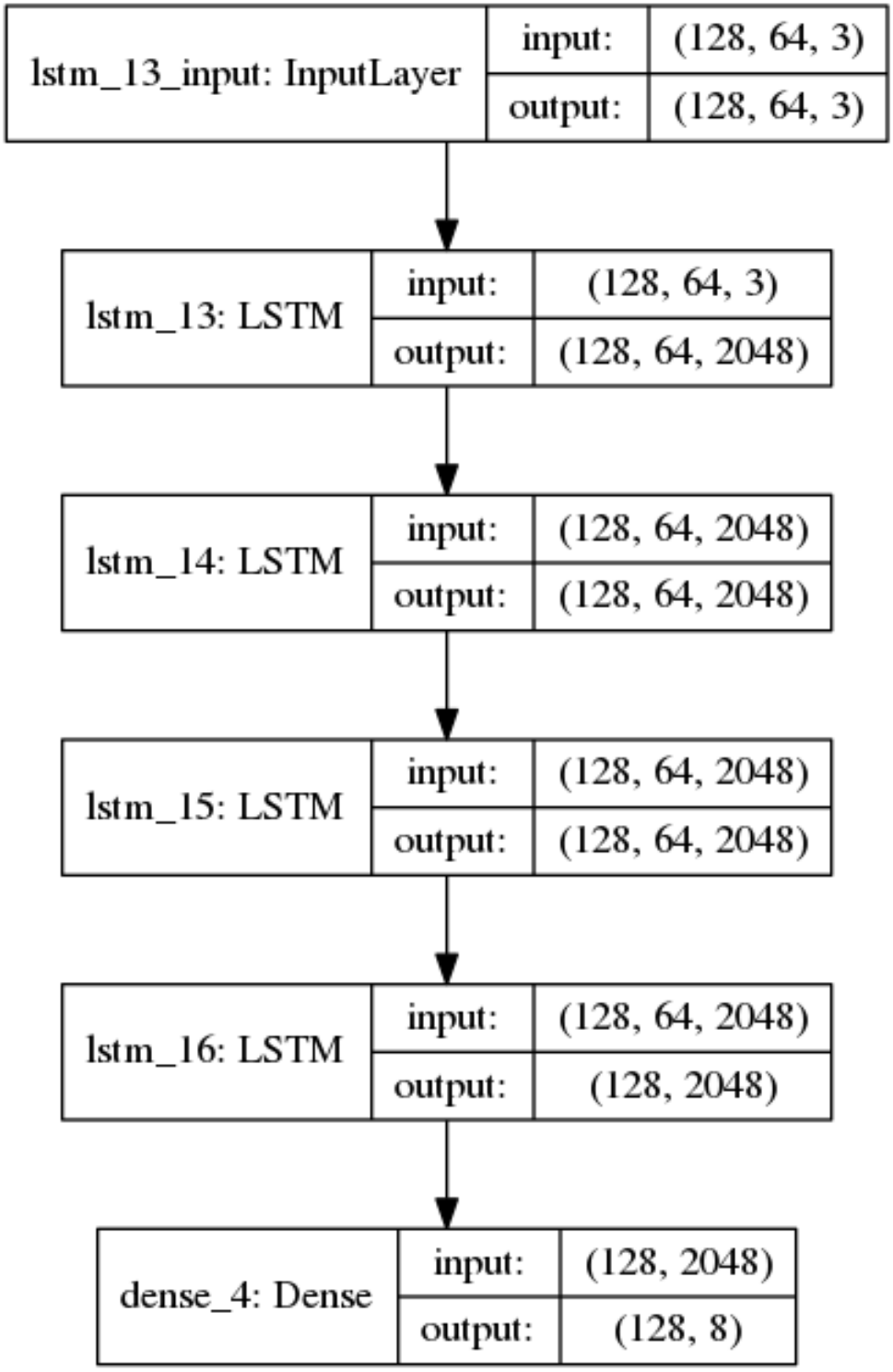

Figure 7.3: best performing architecture M on OPP

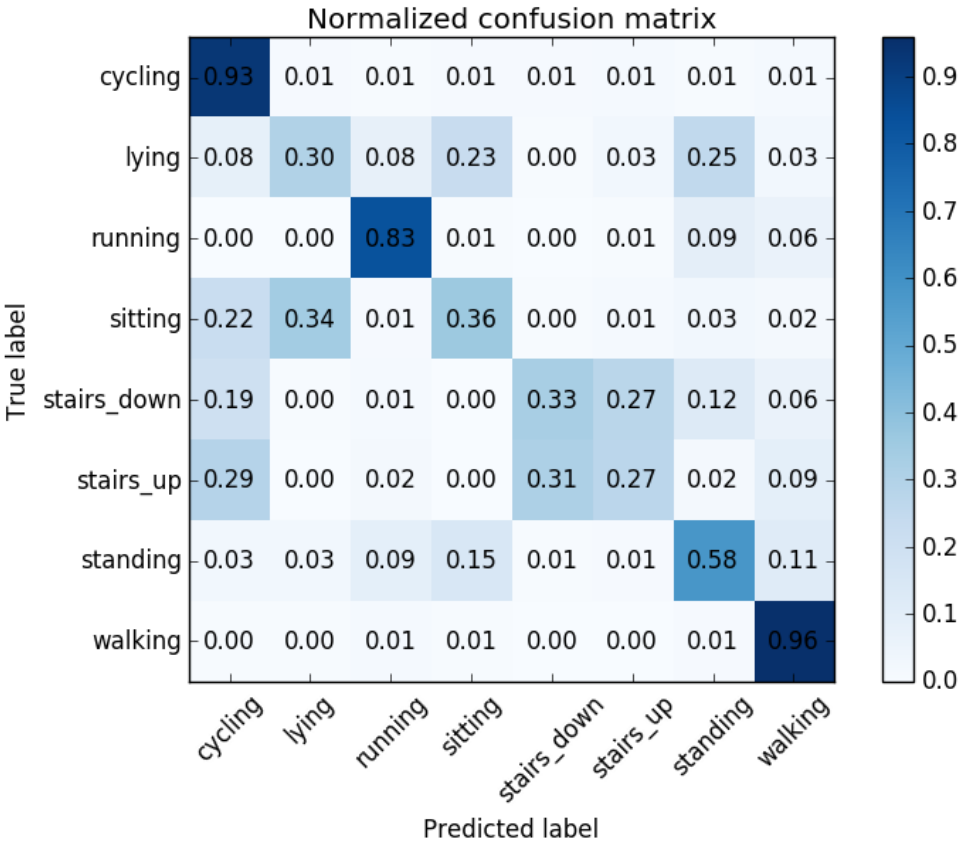

Figure 7.4: best performing architecture on TUM; confusion-matrix of the best fold

Table 7.17: best performing architecture on OPP dataset with 4 LSTM layers and 2048 neurons/layer

	mean	median	std	max	min
$F1_{micro}$	0.852	0.953	0.144	0.955	0.649
$F1_{macro}$	0.851	0.948	0.139	0.951	0.654
$F1_{weighted}$	0.843	0.953	0.157	0.956	0.621

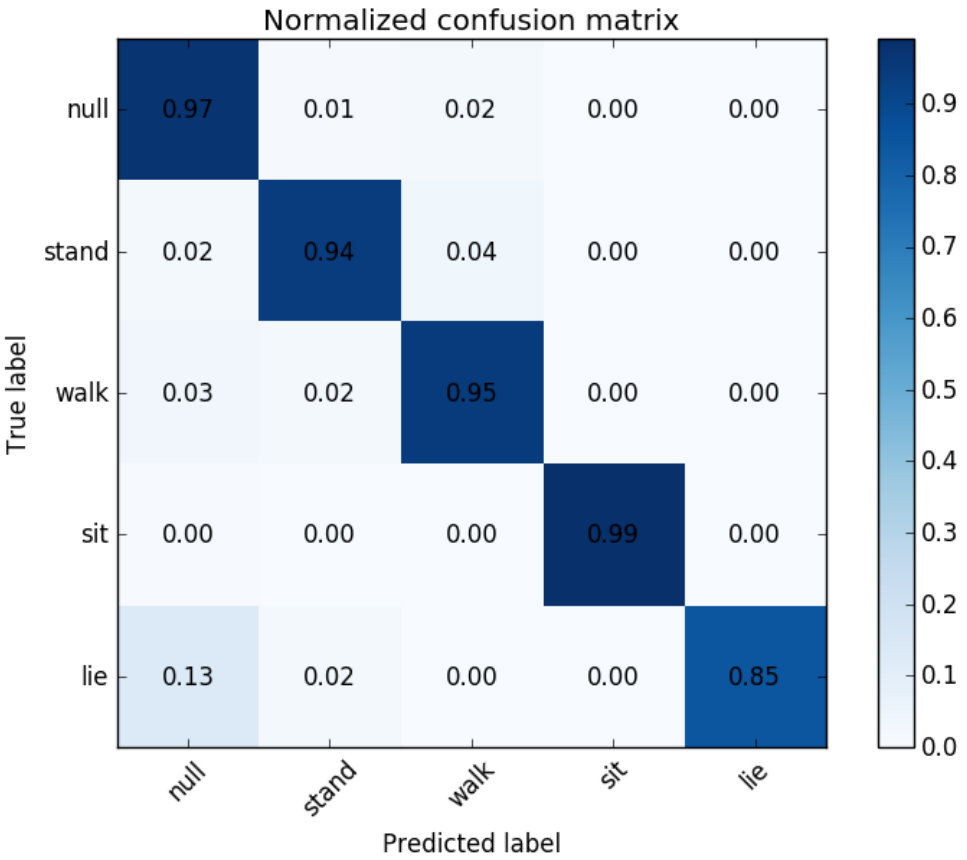

Figure 7.5: best performing architecture on OPP; confusion-matrix of the best fold

The confusion matrix shown in Figure 7.5 shows that only the "lie" class suffers from significant miss-classification with the generic "null" class. This confusion matrix is representing the best fold of three. The worst of three folds shown in Figure 7.6 is of interesting too, as it can partially explain what activities of the one individual in the OPP-dataset are problematic for the classifier. Especially the classes "walk" and "stand" are confused. The "null" class also has issues. Further experiments are needed to determine the root of that issue.

7.4.7 Sensor Position

This experiment evaluates how the sensor locations affects the performance of a fixed network architecture. In most studies where a recommendation for a sensor placement is made, the decision is based on the performance of given features and a classification method. While the weighting of the predefined features is performed by a classifier (e.g. Random Forest [110], Naive Bayes [35], NN [86]), the features themselves are immutable.

This approach has the freedom to learn specific features for each sensor location and allows the classifier to weight any features individually for a given sensor position. This allows us to identify at which position the NN can extract the most relevant information for the classification task.

Table 7.18 shows that the default test architecture archives the best performance on the right wrist and the worst performance on the left ankle.

Table 7.18: $F1_{weighted}$ for various sensor positions on TUM dataset

position	mean	median	std	max	min
H	**0.619**	**0.628**	0.019	0.634	0.587
T	0.617	0.618	0.022	0.647	0.586
R	0.616	0.617	0.039	0.669	0.562
L	0.579	0.575	0.086	0.702	0.463

Confusion matrices and learning curves of the networks for each sensor are presented in A.4 for further study.

7.5 Conclusions & Outlook

The direct comparison of the feature-engineering approach with the feature-learning approach shows, that despite the progress made in the field of neural networks the classic method still remains competitive. On the TUM-dataset the classic method beats the DeepConvLSTM by a good margin, despite no special adaption towards the dataset. On the OPP-dataset on the other hand, the average of all performance metrics of the classic approach falls behind DeepConvLSTM. Though a look at the median, min and max values indicate, that the issue lies with one fold. A behavior that is also observed in experiments

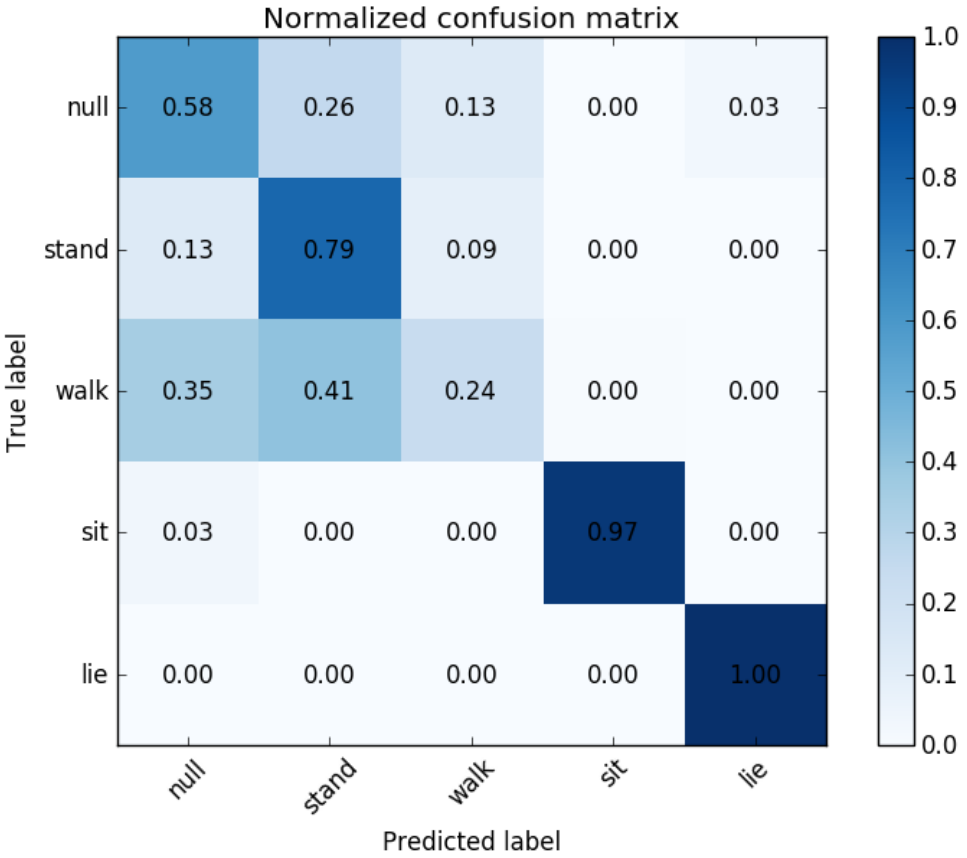

Figure 7.6: best performing architecture on OPP; confusion-matrix of the worst fold

with alternative neural network architectures (e.g. in Section 7.4.3 and 7.4.2). Further analysis of the dataset is needed to make a definite statement about the issues with one of the individuals and what behavior in particular takes the traditional method and the neural networks off track.

Network architectures and feature-engineered methods are transfered to new datasets. While the classic approach is performing well without further changes, neural networks must be adapted to changed dimensionality and as a result retrained.

Though the architecture deducted from the hyper-parameter-evaluation does not beat DeepConvLSTM in all instances, the performance is on par. Notable is that the new architecture is actually outperforming DeepConvLSTM on the TUM-dataset, despite not using 2D convolutional layer. This indicates, that convolutions are not a requirement for a good performance in HAR. Using 1D convolutional layers in the architecture is shown to be detrimental to the classification performance. Combining the deducted architecture with 2D convolutions could provide interesting results in future work.

Another notable difference between DeepConvLSTM and the evaluated architectures is the focus of DeepConvLSTM on the last observation in a time-series. Architectures evaluated in this work on the other hand, are using the whole time-series to predict the activity-class. That creates issues with the windows that contain observations of multiple activities. The DeepConvLSTM-architecture seems to cope better with this issue, though further evaluation of theses edge-cases must be made.

An issue that is largely ignored by this work, is variance created by neural network training. Initialization of weight, batch-composition and the gradient descent induce randomness into the training of a model. To determine the degree of the variance, multiple models of the same architecture must be evaluated. That is an interesting topic for future work. Also the model-complexity and training-time in the form of seconds or epochs is not considered by this work. Future evaluations can cover that.

The experiments with 4 different sensor-placements in the TUM-dataset show, that the best results can be reached if the sensor is worn on the wrist.

Further research can be done on the personalization of models. In this evaluation the individuals are already separated into train- and test-dataset. The magnitude of that separation compared to a dataset where the individuals are mixed in both sets and the potential of tailoring a pre-trained model to a specific individual are certainly interesting topics.

Chapter 8

Conclusions and Outlook

The previous Chapters introduced the management paradigm of Strategic Management, outlining some of the methods used in that paradigm. Due to digitalization, the Worl Wide Web and Big Data, the support of these methods using automated data analysis is feasible. In particular the task of environmental scanning, which describes the analysis of external factors in the meso- and marco-environment is an attractive candidate for automatization. This is because of the free availability of the data needed for such an analysis and the large amount of information that is available for processing.

Separating the time-horizon into three different ranges short-, medium- and long-term, allows the identification of different data-sources for analysis. While there is certainly an overlap, e.g. between the Web Data and the Patent Data in terms of time-horizon, volatile trends are more readily available in short-lived data e.g. Web Data. Entry-cost into a data-set (e.g. patent application fees, work of collecting accurate data on state level) fulfill the role of a filter that controls access to a dataset. An entry-cost-based model for datasets and their volatility in regards of trend-detection and monitoring could be helpful in identifying datasets for a given time-range. In future work, the generation and evaluation of such a model could provide helpful insights for data-set selection and a better understanding of entry-costs.

Following the methods listed in the "Foresight Diamond of Popper" [40], time-series analysis and patent-analysis are evidence based and quantitative methods. Their low demand for expertise makes them a good fit for an automated system. The goal of the automated system is to support a decision maker by filtering information and classification by compressing the information. With that approach, any expert knowledge can still be included in the process, though at a later stage. The work presented in this thesis aims to achieve this supply of filtered and compressed information by providing multiple metrics (e.g. Web Data in Chapter 4) or by presenting lists sorted by probability (e.g. Patent Classification in Chapter 5) to the analyst.

For the short-term time-range, experiments show that Web Data is a good source of

information. The proposed method in Chapter 4 allows the tracking of fast-paced trends by monitoring Web documents over time. A statistical analysis provides various metrics that indicate trends, reach and activity regarding a specific topic defined by keywords. Integrated machine translation helps to identify trends in other languages. The dimension "language" is combined with the dimension "country" so that a regional analysis on a country-level is possible. Primarily targeted at the short-term outlook, the proposed approach scales linear the number of observations. For medium- to long-term analysis enhancements that scale sub-linear over time are a field for future work. The addition of further metrics for trend detection in combination with a model utilizing all these metrics could improve the utility. Semantic methods, e.g. entity detection and part-of-speech tagging would be extremely helpful in the identification of trends where the subject consists of multiple tokens. Though this work focuses on the support of multiple languages, the integration of semantic methods is future work. The proposed method represents trends and other metrics, e.g. reach as time-series. Unsupervised clustering on the time-series could be used to identify similar time-series in order to allow a more focused approach by the analyst. Depending on the quality of the clustering, the extracted data might be usable to create a prediction about the behavior of an emerging trend.

The nature of patents ensures, that only documents which contain an innovation and for which the registering organization is willing to pay several thousand dollars are part of the corpus. Following the assumption of decreasing noise with increasing entry-cost and the fact that applications for patents are only followed after a process of R&D, they are not ideal for monitoring of short-term developments. For the medium-term outlook though, patent analysis provides an attractive database as the documents are rich in meta-data, follow an agreed upon structure, and are categorized into a system of classes that allows fine-grained monitoring of fields of interest as detailed in Chapter 5. In this work, methods are proposed that enhance the utility of patent application documents, by enhancing automatic patent classification. That reduces the lag in which a document is available to the public, but it is not clear to which classes it is relevant. The proposed methods follow two different approaches: meta-data-usage and a enhanced text-representation. Section 5.4 takes advantage of the formation of Industrial Clusters that concentrate certain industries in a geographic region. Concluding from the inventors addresses, the predetermined patent-class distribution of a region is used to enhance the predictions of an ordinary text-classifier. Experiments show, that the ensemble-approach increases the classification-accuracy. The automatic detection of such clusters and their size is a open research question. Also the evaluation of the movement and development of these clusters over time provides interesting future work.

Part of the ensemble proposed in Section 5.4 is a text-classifier. The state-of-the-art in patent classification is using BOW-approaches and SVMs. In Section 5.5 this thesis evaluates multiple text-representations coupled with standard classification methods. By using a semantic representation and a deep neural network for classification, the performance can be significantly enhanced in comparison to BOW methods. The semantic structure of a patent document is projected onto a hierarchy of context models. The vector resulting

from a depth-first traversal of this hierarchy is referred to as FHV. This representation of the context-hierarchy models the reading-flow of a human through the document and allows the usage of sequence learning. This work shows, that the sequence-representation and the usage of various context-granularities exceeds the performance achievable with other document representations. A current limitation of that approach is the need for a fixed semantic structure in all elements of a collection. Adapting that approach for a flexible sequence-length while maintaining the semantic association remains future work. Evaluations of the FHV on other data types e.g. videos or images are also to be made.

In this thesis it is shown, that automatic patent classification can be enhanced by multiple approaches. In the end though, an ensemble approach is needed for best results. That would combine multiple classification methods such as co-citation-analysis, GeoDAT and text-classification utilizing FHV. The improved classification-performance could give a business a competitive edge in patent-analysis.
In Chapter 6, marco-indicators are used to monitor the long-term time-horizon. A web based system is proposed that can integrate various external indicator sources and provides functionality relevant to an analyst. This functionality includes visualization, export of data and simple arithmetic operations with indicators. A new metric is proposed to gauge the quality of an indicator in relation to completeness, coverage of countries and continuity of observations. Weighting of these aspects model an individual preference of the analyst. Experiments with this metric on the World Bank Data Catalog show, that it is discriminative and limits the amount of indicators with a acceptable score significantly. To support the task of time-series extrapolation, BestFit is introduced. The method evaluates the performance of various regressors on a time-series to determine which of them performs best. The evaluation of BestFit shows, that multiple regressors are used. Further research is necessary for a quantitative analysis of the prediction quality. The dataset used for the experiments needs expansion by the scale-type so that prediction-performance between two indicators can be compared.

Evaluation of Deep Learning techniques and the direct comparison to feature-engineering in Chapter 7 shows that there is great potential for time-series processing with RNNs. The supervised classification task with feature-learning is on par with the traditional feature-engineering. Hyper-parameter evaluation shows where the greatest gains in regards of classification-performance can be had. Comparison of various network-architectures confirms that in the experimental setting RNNs are superior to feed forward networks. Even if convolutions are omitted, the NNs perform reasonably well. A competitive architecture is derived from the results of the hyper-parameter evaluation and tested against a state-of-the-art network. This results in an architecture that outperforms the reference methods in at least one dataset. Further evaluation of convolutions must be made. There are also special characteristics of one of the datasets that would warrant further research. In relation to the task of time-series classification, Deep Learning seems an appropriate technology when sufficiently labeled data is available. This observation can also be made in Section 5.5 where the classification is also performed by an RNN. In fields where little feature-engineering has been done, Deep Learning could prove extremely helpful as features are

learned on the fly. The task of trend-detection and classification would be an interesting application. The potential of integrating feature-learning into the analysis of Web Data in future work looks promising. The same goes for macro-indicators and models that depend on them.

While this thesis proposed methods for three different time-ranges, the integration of all these approaches to a combined tool for environmental scanning is sensible. Scanning projects in such a setting are defined by tokens, patent-classes, macro-indicators and related models. The time-series processing in such a system can be supported by Deep Learning, as this thesis has shown that due to feature-learning little expert-knowledge is required while still performing reasonably well.

Appendix A

Human Activity Recognition

A.1 Number of Neurons

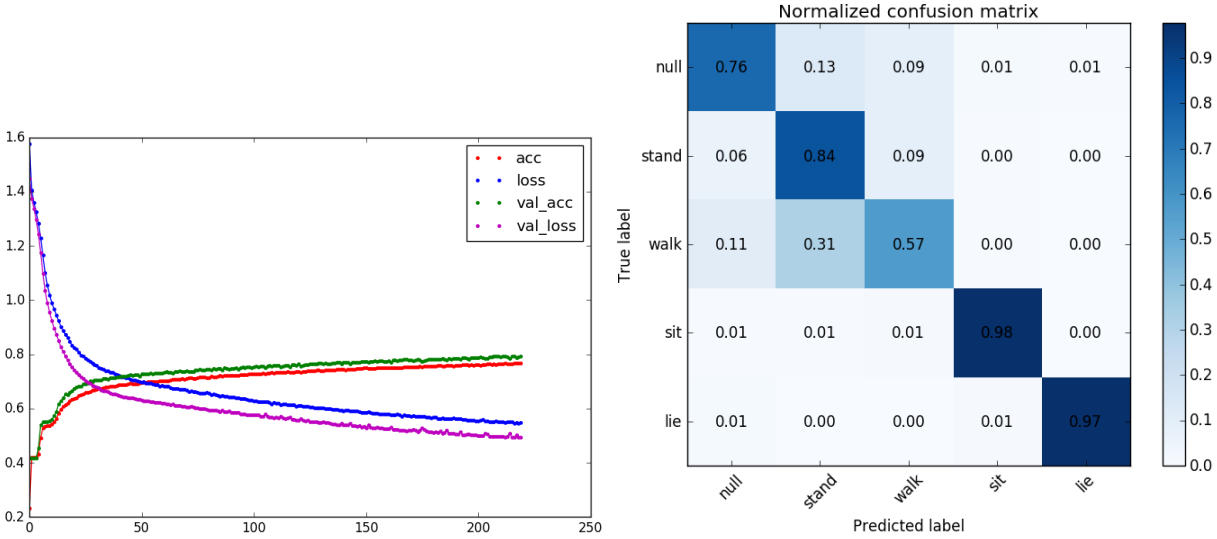

Figure A.1: dataset: OPP, w=16, neurons=128

Table A.1: dataset: OPP, w=16, neurons=128

	mean	median	std	max	min
$F1_{micro}$	0.746	0.791	0.065	0.792	0.654
$F1_{macro}$	0.709	0.805	0.149	0.823	0.498
$F1_{weighted}$	0.730	0.787	0.082	0.788	0.615

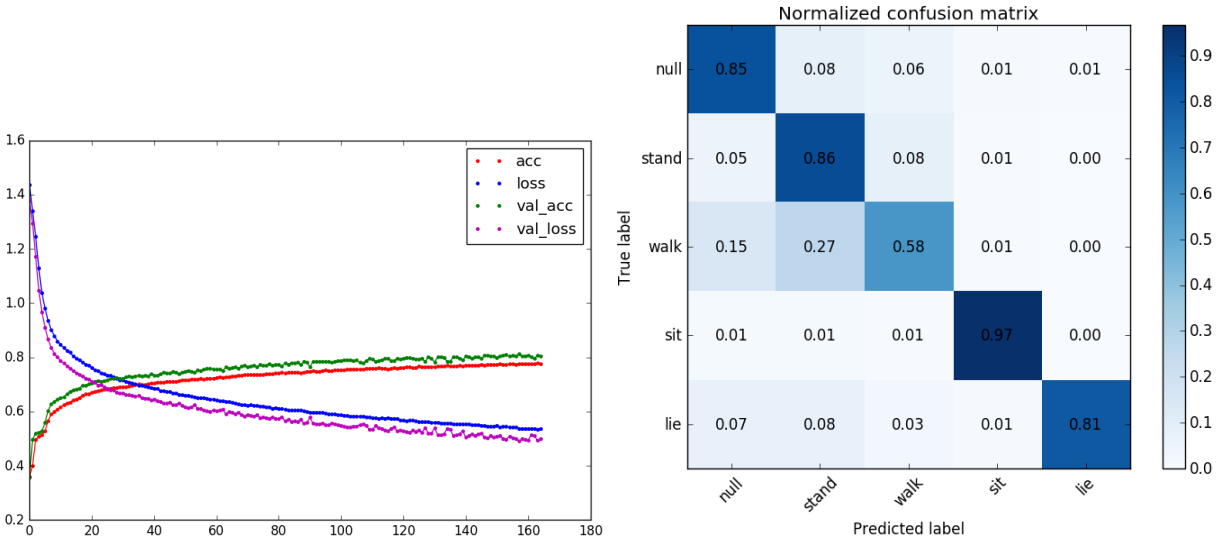

Figure A.2: dataset: OPP, w=16, neurons=256

Table A.2: dataset: OPP, w=16, neurons=256

	mean	median	std	max	min
$F1_{micro}$	0.701	0.802	0.150	0.812	0.490
$F1_{macro}$	0.656	0.823	0.236	0.823	0.323
$F1_{weighted}$	0.673	0.800	0.186	0.808	0.410

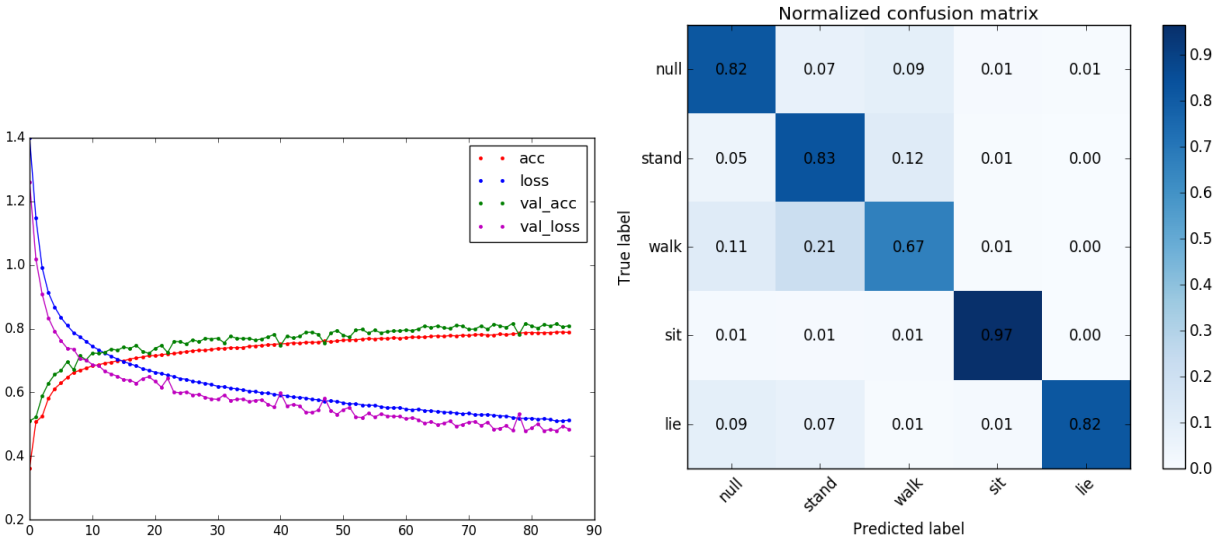

Figure A.3: dataset: OPP, w=16, neurons=512

Table A.3: dataset: OPP, w=16, neurons=512

	mean	median	std	max	min
$F1_{micro}$	0.723	0.817	0.134	0.819	0.534
$F1_{macro}$	0.681	0.830	0.220	0.843	0.369
$F1_{weighted}$	0.703	0.815	0.159	0.816	0.478

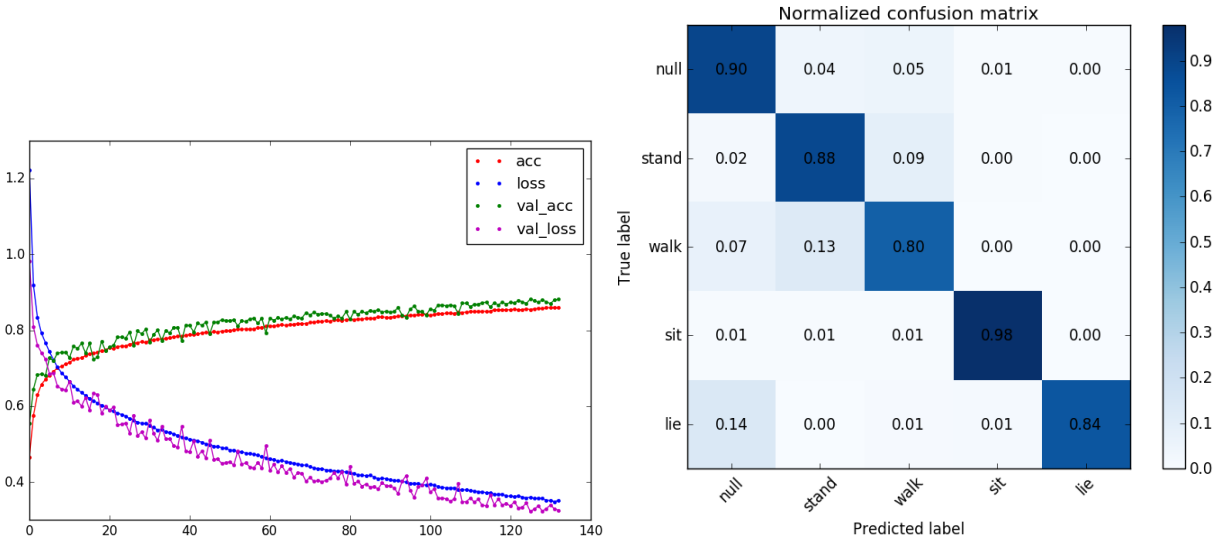

Figure A.4: dataset: OPP, w=16, neurons=1024

Table A.4: dataset: OPP, w=16, neurons=1024

	mean	median	std	max	min
$F1_{micro}$	0.795	0.846	0.099	0.882	0.658
$F1_{macro}$	0.783	0.861	0.128	0.886	0.603
$F1_{weighted}$	0.786	0.845	0.111	0.882	0.630

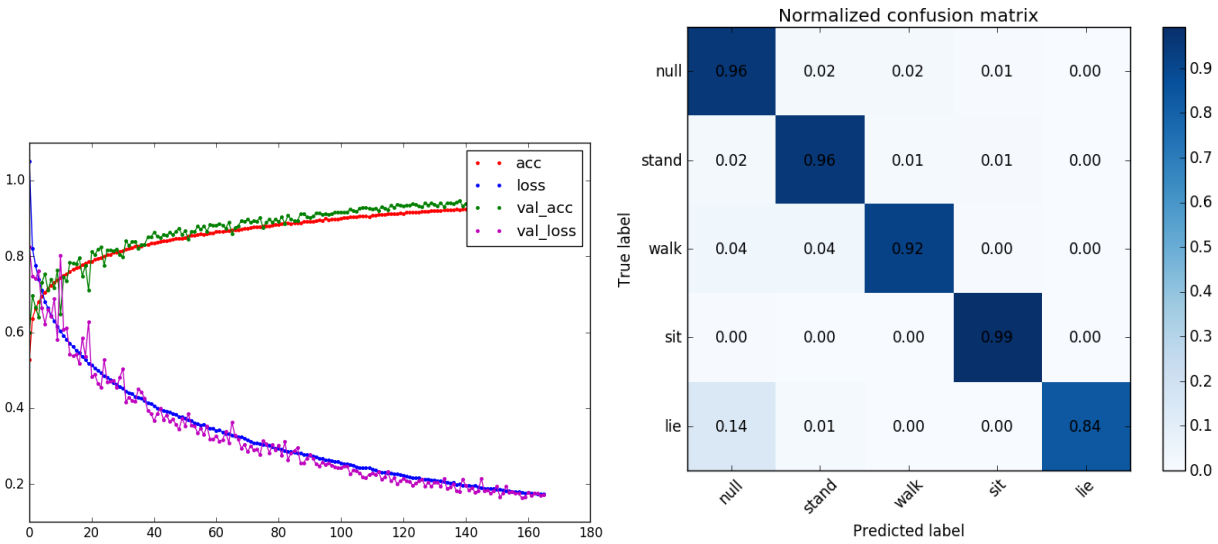

Figure A.5: dataset: OPP, w=16, neurons=2048

Table A.5: dataset: OPP, w=16, neurons=2048

	mean	median	std	max	min
$F1_{micro}$	0.852	0.886	0.100	0.954	0.716
$F1_{macro}$	0.840	0.895	0.115	0.944	0.680
$F1_{weighted}$	0.836	0.886	0.122	0.954	0.668

A.2 Feature Engineering Approach

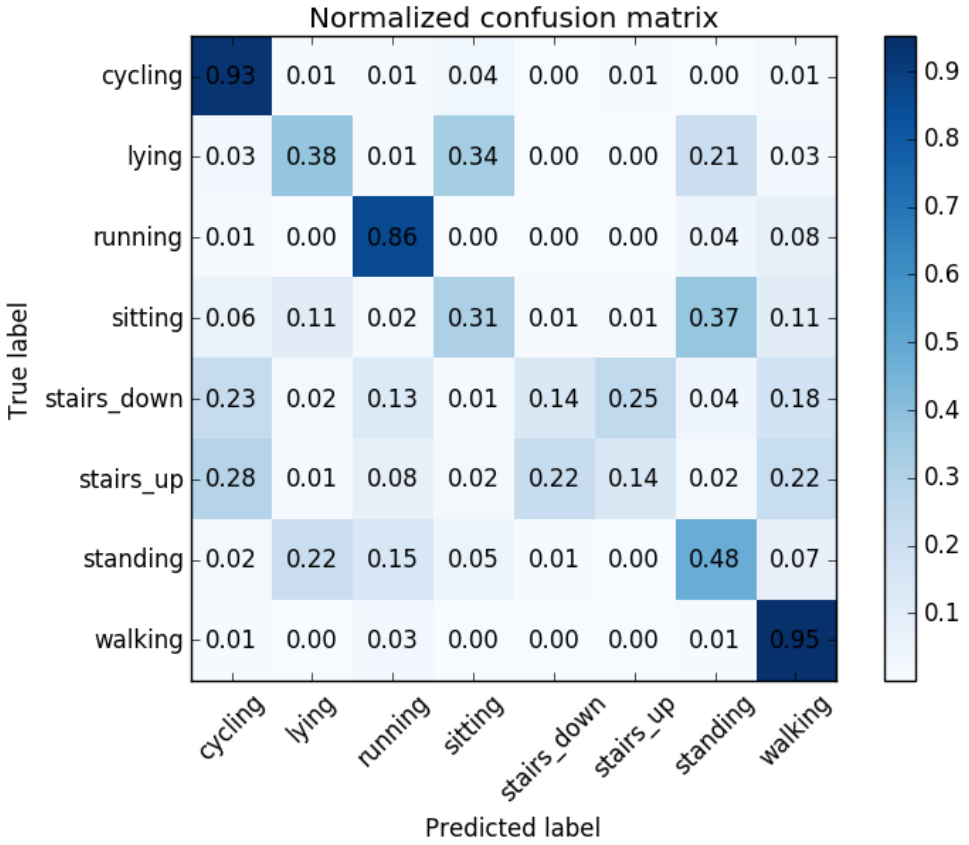

Figure A.6: dataset: TUM, sensor=H, w=32, feature engineering approach

Table A.6: dataset: TUM, sensor=H, w=32, feature engineering approach

	mean	median	std	max	min
$F1_{micro}$	0.660	0.665	0.026	0.689	0.620
$F1_{macro}$	0.506	0.518	0.024	0.521	0.464
$F1_{weighted}$	0.642	0.647	0.021	0.664	0.608

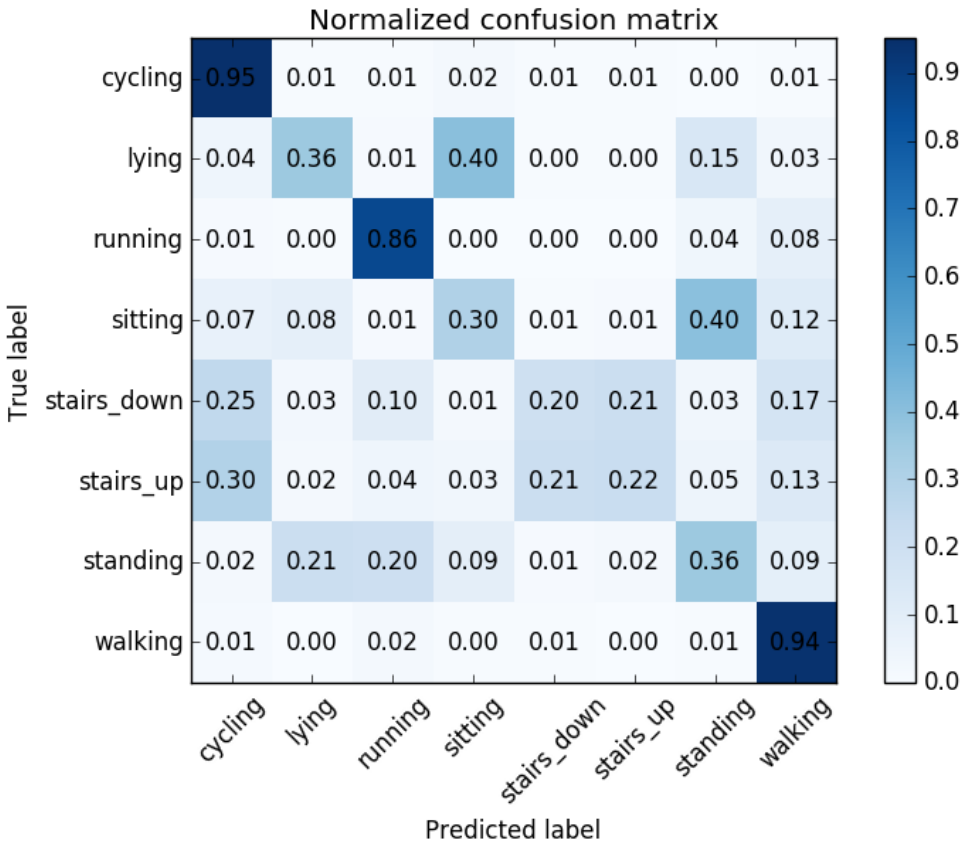

Figure A.7: dataset: TUM, sensor=H, w=64, feature engineering approach

Table A.7: dataset: TUM, sensor=H, w=64, feature engineering approach

	mean	median	std	max	min
$F1_{micro}$	0.662	0.665	0.025	0.689	0.620
$F1_{macro}$	0.508	0.518	0.022	0.531	0.464
$F1_{weighted}$	0.645	0.647	0.020	0.666	0.608

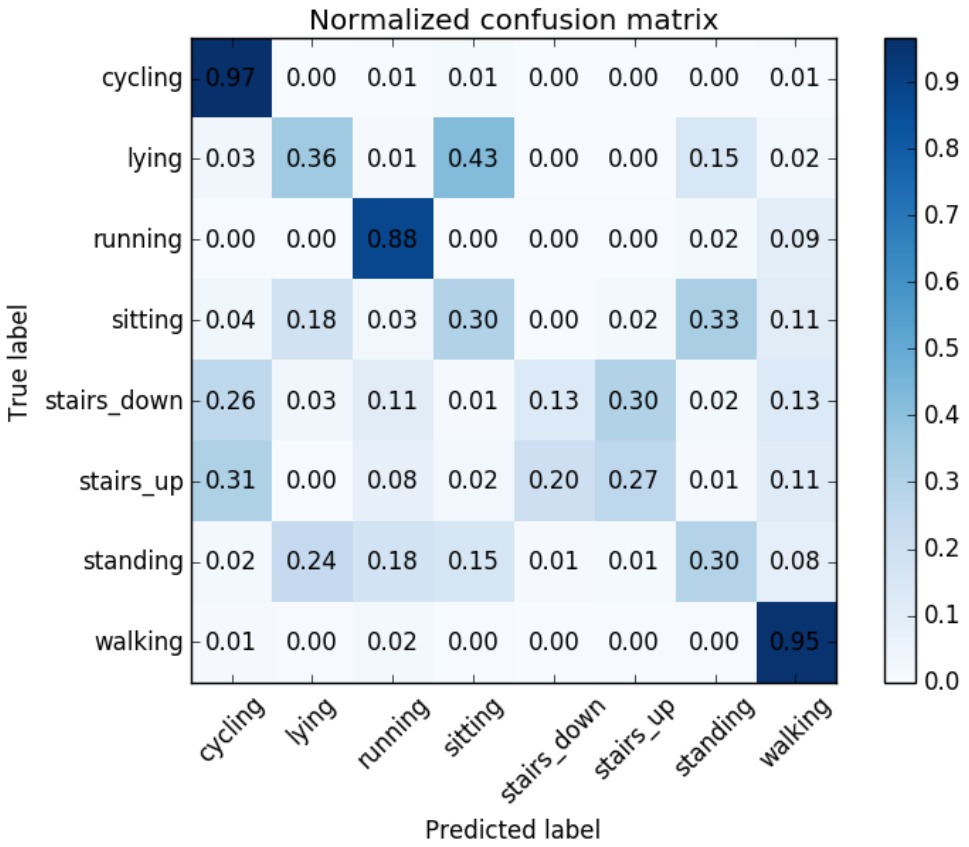

Figure A.8: dataset: TUM, sensor=H, w=128, feature engineering approach

Table A.8: dataset: TUM, sensor=H, w=128, feature engineering approach

	mean	median	std	max	min
$F1_{micro}$	0.665	0.672	0.023	0.694	0.620
$F1_{macro}$	0.510	0.520	0.020	0.531	0.464
$F1_{weighted}$	0.648	0.654	0.018	0.670	0.608

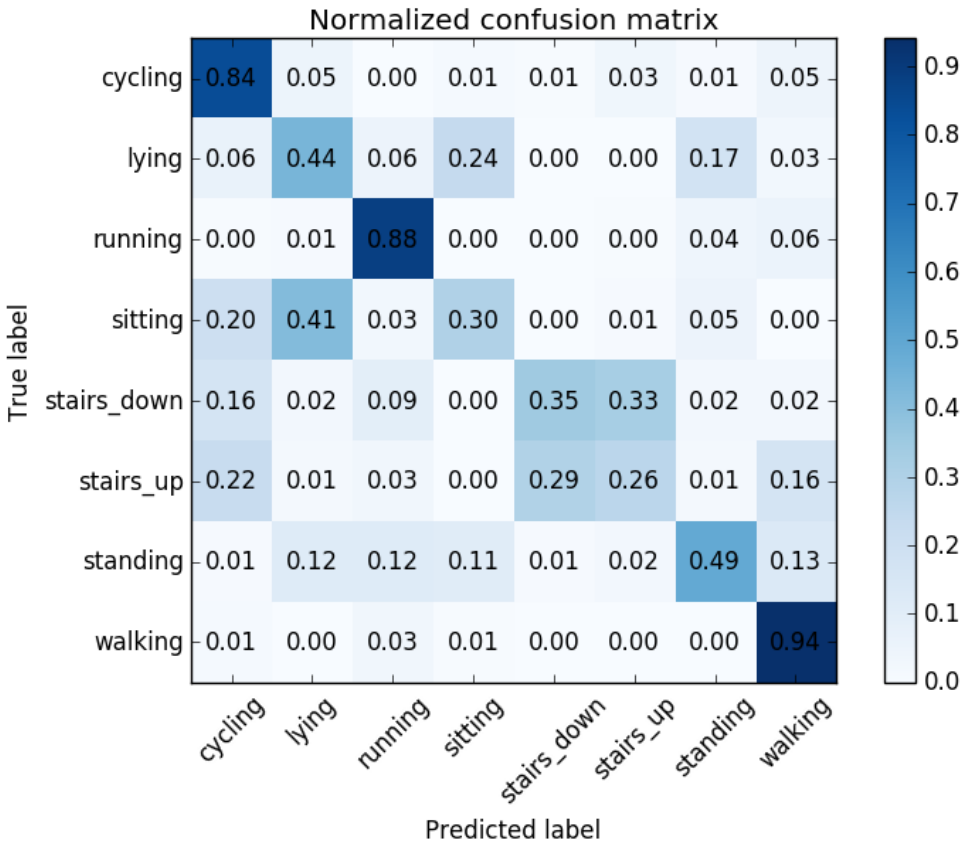

Figure A.9: dataset: TUM, sensor=H, w=256, feature engineering approach

Table A.9: dataset: TUM, sensor=H, w=256, feature engineering approach

	mean	median	std	max	min
$F1_{micro}$	0.672	0.676	0.024	0.707	0.620
$F1_{macro}$	0.516	0.521	0.024	0.567	0.464
$F1_{weighted}$	0.655	0.655	0.022	0.696	0.608

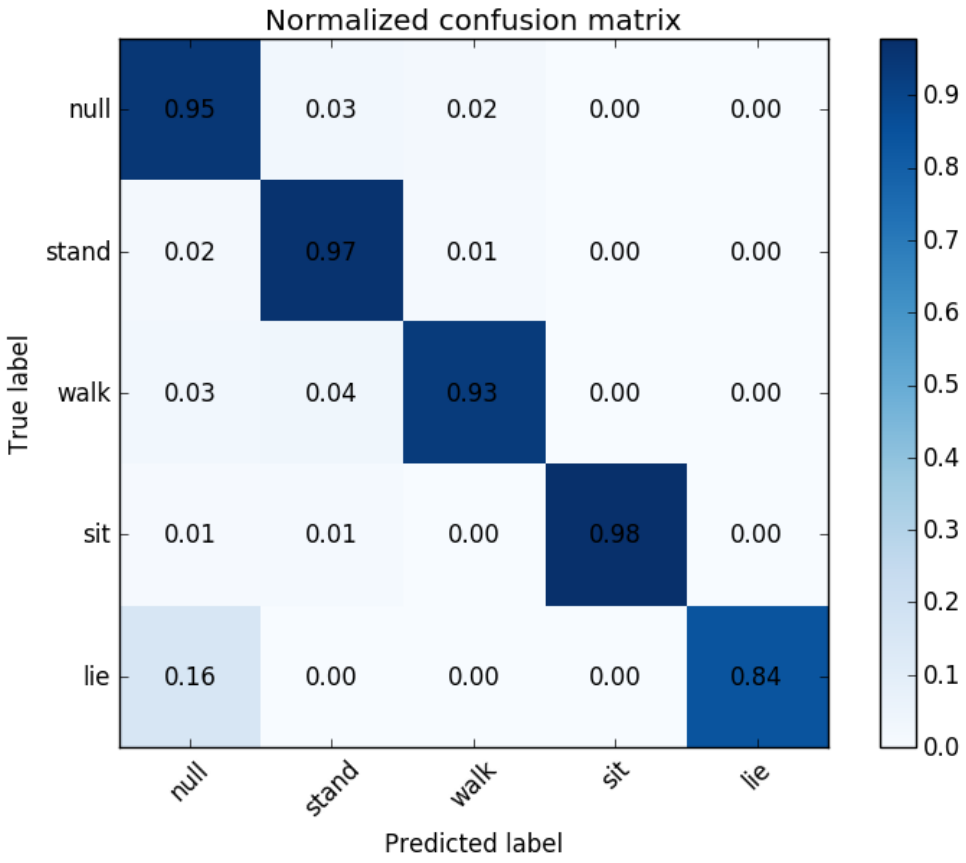

Figure A.10: dataset: OPP, w=4, feature engineering approach

Table A.10: dataset: OPP, w=4, feature engineering approach

	mean	median	std	max	min
$F1_{micro}$	0.816	0.949	0.192	0.954	0.543
$F1_{macro}$	0.823	0.947	0.175	0.947	0.575
$F1_{weighted}$	0.806	0.949	0.206	0.954	0.514

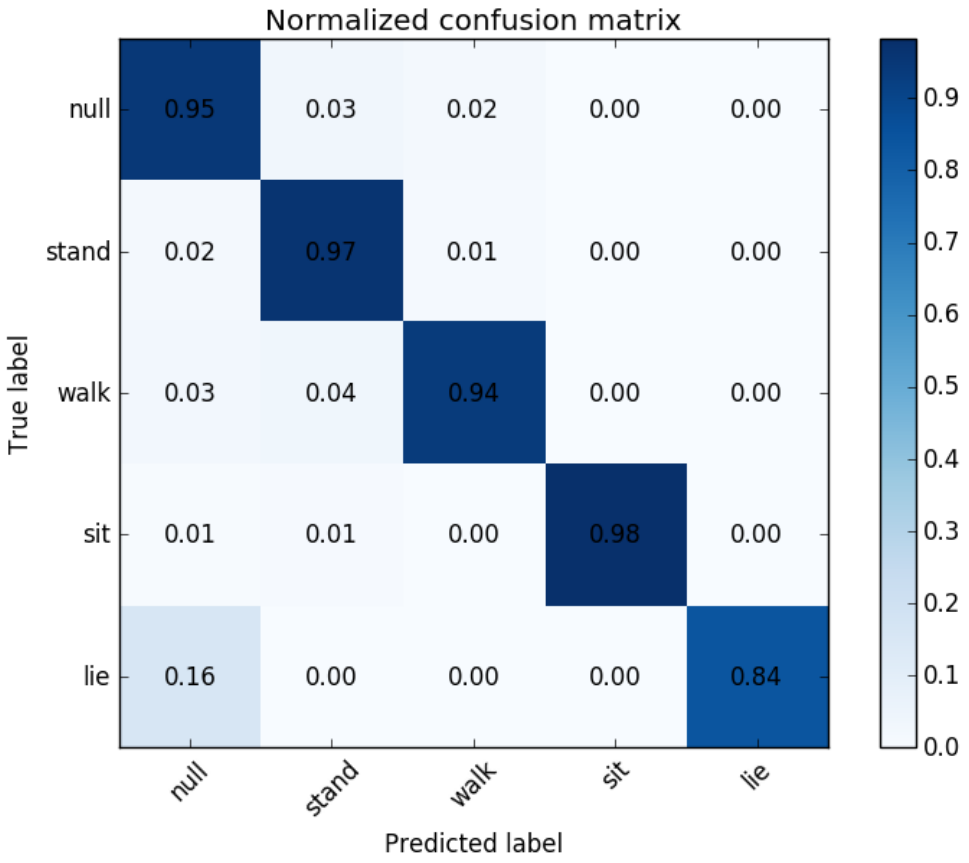

Figure A.11: dataset: OPP, w=8, feature engineering approach

Table A.11: dataset: OPP, w=8, feature engineering approach

	mean	median	std	max	min
$F1_{micro}$	0.815	0.952	0.196	0.956	0.530
$F1_{macro}$	0.822	0.947	0.180	0.952	0.560
$F1_{weighted}$	0.805	0.952	0.211	0.956	0.499

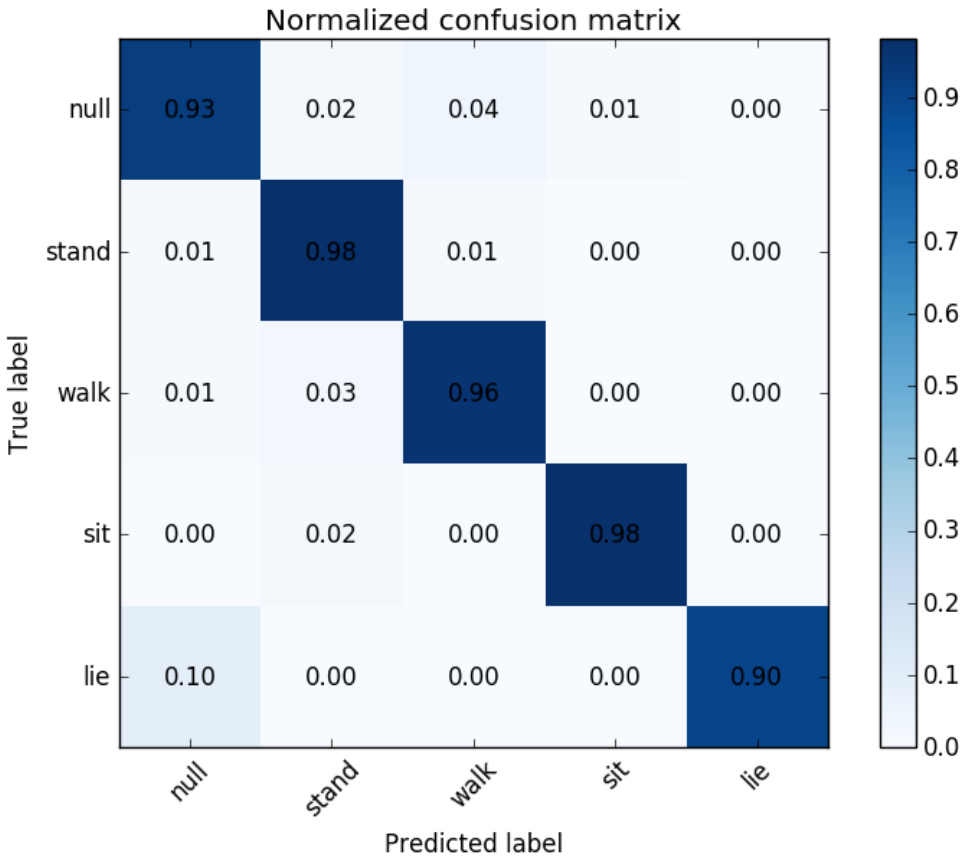

Figure A.12: dataset: OPP, w=16, feature engineering approach

Table A.12: dataset: OPP, w=16, feature engineering approach

	mean	median	std	max	min
$F1_{micro}$	0.813	0.954	0.202	0.965	0.511
$F1_{macro}$	0.819	0.947	0.187	0.960	0.530
$F1_{weighted}$	0.802	0.954	0.218	0.964	0.468

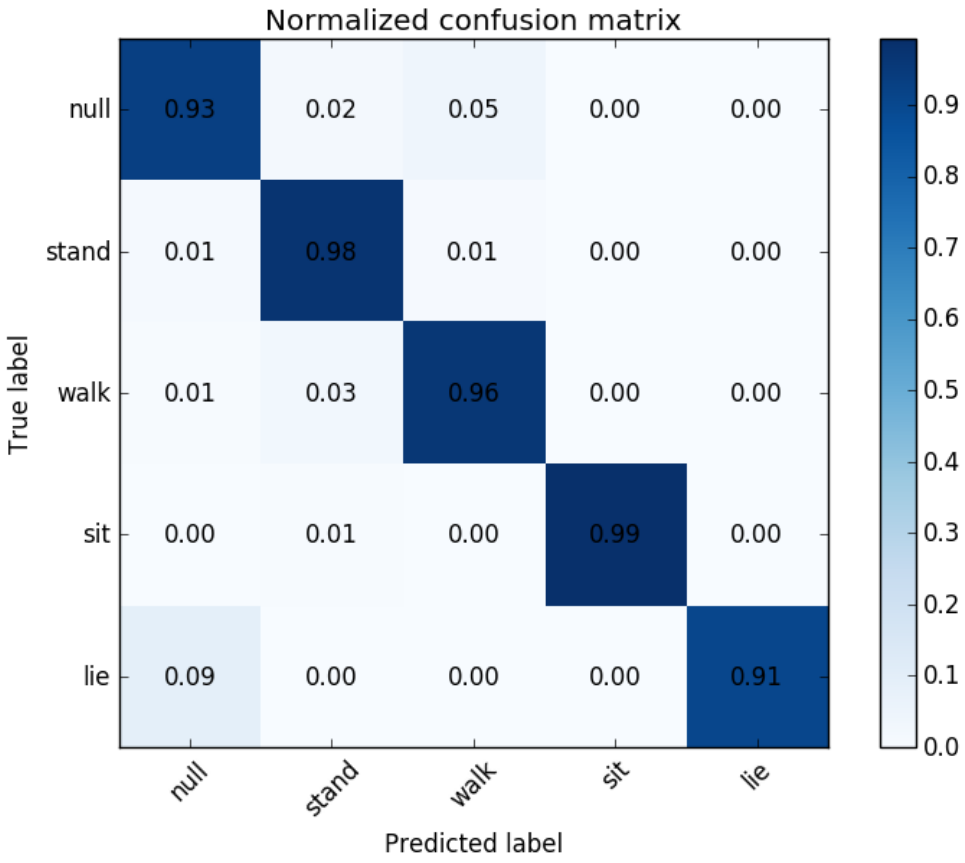

Figure A.13: dataset: OPP, w=32, feature engineering approach

Table A.13: dataset: OPP, w=32, feature engineering approach

	mean	median	std	max	min
$F1_{micro}$	0.823	0.954	0.191	0.967	0.511
$F1_{macro}$	0.828	0.948	0.177	0.963	0.530
$F1_{weighted}$	0.813	0.954	0.206	0.967	0.468

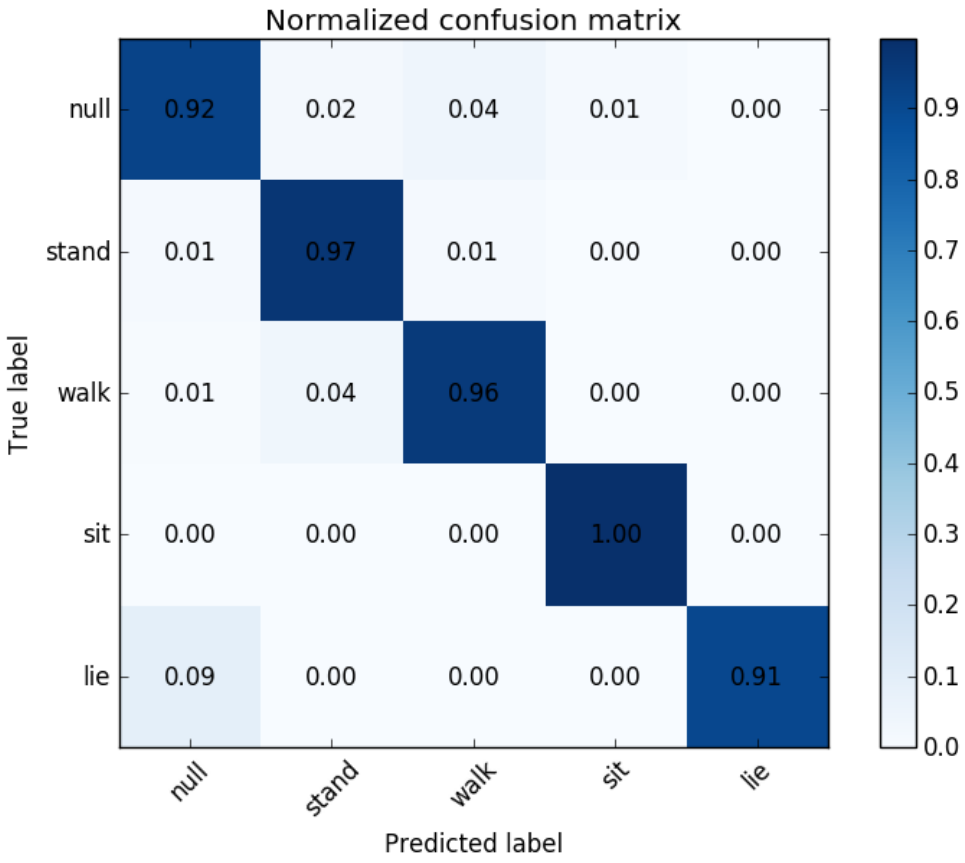

Figure A.14: dataset: OPP, w=64, feature engineering approach

Table A.14: dataset: OPP, w=64, feature engineering approach

	mean	median	std	max	min
$F1_{micro}$	0.820	0.955	0.196	0.967	0.510
$F1_{macro}$	0.820	0.948	0.191	0.963	0.456
$F1_{weighted}$	0.812	0.955	0.208	0.967	0.468

A.3 Window Size

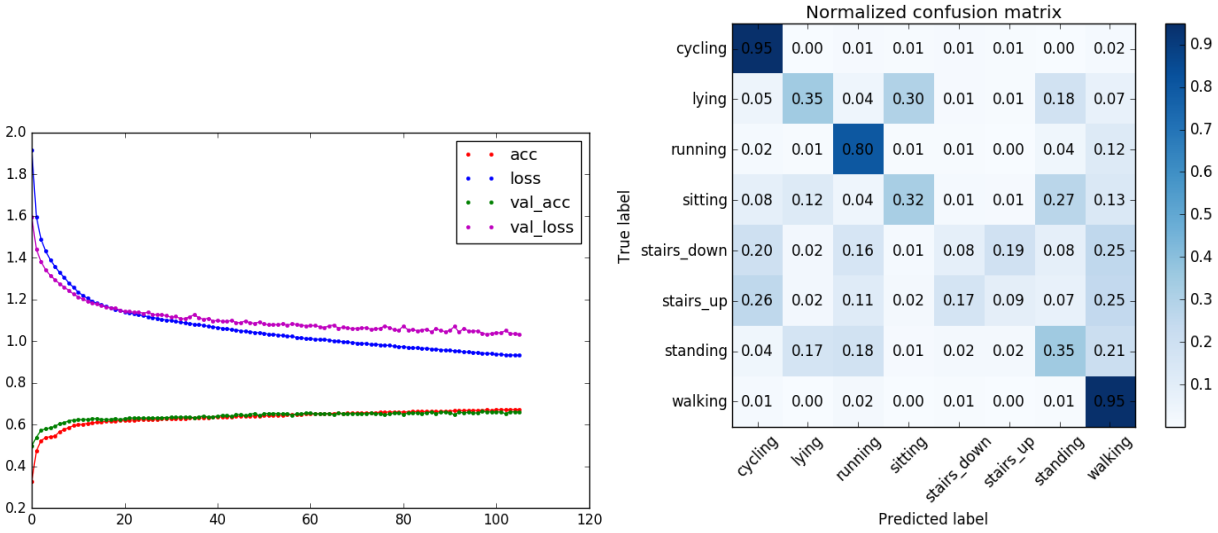

Figure A.15: dataset: TUM, sensor=H, w=32, neurons=512

Table A.15: dataset: TUM, sensor=H, w=32, neurons=512

	mean	median	std	max	min
$F1_{micro}$	0.630	0.632	0.026	0.663	0.592
$F1_{macro}$	0.464	0.478	0.027	0.483	0.417
$F1_{weighted}$	0.602	0.606	0.022	0.628	0.568

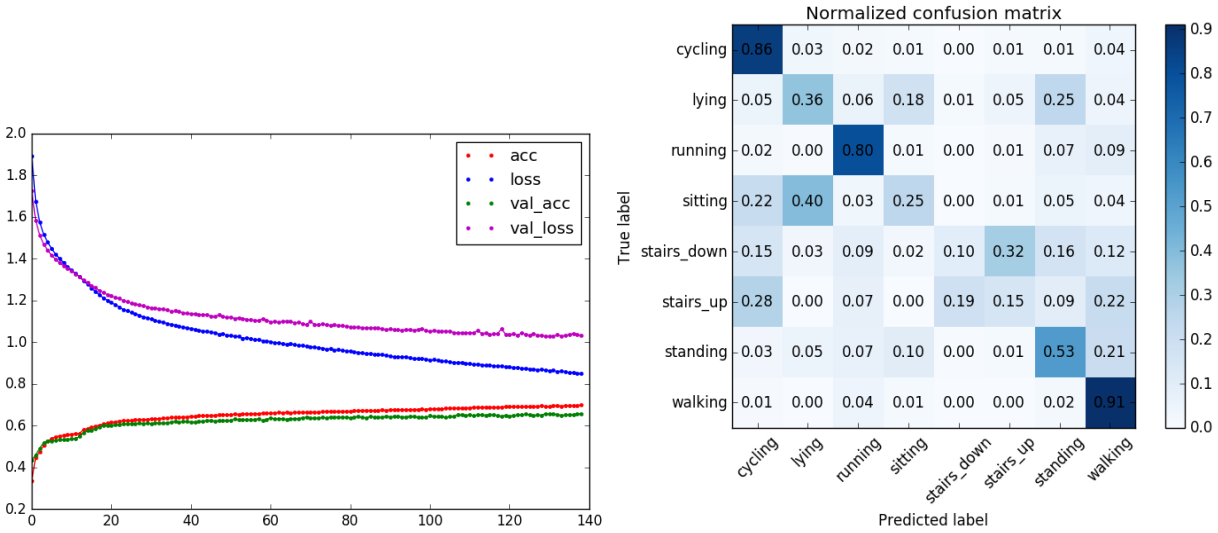

Figure A.16: dataset: TUM, sensor=H, w=64, neurons=512

Table A.16: dataset: TUM, sensor=H, w=64, neurons=512

	mean	median	std	max	min
$F1_{micro}$	0.643	0.652	0.023	0.662	0.604
$F1_{macro}$	0.476	0.484	0.026	0.502	0.435
$F1_{weighted}$	0.619	0.628	0.019	0.634	0.587

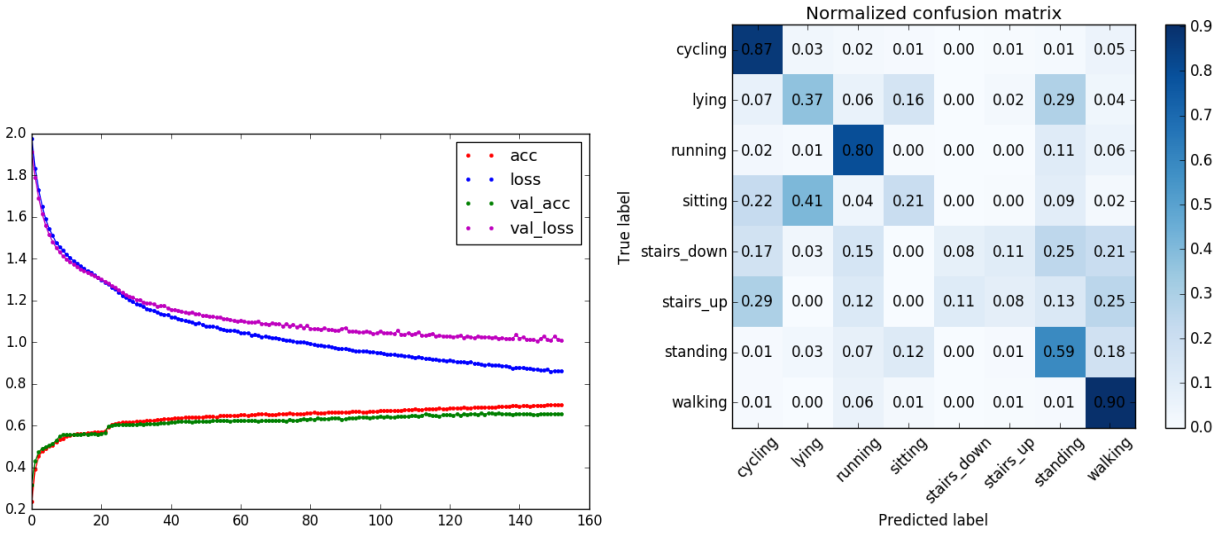

Figure A.17: dataset: TUM, sensor=H, w=128, neurons=512

Table A.17: dataset: TUM, sensor=H, w=128, neurons=512

	mean	median	std	max	min
$F1_{micro}$	0.640	0.645	0.023	0.661	0.607
$F1_{macro}$	0.449	0.453	0.025	0.479	0.413
$F1_{weighted}$	0.607	0.608	0.017	0.629	0.582

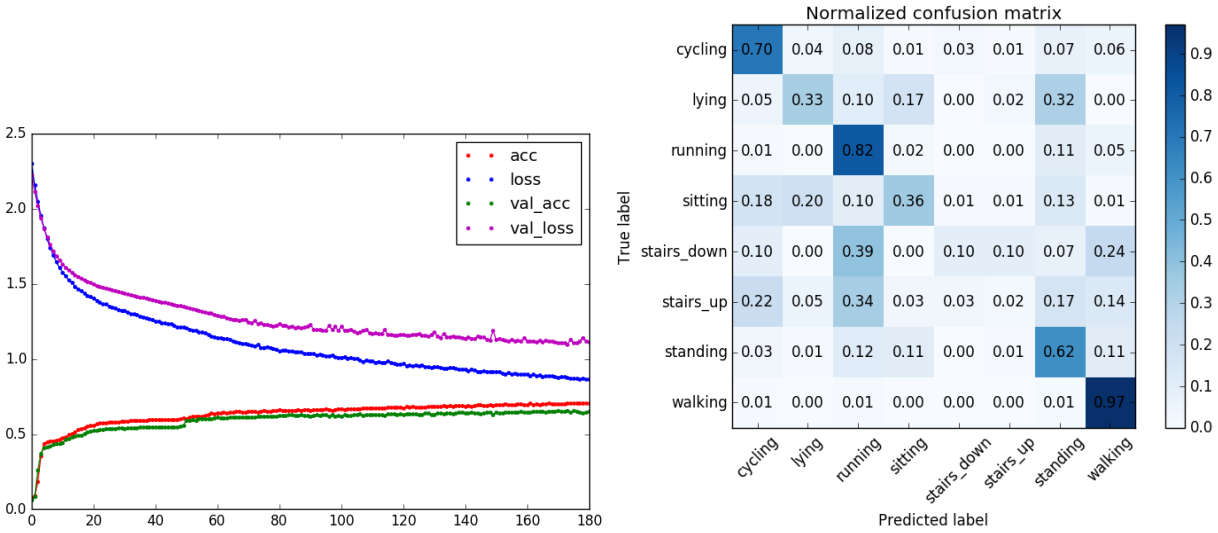

Figure A.18: dataset: TUM, sensor=H, w=256, neurons=512

Table A.18: dataset: TUM, sensor=H, w=256, neurons=512

	mean	median	std	max	min
$F1_{micro}$	0.655	0.663	0.019	0.672	0.623
$F1_{macro}$	0.450	0.457	0.028	0.480	0.404
$F1_{weighted}$	0.620	0.625	0.013	0.634	0.599

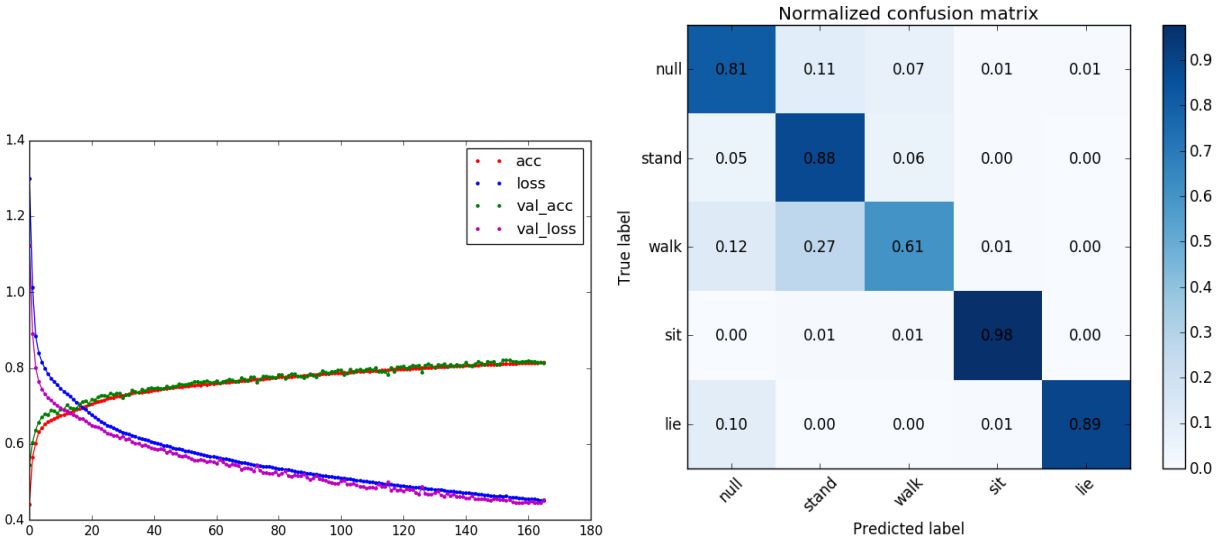

Figure A.19: dataset: OPP, w=4, neurons=512

Table A.19: dataset: OPP, w=4, neurons=512

	mean	median	std	max	min
$F1_{micro}$	0.730	0.758	0.088	0.821	0.612
$F1_{macro}$	0.688	0.777	0.171	0.839	0.450
$F1_{weighted}$	0.713	0.753	0.105	0.818	0.569

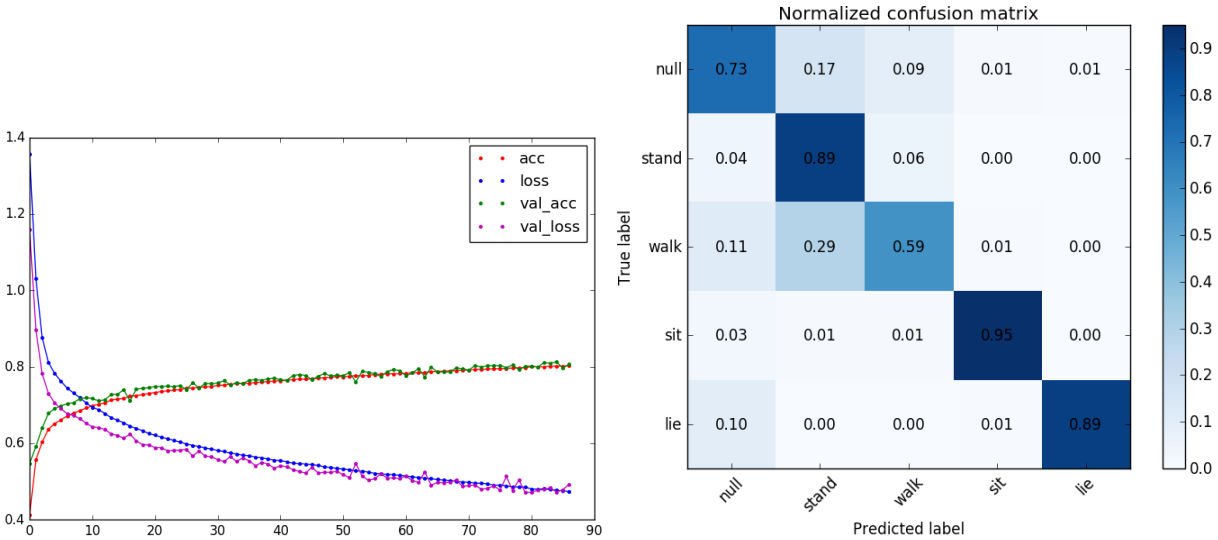

Figure A.20: dataset: OPP, w=8, neurons=512

Table A.20: dataset: OPP, w=8, neurons=512

	mean	median	std	max	min
$F1_{micro}$	0.736	0.790	0.087	0.805	0.614
$F1_{macro}$	0.697	0.804	0.164	0.822	0.465
$F1_{weighted}$	0.721	0.784	0.101	0.801	0.578

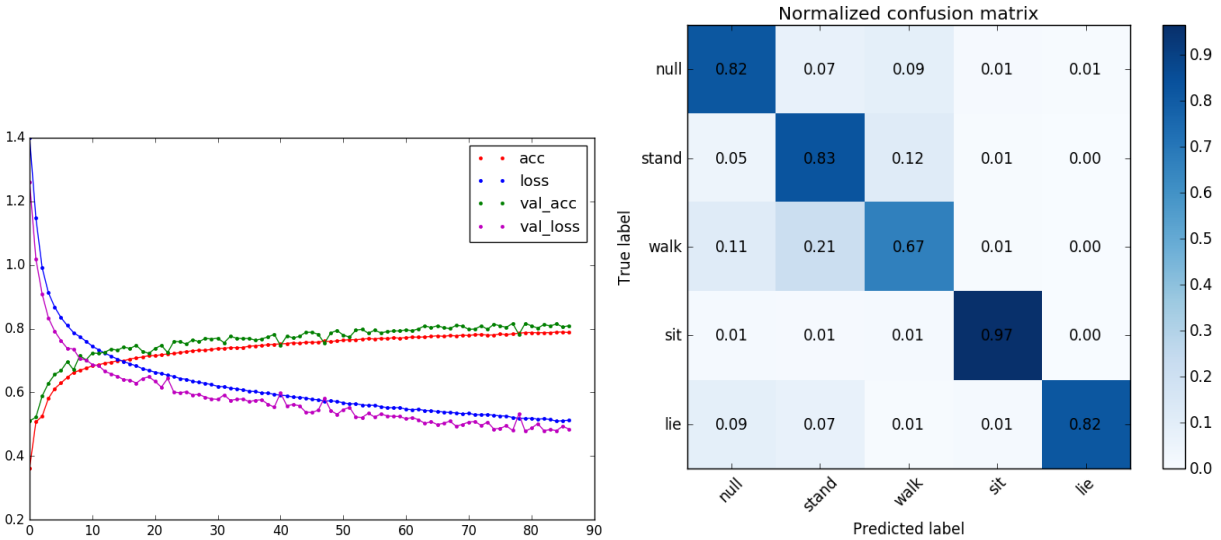

Figure A.21: dataset: OPP, w=16, neurons=512

Table A.21: dataset: OPP, w=16, neurons=512

	mean	median	std	max	min
$F1_{micro}$	0.723	0.817	0.134	0.819	0.534
$F1_{macro}$	0.681	0.830	0.220	0.843	0.369
$F1_{weighted}$	0.703	0.815	0.159	0.816	0.478

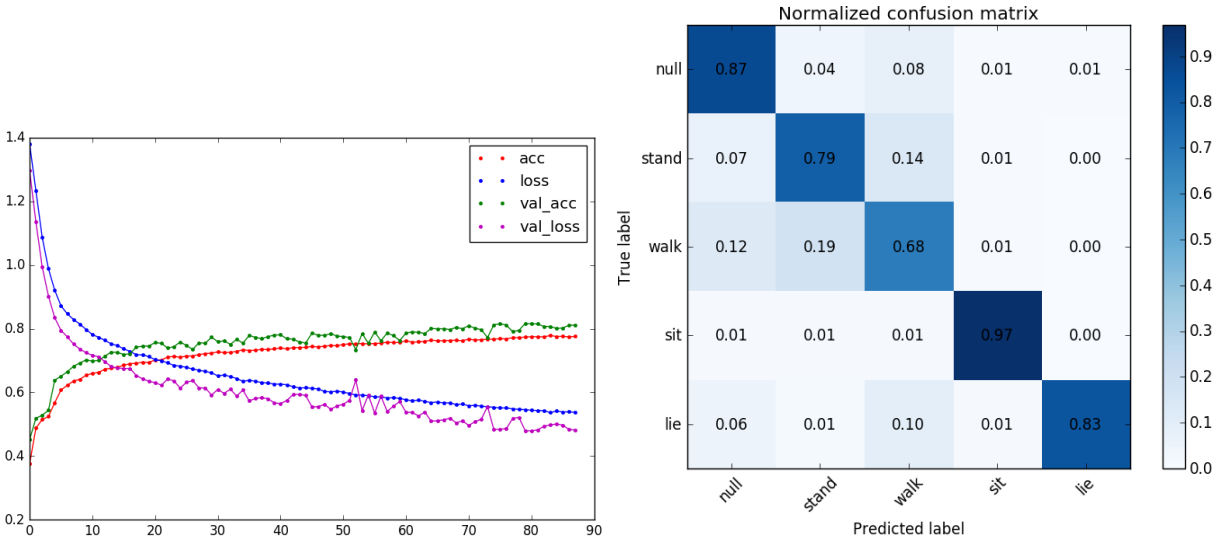

Figure A.22: dataset: OPP, w=32, neurons=512

Table A.22: dataset: OPP, w=32, neurons=512

$F1_{micro}$	0.715	0.813	0.140	0.816	0.518
$F1_{macro}$	0.666	0.831	0.236	0.835	0.332
$F1_{weighted}$	0.684	0.810	0.183	0.816	0.426

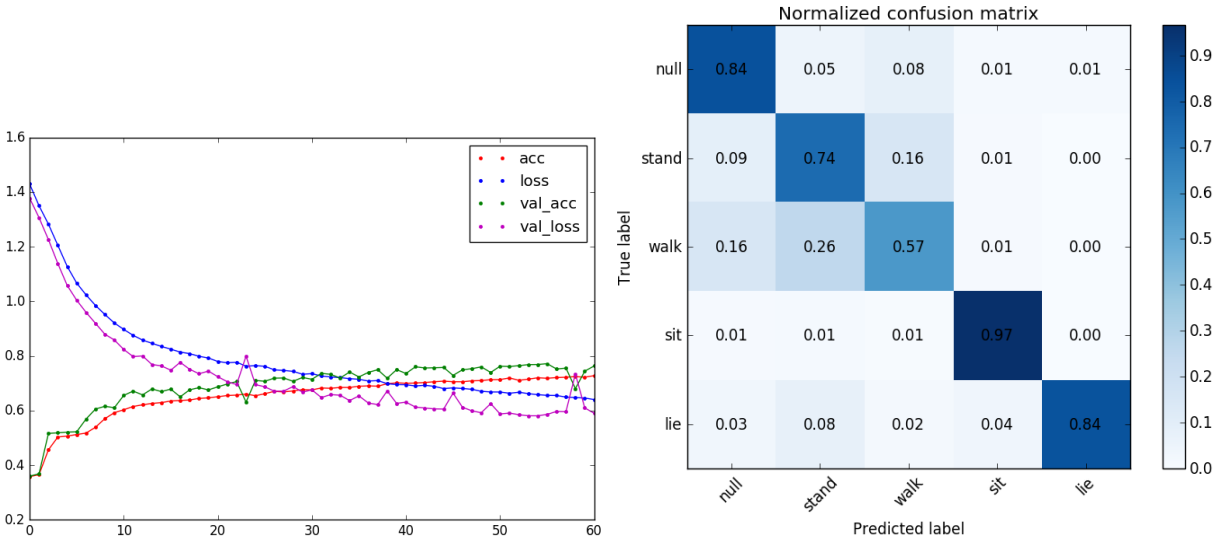

Figure A.23: dataset: OPP, w=64, neurons=512

Table A.23: dataset: OPP, w=64, neurons=512

$F1_{micro}$	0.749	0.756	0.020	0.768	0.722
$F1_{macro}$	0.691	0.784	0.135	0.788	0.500
$F1_{weighted}$	0.720	0.752	0.056	0.766	0.641

A.4 Sensor Position

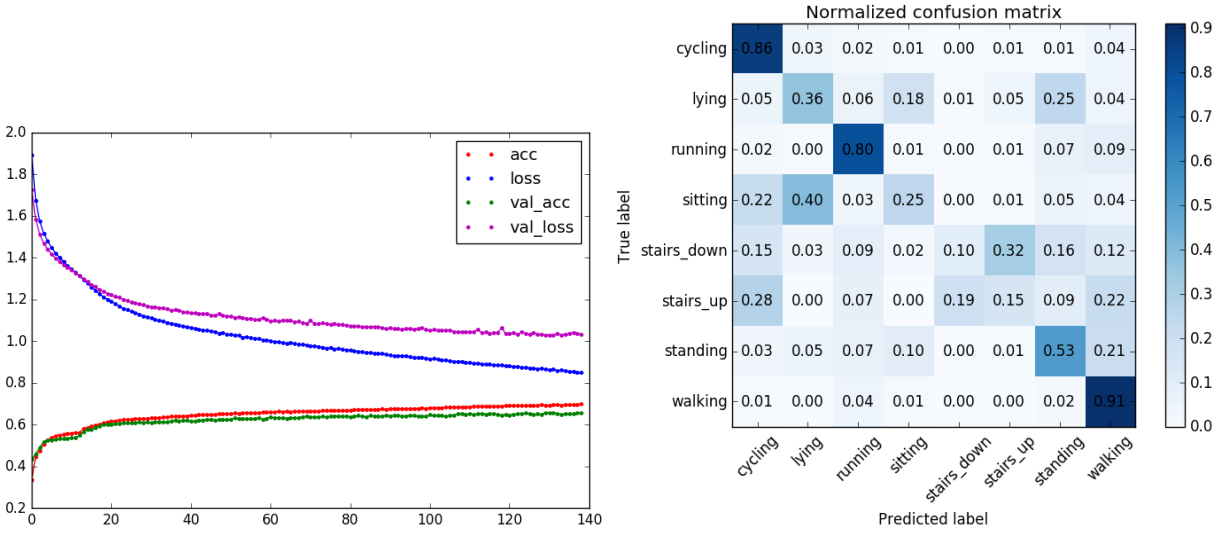

Figure A.24: dataset: TUM, sensor=H, w=64, neurons=512

Table A.24: dataset: TUM, sensor=H, w=64, neurons=512

	mean	median	std	max	min
$F1_{micro}$	0.643	0.652	0.023	0.662	0.604
$F1_{macro}$	0.476	0.484	0.026	0.502	0.435
$F1_{weighted}$	0.619	0.628	0.019	0.634	0.587

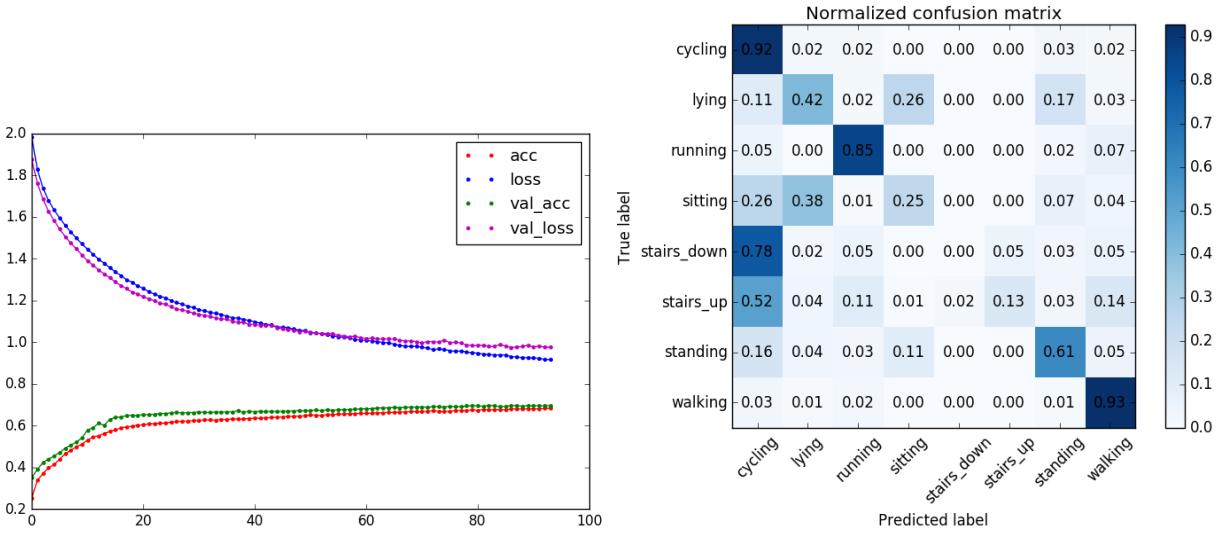

Figure A.25: dataset: TUM, sensor=R, w=64, neurons=512

Table A.25: dataset: TUM, sensor=R, w=64, neurons=512

	mean	median	std	max	min
$F1_{micro}$	0.646	0.645	0.038	0.698	0.596
$F1_{macro}$	0.447	0.450	0.048	0.511	0.376
$F1_{weighted}$	0.616	0.617	0.039	0.669	0.562

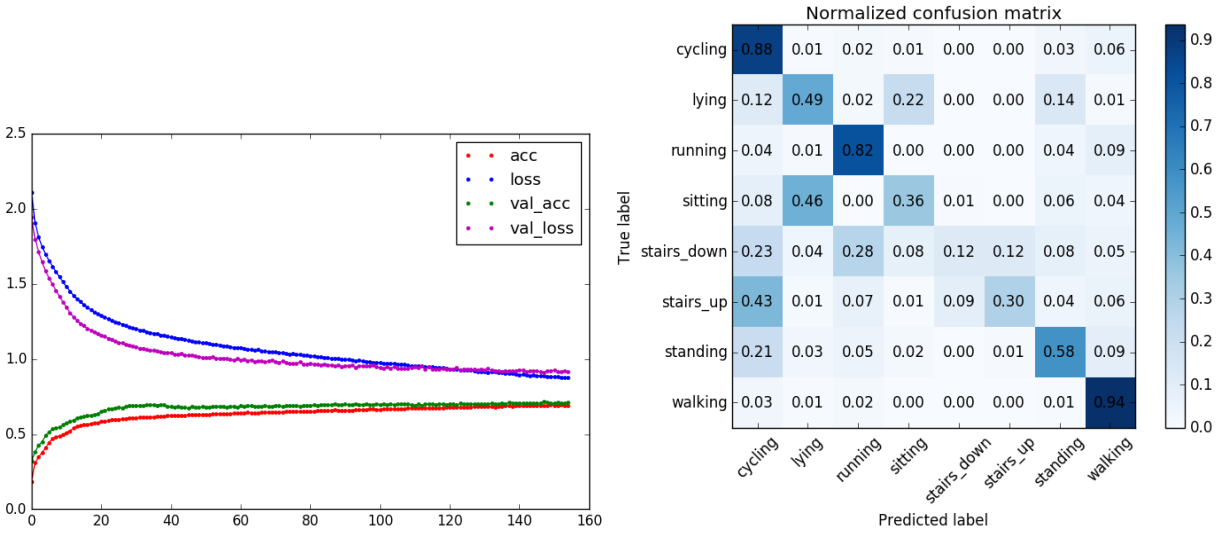

Figure A.26: dataset: TUM, sensor=L, w=64, neurons=512

Table A.26: dataset: TUM, sensor=L, w=64, neurons=512

	mean	median	std	max	min
$F1_{micro}$	0.599	0.588	0.082	0.718	0.503
$F1_{macro}$	0.435	0.428	0.097	0.577	0.305
$F1_{weighted}$	0.579	0.575	0.086	0.702	0.463

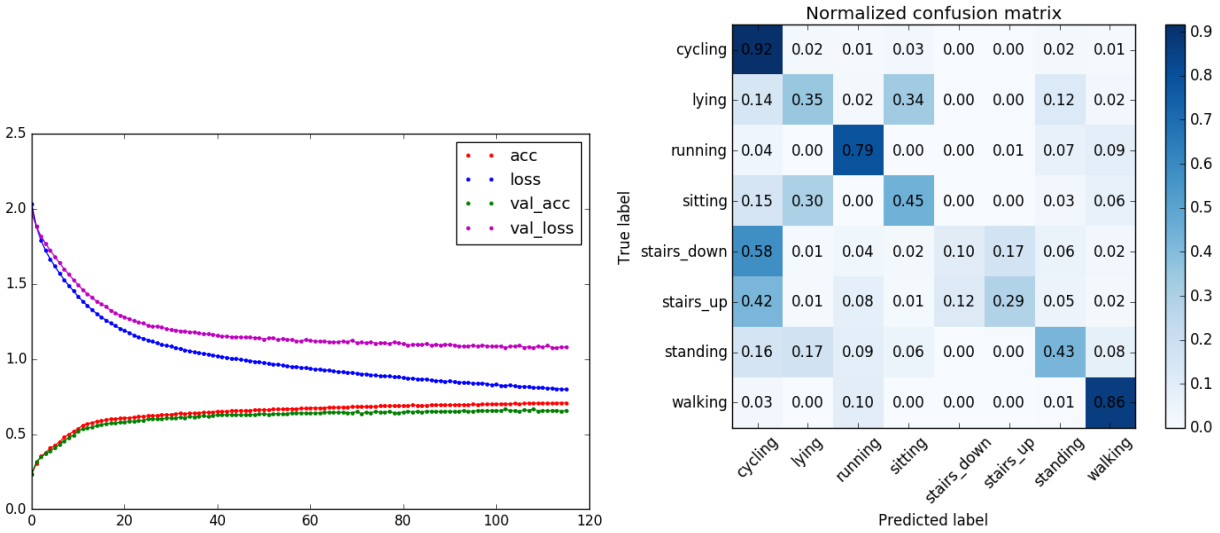

Figure A.27: dataset: TUM, sensor=T, w=64, neurons=512

Table A.27: dataset: TUM, sensor=T, w=64, neurons=512

	mean	median	std	max	min
$F1_{micro}$	0.641	0.644	0.021	0.666	0.609
$F1_{macro}$	0.482	0.472	0.031	0.530	0.453
$F1_{weighted}$	0.617	0.618	0.022	0.647	0.586

List of Figures

List of Tables

Bibliography

[1] Bm25 the next generation of lucene relevance. http://opensourceconnections.com/blog/2015/10/16/bm25-the-next-generation-of-lucene-relevation/. Accessed: 2017-05-11.

[2] Cambrige analytica mission statement. https://cambridgeanalytica.org/. Accessed: 2017-06-18.

[3] Die mama von opera analysiert das web. https://www.heise.de/newsticker/meldung/Die-MAMA-von-Opera-analysiert-das-Web-211784.html. Accessed: 2017-06-18.

[4] Free-falling in milwaukee: A close-up on one city's middle-class decline. https://www.theatlantic.com/business/archive/2011/12/free-falling-in-milwaukee-a-close-up-on-one-citys-middle-class-decline/250100/. Accessed: 2017-07-6.

[5] Google's 200 ranking factors: The complete list. http://backlinko.com/google-ranking-factors. Accessed: 2017-06-18.

[6] The great british brexit robbery: how our democracy was hijacked. https://www.theguardian.com/technology/2017/may/07/the-great-british-brexit-robbery-hijacked-democracy. Accessed: 2017-06-18.

[7] Internet world users by language. http://www.internetworldstats.com/stats7.htm. Accessed: 2017-06-18.

[8] Lawsuits in the mobile business (redux). http://news.designlanguage.com/post/1473307539. Accessed: 2017-05-11.

[9] Merriam webster - def. indicator. https://www.merriam-webster.com/dictionary/indicator. Accessed: 2017-07-18.

[10] Organisation for economic co-operation and development. http://stats.oecd.org/mei/default.asp?rev=4. Accessed: 2017-07-18.

[11] Pew research center. http://www.pewglobal.org/database/. Accessed: 2017-07-18.

[12] Polity iv dataset. http://www.systemicpeace.org/polityproject.html. Accessed: 2017-07-18.

[13] The size of the world wide web (the internet). http://www.worldwidewebsize.com/. Accessed: 2017-06-18.

[14] United nations statistics division. https://unstats.un.org/home/. Accessed: 2017-07-18.

[15] Uspto performance and accountability report 2016. https://www.uspto.gov/sites/default/files/documents/USPTOFY16PAR.pdf. Accessed: 2017-06-18.

[16] World bank data catalog. https://datahelpdesk.worldbank.org/knowledgebase/articles/902049-data-catalog-api. Accessed: 2017-07-18.

[17] World development indicators 2017. http://data.worldbank.org/data-catalog/world-development-indicators. Accessed: 2017-07-18.

[18] World happiness report 2017. http://worldhappiness.report/ed/2017/. Accessed: 2017-07-18.

[19] Z. J. Acs, L. Anselin, and A. Varga. Patents and innovation counts as measures of regional production of new knowledge. *Research policy*, 31(7):1069–1085, 2002.

[20] M. Aizerman, E. M. Braverman, and L. Rozonoer. Theoretical foundations of potential function method in pattern recognition. *Automation and Remote Control*, 25:917–936, 1964.

[21] L. AlSumait, D. Barbará, and C. Domeniconi. On-line lda: Adaptive topic models for mining text streams with applications to topic detection and tracking. In *2008 Eighth IEEE International Conference on Data Mining*, pages 3–12. IEEE Computer Society, Dec 2008.

[22] K. R. Andrews et al. Concept of corporate strategy. 1971.

[23] N. Argyres and A. M. McGahan. An interview with michael porter. *The Academy of Management Executive*, 16(2):43–52, 2002.

[24] K. Arto, B. Hu, A. Leopold, S. Meyer-Nieberg, G. Mihelcic, J. Stutzki, and T. Tepel. Modellbasierende früherkennung politischer krisen: Prototyp einer serviceorientierten plattform. *Multikonferenz der Wirtschaftsinformatik, Paderborn*, 26:28–44, 2014.

[25] B. R. Barringer and A. C. Bluedorn. The relationship between corporate entrepreneurship and strategic management. *Strategic Management Journal*, pages 421–444, 1999.

[26] J. Beel, B. Gipp, S. Langer, and C. Breitinger. Research-paper recommender systems: a literature survey. *International Journal on Digital Libraries*, 17(4):305–338, 2016.

[27] Y. Bengio. Practical recommendations for gradient-based training of deep architectures. In *Neural Networks: Tricks of the Trade*, pages 437–478. Springer, 2012.

[28] Y. Bengio, R. Ducharme, P. Vincent, and C. Jauvin. A neural probabilistic language model. *Journal of machine learning research*, 3(Feb):1137–1155, 2003.

[29] Y. Bengio, R. Ducharme, P. Vincent, and C. Jauvin. A neural probabilistic language model. *Journal of Machine Learning Research*, 3(Feb):1137–1155, 2003.

[30] Y. Bengio, P. Simard, and P. Frasconi. Learning long-term dependencies with gradient descent is difficult. *IEEE Transactions on Neural Networks*, 5(2):157–166, 1994.

[31] J. Benhardus and J. Kalita. Streaming trend detection in twitter. *International Journal of Web Based Communities*, 9(1):122–139, 2013.

[32] J. Bentley. [a]multidimensional binary search trees used for associative searching, [o] comm, 1975.

[33] K. Benzineb and J. Guyot. Automated patent classification. In *Current challenges in patent information retrieval*, pages 239–261. Springer, 2011.

[34] J. Bergstra and Y. Bengio. Random search for hyper-parameter optimization. *Journal of Machine Learning Research*, 13(Feb):281–305, 2012.

[35] T. Bernecker, F. Graf, H.-P. Kriegel, C. Moennig, D. Dill, and C. Tuermer. Activity recognition on 3d accelerometer data (technical report). 2012.

[36] K. Beuls, B. Pflugfelder, and A. Hanbury. Comparative analysis of balanced winnow and svm in large scale patent categorization. In *Proceedings of Dutch-Belgian Information Retrieval Workshop (DIR)*, pages 8–15, 2010.

[37] B. Bird. Implementing entrepreneurial ideas: The case for intention. *Academy of management Review*, 13(3):442–453, 1988.

[38] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3(Jan):993–1022, 2003.

[39] J. N. Britton. Network structure of an industrial cluster: electronics in toronto. *Environment and Planning A*, 35(6):983–1006, 2003.

[40] M. Butter, F. Brandes, M. Keenan, R. Popper, and R. Popper. How are foresight methods selected? *foresight*, 10(6):62–89, 2008.

[41] R. Caruana, S. Lawrence, and C. L. Giles. Overfitting in neural nets: Backpropagation, conjugate gradient, and early stopping. In *Advances in neural information processing systems*, pages 402–408, 2001.

[42] A. D. Chandler. *Strategy and structure: Chapters in the history of the industrial enterprise*, volume 120. MIT press, 1990.

[43] S. Chari. The agrarian origins of the knitwear industrial cluster in tiruppur, india. *World development*, 28(3):579–599, 2000.

[44] J. A. Cheibub, J. Gandhi, and J. R. Vreeland. Democracy and dictatorship revisited. *Public choice*, 143(1-2):67–101, 2010.

[45] K. Cho, B. Van Merriënboer, D. Bahdanau, and Y. Bengio. On the properties of neural machine translation: Encoder-decoder approaches. *arXiv preprint arXiv:1409.1259*, 2014.

[46] H. Choi and H. Varian. Predicting the present with google trends. *Economic Record*, 88(s1):2–9, 2012.

[47] F. Chollet. keras. `https://github.com/fchollet/keras`, 2015.

[48] C. W. Choo. *Information management for the intelligent organization: the art of scanning the environment*. Information Today, Inc., 2002.

[49] C. W. Choo and E. Auster. Environmental scanning: Acquisition and use of information by managers. *Annual Review of information Science and technology (Arist)*, 28:279–314, 1993.

[50] Z. Chu, S. Gianvecchio, H. Wang, and S. Jajodia. Who is tweeting on twitter: human, bot, or cyborg? In *Proceedings of the 26th annual computer security applications conference*, pages 21–30. ACM, 2010.

[51] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014.

[52] C. Cortes and V. Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

[53] I. Costantea, R. I. Boţ, and G. Wanka. *Patent document classification based on mutual information feature selection*. Technische Universität Chemnitz. Fakultät für Mathematik, 2004.

[54] A. Cutler, D. R. Cutler, and J. R. Stevens. Random forests. In *Ensemble machine learning*, pages 157–175. Springer, 2012.

[55] J. E. Cutting and L. T. Kozlowski. Recognizing friends by their walk: Gait perception without familiarity cues. *Bulletin of the psychonomic society*, 9(5):353–356, 1977.

[56] I. P. Cvijikj and F. Michahelles. Monitoring trends on facebook. In *Dependable, Autonomic and Secure Computing (DASC), 2011 IEEE Ninth International Conference on*, pages 895–902. IEEE, 2011.

[57] R. Díaz-Uriarte and S. A. De Andres. Gene selection and classification of microarray data using random forest. *BMC bioinformatics*, 7(1):3, 2006.

[58] M. Edel and E. Köppe. Binarized-blstm-rnn based human activity recognition. In *Indoor Positioning and Indoor Navigation (IPIN), 2016 International Conference on*, pages 1–7. IEEE, 2016.

[59] L. Egghe. Untangling herdan's law and heaps' law: Mathematical and informetric arguments. *Journal of the American Society for Information Science and Technology*, 58(5):702–709, 2007.

[60] D. S. Elenkov. Strategic uncertainty and environmental scanning: The case for institutional influences on scanning behavior. *Strategic management journal*, pages 287–302, 1997.

[61] L. Elmer. Hierarchical paragraph vectors. Master's thesis, Eidgenössische Technische Hochschule Zürich, 2015.

[62] C. J. Fall, A. Törcsvári, K. Benzineb, and G. Karetka. Automated categorization in the international patent classification. *SIGIR Forum*, 37(1):10–25, 2003.

[63] A. Fujii. Enhancing patent retrieval by citation analysis. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 793–794. ACM, 2007.

[64] A. Fujii, M. Iwayama, and N. Kando. Overview of the patent retrieval task at the ntcir-6 workshop. In *NTCIR*, 2007.

[65] P. Fung. Extracting key terms from chinese and japanese texts. *Computer Processing of Oriental Languages*, 12(1):99–121, 1998.

[66] Y. Gal. A theoretically grounded application of dropout in recurrent neural networks. *arXiv preprint arXiv:1512.05287*, 2015.

[67] R. C. S. L. L. Giles. Overfitting in neural nets: Backpropagation, conjugate gradient, and early stopping. In *Advances in Neural Information Processing Systems 13: Proceedings of the 2000 Conference*, volume 13, page 402. MIT Press, 2001.

[68] X. Glorot, A. Bordes, and Y. Bengio. Deep sparse rectifier neural networks. In *Aistats*, volume 15, page 275, 2011.

[69] J. A. Goldstone, R. H. Bates, D. L. Epstein, T. R. Gurr, M. B. Lustik, M. G. Marshall, J. Ulfelder, and M. Woodward. A global model for forecasting political instability. *American Journal of Political Science*, 54(1):190–208, 2010.

[70] J. A. Goldstone, T. R. Gurr, B. Harff, M. A. Levy, M. G. Marshall, R. H. Bates, D. L. Epstein, C. H. Kahl, P. T. Surko, J. C. Ulfelder, et al. State failure task force report: Phase iii findings. *McLean, VA: Science Applications International Corporation*, 30, 2000.

[71] A. Graves. *Supervised sequence labeling*. Springer, 2012.

[72] A. Graves, A.-r. Mohamed, and G. Hinton. Speech recognition with deep recurrent neural networks. In *Acoustics, speech and signal processing (icassp), 2013 ieee international conference on*, pages 6645–6649. IEEE, 2013.

[73] Z. Griliches. Patent statistics as economic indicators: a survey. Technical report, National Bureau of Economic Research, 1990.

[74] N. Y. Hammerla, S. Halloran, and T. Ploetz. Deep, convolutional, and recurrent models for human activity recognition using wearables. *arXiv preprint arXiv:1604.08880*, 2016.

[75] D. M. Hawkins. The problem of overfitting. *Journal of chemical information and computer sciences*, 44(1):1–12, 2004.

[76] K. He, X. Zhang, S. Ren, and J. Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1026–1034, 2015.

[77] K. Hechenbichler and K. Schliep. Weighted k-nearest-neighbor techniques and ordinal classification. 2004.

[78] G. E. Hinton and R. R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *science*, 313(5786):504–507, 2006.

[79] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[80] C. Hutter and E. Weber. Constructing a new leading indicator for unemployment from a survey among german employment agencies. *Applied Economics*, 47(33):3540–3558, 2015.

[81] R. J. Hyndman, Y. Khandakar, et al. *Automatic time series for forecasting: the forecast package for R*. Number 6/07. Monash University, Department of Econometrics and Business Statistics, 2007.

[82] Y.-S. Jeong, M. K. Jeong, and O. A. Omitaomu. Weighted dynamic time warping for time series classification. *Pattern Recognition*, 44(9):2231–2240, 2011.

[83] R. S. Kaplan and D. P. Norton. The balanced scorecard: measures that drive performance. *Harvard Business Review*, 1992.

[84] R. S. Kaplan and D. P. Norton. *The balanced scorecard: translating strategy into action*. Harvard Business Press, 1996.

[85] M. Karlsson. The immediacy of online news, the visibility of journalistic processes and a restructuring of journalistic authority. *Journalism*, 12(3):279–295, 2011.

[86] A. M. Khan, Y.-K. Lee, S. Y. Lee, and T.-S. Kim. A triaxial accelerometer-based physical-activity recognition via augmented-signal features and a hierarchical recognizer. *IEEE transactions on information technology in biomedicine*, 14(5):1166–1172, 2010.

[87] E. Kim, S. Helal, and D. Cook. Human activity recognition and pattern discovery. *IEEE Pervasive Computing*, 9(1), 2010.

[88] C. H. A. Koster, M. Seutter, and J. Beney. Classifying patent applications with winnow.

[89] J. R. Kwapisz, G. M. Weiss, and S. A. Moore. Activity recognition using cell phone accelerometers. *ACM SigKDD Explorations Newsletter*, 12(2):74–82, 2011.

[90] K.-K. Lai and S.-J. Wu. Using the patent co-citation approach to establish a new patent classification system. *Information processing & management*, 41(2):313–330, 2005.

[91] O. D. Lara and M. A. Labrador. A survey on human activity recognition using wearable sensors. *IEEE Communications Surveys & Tutorials*, 15(3):1192–1209, 2013.

[92] L. S. Larkey. A patent search and classification system. In *Proceedings of the Fourth ACM conference on Digital Libraries, August 11-14, 1999, Berkeley, CA, USA*, pages 179–187. ACM, 1999.

[93] P. R. Lawrence and J. W. Lorsch. Differentiation and integration in complex organizations. *Administrative science quarterly*, pages 1–47, 1967.

[94] G. Lawrie and I. Cobbold. Third-generation balanced scorecard: evolution of an effective strategic control tool. *International Journal of Productivity and Performance Management*, 53(7):611–623, 2004.

[95] Q. V. Le and T. Mikolov. Distributed representations of sentences and documents. In *ICML*, volume 14, pages 1188–1196, 2014.

[96] Y. Li, C. Fan, Y. Li, and Q. Wu. Improving deep neural network with multiple parametric exponential linear units. *arXiv preprint arXiv:1606.00305*, 2016.

[97] N. Littlestone. Learning quickly when irrelevant attributes abound: A new linear-threshold algorithm. *Machine Learning*, 2(4):285–318, 1987.

[98] J. B. Lovins. Development of a stemming algorithm. *Mech. Translat. & Comp. Linguistics*, 11(1-2):22–31, 1968.

[99] L. v. d. Maaten and G. Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(Nov):2579–2605, 2008.

[100] C. D. Manning, P. Raghavan, H. Schütze, et al. *Introduction to information retrieval*, volume 1. Cambridge university press Cambridge, 2008.

[101] J. Mantyjarvi, M. Lindholm, E. Vildjiounaite, S.-M. Makela, and H. Ailisto. Identifying users of portable devices from gait pattern with accelerometers. In *Proceedings.(ICASSP'05). IEEE International Conference on Acoustics, Speech, and Signal Processing, 2005.*, volume 2, pages ii–973. IEEE, 2005.

[102] D. P.-U. MARKENAMT. Guidelines for the classification of patent and utility model applications. 2014.

[103] F. Martínez-Álvarez, A. Troncoso, G. Asencio-Cortés, and J. C. Riquelme. A survey on data mining techniques applied to electricity-related time series forecasting. *Energies*, 8(11):13162–13193, 2015.

[104] M. Mathioudakis and N. Koudas. Twittermonitor: trend detection over the twitter stream. In *Proceedings of the 2010 ACM SIGMOD International Conference on Management of data*, pages 1155–1158. ACM, 2010.

[105] F. McCown and M. L. Nelson. Search engines and their public interfaces: which apis are the most synchronized? In *Proceedings of the 16th international conference on World Wide Web*, pages 1197–1198. ACM, 2007.

[106] S. Mika, G. Ratsch, J. Weston, B. Scholkopf, and K.-R. Mullers. Fisher discriminant analysis with kernels. In *Neural Networks for Signal Processing IX, 1999. Proceedings of the 1999 IEEE Signal Processing Society Workshop.*, pages 41–48. IEEE, 1999.

[107] T. Mikolov, K. Chen, G. Corrado, and J. Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[108] H. Mintzberg. The strategy concept i: Five ps for strategy. *California management review*, 30(1):11–24, 1987.

[109] J. L. Morrison. Environmental scanning. *A primer for new institutional researchers*, pages 86–99, 1992.

[110] M. Nguyen, L. Fan, and C. Shahabi. Activity recognition using wrist-worn sensors for human performance evaluation. In *2015 IEEE International Conference on Data Mining Workshop (ICDMW)*, pages 164–169. IEEE, 2015.

[111] J. Nishimura and H. Okamuro. R&d productivity and the organization of cluster policy: An empirical evaluation of the industrial cluster project in japan. *The Journal of Technology Transfer*, 36(2):117–144, 2011.

[112] F. J. Ordóñez and D. Roggen. Deep convolutional and lstm recurrent neural networks for multimodal wearable activity recognition. *Sensors*, 16(1):115, 2016.

[113] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 311–318. Association for Computational Linguistics, 2002.

[114] U. S. Patent and T. Office. Aia cpc xml documentation, 2013.

[115] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[116] A. J. Perotte, F. Wood, N. Elhadad, and N. Bartlett. Hierarchically supervised latent dirichlet allocation. In *Advances in Neural Information Processing Systems*, pages 2609–2617, 2011.

[117] M. E. Porter. *Clusters and the new economics of competition*, volume 76. Harvard Business Review Boston, 1998.

[118] R. Řehůřek and P. Sojka. Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, pages 45–50, Valletta, Malta, May 2010. ELRA.

[119] G. Richter and A. MacFarlane. The impact of metadata on the accuracy of automated patent classification. *World Patent Information*, 27(1):13–26, 2005.

[120] S. E. Robertson, S. Walker, S. Jones, M. M. Hancock-Beaulieu, M. Gatford, et al. Okapi at trec-3. *NIST SPECIAL PUBLICATION SP*, 109:109, 1995.

[121] D. Roggen, A. Calatroni, M. Rossi, T. Holleczek, K. Förster, G. Tröster, P. Lukowicz, D. Bannach, G. Pirkl, A. Ferscha, et al. Collecting complex activity datasets in highly rich networked sensor environments. In *Networked Sensing Systems (INSS), 2010 Seventh International Conference on*, pages 233–240. IEEE, 2010.

[122] T. N. Sainath, O. Vinyals, A. Senior, and H. Sak. Convolutional, long short-term memory, fully connected deep neural networks. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4580–4584. IEEE, 2015.

[123] Y. Sano, H. Takayasu, and M. Takayasu. Zipf's law and heaps law can predict the size of potential words. *Progress of Theoretical Physics Supplement*, 194:202–209, 2012.

[124] N. I. Sapankevych and R. Sankar. Time series prediction using support vector machines: a survey. *IEEE Computational Intelligence Magazine*, 4(2), 2009.

[125] B. Scholkopf and A. J. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press, Cambridge, MA, USA, 2001.

[126] E. Schubert, M. Weiler, and H.-P. Kriegel. Signitrend: scalable detection of emerging topics in textual streams by hashed significance thresholds. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 871–880. ACM, 2014.

[127] H. L. Seal. *The historical development of the Gauss linear model*. Yale University, 1968.

[128] E. Seglem, F. Borutta, J. Stutzki, E. Faerman, A. Züfle, and M. Schubert. On privacy in spatio-temporal data: User identification using microblog data. In *Submitted to be published in Proceedings of the 15th International Symphosium on Spatial & Temporal Databases*, 2017.

[129] C. Silva and B. Ribeiro. The importance of stop word removal on recall values in text categorization. In *Neural Networks, 2003. Proceedings of the International Joint Conference on*, volume 3, pages 1661–1666. IEEE, 2003.

[130] D. F. Silva and G. E. Batista. Speeding up all-pairwise dynamic time warping matrix calculation. In *Proceedings of the 2016 SIAM International Conference on Data Mining*, pages 837–845. SIAM, 2016.

[131] N. Srivastava, G. E. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

[132] J. H. Stock and M. W. Watson. How did leading indicator forecasts perform during the 2001 recession? 2003.

[133] J. Stutzki. Multilingual trend detection in the web. In *OASIcs-OpenAccess Series in Informatics*, volume 37, pages 23–32. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2014.

[134] J. Stutzki and M. Schubert. Geodata supported classification of patent applications. In *Proceedings of the Third International ACM SIGMOD Workshop on Managing and Mining Enriched Geo-Spatial Data*, page 4. ACM, 2016.

[135] Y. Sun, Z. Zhuang, and C. L. Giles. A large-scale study of robots. txt. In *Proceedings of the 16th international conference on World Wide Web*, pages 1123–1124. ACM, 2007.

[136] T. Tieleman and G. Hinton. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural Networks for Machine Learning*, 4(2), 2012.

[137] S. C. Wangberg, H. K. Andreassen, H.-U. Prokosch, S. M. V. Santana, T. Sørensen, and C. E. Chronaki. Relations between internet use, socio-economic status (ses), social support and subjective health. *Health promotion international*, 23(1):70–77, 2008.

[138] D. Wu and P. Fung. Improving chinese tokenization with linguistic filters on statistical lexical acquisition. In *Proceedings of the fourth conference on Applied natural language processing*, pages 180–181. Association for Computational Linguistics, 1994.

[139] B. Xu, N. Wang, T. Chen, and M. Li. Empirical evaluation of rectified activations in convolutional network. *arXiv preprint arXiv:1505.00853*, 2015.

[140] M. D. Zeiler and R. Fergus. Visualizing and understanding convolutional networks. In *European Conference on Computer Vision*, pages 818–833. Springer, 2014.