Error propagation

Lê, Minh Ngoc

2021

document version
Publisher's PDF, also known as Version of record

The research reported in this thesis has been funded by the Netherlands Organization for Scientific Research (NWO) via the Spinoza fund, which was granted to prof.dr. Piek Vossen in 2013.

VRIJE UNIVERSITEIT

ERROR PROPAGATION

ACADEMISCH PROEFSCHRIFT

ter verkrijging van de graad Doctor of Philosophy aan
de Vrije Universiteit Amsterdam,
op gezag van de rector magnificus
prof.dr. V. Subramaniam,
in het openbaar te verdedigen
ten overstaan van de promotiecommissie
van de Faculteit der Geesteswetenschappen
op vrijdag 28 mei 2021 om 13.45 uur
in de online bijeenkomst van de universiteit,
De Boelelaan 1105

door
Minh Ngọc Lê
geboren te Hanoi, Vietnam

promotor:	prof.dr. P.T.J.M. Vossen
copromotor:	dr. A.S. Fokkens

reading committee:

- prof.dr. Antal van den Bosch
- prof.dr. Martine Coene
- prof.dr. Malvina Nissim
- prof.dr. Stephan Oepen
- dr. German Rigau

For my grandfather Biên.

Contents

Foreword 1

1 Introduction 5

1.1 Research Questions 7

1.2 Contributions 8

1.3 The Organization of this Thesis 9

2 Error Propagation 13

2.1 Types of Error Propagation 14

2.1.1 Error Propagation Within and Across Tasks 15

2.1.2 Hard and Soft Error Propagation 16

2.1.3 Error Propagation in Discrete and Continuous Representations 18

2.2 Responding to Errors 20

2.3 Existing Approaches 21

2.3.1 Distributed Representations (S1) 21

2.3.2 Easy-first (S2) 23

2.3.3 Joint Processing (S2/S3) 23

2.3.4 Global Inference (S3) 27

2.3.5 End-to-end Learning (S3) 27

2.3.6 Training for Shorter Transition Sequences (S3) . . 28

2.3.7 Training on Automatic Annotations (S4) 29

2.3.8 Multiple Outputs (S4) 29

2.3.9 Error Detection and Correction (S5) 30

2.3.10 Reranking (S5) 31

2.4 The Importance of World Knowledge 31

2.5 Overview of Papers 34

2.5.1 Errors in Distributed Representations (Q1, S1) . . . 34

2.5.2 Conceptualized Word Embeddings and (Q1/Q2, S1) 36

2.5.3 Reinforced Transition-based Dependency Parsing (Q2, S4) 37

2.5.4 The Value of Context in Coreference Resolution (Q3, S5) 38

2.5.5 Modeling Selectional Preference for Implicit Semantic Role Labeling (Q3, S5) 39

3 Mitigation 47

3.1 Taxonomy Beats Corpus in Similarity Identification, but Does It Matter? 53

3.2 A Deep Dive into Word Sense Disambiguation with LSTM 71

4 Adaptation 95

4.1 Tackling Error Propagation through Reinforcement Learning: A Case of Greedy Dependency Parsing 98

5 Correction 121

5.1 An Input Ablation Analysis of Coreference Resolution 126

5.2 Neural Models of Selectional Preferences for Implicit Semantic Role Labeling 152

6 Conclusions 167

6.1 What We Have Learned 168

6.1.1 Types of Error Propagation 168

6.1.2 Case Studies: Errors in Word Embeddings 170

6.1.3 Existing Remedies 171
6.1.4 Proposed Solutions 172
6.2 Challenges 173
6.3 Future Research 176

Bibliography 178

Index 223

Foreword

> "Would you tell me, please, which way I ought to go from here?"
> "That depends a good deal on where you want to get to," said the Cat.
> "I don't much care where—" said Alice.
> "Then it doesn't matter which way you go," said the Cat.
> "—so long as I get somewhere," Alice added as an explanation.
> "Oh, you're sure to do that," said the Cat, "if you only walk long enough."
>
> *Alice in Wonderland (Lewis Carroll)*

When I started my PhD seven years ago, Natural Language Processing (NLP) was very different from what it is now. Pipelines still ruled leader boards, end-to-end was the new kid on the block, and Bert was just a character in Sesame Street. Despite their popularity, pipelines were known to have a big weakness: error propagation, i.e. the chain reaction of errors causing more errors after each step of processing. Having just finished two years of cognitive sciences for my master's degree, I was convinced that NLP needs non-pipeline architectures – not just because the brain does not resemble a directed acyclic graph of modules, but also because feedback loops allow it to use rich, high-level information to detect and correct errors and hence, enable accurate and robust language comprehension.

In an unexpected way, recent developments have vindicated my belief in non-pipeline architectures: pipelines have given way

to deep learning end-to-end systems that work directly on raw text. In syntactic parsing, semantic role labeling, natural language inference, sentiment analysis, and more, records previously held by pipelines were surpassed by significant margins by end-to-end systems, especially when combined with pretrained models such as BERT. The meaning of the word *pipeline* itself has drifted: in Spacy, a popular modern NLP library, a pipeline is a collection of modules, without any dependency, that are run sequentially out of convenience.[1]

Yet, the problem of error propagation persists in two ways. First, end-to-end architectures themselves involve the step-by-step construction of representations whose distributed nature makes them less error-prone but certainly not error-free. Second, systems that make sequential decisions, as commonly found in syntactic parsing and coreference resolution, are always vulnerable to cascading errors regardless of what machine learning model is employed. Shortening the decision chain is good to reduce error propagation but to solve it, I believe, we need to capture progressively richer knowledge to enable error detection and correction. Around the time I started, Cambria and White (2014) envisioned a new wave of semantics-focused NLP that dwells in "information commonly associated with real-world objects, actions, events, and people" including "common sense and common knowledge". They posit that it will take more than 30 years to achieve this milestone.

If only I could say that tardy progress and unexpected developments were new to me. Since following Curiosity to NLP-land, between three and five seemingly brilliant ideas have turned out wrong for every paper I published. If the chapters in this thesis appear only loosely connected to each other, it is because I had a clearer idea about where to go away *from* than where to go *to*.

[1] See question "Does the order of pipeline components matter?" at `https://spacy.io/usage/processing-pipelines`

As will become apparent in the next chapters, error propagation is pervasive in the vast and ever-changing landscape of NLP and my desire to find not just a solution but *the* solution, not just remedies but the cure, often leads me to paths too long and windy. However difficult it was, I am grateful for the journey of the last seven years. It has certainly gotten me *somewhere* and, more importantly, made me a better researcher.

I could not have finished this PhD without the help of a lot of people. First and foremost, I want to thank my supervisors. Piek and Antske's generosity and support have allowed me to pursue the research directions closest to my heart. They are the best teachers I ever had: knowledgeable, articulate, dedicated, and sympathetic. From the very beginning of the project, they have become big friends that I counted on when I was most vulnerable. Other big friends without whom I would not have started the PhD in the first place are my parents. Since I was a small boy, they have always provided me with a stable launchpad for my endeavors and granted me the freedom to follow my curiosity. I was also fortunate to find a soulmate in Huong Pham who has not only walked by my side for the last fifteen years but also challenged me to move forward.

Through the course of the PhD, my colleagues have taught me a great deal about research and collaboration. Marten Postma, whose great collaboration made possible one of the papers in this book, has taught me intricate details of word sense disambiguation and inspired me to become more organized and professional. During the same project, Jacopo Urbani has given me valuable tips on experimentation and writing which I am still using today. I have shared countless intellectually stimulating discussions with my CLTL colleagues who are also unfailingly kind and generous: Rubén Izquierdo Beviá, Tommaso Caselli, Filip Ilievski, Isa Maks, Emiel van Miltenburg, Pia Sommerauer, Chantal van Son, Roxane Segers, Roser Morante Vallejo, and Hennie van der Vliet.

My gratitude also goes to my Elsevier colleagues: Finne Boonen, Chinh Bui, Maya Hristakeva, Deep Kayal, Elaine van Ommen Kloeke, Tuhin Mitra, and Reinder Verlinde, who have broadened my mind to the real-world opportunities and challenges of machine learning, but also to the quirks of various cultures, medieval fashion and culinary, European geography, Indian gods, and more. Our joyful chats have proven therapeutic whenever I was in despair.

The last words of this chapter are dedicated to my beloved maternal grandfather who has passed away before I could finish this PhD. He was a very special grandpa, not least because he allowed 6-year-old me to play with hammers and metal scraps, and told 15-year-old me to always work on two projects at the same time. Many years ago, somewhere between his overflowed bookshelves and his workshop full of tools, materials, and machine parts, he has kindled in me a love of science, engineering, and invention. Now that his flame no longer burns, it is my task to carry on our love and share it with others.

Utrecht, 2021

1. Introduction

Human language and the cognitive processes that enable it are complex, multifaceted phenomena. As you read this thesis, your brain will be busy performing a lot of subconscious calculations to figure out the meaning of words, how they are organized into structures, how they refer to entities in the world, etc. As a scientific endeavor to process and understand human language with the mechanistic processes of the computer, Natural Language Processing (NLP) necessarily deals with this complexity.

Over the decades that NLP exists, divide-and-conquer has been the main strategy. A recent survey has listed more than 40 main tasks and many more exist in the literature.[1] A traditional NLP system is constructed from modules, each performing one of those tasks, assembled in a "pipeline" structure where the output of one becomes the input of another. A fundamental problem with this approach is error propagation. Like a game of "Chinese whispers", as one module passes its interpretation to the next, it also passes on mistakes. The errors pile up until, at the end of the line, one can hardly understand the original message. Until today NLP still lacks a way of integrating and cross-checking information, therefore layers of interpretation produced by error-prone modules are simply juxtaposed and fed to the next module as-is.

The chain needs not be long for a snowball effect to take place. It has been reported that half of the errors in one step of shallow semantic parsing is due to errors in syntactic parsing

[1] See `https://github.com/Kyubyong/nlp_tasks`.

(Pradhan et al., 2005). It is even more problematic when modules are omitted altogether because the rate of errors is deemed too high. This is the case for word sense disambiguation, a task that tries to pinpoint a particular meaning of a word given the context it appears in. Even though it is intuitive that one should differentiate between different senses of a word, practical systems of syntactic and semantic analysis do not make use of word sense analysis because it is too noisy. NLP itself, in turn, can become a module that is omitted from downstream applications due to quality concerns. Digital humanities literature reports cases where researchers resort to manual annotation (Strange et al., 2014) and, in information retrieval, researchers have found that applying NLP technologies brings improvements "too small" or "discouraging" (Baeza-Yates, 2004). Even when an NLP system performs well on canonical test sets, it is crippled by errors in the digitization process (Jiang, 2008; Plank, 2016; Walker et al., 2010; Kumar et al., 2015), limiting its scope of application.

Recently, advances in deep learning have enabled alternatives to pipelines in the form of end-to-end architectures (e.g. Vaswani et al. 2017; Raffel et al. 2019; Li et al. 2020b). In this paradigm, tasks are performed by independent modules that work directly on raw text. Intermediate results are no longer discrete predictions of NLP modules but real-valued vectors generated by deep neural networks trained on vast amounts of data. Although end-to-end models reduce error propagation to some extent, they have some of the same vulnerabilities as their pipeline counterparts. First, the new systems suffer from the same types of error propagation whenever they attempt to make sequential decisions. This has been demonstrated in language modeling (Ranzato et al., 2016) and coreference resolution (Fei et al., 2019). Second, error propagation can also occur along the layers of deep neural networks. Though harder to quantify than traditional systems, special cases have been described such as reconstruction errors

in pruned networks (Dong et al., 2017), quantization error in quantized networks (Wang et al., 2018), and the perturbation of latent representations in regular networks in response to adversarial perturbation (e.g. Cissé et al. 2017). The above observations suggest that being end-to-end is not a silver bullet against error propagation.

The high prevalence and serious consequences of error propagation justify a systematic study dedicated to the phenomenon. To the best of my knowledge, such a study has not been done before. Although the topic is too expansive to be covered by a single doctoral thesis, I will try here to cover as much ground as possible in a systematic way. It is my hope that the synthesis of the problem and its remedies into a self-contained document will serve as a stepping stone and a source of motivation for future research.

1.1 Research Questions

An informal definition of error propagation is when "one error leads to another". This definition captures human intuition well but leaves many questions unanswered: *What is an error? What concept of causation applies in our context? What are the types of errors and how do their effects differ?* Seeking the answers to these questions, I discovered that the seemingly simple concept of error propagation is a complex of various phenomena. Therefore, the first research question this thesis will address is:

Q1: What types of error propagation are there?

With a clear understanding of the problem, we will investigate how to reduce the chance of error propagation with the ultimate aim of improving the performance of NLP systems. The second research question, therefore, is:

Q2: How can we avoid error propagation?

Given the boundless creativity of language and the vast complexity of the world it refers to, some amount of errors might be inevitable. There is also a limit to reducing error propagation as we cannot make everything independent of each other. It is likely that we will need a way to actively look out for errors and fix them as they arrive. Effectively fixing errors would curb error propagation and bring NLP closer to human language ability. Hence, the third research question is:

Q3: How can we detect and correct errors?

1.2 Contributions

As part of my Ph.D., I have published four first-authored papers at medium- to high-ranking conferences, with one more under review (Table 1.1, p. 11). The papers are accompanied by publicly available source code and datasets. The contributions of the thesis are the following:

1. I have **reviewed** the literature around error propagation. Reports of error propagation in different tasks are tabularized; approaches scattered in the literature are categorized, and their relationship, strength, and weakness discussed.
2. Remedies for error propagation were **described** and **categorized** into five strategies and three broad categories. The scope of their effects were discussed.
3. NLP modules were **evaluated** with respect to their contributions to error propagation. Word embeddings models, the first component to be executed in many pipelines and neural networks, were evaluated because the quality of their output is important for later processes. At the other end of pipelines, the use of contextual information in coreference resolution systems was examined. To be robust against errors passed on by upstream processes, it is essential for a coreference resolution system to integrate multiple views

of the text. I found that this is not the case for various high-performing systems. For both investigations, new evaluation methods were proposed.

4. I have proposed **models** for a type of world knowledge potentially useful in error detection and correction. Experiments show that the models can capture the dependency between semantic roles. In the future, this capability might be used to detect inconsistency in semantic analysis.
5. I have designed a new reinforcement learning **algorithm** to reduce error propagation in syntactic parsing. I showed that the algorithm improved the accuracy of studied systems of incremental syntactic parsing and, moreover, the improvement is partly due to a reduction of error propagation.

1.3 The Organization of this Thesis

The bulk of this thesis is contained in five papers. As readers will soon find out, the motivation of the papers and the interpretation of their results are sometimes different from that of other parts of the thesis. This is mostly because the research reported in this thesis spans six years during which NLP was revolutionized by deep learning language models. Likewise, my understanding of the thesis topic (and the topic itself) has changed in important ways. Together with time, the task of rewriting the papers has grown so large that it is impractical given my circumstances.

Instead, I decided to reproduce the papers exactly as they are published except some presentation details. To provide context and situate the results in a common theme, summaries and discussions will be provided. It is also essential to provide a conceptual framework about error propagation and a review of existing literature. A chapter will be dedicated to this task.

The thesis is organized as follows: Chapter 2 presents background concepts in error propagation and provides an overview of

the literature and the current research. The next chapters follow three broad responses to error: mitigation (Chapter 3), adaptation (Chapter 4), and correction (Chapter 5). Each of the three chapters contains an introduction and one or two papers. Finally, conclusions are given in Chapter 6.

Article	Conference	Ranking	Contributions	Resources
Minh Le & Antske Fokkens (2015). Taxonomy Beats Corpus in Similarity Identification, but Does It Matter?	Recent Advances in Natural Language Processing (RANLP)	C	Came up with main ideas, developed experiments, performed one of the two analyses	bitbucket.org/cltl/kcsim
Minh Le & Antske Fokkens (2017). Tackling Error Propagation through Reinforcement Learning: A Case of Greedy Dependency Parsing.	Conference of the European Chapter of the Association for Computational Linguistics (EACL)	A	Came up with main ideas, developed experiments, provided technical support for analyses	bitbucket.org/cltl/redep-java
Minh Le & Antske Fokkens (2018). Neural Models of Selectional Preferences for Implicit Semantic Role Labeling.	International Conference on Language Resources and Evaluation (LREC)	C	Came up with ideas, developed experiments, and performed analyses	bitbucket.org/cltl/isrl-sp
Minh Le, Marten Postma, Jacopo Urbani, & Piek Vossen (2018). A Deep Dive into Word Sense Disambiguation with LSTM.	International Conference on Computational Linguistics (COLING)	A	Implemented experiments, provided technical support for analyses	github.com/cltl/wsd-dynamic-sense-vector
Minh Le, Antske Fokkens, & Piek Vossen (submitted). An Input Ablation Analysis of Coreference Resolution.			Came up with main ideas, developed experiments, performed analyses.	bitbucket.org/cltl/even

Table 1.1: A summary of contributions in chronological order. Rankings are retrieved from CORE2018 (`portal.core.edu.au/conf-ranks`), provided by The Computing Research and Education Association of Australasia. The CORE rankings closest to the year of publication are the same as CORE2018.

2. Error Propagation

The inevitability of error propagation is demonstrated by language comprehension in humans. Given the garden-path sentence: *The girl told the story cried*, a reader typically tumbles on the first error by considering *told* the main verb of the sentence. This leads to a situation where the second verb, *cried*, is unresolvable. Research in neurolinguistics has shown that people invariably make this type of errors and then revise their analysis to reach a satisfying solution (Rayner and Frazier, 1982).

This is not just a quirk of a wetware implementation but reflects a fundamental issue in any system of language comprehension. All languages consist of progressively larger structures: letters, words, sentences, paragraphs, and documents. Due to combinatorial explosion, the number of interpretations is so large that no system can consider a whole document or even a whole sentence at once. As they make sequential decisions to build up an interpretation, errors occur and cascade.

Pipelines, a long tradition in NLP, are particularly vulnerable to error accumulation. The high-level semantic and pragmatic knowledge that only becomes available later in a pipeline cannot be fed back to lower-level modules, forcing them to make decisions based on very little and localized information. Recently end-to-end neural networks have alleviated the issue by allowing under-specified representations but they are still one-directional. The flow of information is always from small to large (i.e. tokens to phrases, clauses, and sentences) and from simple to complex (i.e. morphology to syntax and semantic) (Tenney et al., 2019).

This is in stark contrast to the brain which fashions frequent integration and reconciliation of high-level interpretation and world knowledge with the lowest levels of processing (Kutas and Federmeier, 2011).

Given this background, it should not come as a surprise that error propagation is widespread in NLP. The issue is well-known and has been mentioned many times with different names: *error propagation* (Dell'Orletta, 2009), *cascading errors* (Finkel et al., 2006; Guu et al., 2015), *compounding errors* (Li et al., 2016b; Chu-Carroll et al., 2003), and *accumulation of errors* (McCallum, 2009; Zhikov et al., 2013). What the field is still lacking is a systematic study of the phenomenon. This thesis is intended to fill in this void.

I set out to understand what error propagation is and how to deal with it. In this chapter, we will learn that it can occur not only at task boundaries in a pipeline but also within a task. Considering the type of causality involved, we can divide it into three types: discrete-deterministic, discrete-probabilistic, and continuous. We will learn about five strategies to reduce error propagation: reducing upstream errors, reordering, making steps independent, increasing robustness, and correction. These strategies can be grouped into mitigation (the first three), adaptation (the forth strategy), and correction (the last). This categorization provides a backbone based on which we will review existing literature on reducing error propagation and summarize the papers in this thesis.

2.1 Types of Error Propagation

I have begun our discussion with NLP pipelines because error propagation in pipelines is the archetype of the phenomenon. It is easy to picture in your mind the influence of one module on the next and, indeed, most documented cases of error propagation

in the literature have been of this sort (see Table 2.1, p. 42). However, this is not the only scenario: error propagation can also occur within one task and, regardless of locality, can take different forms. This section will study the types of error propagation in three contrasting pairs: within v. across tasks, hard (deterministic) v. soft (probabilistic), and discrete v. continuous.

2.1.1 Error Propagation Within and Across Tasks

Dridan and Oepen (2013) demonstrated that even the simplest tasks cause error propagation: incorrect tokenization and sentence segmentation leads to 0.6% to 4.5% decrease in syntactic parsing performance which, they remarked, is larger than incremental improvements frequently reported in the literature. Syntactic analyses are, in turn, the input to many other NLP modules so syntactic errors similarly cause performance degradation there.

However, errors can already accumulate within a task in the consecutive steps taken to accomplish it. Let us consider the following syntactic analysis:[1]

(1) (GOLD-STANDARD) (((ABC 's) (Bob Woodruff)) (is (in Belgrade) tonight) .)

The short sentence above is parsed with six nesting pairs of parentheses plus corresponding labels of grammatical roles (not written here for readability). Given the complexity of the structure, it is natural to attempt to build it up piece-by-piece from words to the whole sentence. Along this incremental process, there is plenty of chance for error propagation to throw the parser off balance.

McDonald and Nivre (2007); Kummerfeld et al. (2012) and Clark (2015) argued that syntactic parsing and coreference resolution systems that work incrementally suffer from error propaga-

[1] The sentence can be found in LDC's OntoNotes distribution (Weischedel et al., 2013a), in the file `data/files/data/english/annotations/bn/abc/00/abc_0003.onf`.

tion. Ng and Curran (2015a) went a step further by quantifying the propagated errors caused by incorrect NP and punctuation attachment in dependency parsing (among other types of errors). Ranzato et al. (2016) showed that error propagation occurs in language modeling using recurrent neural networks.

2.1.2 Hard and Soft Error Propagation

Error propagation can be categorized by not only *where* but also *how* they occur. In some cases, an error distorts the very framework on which later processing takes place, making it impossible to obtain a correct analysis. In other cases, an error introduces misleading information that increases the chance of further errors. I will call the former *hard (or deterministic) error propagation* and the latter *soft (or probabilistic) error propagation*.

To illustrate the concepts above, let us return to Example (1). A correct semantic analysis would assign to *is* the first sense of *to be* as prescribed by PropBank (Palmer et al., 2005) with three arguments: *ABC's Bob Woodruff* as a "topic", *in Belgrade* – a "comment" about the "topic", and *tonight* – auxiliary information about time. The analysis can be compactly expressed as follows:

(2) (GOLD-STANDARD) [$_{\text{ARG1}}$ ABC's Bob Woodruff] [$_{\text{be.01}}$ is] [$_{\text{ARG2}}$ in Belgrade] [$_{\text{ARGM-TMP}}$ tonight].

In a typical pipeline, a syntactic parser is executed and then the result is used as input for semantic analysis, therefore a correct semantic analysis requires correct chunks. However, sometimes parsers err. The following incorrect analysis is produced by a high-performance parser (the gold standard is repeated right after for comparison):[2]

[2]The parser is an implementation of Chen and Manning (2014) and achieves 90% accuracy on financial news. The dependency tree is greatly simplified and converted into chunks for presentation purposes.

(3) (SYSTEM-OUTPUT) *(((ABC 's) (Bob Woodruff)) (is) (in (Belgrade tonight)).)

(4) (GOLD-STANDARD) (((ABC 's) (Bob Woodruff)) (is (in Belgrade) tonight).)

The faulty chunks mean the semantic parser cannot get the right solution for ARG2 and ARGM-TMP roles. This is a case of hard error propagation.

In the same document, I also find this excerpt:

(5) (GOLD-STANDARD) ... tonight he [$_{go.02}$ went] [$_{ARG1}$ on national television] [$_{ARGM-PRP}$ to take direct questions from the viewers] – [$_{ARGM-PRD}$ *something Milosevic never dared to do*].

For a correct semantic analysis of the ARGM-PRD argument (in italic), the correct span must be identified as a chunk by a syntactic parser. The parser mentioned above correctly identifies this chunk but introduces a more subtle error (an underline is used to mark the head of a chunk):

(6) (SYSTEM-OUTPUT) *... tonight he went (on national television) (to take direct questions from (the viewers – (something Milosevic never dared to do))).

(7) (GOLD-STANDARD) ... tonight he went (on national television) (to take direct questions from the viewers) – (something Milosevic never dared to do).

Although the parser correctly identifies the span that would become ARGM-PRD, the annotation shown in Example (6) is wrong in two ways. First, the head of the span switches to a verb instead of the pronoun. Second, the span becomes a modifier of *the viewers* instead of the main verb. These errors might be picked up by a semantic parser and lead to the wrong prediction, which would

become a case of soft error propagation.

2.1.3 Error Propagation in Discrete and Continuous Representations

For the most part of the history of NLP, discrete representations, in the form of labels and graphs, have been in charge. This has changed in recent years when word embeddings and, later, end-to-end neural networks have used distributed representations to vastly increase performance. Models such as BERT (Devlin et al., 2019) and GPT (Radford et al., 2019) associate each token in a sentence with real-valued vectors (one for each layer) which encode diverse information about the token and the surrounding context (Tenney et al., 2019).

If we have a predicted vector representation $v \in \mathbb{R}^n$ and the desired representation $\hat{v} \in \mathbb{R}^n$, error can be calculated using, for example, Euclidean distance $\mathrm{d}(v, \hat{v})$ or cosine distance $(1 - \cos\theta)$ where θ is the angle between v and $\hat{v}$. However, different from discrete annotations, what distributed representations stand for is largely unknown and, therefore, no gold standard can be offered for error calculation. Future research might shed light into this question. For now, we must content with some special cases where it is possible to define an ideal vector and, hence, the calculation of error. One such case is neural networks on adversarial perturbation.

Szegedy et al. (2014) reported a phenomenon known as adversarial examples: by adding an imperceptibly small perturbation to an image, they could change the prediction of a model to any other class (see Figure 2.1a-c, p. 43 for some examples). Since the discovery, a growing literature has documented the instability of neural networks in computer vision (e.g. Carlini and Wagner, 2017; Akhtar and Mian, 2018; Ghiasi et al., 2020) and later NLP (e.g. Ribeiro et al., 2018; Zhang et al., 2020).

Whereas most of the research in adversarial examples has

focused on the output, the internal representations might be more interesting. Ideally, we would like a model to be resilient in the face of noise, i.e. error, defined as the difference between activation v on noisy input and activation $\hat{v}$ on natural input, should be as small as possible. If each layer of a neural network is robust, we can expect that the magnitude of error decreases as activation propagates through layers. In other words, deeper networks should be more robust. In reality, the opposite happens.

As neural networks grow deeper, they seem to sacrifice robustness for accuracy. When Goodfellow et al. (2015) and Tanay and Griffin (2016) plot adversarial examples of shallow linear models (making a model predict "3" for "7" and vice versa), the required perturbation is conspicuous to human eyes and displays similarity to the target class. This is in stark contrast with numerous reports of imperceptible attacks on deep models such as ε-bounded attacks (Kurakin et al., 2017; Carlini and Wagner, 2017, among many others) and shadow attack (Ghiasi et al., 2020); and ones that are visible but small such as sticker attack (Eykholt et al., 2018).

One possible explanation is that perturbation is propagated and amplified through the many layers of deep neural networks in a way analogous to error propagation in a pipeline. Figure 2.1d (p. 43) illustrates this mechanism at work. Starting at a small change in the input, the error gets bigger through the layers (statistically significant) until it seems to reach an upper bound at an advanced layer when it is already big enough to cause a wrong but highly confident prediction.

Deep learning research in NLP imports many aspects of its vision counterpart, including adversarial examples. It has been shown that NLP models based on neural networks change their prediction in unexpected ways in response to small or irrelevant changes in input (see Zhang et al. (2020) for a review). It is also possible that naturally occurring imperfection of embeddings (for

example, due to the small number of training examples for low-frequency words) can distort the representations of later layers and negatively affect predictions.

2.2 Responding to Errors

A reason for the inevitability and prevalence of error propagation is its simple anatomy: wherever two or more dependent decisions are made, error propagation becomes a possibility and the longer the chain of decisions is, the more likely error propagation occurs. This initial analysis hints at a few strategies to alleviate the problem (see Figure 2.2, p. 43 for a schematic illustration):

S1: improve the accuracy of individual steps, especially near the beginning of the chain because they are likely the biggest source of propagated errors,

S2: reorder the steps such that decisions with a higher chance of being correct are executed first,

S3: reduce the length of the decision chain, for example by making some steps independent of others or removing some steps,

S4: enhance robustness of individual steps, i.e. increase the chance of correct analysis in the presence of errors, and

S5: detect and fix errors as they arise.

Accepting the inevitability of errors also changes one's perspective to NLP. While research on how to reduce independent errors still plays an important role, it is no longer enough. More emphasis should be placed on managing the impact of errors and responding to their occurrence. Seen from this perspective, the strategies can be grouped into three main ideas:

Mitigation: reduce the error load prior to a decision (S1,2,3)

Adaptation: ready a system to operate in the presence of errors (S4), and

Correction: iteratively fix errors, hoping that the output will eventually become error-free (S5).

2.3 Existing Approaches

To understand response strategies better, I will use this section to enumerate relevant approaches reported in the literature. Research that potentially has implications on error propagation is numerous, but not always reported as such. For example, all reported improvements in upstream tasks such as word embeddings, tokenization, and part-of-speech tagging can be classified into S1 even though authors do not always mention error propagation and rarely analyze the effect on downstream tasks. Therefore, I select papers based on their expected, rather than reported, effect and classify them into one or more of the strategies. Due to the vast number of publications on this topic, I selected a few papers per approach that illustrate the main ideas clearly.

2.3.1 Distributed Representations (S1)

If citation count is any indication of impact, Word2vec (Mikolov et al., 2013a) has been a huge success. Published 7 years ago, the reference paper has been cited more than 14,000 times according to Google Scholar. Vector space models, for which word2vec is a representative, improve upon symbolic representations by means of generalization power (Goldberg, 2016) and coverage. Whereas symbolic lexical representations need to be curated by experts or trained on annotated corpora, embeddings can be "harvested" from vastly larger unlabelled data. The set of pretrained Word2vec embeddings released by Google contains 3 million unique tokens and phrases whereas an annotated corpus of 1 million non-unique tokens is already rather rare and expensive.[3] Word embeddings also allow arbitrary combinations of words to be represented and their meaning inferred (if somewhat fuzzily),

whereas in symbolic representations, storing larger and larger *n*-grams quickly becomes infeasible due to combinatorial explosion. The end result is improvements in tasks as varied as dependency parsing (Bansal et al., 2014), named-entity recognition (Cherry and Guo, 2015), and discourse relation classification (Braud and Denis, 2015).

Whereas the first wave of vector space models improves recall in terms of tokens and *n*-grams, the second wave targets senses. Word embeddings models are look-up tables that always return the same vector for a given word. They do not distinguish between different meanings a word can take, for example, *bank* the monetary institution and *bank* the side of a river. Camacho-Collados and Pilehvar (2018) call this issue *meaning conflation deficiency*. As a remedy, recent contextualized word vector models use powerful neural network-based language models to generate word vectors on the fly. Peters et al. (2018) show that concatenating static embeddings with context-dependent vectors generated by a bi-directional LSTM improves downstream performance. Similarly, models such as ULMFiT (Howard and Ruder, 2018), and BERT (Devlin et al., 2019) have improved the performance of natural language inference (Devlin et al., 2019), question answering and sentiment analysis (Peters et al., 2018), dependency parsing (Li et al., 2019), coreference resolution (Joshi et al., 2020), among other tasks.

Together, the two generations of word vector models prove that Strategy S1 can drastically improve the performance of NLP pipelines.

[3]The newest version of the popular PENN Treebank, for example, contains 1.2 million non-unique tokens. See `https://catalog.ldc.upenn.edu/LDC2015T13`.

2.3.2 Easy-first (S2)

In a sense, all pipelines operate under an easy-first premise: we first perform part-of-speech tagging (with an accuracy around 97%) then syntactic parsing (accuracy around 90%) but not the other way around. Within a task, however, which step is easier is less clear. In dependency parsing, a non-directional parser attaches words to a head in a particular order and the most natural way – working from left to right – might not be the most optimal. Goldberg and Elhadad (2010) showed that teaching a non-directional system to determine the order of target words leads to results competitive to other paradigms such as graph-based and transition-based parsing. Later work has refined this approach and improves its performance (Tratz and Hovy, 2011; Ma et al., 2012, 2013; Kiperwasser and Goldberg, 2016a).

In a similar vein, a strong baseline for coreference resolution is a rule-based system called multi-sieve (Raghunathan et al., 2010). The algorithm works as follows: the first "sieve" passes through the document once, linking all repeating references; the second "sieve" looks at constructs that are known to indicate coreference such as appositive (e.g. *the head of the department, Dr. Kristal*); followed by five other passes in the order of decreasing precision and increasing recall. Stoyanov and Eisner (2012) and Xie et al. (2015) added a machine learning component and showed that an easy-first approach can achieve competitive performance in coreference resolution.

2.3.3 Joint Processing (S2/S3)

In joint processing (also known as joint inference), two or more modules are fused together and one structure that encompasses all pertinent tasks is optimized and predicted. In this thesis, I only consider a method joint processing if two or more tasks are performed in a meaningfully inter-dependent manner. Thus, multi-task neural networks whose outputs are independent given

the internal representations are excluded. Joint processing can be preceded by joint training (such as in probabilistic approach) or not (such as in approaches using integer linear programming). It has been argued that joint inference can reduce or avoid error propagation by enabling across-task bi-directional communication and predicting many variables at once (Singh et al., 2009).

Joint processing is a well-known approach in NLP. A quick pass through the literature reveals innumerable papers with different combinations of tasks: multi-word expression recognition and syntactic parsing (Constant and Nivre, 2016), part-of-speech tagging and syntactic parsing (Bohnet and Nivre, 2012), word sense disambiguation and semantic role labeling (Che and Liu, 2010), coreference resolution and named-entity linking (Hajishirzi et al., 2013), among others. Work on joint processing that was explicitly motivated by error propagation includes Finkel (2010), Zhikov et al. (2013), Li et al. (2016b), and Singh et al. (2009).

To understand how joint processing helps reduce error propagation, it is necessary to differentiate two paradigms in structure prediction (borrowing from the terminology of syntactic parsing): graph-based and transition-based.

In graph-based processing, one learns a scoring function that returns a scalar value for each pair of input-output: $s(x,y)$. Here, $y \in \mathscr{Y}$ is a complex structure that is generally represented by a graph (hence the name). To parse a new input is to find a structure that maximizes the scoring function.

Obviously, the size of the output space $\mathscr{Y}$ is huge due to combinatorial explosion. To keep inference tractable, it is therefore required that graphs are split (or factorized) into smaller components. Various formulations exist to define graphs and their optimization: Markov logic network (Poon and Domingos, 2007), imperatively-defined factor graphs (Singh et al., 2009), conditional random fields (Toutanova et al., 2008), and many others. A close examination of graph-based joint processing methods is be-

yond the scope of the current thesis. For demonstration purpose, consider a simple factorization method based on McDonald et al. (2005a):

$$y^* = \arg\max_{y \in \mathscr{Y}} s(x,y) = \arg\max_{\{y_i\}} \sum_i s(x,y_i) \tag{2.1}$$

In the formula above, a complex structure y is defined as the totality of atomic structures $\{y_i\}$. For example, in dependency parsing, y is a parse tree and $\{y_i\}$ can be edges. The restriction put on $\{y_i\}$ limits the types of possible features but enables specialized algorithms to find the globally optimum structure. For example, in projective dependency parsing, Eisner (1996)'s algorithm allows us to find the globally optimum graph with respect to a scoring function that is defined on individual edges.

While graph-based processing trades feature complexity for globally optimal inference, transition-based processing embraces local inexact inference for the sake of rich non-local features. In this paradigm, the output is built up gradually via a series of actions $\{a_t \in \mathscr{A}\}$: $\{\emptyset = y_0, y_1, y_2, \ldots, y_n = y\}$, $y_{t+1} = a_t(y_t) \forall t = 0, \ldots, n-1$. These actions are chosen to maximize a scoring function ϕ that operates on the rich partial structure that is available at each time step:

$$a_t = \arg\max_{a \in \mathscr{A}} \phi(x, y_t, a) \tag{2.2}$$

Integrating two individual modules into one graph-based system is essentially converting a two-step process into a single decision, therefore eliminating error propagation (Strategy S3). On the other hand, joining two transition-based systems into one is analogous to rearranging a multi-step process (Strategy S2). Although the decision sequence is just as long, we can hope that

each step will be more accurate because of the rich multi-task feature representations that it enables.

Building a joint-processing system is a complex, labor-intensive endeavor. Take Constant and Nivre (2016) as an example: the authors adapt transition-based syntactic parser by adding new operations that annotate multi-word expressions. To build this new system, they need to modify the code for data representation, data preprocessing, model representation/training/serialization, and post-processing. In short, converting a pipeline into joint-processing implies a complete overhaul of the system. Adding to that, in the case of graph-based processing, one might also need to find an inference algorithm that works on the combined graph.

Yet, an even more formidable challenge is data availability. Constant and Nivre (2016) can only train their system on datasets that annotate both syntactic structures and multi-word expressions but this type of datasets is not always available for all combinations of tasks. Table 2.2 (p. 44) summarizes what types of annotations are available for a few well-known corpora. At a glance, it is clear that the table is very sparse. The most heavily annotated corpora (Wall Street Journal and OntoNotes) are not annotated for all the tasks listed.

For joint-learning to replace pipeline architectures, we need a generic framework that enables efficient definition and construction of joint systems and a way to bootstrap from existing annotated data. Between the two paradigms described in this section, the transition-based paradigm is more suitable for this goal because a generic inference procedure is already widely used and it is rather easy to define new actions. Human language processing is arguably also transition-based and researchers has been investigating and improving the cognitive plausibility of transition-based approaches (Nivre, 2004; Boston et al., 2008).

2.3.4 Global Inference (S3)

In the context of transition-based processing, global inference is a method turns a system that makes a sequence of decisions iteratively into one that optimizes complex structures as a whole and makes a single final decision.

Andor et al. (2016) proposed global normalization for dependency parsing and showed via a mathematical proof that it is better than local normalization. Their implementation surpasses the previous state-of-the-art in performance and is later applied for many different languages (Alberti et al., 2017). Similar to joint processing, upgrading a local normalization system to a global one requires substantial engineering effort.

2.3.5 End-to-end Learning (S3)

To some extent, neural network-based end-to-end models avoid the trade-off between feature richness and robustness by replacing discrete, explicit features by distributed, under-specified ones. Like joint processing, end-to-end models are trained on multiple tasks simultaneously. At test time, however, the model might solve tasks jointly or individually with exactly the same behavior.

Traditionally, part-of-speech tagging and syntactic parsing are necessary inputs for many tasks. With the advent of deep neural networks, it is suddenly possible to bypass those steps. SENNA (Collobert and Weston, 2008) is an influential model that uses neural networks to replace discrete words and syntactic structures with distributed representations. One single model based on convolutional neural networks is trained to simultaneously perform four tasks: part-of-speech tagging, chunking, semantic role labeling, and language modeling. More importantly, the predictions of each task is conditionally independent to others' given the latent representation in the neural network's hidden layers. This setting effectively turns one 4-stage pipeline into four independent systems, eliminating the chance of cascading errors.

An end-to-end design can also be used to construct single-task systems that do away with preprocessing pipelines. In coreference resolution, for example, the system of Lee et al. (2017) predicts co-referring spans based on distributed representations computed directly on raw input. This idea has been successfully applied to many tasks such as named-entity recognition (Ma and Hovy, 2016), dependency parsing (Kiperwasser and Goldberg, 2016b), semantic role labeling (Zhou and Xu, 2015; Marcheggiani et al., 2017), semantic parsing (Foland and Martin, 2016).

Despite their successes, end-to-end models share some weaknesses with all NLP so far. They are vulnerable to error propagation when making consecutive decisions and so far lack a notion of reference, episodic memory, and a situation model (Van Dijk et al., 1983). As noted by Lee et al. (2017), their coreference resolution system tends to "conflate paraphrasing with relatedness and similarity" and does little in modeling common sense knowledge. Indeed, harmonizing end-to-end learning and multi-step reasoning is a challenge that only recently starts to gain research interest (Dua et al., 2019).

2.3.6 Training for Shorter Transition Sequences (S3)

A remarkable demonstration of Strategy S3 is Nivre et al. (2009). The paper concerns with transition-based dependency parsing, in which a parser builds up a dependency tree by executing actions one by one: an action might be "create a link to the word in the left" or "create a link to the word in the right", but there are also actions to shift the focus of the system to another word or to move words around. Observing that a parser performs an excessive number of an operation called SWAP, the authors proposed a new training procedure that cuts down on SWAPs by up to 80%. The result is shorter transition sequences and improved performance across five languages without changing the architecture of the model.

2.3.7 Training on Automatic Annotations (S4)

It has long been observed that modules trained on gold labels of upstream tasks perform poorly in a realistic scenario when only predicted labels are available. For this reason, shared tasks often provide both gold and predicted annotations (e.g. Hajič et al. 2009).

Given a pipeline with a part-of-speech tagger and a syntactic parsing, *jackknifing* is a technique that generates automatic POS tags for the purpose of training a syntactic parser despite gold POS tags being available. First, the training set is divided into, for example, 10 folds. A POS tagger is then trained on 9 folds and then used to predict the POS in the remaining part. The process is repeated until automatic POS tags are generated for the whole corpus. The method improves parsing score as much as 3% on Chinese Tree Bank (Che et al., 2012) and has become the de-facto standard in syntactic parsing experiments (Zhang and Nivre, 2011; Li et al., 2014a; Chen and Manning, 2014; Pei et al., 2015).

In the same spirit, a remedy can be defined for within-task error propagation. In *stacked sequential learning* (Cohen and Carvalho, 2005), a base classifier is trained and applied using K-fold cross-validation to generate predictions which are fed, together with the original input, into a second classifier. The composite architecture is shown to outperform the base classifier in various sequential partitioning tasks (Cohen and Carvalho, 2005) because of a reduced mismatch between training and test.

2.3.8 Multiple Outputs (S4)

Knowing that the output of a step might contain errors that negatively affect the next ones in the pipeline, it is natural to ask *What if we pass on more than one solution?* The hope is that one of those alternatives ends up being correct and picked up by downstream modules.

An early attempt along this line is Charniak et al. (1996) which shows that returning multiple parts-of-speech per token leads to improvement in a downstream syntactic parser. Corroborating on this result, Watson (2006), Yoshida et al. (2007), Bunescu (2008), and Henestroza Anguiano and Candito (2012) all found improvement in syntactic parsing and named-entity recognition, especially when the probability distribution of POS tags is incorporated (Venugopal et al., 2008).

Finkel et al. (2006) take this idea one step further by proposing to model the whole pipeline as a Bayesian network. The approach is conceptually simple enough to apply to long pipelines but empirical results have been lacking. To the best of my knowledge, the longest pipeline that it was evaluated on has a length of three (Raman et al., 2013).

2.3.9 Error Detection and Correction (S5)

I did not find any paper exploring the possibility of detecting and correcting errors in multi-module NLP pipelines. From a single-task perspective, syntactic parsing and machine translation have received the most attention. Hall and Novák (2005); Attardi and Ciaramita (2007); Hall and Novák (2011); Delecraz et al. (2017) proposed corrective procedures to improve syntactic parsing. With the purpose of improving the semi-automatic construction of treebanks, Dickinson (2010); Dickinson and Smith (2011) propose procedures to help annotators detect and correct errors of a parser. In machine translation, automatic post-editing is an active research topic (e.g. Simard et al. 2007; Béchara et al. 2011; Pal et al. 2017). The purpose is to modify the output of a "first stage" machine translation system to increase quality (Pal et al., 2017).

Hollingshead and Roark (2007) described a method in which a parsing pipeline is run twice but not for error correction. Related ideas that I found in the literature is detecting human annotators'

errors (Ambati et al., 2011; Ng et al., 2013); and calibrating the probability of a system which might facilitate error detection (Nguyen and O'Connor, 2015; Guo et al., 2017).

2.3.10 Reranking (S5)

In cases where we do not know how to detect errors, can we at least refine the outputs such that the number of errors are statistically reduced? Reranking falls in to this category of statistical error "correction". Instead of one interpretation for each sentence, multiple candidates are proposed. The aim is to reorder them such that those that have more errors are demoted (pushed to lower ranks) whereas better outputs are promoted (pushed towards the top of the list).

Reranking was studied in Charniak and Johnson (2005) and Ge and Mooney (2006). An n-best list of parses is first produced by a local parser. Given the rich information in the parses, a reranker is trained to exploit global features to pick the best parse.

So far, only incremental improvements were found using reranking. A fundamental difficulty with the approach is that the number of possible annotations grows exponentially with respect to sentence or document length. In contrast, the number of candidate outputs has to be small to maintain a reasonable speed, e.g. $n = 50$ (Charniak and Johnson, 2005). Therefore, no guarantee can be made about the quality of the top n outputs.

2.4 The Importance of World Knowledge

As argued at the beginning of this chapter, error propagation is inevitable and requires correction. On what basis do we decide what part of an output is an error and how to correct it? One possible strategy is to use knowledge about the same task to detect and correct errors. Related ideas have been tried out in the context of dependency parsing: Zhang and Clark (2008) combined graph-

based and transition-based parsers to improve performance and Andor et al. (2016) showed that a globally normalizing model is strictly more accurate than a local one. Although papers like these show that improvement is possible, they also demonstrate that what can be achieved within a given level of abstraction is far from human-level performance.

Research in neurolinguistics points at another solution: integration of world knowledge – a type of open-ended knowledge that is shared among the majority of human adults (Allen, 1995, p. 10). When talking to a normal adult English speaker, we can safely assume they know that *open* is the opposite of *closed*; and that certain things are (im)plausible, for example, inanimate objects do not have emotions and things fall down, not up. For decades, countless papers have documented complex integration of meaning as early as 400 milliseconds after reading a word, reflected in a change in electroencephalographic signal known as N400 effect (see Kutas and Federmeier (2011) for a review). Hagoort et al. (2004) have shown that N400 occurs when a reader detects violations of linguistic constraints and world knowledge. Nieuwland and van Berkum (2006) extended the purview of N400 by showing that it responds to the preceding narration such that new knowledge can be introduced to override pre-existing knowledge. Although N400 is normally observed for implausible sentences such as *The peanut is in love*; in their experiments, the effect disappears when preceded by a story about the same peanut behaving like a human.

The following example illustrates how world knowledge can help resolve errors in a natural language processing task. Coreference resolution is the task of collecting referring expressions (also called mentions) into groups, each one is about the same person, thing, or event. After decades of active research, it remains a challenging task.

To demonstrate the difficulty faced by coreference resolution

systems, I took a random news article,[4] and ran a highly optimized neural network-based model to obtained the analysis depicted in Figure 2.3 (p. 45).[5] A quick scan reveals two errors (marked by red letters in a circle): *Bro* is mistakenly marked as coreferring with *I*, and *Brady's production* is put into the same bucket with *it*. Cases like these are difficult for models that consider expressions in isolation but should be obvious to one that takes into account the interaction between entities and events. In the first case, *Bro*, and the latter word *you*, refers to Evans's listener and therefore cannot be coreferring with Evans himself. In the second case, the expressions *it* and *Brady's production* stand in a cause-effect relationship and therefore cannot co-refer. A correct solution would point *it* to *a lack of talent around him* a few lines above.

Li et al. (2016b) made a similar observation in information extraction. Among 200 randomly selected errors that his state-of-the-art system made, it was found that 38.82% is due to the lack of world knowledge, 11.18% "requires sophisticated semantic inference", and another 23.53% is caused by out-of-vocabulary words which can be partly explained by an inability to infer lexical meaning from context.

As the examples above make clear, the ability to encode world knowledge and reason with it is beneficial for NLP. There has been some work on establishing benchmarks (Levesque et al., 2011), collecting data (Speer and Havasi, 2012), and developing models (Ostermann et al., 2018), but big challenges remain and a solution appears far away. One big inspiration of the research described in this thesis is to demonstrate the necessity of and make progress towards effective modeling and application of

[4] From *www.cbssports.com/nfl/news/mike-evans-reacts-to-tom-brady-signing-with-the-buccaneers-and-its-safe-to-say-hes-very-very-excited*, retrieved on 23 March 2020.

[5] I used the demo at *demo.allennlp.org/coreference-resolution*, accessed on 23 March 2020. As described on the website, the underlying model is the end-to-end neural network by Lee et al. (2017) modified to use BERT embeddings.

world knowledge.

2.5 Overview of Papers

In previous sections, the groundwork has been laid covering the definition and classification of error propagation, the categorization of alleviation strategies and existing approaches, and the singular importance of world knowledge. In this section, I will give an overview of my research. Because the papers are spread across various tasks in NLP, I will also provide in this section the necessary background to understand them.

I chose tasks to study according to three criteria. First, they should be well-established so that impact can be measured quantitatively and relevance can be established with respect to the broader NLP ecosystem. Second, the tasks need to have big potential impact on other tasks such that its error propagation effect is worth studying. And last but not least, it is desirable to choose tasks that span the interpretive spectrum: from token-bound, superficial tasks to document-wide, deep understanding ones. The chosen tasks are summarized in Figure 2.4 and described in more detail in the rest of this section.

2.5.1 Errors in Distributed Representations (Q1, S1)

Some NLP tasks make a bigger contribution towards the overall performance than others and perhaps the best example is word representation learning. As described in Section 2.3.1, word embeddings have been shown to drastically improve performance of many NLP tasks over discrete representations. To understand how word embeddings work, let us consider the task of predicting fillers for the semantic roles of the word *pay* in the following sentence:[6]

[6]Taken from document `nw/wsj/15/wsj_1556.onf` in OntoNotes 5.0.

(8) Mr. Morrissey, the Kentucky Fried Chicken franchisee, notes that most franchise owners must absorb increases in expenses without any cut in the royalties, or portion of sales, that they must pay franchisers.

Let us further assume that the token *royalties* (or any inflection of the word) does not appear in training data. Instead of treating it as a wild card, a model could consult its embeddings table and, because such a table is harvested from vast unlabeled data, a match is returned with high likelihood. More importantly, word embeddings are trained in such a way that neighbors of the word *royalties* in the vector space tend to have a similar meaning such as *fee*, *toll*, etc., one of which the model has observed in its training set. Because the seen and unseen words are similar, the model can draw on its experience to predict that *royalties* fills the role ARG1 of *pay*.

Or at least that is the theory. In practice, embeddings are noisy and exhibit structural biases. It has been shown that, for example, various vector space models capture similarity only poorly while being good at relatedness (for example, *royalties* is related to *artist*, *music*, and *license*) (Hill et al., 2015).

One of the main evaluations of embeddings models is via a lexical similarity dataset: a set of word pairs with manually sourced similarity scores. The correlation (e.g. Spearman's ρ) between the scores a model predicts and those from humans is then calculated, allowing us to gauge the overall quality of a model. Using such a dataset, my first paper, **Taxonomy beats corpus in similarity identification, but does it matter?**, shows that word vector models underperform taxonomy-based models. We proposed two ways to break down the dataset into different granularities and showed that word vector models are consistently less accurate across sub-datasets. For this thesis, I additionally examined erroneous word pairs and classify them into three types

of errors: wrong cluster, bad cluster, and wrong word sense. Overall, the results show that although word vector models have good coverage and positive impact on downstream models, they can be improved by reducing noise in their clusters, improving cluster placement, and dealing with word senses.

2.5.2 Conceptualized Word Embeddings and (Q1/Q2, S1)

As discussed in Section 2.3.1, the second wave of word vector space models replaces lookup tables with neural networks that compute embeddings on-the-fly, allowing models to take into account variation of word meaning.

A direct application of contextualized word embeddings is Word Sense Disambiguation (WSD). A sense is a meaning, among possibly many meanings, of a word. Senses are documented in sense inventories such as WordNet (Fellbaum, 1998). In WSD, a system assigns senses to words in a document according to a certain inventory. This information is rarely used in later processing due to error propagation concerns but the performance of WSD systems has been steadily improving. During the time of my research, the state-of-the-art is achieved by Yuan et al. (2016a), a model that builds contextualized word embeddings using an LSTM and compares them to memorized embeddings to disambiguate.

My second paper, **A Deep Dive into Word Sense Disambiguation with LSTM**, focuses on the reproduction of Yuan et al.'s results. In the context of this thesis, the paper serves two purposes: first, WSD can be considered a probing task of the ability of contextualized word embeddings to model senses. Second, I sought to understand how far we can reduce errors by scaling up this model and what the limiting factors are. The results show that although the model performs better than a most-frequent sense baseline, the improvement is small. On the second research question, increasing the amount of (unlabeled) data and model

size quickly leads to diminishing returns. Taken together, the results indicate that alternative architectures are needed to model word senses properly. As part of that research, I reimplemented Yuan et al. (2016a)'s algorithm and made it publicly available.

2.5.3 Reinforced Transition-based Dependency Parsing (Q2, S4)

As discussed in Section 2.1.1 and Section 2.2, one of the alleviation strategies is to improve the performance of a system at the presence of errors (S4) and we can validate this idea using within-task error propagation, assuming that any learned lesson will be applicable for the across-task variety too.

Dependency grammars analyze a sentence through identifying dependency relations between its words. As an example, Figure 2.5 shows a part of the dependency analysis for Excerpt (8) in p. 35. Transition-based parsing is a paradigm in dependency parsing in which a system builds up an analysis via many sequential steps. It is therefore ideal to study within-task error propagation.

Previous work has shown that when there is a discrepancy between training and testing, the performance of a dependency parser suffers. Zhang and Nivre (2012) shown that when the beam size of a parser (the number of parses attempted per sentence) at testing is different from the beam size used during training, performance decreases. Similarly, a train-test discrepancy also occurs during error propagation because parsers are trained on all-correct input while, at test time, errors inevitably occur.

Reinforcement learning is useful in combating this problem. Instead of training a system by a gold-standard set of situations paired with expected behaviors, reinforcement learning lets a system execute actions, gather experiences, and improve itself based on feedback signals (Simeone, 2018). In our case, the agent is a parser which, at any time step, can choose between

creating a dependency link or changing its internal state. The parser receives a numerical reward proportional to how close its output resembles the correct analyses. Because the training examples originate from the parser's own actions, the train-test discrepancy is minimized.

The paper **Tackling Error Propagation through Reinforcement Learning: A Case of Greedy Dependency Parsing** is devoted to investigating the effect of reinforcement learning for a greedy transition-based dependency parser. We show that our reinforcement learning algorithm succeeds in increasing the performance of different parser formulations fed with English and German texts. Moreover, we show that the effect is partially due to a reduced rate of error propagation.

2.5.4 The Value of Context in Coreference Resolution (Q3, S5)

We have taken a glimpse at coreference resolution in Section 2.4. In the paper **An Input Ablation Analysis of Coreference Resolution**, I investigate in more detail the source of performance in coreference resolution systems and point out potential improvement by incorporating world knowledge.

Coreference resolution is a high-level semantic task that is often treated by low-level methods. It is long known that a strong baseline of the task is string matching: putting all mentions with the same surface form to the same cluster. Durrett and Klein (2013) drive this point home by demonstrating that a simple system using only surface features can outperform existing systems that use rich syntactico-semantic features and carefully crafted heuristics. The obvious mismatch proves hard to bridge. Efforts to incorporate semantic roles, external knowledge sources, and event information lead to incremental improvement and a big gap remains between the state-of-the-art results and human performance (Peng and Roth, 2016; Ponzetto and Strube, 2006).

I hope to show that, despite well-recognized difficulties, modeling world knowledge is essential to reach human-level performance in coreference resolution. By modifying the input documents to control the layers of information that they expose, we show that a representative sample of existing methods does not make use of context. Furthermore, by observing human annotators working on the same modulated documents, we illustrate that humans use world knowledge and complex reasoning to retain high accuracy on challenging documents.

2.5.5 Modeling Selectional Preference for Implicit Semantic Role Labeling (Q3, S5)

In Section 2.4, I have argued that an effective model of world knowledge is essential to solve error propagation. To perform reasoning over the content of a document, we first need to be able to connect pieces of the text into a coherent story. Implicit semantic role labeling (Ruppenhofer et al., 2009) is one of a few tasks that meet this need.

Roughly speaking, Semantic Role Labeling (SRL) is the task of answering the question *Who does what when where and how?*. A frame is a linguistic structure akin to *schema*, *script*, or *scenario*, which, once evoked, elicits a fixed set of elements (Fillmore, 1982). A frame consists of a predicate and multiple roles where the predicate is typically an action or attribute and the roles describe its various aspects. Based on an inventory of frames (Palmer et al., 2005; Fillmore et al., 2003), one can identify instantiations of predicates in a document and expressions that fill their roles (also called fillers). As defined in the NLP literature, SRL only considers expressions that reside in the same sentence as the predicate. Implicit semantic role labeling (iSRL) focuses on finding the missing fillers: ones that are in other sentences. Consider the following example:[7]

[7]Taken from "The divided house", a novel by Mary Raymond, accessed via

(9) "Tell me about it," Sara said. Jenny put her head in her pillow again. Her voice came muffled, "I don't want to talk about it."

For the underlined word, following PropBank (Palmer et al., 2005) formulation, an SRL system would annotate the following predicates and arguments:

(10) Predicate: talk.01, ARG0: *I*, ARG1-PPT: *it*

An iSRL system, on the other hand, will look beyond sentence boundary to identify fillers for the roles that are not already filled:

(11) Predicate: talk.01, ARG1-GOL: *Sara*

Implicit semantic role labeling is both interesting and challenging because it deprives us of traditional features coming from local syntactic structures, forcing us to model global coherence and develop a deeper understanding of the series of events in a document. This is an area where NLP has typically struggled with.

Selectional preference is a type of world knowledge that captures the plausibility of expressions to be the filler of a semantic role. The predicate talk.01 expects an animate object as a filler of the role ARG2-GOL (hearer). This makes *Sara* a more likely filler than *her pillow*. Exceptions do occur and selectional preference is meant to be combined with other signals, but it has been shown that a model of selectional preference can contribute to the performance of semantic role labeling (Zapirain et al., 2013). A natural extension is modeling the constraints between roles.

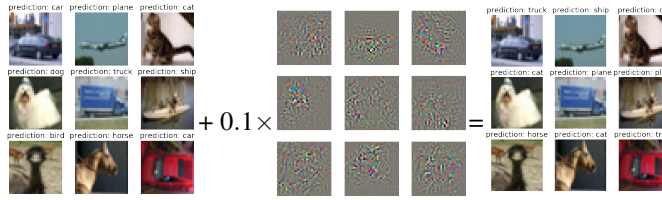
My last paper, **Neural Models of Selectional Preferences for Implicit Semantic Role Labeling**, investigates how we can train a selectional preference model to capture not only the inter-

the British National Corpus (BNC, 2007).

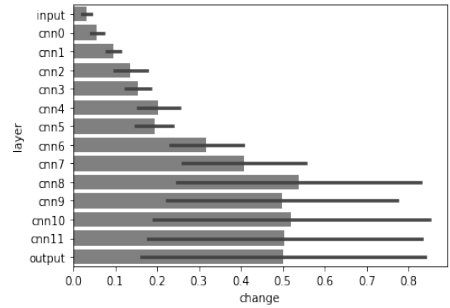
action between predicates and roles and between roles themselves with the aim to improve implicit semantic role labeling. We did not find an improvement over the baselines, probably because of a misalignment between training and test data. We release our implementation of existing methods and an evaluation framework to support further development of the area.

	Ck.	Syn.	SRL	CR	SA	OM	TE	MT
Sent.		1						
Tok		1, 2, 3						
POS	4	5, 6, 7		24				10
NER				20, 24				
Syn.		25, 26, 27, 28, 31	8, 13, 15, 16, 17, 29, 32	18, 20	19			10, 11, 12
SRL							9	
CR				21, 22, 23			9	
OM						14		
MT								11, 30

Table 2.1: Overview of reports about error propagation from one task to another. Source tasks are written in the leftmost column while target tasks are in the first row. Entries that have the same header on the row and the column (e.g. CR-CR) are error propagation from one subtask to another within a bigger task or that between steps in the same task. For brevity, I do not list all subtasks (i.e. mention identification and mention clustering are two steps in coreference resolution). Abbreviations: Ck. = Chunking, Syn. = Syntax parsing, Sent. = Sentence boundary detection, POS = Part-of-speech tagging, NER = Named-Entity Recognition, Syntax = Syntactic Parsing, SRL = Semantic Role Labeling, CR = Coreference Resolution, SA = Sentiment analysis, OM = Opinion mining, TE = Timeline extraction, MT = Machine Translation. List of references: 1. Dridan and Oepen (2013), 2. Forst and Kaplan (2006), 3. Foster (2010), 4. Song et al. (2012), 5. Foster et al. (2011), 6. Lease and Charniak (2005), 7. Kong and Smith (2014), 8. Pradhan et al. (2005), 9. Caselli et al. (2015), 10. Han et al. (2012), 11. Han et al. (2013), 12. Quirk and Corston-Oliver (2006), 13. Gildea and Palmer (2002), 14. Yang and Cardie (2013), 15. Carreras and Màrquez (2004), 16. Favre et al. (2010), 17. He et al. (2017), 18. Lee et al. (2013), 19. Gómez-Rodríguez et al. (2017), 20. Lee et al. (2013), 21. Peng et al. (2015), 22. Bengtson and Roth (2008), 23. Ng and Cardie (2002a), 24. Soon et al. (2001), 25. Nivre et al. (2009), 26. Ng and Curran (2015a), 27. McDonald and Nivre (2007), 28. Kummerfeld et al. (2012), 29. Sagae (2010), 30. Venugopal et al. (2008), 31. Zhang and Nivre (2012), 32. Haghighi et al. (2005).



(a) Original **(b)** Perturbation **(c)** Adversarial



(d) Change in activation

Figure 2.1: Error propagation in layers of a VGG-16 deep neural network (Simonyan and Zisserman, 2015). Images are from CIFAR-10 (Krizhevsky, 2009), perturbation is generated by FGSM ($\epsilon_{L_2} = 2$) (Goodfellow et al., 2015). The change in the input and layer activation is computed as $\frac{\|v-\hat{v}\|_2}{\|\hat{v}\|_2}$, averaged across 2,000 images, with $\hat{v}$ is the input or layer activation in the case of natural images and v is the input and activation in the case of perturbed images. Error bars represent standard deviation.

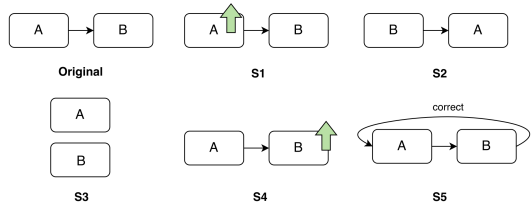


Figure 2.2: A schematic representation of five strategies to reduce error propagation. Given a minimal pipeline of two steps: A and B, Strategy S1 improves a step individually, S2 reorder the steps, S3 reduces the length of the decision chain (by breaking it into two independent chains of length one in this case), S4 makes the latter step more robust to errors, and S5 applies error correction.

	POS	Syn.	NER	NED /EL	SRL	iSRL	Entity Coref.	Event Coref.	WSD	Quote Attrib.
TAC				1						
IITB					2					
KORE50				3						
Reuters			4	5			6	6		
WSJ	8	8	9		10, 11, 12		9		14	15
FrameNet					17		17			
OntoNotes		16			16	13	16		16	
Sherlock Holmes						18	18			
Brown	20	8			19					
SemCor (part of Brown)	20								21	
WSMT									22	
RSS-500			23	23						

Table 2.2: A summary of available annotations: each row is a corpus, each column is an NLP task. Abbreviations: POS = part-of-speech tagging, Syn. = syntactic parsing, NER = named-entity recognition, NED/EL = named-entity disambiguation/entity linking, SRL = semantic role labeling, iSRL = implicit semantic role labeling, Entity/Event Coref. = entity/event coreference, WSD = word sense disambiguation, Quote Attrib. = quote attribution. 1. Getman et al. (2018), 2. Kulkarni et al. (2009), 3. Hoffart et al. (2012), 4. Tjong Kim Sang (2002), 5. Hoffart et al. (2011), 6. Hasler et al. (2006), 8. Marcus et al. (1993), 9. Weischedel and Brunstein (2005), 10. Palmer et al. (2005), 11. Meyers et al. (2004), 12. Surdeanu et al. (2008), 13. Moor et al. (2013b), 14. Pradhan et al. (2007), 15. Pareti (2011), 16. Hovy et al. (2006), 17. Baker et al. (1998), 18. Ruppenhofer et al. (2009), 19. Carreras and Màrquez (2004), 20. Kučera and Francis (1967) 21. Miller et al. (1993b) 22. Navigli et al. (2013a), 23. Röder et al. (2014)

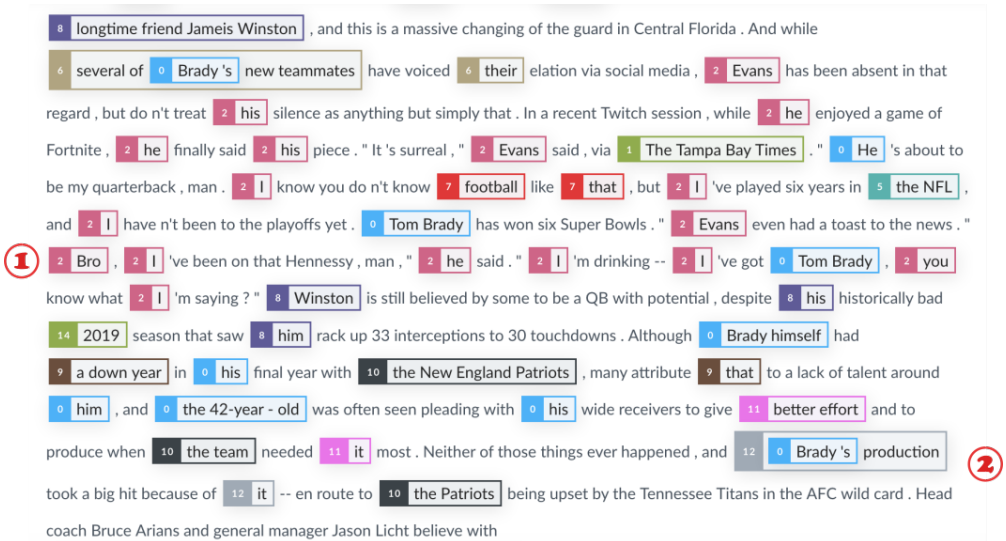


Figure 2.3: Excerpt from a news article with coreference annotated by AllenNLP. Expressions that refer to the same entity or event are marked with the same color and number.

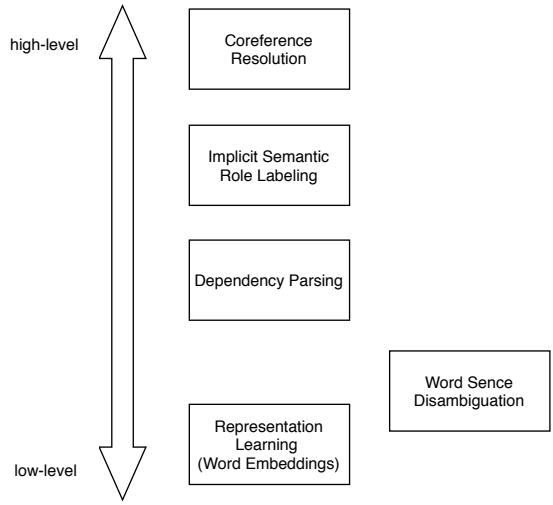


Figure 2.4: Natural language processing tasks studied in the current thesis.

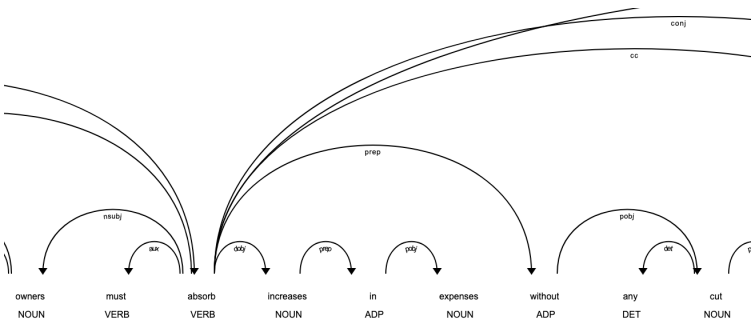


Figure 2.5: Part of a dependency analysis of Excerpt (8). Arrows point from a head to its dependents and the type of dependency is noted on edge labels.

3. Mitigation

As is the case with any kind of negative events, prevention is pivotal. The previous chapter has discussed Strategy S1 (reducing errors in earlier steps). In the context of pipelines, this means getting the highest possible accuracy on foundational tasks while in end-to-end models based on neural networks, we would want to have the best representations of words in the first hidden layer. To facilitate the preventation of error propagation, this chapter will analyze errors in lexical representations that have become popular in recent years: static and contextualized word embeddings.

Word embeddings are often evaluated at the vocabulary level, either intrinsically with similarity datasets or extrinsically via downstream tasks. While useful, those methods do not accommodate the interpretation of individual decisions made by an NLP system. On the other hand, evaluating individual vectors is challenging because of their unsupervised nature – there is no gold-standard set of embeddings. It might not be obvious what a vector error even means. To get a sense of what a vector error looks like, consider the case of the word *royalties* previously mentioned in Section 2.5.1. The vector for *royalties* as depicted in Figure 3.1 is an error because it is far away from the most similar words (*fee*, *toll*, *bill*), relative to less similar one such as *singer* and *license*.

In the first paper, **Taxonomy beats corpus in similarity identification, but does it matter?**, we propose an evaluation method based on similarity score comparison that allow the identification of vectors that might be an error. The method is called

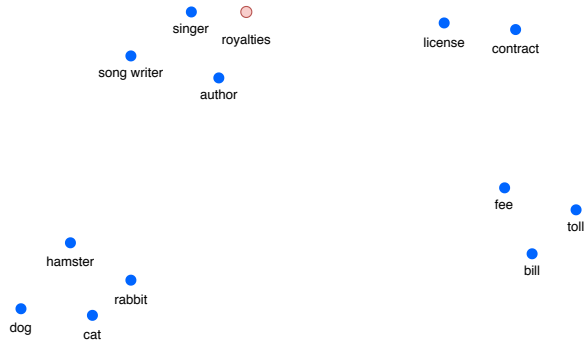


Figure 3.1: An error in vector space: the word *royalties* is placed in the wrong cluster.

ordering accuracy and works as follows. Two gold-standard similarity scores are compared. Without loss of generality, let us assume they are $\text{sim}(A) > \text{sim}(B)$ where A, B are two pairs of words. Given a word vector model M and similarity function f, e.g. the popular cosine similarity, the model is said to make an error if $f_M(A) < f_M(B)$. The method also allows us to differentiate between fine-grained errors, in which a model fails to make subtle distinctions that are sometimes hard for even humans, and coarse-grained errors in which the model is clearly at fault. In the current discussion, we will use exclusively the dataset SimLex-999 (Hill et al., 2015) which has scores in a 0-10 scale. Let us define coarse-grained errors to be erroneous cases that have $|\text{sim}(A) - \text{sim}(B)| \geq 6$. For example, two word pairs found in SimLex-999 are *mouse-management* (score: 0.48) and *journey-trip* (score: 8.88). If a model rates the first pair higher than the latter, it has made a coarse-grained error.

Applying this method on a set of pretrained word2vec embeddings (Mikolov et al., 2013a) allows me to pinpoint the following vectors that seem out of place:

reality: closest vectors are *Reality, realities, Emblematic_examples, Jaap_Amesz_Dutch, Germany_Dieter_Wiefelspuetz, raj_....._i,*

saucier_Miracle_Jones, Pennsylvania_prophetic_rodent, Wife_Swap_Larimer, Octo_mum

sense: nearest neighbors are *humor_Jena_Elayan, deceiving_Hondurans, feel, feeling, Dr._Valerie_Voon, Prosecutor_Ann_Swegle, sensibility, stunner_Afleet_Alex, groundedness, sence*

zone: nearest neighbors are *zones, Adjust_tuning, Zone, #.###_Chalcocite_covellite, toxic_gook, lofting_fade, endzone, insistence_Arata, grade_La_Mascota, BRUSSELS_AFX_Euro*

suds: instead of semantically similar words such as *foam* and *bubbles*, its closest neighbors are beer-related words: *beer, brewskies, brewski, brewskis, Yuengling_Lager, Natty_Light, microbrew, Brooklyn_Lager, beers, craft_brews*

bread: except an inflection and a few types of bread, all nearest neighbors are words related to the making or consumption of bread: *butter, rye_sourdough, breads, loaf, flour, baladi_bread, loaves, raisin_bread, stale_bread, wheaten_flour*

Extracting words from coarse-grained errors generates a whooping 449 words in a total of 751. Upon closer examination though, most of the errors are due to systematic discrepancy between human annotators and systems in the treatment of antonyms (e.g. *north-south*), relational antonyms (e.g. *husband-wife*), and alternatives (e.g. *top-side*). Human annotators consistently give low scores for those cases whereas word embeddings models tend to give high scores. While conceptually dissimilar, it makes sense that the nearest neighbors of *north*, for example, include *south, west*, and *east* because all of them are geographical directions.

Another reason for ordering errors is what might be called cluster displacement: when all the words involved are correctly placed next to their most similar words but the distance between those clusters is wrong. We will not deal with this problem to focus on individual vectors which can be classified into:

Correct: Considering up to 20 of nearest neighbors, the word is placed in the right cluster.

Wrong sense: Nearest neighbors can be considered semantically similar to the target word but only if it is read in a sense different from what a human would pick given the context of the dataset.

Bad cluster: The word is placed in a cluster that includes some similar words and some unrelated words with roughly equal proportions.

Wrong cluster: The word is placed next to mostly unrelated words.

I examined 50 words picked randomly from the 449 words that are involved in coarse-grained errors. Assuming that the remaining 302 words are all correct, the distribution of errors can be estimated as follows: 3.6% of words are represented in the wrong sense, 8.4% have to settle with a bad cluster, and 7.2% are assigned to the wrong cluster. Given that the quality of embeddings correlates with the amount of training data (Herbelot and Baroni, 2017) and most words in SimLex-999 are relatively frequent, it can be expected that the true error rates are higher. This result corroborates with the observation in our paper that word embeddings are worse than taxonomies at capturing similarity (as of 2015) and suggests that bad vectors might have a noticeable effect in increasing downstream errors.

Among the cases that I have examined, a word pair that receives lower-than-expected score is *wisdom-intelligence*. The Word2vec vector for *intelligence* counts among its nearest neighbors *intel*, *counterintelligence*, *counterterrorism*, and *humint* (intel gathered through interpersonal means). The model happens to encode a sense of *intelligence* that, though valid, is not what humans tend to think given the context. Contextualized word embeddings promise to resolve this problem by generating dynamic word vectors that vary depending on the surrounding words. In the second paper, **A Deep Dive into Word Sense Disambiguation with LSTM**, my colleagues and I explore how well such

models work and how much their performance depends on the amount of training data.

The existence of sense annotated datasets allows the direct evaluation of individual contextualized vectors. Given a dataset consisting of sentences annotated with word senses, a contextualized word embeddings model $\mathscr{E}$, and a simple classification model $\mathscr{M}$ that takes a word vector and predicts the sense of the given word, the accuracy of the classifier $\mathscr{M}$ can be thought of as reflecting the amount of sense information in the embeddings model $\mathscr{E}$. In other words, word sense disambiguation can be used as a probing task (Conneau et al., 2018) to evaluate contextualized word embedding models.

At one extreme, if $\mathscr{E}$ always predicts the same vector (as an embeddings lookup table does), the accuracy will be upper-bounded by the prevalence of the most frequent sense. At the other extreme, if $\mathscr{E}$ always return vectors that reflect the right sense in context, the accuracy is 100%. It is important that the classifier $\mathscr{M}$ is simple so that any gain in accuracy is explained by $\mathscr{E}$ only. For this reason, I will focus on a model called *averaging* which computes a prototype vector for each sense by taking the average of relevant vectors on a training set and uses such prototypes and simple cosine similarity for prediction.

The results show that the disambiguating ability of LSTM models (a type of recurrent neural networks) improve upon the most frequent sense baseline (MFS) but leaves much room for improvement. On senseval2 (an all-word dataset with 2,282 test instances), the model scores between 67.5% and 72.0% compared to 66.8% of MFS. On semeval2013 (a noun-only dataset with 1,644 annotations), its performance ranges from 64.7% to 65.6% compared to 63.0% of MFS. As expected, performance is dependent on the amount of unannotated data used for training. However, in the range that we test (18 million to 1.8 billion words), the improvement is small and the amount of data needed

for 1% improvement increases exponentially.

All in all, the research in this chapter suggests that a lot of work still needs to be done to improve the ability of embedding models in representing lexical similarity and polysemy, and doing so will reduce error propagation downstream. Since my research was concluded, more complex and performant models have been devised but they are also mostly evaluated at vocabulary- or task-level only. To open the blackboxes that NLP systems are today, it will be beneficial to continue the study of individual vector errors and how they affect downstream decisions.

Taxonomy Beats Corpus in Similarity Identification, but Does It Matter?

Minh Ngoc Le and **Antske Fokkens**
VU University Amsterdam
{m.n.le,antske.fokkens}@vu.nl

Abstract

We present extensive evaluations comparing the performance of taxonomy-based and corpus-based approaches on SimLex-999. The results confirm our hypothesis that taxonomy-based approaches are more suitable to identify similarity. We introduce two new measures of evaluation that show that all measures perform well on a coarse-grained evaluation and that it is not always clear which approach is most suitable when a similarity score is used as a threshold. This leads us to conclude that the inferior performance of corpus-based approaches may not (always) matter.

1. Introduction

Similarity measures are used in a wide variety of Natural Language Processing (NLP) tasks (see Pilehvar et al. (2013), among others for examples). They may be used, e.g. to increase coverage of an approach by using information from similar words for unseen data, or to establish average similarity between a question and a potential answer.

Due to its importance, similarity measures have received steady attention in computational linguistics. There are two

widely followed, but different, schools: taxonomy-based approaches and distributional, or corpus-based, approaches. Apart from a few exceptions, these approaches have mostly been studied separately.

Our main goal is to examine how the approaches perform when identifying true similarity, in contrast to the more general relatedness, which also includes association, between word-pairs. We evaluate the approaches on the new gold-standard SimLex-999 (Hill et al., 2014b). We compare taxonomy-based approaches that use WordNet (Fellbaum, 1998) to the corpus-based approaches that performed best on SimLex-999 in Hill et al. (2014a). We hypothesize that taxonomy-based approaches outperform corpus-based approaches on a true similarity set, because corpus-based approaches tend to mix-up similarity and association.

We carry out several evaluations which investigate (i) the difference in performance on pure similarity sets and sets that combine similarity and association, (ii) the influence of associative pairs while identifying true similarity, and (iii) various evaluation metrics that compare similarity measures to the gold standard of SimLex-999.

We perform more than one evaluation metric for two reasons. First, different ranking coefficients can lead to a completely different outcome when evaluating similarity scores (Fokkens et al., 2013a). Second, we want to gain more insight into the differences between individual measures. To do so, we introduced two new, more flexible, evaluation methods which reveal high results for all similarity measures. We argue that these new evaluations provide a better insight into how suitable similarity measures are to be used in NLP tasks than the commonly used Spearman's correlation (henceforth Spearman ρ).

Our results show that most of the evaluations confirm our hypothesis. The few cases where corpus-based methods outperformed taxonomy-based approaches reveal much smaller differ-

ences than the many cases where taxonomy-based approaches have higher results. However, all similarity measures perform very well when they are evaluated on the relative ranking of word-pairs that are further apart in the gold-standard. We therefore conclude that, even though taxonomy-based are better at identifying similarity than corpus-based approaches, this may not (always) matter.

The rest of this paper is structured as follows. In Section 2, we motivate our approach and address related work. Section 3 describes the similarity measures we investigate. In Section 4, we outline our experimental methodology, including used datasets and evaluation methods. The results are presented in Section 5, and our conclusions and future work in Section 6.

2. Background and Motivation

Several gold-standards have been created that rank word-pairs based on their similarity. Agirre et al. (2009) point out that association and similarity are mixed up in these sets, where associated pairs such as *coffee* and *cup* rank higher than truly similar pairs such as *car* and *train*. The confusion directly influences the performance of corpus-based approaches, which also tend to have difficulties distinguishing association from similarity (Hill et al., 2014a).

Hill et al. (2014b) introduce a new gold standard dataset that is annotated with pure semantic similarity and larger than previously created similarity sets, such as Rubenstein and Goodenough (1965) and Agirre et al. (2009)'s sets. Hill et al. (2014a) evaluate corpus-based approaches and show that they indeed have trouble identifying similarity, performing well-below the upperbound of agreement between human annotators.

It is not surprising that corpus-based approaches confuse similarity and association: semantically related words tend to occur close to each other and hence in similar contexts. Approaches

that make use of a relatively narrow context window perform slightly better, because they can capture more subtle differences in context to some extend.

Taxonomies represent word meanings in hypernym and hyponym hierarchies, directly capturing their similarity. The closer two terms are in the hierarchy, the more similar they are. Similarity measures that make use of this structure are less likely to confuse whether two terms are similar or related in some other way.

These well-known properties of corpus-based and taxonomy-based approaches led to the following hypothesis:

> Taxonomy-based approaches are better suited to identify similarity than corpus-based approaches

Agirre et al. (2009) seem to contradict this hypothesis showing that corpus-based approaches can be as good at identifying similarity (when the right model is based on enough data). However, Hill et al. (2014b) point out that Agirre et al.'s evaluation set does not form a representative set for measuring similarity, even after they made an alternative set that separates association and similarity. We therefore expected that the hypothesis would nevertheless hold on SimLex-999.

The outcome of our experiments confirmed our hypothesis, thus contradicting Agirre et al. (2009)'s results and being, to our knowledge, the first to show this on such a large and reliable benchmark. Banjade et al. (2015) also applies WordNet-based and corpus-based similarity measures to SimLex-999, but do not examine or discuss the difference between taxonomy-based approaches and corpus-based approaches in detail. Instead, they focus on the strength of combining several approaches to yield better results.[1] We investigate the difference between the approaches in various evaluations showing that taxonomy-based

[1] We independently confirmed this result in our own experiments, but decided

approaches outperform corpus-based approaches, a conclusion that cannot be drawn (clearly) from Banjade et al. (2015)'s results. It should be noted that our conclusions only apply to the task of identifying pure similarity. Markert and Nissim (2005) show, for instance, that a corpus-based approach with sufficiently large corpus works better than WordNet for anaphora resolution.

The next step in our investigation was to determine the strengths and weaknesses of each approach. The original idea was to investigate pairs that are ranked more or less correctly by one approach, but are far off in the other to identify patterns of errors in each approach. We did not find such patterns, partially because the examples that have large differences in ranking compared to the gold are relatively rare.

We therefore developed two alternative evaluation methods that are less sensitive to minor differences in ranking. The first evaluation directly tests the comparison of pairs and, more importantly, allows us to study the contribution of partitions of the dataset. The second evaluation revolves around thresholds for similarity. In this evaluation, we set thresholds to establish a binary distinction between highly similar pairs and other pairs. The pairs above the similarity threshold are compared to those falling above the threshold in the gold (see Section 2).

Many studies compare similarity measures (see Baroni et al. (2014) and Pedersen (2010), among others) but, to our knowledge, Agirre et al. (2009) and Banjade et al. (2015) are the only ones that look at both taxonomy-based approaches and distributional approaches. As mentioned above, they do not dive into the details of the differences between the two. Furthermore, apart from Fokkens et al. (2013a), who do not propose new rankings, we are not aware of studies applying multiple evaluation metrics for

to leave it out of this paper because our results did not add much to Banjade et al. (2015).

similarity-based rankings.

3. Similarity Measures

This section describes the similarity measures compared in this paper.

3.1. Taxonomy-based Similarity Measures

WordNet (Fellbaum, 1998) organizes nouns and verbs in hierarchies of hypernym-hyponym relations. We selected WordNet for our taxonomy-based experiments, because it is widely used and probably the most popular taxonomy when it comes to determining word similarity. Many measures of similarity based on WordNet have been proposed over the years. Early work (Rada et al., 1989) advocates the use of *is-a* hierarchy and later approaches continue to use it heavily. In order to make a clean comparison between WordNet and distributional models, we do not include in our study measures that make use of a corpus such as Resnik (1995) and Jiang and Conrath (1997).

Path length similarity takes the inverse of the path length (i.e. the distance in number of nodes) from s_1 to s_2 plus one.

$$\text{PL} = \frac{1}{\text{d}(s_1, s_2) + 1}$$

Wu and Palmer's similarity (Wu and Palmer, 1994) takes the fact into account that senses deeper in the hierarchy tend to be more specific than those high up. It therefore incorporates the depth of the hierarchy in their similarity calculation:

$$\text{WUP} = \frac{2\text{depth}(lcs)}{\text{d}(s_1, lcs) + \text{d}(s_2, lcs) + 2\text{depth}(lcs)}$$

Leacock and Chodorow's similarity (Leacock and Chodorow, 1998) normalizes path-based scores by the maximum depth D of the hierarchy. This corrects for the difference in the depth of verb

and noun hierarchy:

$$\text{LCH} = -\log \frac{\text{d}(s_1, s_2) + 1}{2D}$$

3.2. Distributional Semantic Models

We selected two representative models from the large and growing literature on corpus-based models of lexical semantics: Word2vec (Mikolov et al., 2013b, W2V) and dependency-based word embeddings (Levy and Goldberg, 2014a, DEPS).

Word2vec is the first model to use a Skip-Gram with Negative Sampling (SGNN) algorithm for constructing semantic models and performed best on SimLex-999 in Hill et al. (2014a). Levy and Goldberg (2014b) argue that SGNN implicitly factorizes a shifted positive mutual information word-context matrix, not unlike traditional distributional semantic models. The use of a small window size and the weighting scheme that favors nearby contexts are supported by a systematic study of Kiela and Clark (2014) that shows the superiority of small windows. Moreover, Sahlgren (2006) presents empirical evidence that smaller windows lead to a cleaner distinction between syntagmatic and paradigmatic relations (which can be considered the linguistic version of similarity and association).

Levy and Goldberg (2014a) extend SGNN to work with arbitrary contexts and experiment with dependency structures. It is generally believed that dependency structures are better at capturing similarity (Padó and Lapata, 2007) although Kiela and Clark (2014) found mixed results.

The Skip-gram model captures the distribution $p(c|t)$ of a context word c within a certain window around a target word t. For a vocabulary of millions, computing normalized probabilities (i.e. summing to one) for each example can be prohibitively expensive. Negative sampling was used to avoid the cost.

For each context-target pair (c,t) taken from training data, we replace the context by random words drawn from the vocabulary to obtain new pairs $\{(c',t)\}$. We call $D \ni (c,t)$ *positive distribution* and $N \ni (c',t)$ *negative distribution*. The task of the model is to identify which pairs come from D and which from N. Formally. that is to maximize the negative log likelihood:

$$\ell = -\left(\sum \log p(D|c,t) + \sum \log p(N|c',t)\right)$$

The probability is calculated using *target embeddings* $e_t \in \mathbb{R}^d$ and *context embeddings* $\hat{e}_c \in \mathbb{R}^d$ such that:

$$p(D|c,t) = \sigma(e_t \cdot \hat{e}_c),$$

where $\sigma(x) = 1/(1+e^{-x})$ is a monotonic function that maps any value in $(-\infty,+\infty)$ to a valid probability.

The training objective encourages to increase $p(D|c,t)$ which can be achieved by aligning e_t and $\hat{e}_c$ in similar directions. On the other hand, the objective also encourages a small $p(N|c,t)$, creating an uniform "repelling force" between all pairs of words. After a lot of updating iterations, similar words come close together while dissimilar words are pulled apart.

We used the trained embeddings from Mikolov et al. (2013b) and Levy and Goldberg (2014a).[2] Word2vec embeddings are 300-dimensional vectors obtained by training on 100 billion words of Google News dataset. Dependency-based embeddings were harvested from English Wikipedia automatically annotated with dependency structures. Although the dependency-based model was trained on a significantly smaller corpus, it achieves compa-

[2]The models are available at: `https://code.google.com/p/word2vec/` and `https://levyomer.wordpress.com/2014/04/25/dependency-based-word-embeddings`

rable results as we will show in Section 5.

4. Experimental Setup

In this section, we describe the experimental setup used in our evaluations. We first describe the datasets and then the evaluation metrics we use.

4.1. Gold-standard Datasets

We evaluate the approaches on three datasets. WordSim-353 and MEN allow us to compare performance on sets that mix association and similarity. SimLex-999's ranking is based on similarity only.

WordSim-353 (Finkelstein et al., 2001) includes 353 word pairs scored for *relatedness* on a scale from 0 to 10 by 13 or 16 subjects. The inter-annotator agreement is 0.611 defined as the average pairwise Spearman's correlation. Researchers have reported correlation as high as 0.81 (Yih and Qazvinian, 2012). Agirre et al. (2009) later divided WordSim-353 into a "similarity" and "relatedness" set. However, Hill et al. (2014b) rightly point out that both remain relatedness datasets, because this is what the annotators rated.

MEN (Bruni et al., 2012) is composed of 3,000 word pairs, sampled to include a balanced range of relatedness. Annotators were asked to choose which of two pairs of words is more related, an arguably more intuitive task than assigning a score.

SimLex-999 (Hill et al., 2014b) carefully distinguishes between similarity and association and provides a balanced range of similarity, concreteness and parts-of-speech. The authors sampled 900 associated pairs from the University of South Florida Free Association Database (Nelson et al., 2004) and randomly coupled them to create 999 unassociated pairs. Subjects were asked to judge the similarity of word pairs on a 0-6 scale. Their answers

were averaged to produce the final score.

All three datasets are lemma-based. The way two words can be compared, however, is more likely via their *senses* (e.g. *queen* is not similar to *princess* when referring to a chess piece). We follow Resnik (1995) in using maximally similar senses in our taxonomy-based approaches.

4.2. Evaluation Metrics

The first evaluation measure we use compares between the gold ranking and a measurement's ranking using **Spearman's** ρ , the most widely used evaluation metric for similarity score.

Hill et al. (2014b) report performance on a subset of highly associated word pairs, but its contribution to the overall performance is unclear. We wish to gain deeper insight into how different subsets in the data contribute to the overall score. This is not possible with Spearman's ρ due to its holistic nature. We overcome this by using **ordering accuracy** following Agirre et al. (2009). The scale is defined as:

$$a = a_{G,G} = \frac{1}{|G|^2} \sum_{(u,v)\in G} \sum_{(x,y)\in G} m_{s,G}(u,v,x,y)$$

where G stands for the gold standard and $m_{s,G}(\cdot)$ is a matching function that returns 1 for those two word-pairs whose relative ranking is the same in the gold standard and in the ranking of the similarity measure and 0 otherwise. We also experiment with a variation of m where ties get half score. As shown in Figure 3.2, ordering accuracy highly correlates with Spearman's ρ.

If G can be partitioned into n subsets g_i (i.e. $\bigcap g_i = \emptyset$ and $\bigcup g_i = G$) then a can be decomposed as the weighted sum of the accuracy on different subsets. The weights are proportional to

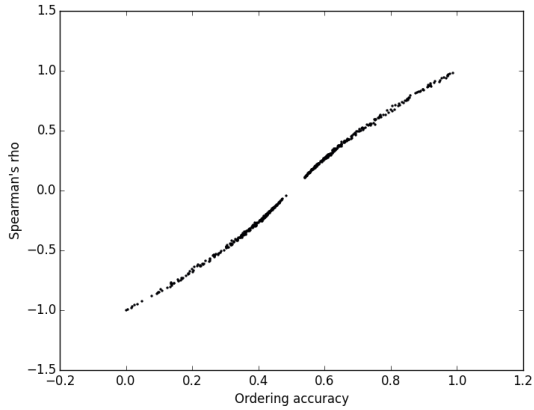


Figure 3.2: Ordering accuracy and Spearman's ρ on a synthesized dataset of 100 word pairs.

their size:

$$a = \frac{1}{|G|^2} \sum_i \sum_j |g_i||g_j| a_{g_i,g_j}$$

The final evaluation measure is based on the observation that many approaches use a threshold to determine which words are similar enough to be used for contributing features or approximations, or to be candidates for lexical substitution (McCarthy and Navigli, 2009; Biran et al., 2011, e.g.). **Threshold accuracy** sets a similarity threshold and determines how many of the n-highest ranking word pairs in a given measurement are also in the top-n pairs of the gold standard. In other words, this evaluation determines whether the right word-pairs would end up above the threshold of being similar.

5. Results

We calculated the similarity scores of all noun and verb pairs in SimLex-999 (a set of 888 pairs), MEN (2,034 pairs), and all pairs

Model	SL-999$_{nv}$	MEN$_{nv}$	WS-353
WUP	0.47	0.39	0.35
PL	0.52	0.39	0.30
LCH	0.55	0.39	0.31
W2V	0.42	0.77	0.70
DEPS	0.45	0.61	0.63

Table 3.1: Spearman's correlation of models to similarity benchmarks.

in WordSim-353 using the measures outlined in Section 3 and ranked the word pairs according to the outcome.

5.1. Spearman's Rank Correlation

Table 3.1 shows the performance of models on all three benchmarks. Taxonomy based approaches perform higher on SimLex-999, whereas corpus-based approaches reveal high performance on MEN and WordSim-353 and score significantly lower on SimLex-999. This result confirms that taxonomy-based approaches capture similarity rather than association, whereas corpus-based approaches do not clearly distinguish the two.

5.2. Ordering Accuracy

Table 3.2 presents the evaluation of our metrics using ordering accuracy. The first column indicates the standard score. The scores in the second and third column are calculated while giving partial credits to ties. Note that this only affects the performance of taxonomy-based approaches, where it is common for word pairs to have identical scores.

Without correction for ties, scores for taxonomy-based and corpus-based measures are highly similar, with the corpus-based DEPS leading to the highest results. Taxonomy-based approaches uniformly beat corpus-based approaches again when we do correct for ties, confirming the outcome of our Spearman ρ evaluation.

We also evaluate on a subset of highly-associated words. The

Model	SL-999 *nv*	SL-999 *nv*	SL-999 *nv,assoc*	Diff. *assoc*
		Using tie corrections		
WUP	64.9	66.6	67.3	+0.7
PL	61.1	68.0	68.2	+0.2
LCH	65.1	69.2	69.1	-0.1
W2V	64.4	64.6	57.5	-7.1
DEPS	65.5	65.6	60.9	-4.7

Table 3.2: Ordering accuracy (percentage) of similarity measures on SimLex-999$_{nv}$.

results are presented in column 3 of Table 3.2. Sizeable decrease is observed in corpus-based measures for highly associated terms while taxonomy-based measures remain largely unaffected. This result confirms our hypothesis once more that taxonomy-based measures are more suited to capture similarity and that corpus-based methods tend to have difficulties separating similarity from association.

5.3. Decomposition of Ordering Accuracy

Palmer et al. (2007) showed that making subtle sense distinction is hard for human subjects leading to evaluations where both coarse-grained and fine-grained word senses are considered (Palmer et al., 2007; Navigli et al., 2007). Similarly, establishing which word-pair is more similar than another is challenging when pairs are close in similarity. This is illustrated by the sample pairs in Table 3.3. The fact that ranking such pairs is highly challenging for humans leads to the question how meaningful differences in performance of similarities measures on these pairs actually are.

To overcome this issue and gain deeper insight into how often low performance is the result of many small errors piling up and how often it is the result of a set of pairs being ranked completely wrongly, we apply our ordering accuracy to a decomposed dataset. We divide SimLex-999$_{nv}$ into five equal similarity ranges $\{g_i\}$ based on SimLex-999's original ranges. The first range g_1 con-

$\Delta = 0$	
pollution-president	forget-learn
take-leave	succeed-try
army-squad	girl-child
emotion-passion	collect-save
sheep-lamb	attention-awareness
$\Delta = 1$	
spoon-cup	argue-differ
remind-sell	apple-candy
book-topic	argument-agreement
corporation-business	kidney-organ
alcohol-wine	beach-island

Table 3.3: Is the pair in the left or in the right more similar? (All pairs are extracted from SimLex-999)

tains highly dissimilar pairs of words with a similarity between 0 and 2. Final set g_5 contains very similar or synonymous pairs with a similarity from 8 to 10.

We use different granularity levels Δ ($\Delta = 0, ..., 4$). Component accuracy is calculated by comparing each pair in g_i to every pair in g_j such that $|i - j| = \Delta$.

The results reported in Figure 3.3 show that all models perform consistently well on coarse-grained similarity while only marginally beating chance-level at the most fine-grained level. Furthermore, taxonomy-based approaches only outperform corpus-based approaches when comparing pairs that are further apart in the gold ranking.

Because the two most fine-grained components ($\Delta = 0$ and $\Delta = 1$) together have a weight of 58%, the ordering accuracy as reported in Table 3.2 is dominated by fine-grained similarity comparison. Spearman's ρ highly correlates with ordering accuracy, indicating that fine-grained differences also had a major impact on previous work. It is questionable whether it is really necessary for these measures to capture the small differences in similarity that are even difficult for humans to find. This outcome shows that similarity measures perform better than they seem to

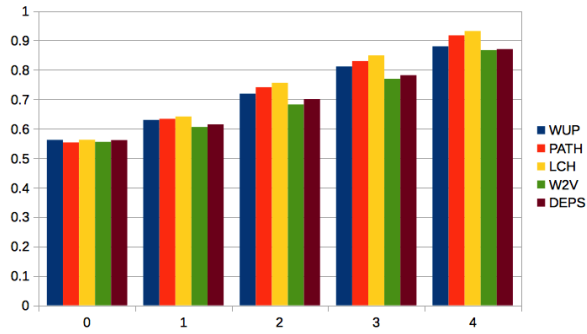


Figure 3.3: Ordering accuracy varies with degrees of granularity on SimLex-999$_{nv}$. $\Delta = 0$ means two pairs fall in the same range of similarity (e.g. 0-2); $\Delta = 1$ means they fall in neighboring ranges of similarity (e.g. 0-2 and 2-4), etc.

do according to recent evaluations in the literature.

5.4. Threshold Evaluation

The final evaluation we carry out is the so-called *threshold* evaluation. It evaluates how well a threshold performs that separates highly similar terms from less similar terms based on a specific score. We use the 10% and 20% most similar terms as a starting point. In a total set of 888 examples, this means we compare the top 89 and top 178 pairs of each measurement's output with the top pairs of the gold data. We report on the accuracy (i.e. percentage of pairs correctly classified as highly similar) of each scores. As mentioned above, taxonomy-based approaches often assign the same score to multiple pairs. If this was the case for the pairs around the threshold, we extended the range of comparison as to include all pairs with an identical score. Table 3.4 provides an overview of the results.

The top-*n* sets increase significantly for taxonomy-based approaches. Because approaches tend to fare better when the size of the group changes, we calculated the scores for W2V and DEPS with the top-*n* ranks found in the taxonomy-based scores.

Model	10%-based		20%-based	
	n	%	*n*	%
WUP	94	42.6	191	50.3
PATH	172	43.5	645	80.8
LCH	172	53.5	305	61.0
W2V	89	32.6	178	38.2
DEPS	89	33.7	178	43.8

Table 3.4: Threshold based evaluation, comparing the set of top-*n* similar pairs

model	*n*-value				
	94	172	191	305	645
W2V	33.0	38.4	39.8	48.5	**82.0**
DEPS	31.9	43.6	42.9	52.8	81.4
taxo.	**42.6**	43.5/**53.5**	**50.3**	**61.0**	80.8

Table 3.5: Scores of corpus-based methods on the *n*-values used for taxonomy-based scores.

Table 3.5 shows the results of this analysis. The scores of the relevant taxonomy-based approach are repeated in the third row.

The threshold based evaluation shows more variation than our other metric. In three out of twelve cases,[3] the corpus-based approach leads to more accurate results than the taxonomy-based score. In combination with the outcome of the accuracy ordering result, this outcome underlines the importance of using a variety of evaluation metrics.

Overall, the outcome seems to confirm that taxonomy-based approaches are better at identifying similarity. First, taxonomy-based approaches outperformed corpus-based approaches on identifying the most accurate pairs. Second, corpus-based approaches only beat taxonomy-based ones in few measures and with comparatively small margins (the largest difference being 1.2%, com-

[3] We compare eight corpus-based outcomes with one taxonomy score and two with two scores for *n*=172, leading to twelve comparisons in total.

pared to differences up to 15.1%).

6. Discussion and Conclusions

This paper investigated the difference in performance of taxonomy-based approaches and corpus-based approaches on identifying similarity. The outcome of our experiments confirmed our hypothesis that taxonomy-based approaches are better at identifying similarity. This is mainly due to the fact that corpus-based approaches have difficulties distinguishing association from similarity, as also noted by Hill et al. (2014a).

We presented several results that confirm our hypothesis by (i) comparing performance of taxonomy-based and corpus-based methods on a dataset designed to capture similarity, (ii) relating this to the results of the same measures on evaluation sets that measure both association and relatedness, and (iii) looking what the influence is of testing against a set that consists of associated terms.

The results show that taxonomy-based approaches excel at identifying similarity whereas corpus-based approaches yield high results when similarity and association are not distinguished. Furthermore, taxonomy-based approaches are not influenced by association between words whereas performance of corpus-based measures drop when their task is to identify similarity.

We applied more than one evaluation to compare the models' performance on SimLex-999. This was done for two reasons. First, different evaluation measures can sometimes lead to different conclusions even if they are meant to address the same question on the same dataset. This also happened in our evaluation, where ordering accuracy without tie-correction and some thresholds led to different results. Second, the evaluation metrics revealed different aspects of the performance. Most notably, the results of our decomposed ordering accuracy showed that all similarity measures are quite good in a coarse-grained setting.

Together with the mixed outcome of the threshold-evaluation, this shows that corpus-based approaches have good potential to be used when similarity needs to be detected. In particular, when taxonomy-based approaches run into coverage issues, they may be the preferred choice. We therefore believe that it will ultimately depend on the application which approach works best. Future work will need to show whether and how these approaches differ when used in actual applications. All our code is published on `https://bitbucket.org/ulm4/kcsim`.

Acknowledgments

The research for this paper was supported by the Netherlands Organisation for Scientific Research (NWO) via the Spinoza-prize Vossen projects (SPI 30-673, 2014-2019) and the BiographyNet project (Nr. 660.011.308), funded by the Netherlands eScience Center (`http://esciencecenter.nl/`). We would like to thank the anonymous reviewers for their feedback.

Appendix: Answers to sample questions in Table 3.3

1. right, 2. right, 3. right, 4. left, 5. right, 6. right, 7. right, 8. left, 9. left, 10. left.

A Deep Dive into Word Sense Disambiguation with LSTM

Minh Le[†], **Marten Postma**[†], **Jacopo Urbani**[‡] and **Piek Vossen**[†]
[†] Department of Language, Literature and Communication
[‡] Department of Computer Science
Vrije Universiteit Amsterdam
{m.n.le,m.c.postma,piek.vossen}@vu.nl, jacopo@cs.vu.nl

Abstract

LSTM-based language models have been shown effective in Word Sense Disambiguation (WSD). In particular, the technique proposed by Yuan et al. (2016b) returned state-of-the-art performance in several benchmarks, but neither the training data nor the source code was released. This paper presents the results of a reproduction study and analysis of this technique using only openly available datasets (GigaWord, SemCor, OMSTI) and software (TensorFlow). Our study showed that similar results can be obtained with much less data than hinted at by Yuan et al. (2016b). Detailed analyses shed light on the strengths and weaknesses of this method. First, adding more unannotated training data is useful, but is subject to diminishing returns. Second, the model can correctly identify both popular and unpopular meanings. Finally, the limited sense coverage in the annotated datasets is a major limitation. All code and trained models are made freely available.

1. Introduction

Word Sense Disambiguation (WSD) is a long-established task in the NLP community (see Navigli (2009) for a survey) which goal

is to annotate lemmas in text with the most appropriate meaning from a lexical database like WordNet (Fellbaum, 1998). Many approaches have been proposed – the more popular ones include the usage of Support Vector Machine (SVM) (Zhong and Ng, 2010), SVM combined with unsupervised trained embeddings (Iacobacci et al., 2016; Rothe and Schütze, 2017), and graph-based approaches (Agirre et al., 2014; Weissenborn et al., 2015).

In recent years, there has been a surge in interest in using Long short-term memory (LSTM) (Hochreiter and Schmidhuber, 1997) to perform WSD (Raganato et al., 2017b; Melamud et al., 2016). These approaches are characterized by their high performance, simplicity and their ability to extract a lot of information from raw text. Among the best-performing ones is the approach by Yuan et al. (2016b), in which an LSTM language model trained on a corpus with 100 billion tokens was coupled with small sense-annotated datasets to achieve state-of-the-art performance in all-words WSD.

Even though the results obtained by Yuan et al. (2016b) outperform the previous state-of-the-art, neither the used datasets nor the constructed models are available to the community. This is unfortunate because this makes the re-application of this technique a non-trivial process, and it hinders further studies for understanding which limitations prevent even higher accuracies. These could be, for instance, of algorithmic nature or relate to the input (either size or quality), and a deeper understanding is crucial for enabling further improvements. In addition, some details are not reported, and this could prevent other attempts from replicating the results.

To address these issues, we reimplemented Yuan et al. (2016b)'s method with the goal of: 1) reproducing and making available the code, trained models, and results and 2) understanding which are the main factors that constitute the strengths and weaknesses of this method. While a full replication is not possible due to the

unavailability of the original data, we nevertheless managed to reproduce their approach with other public text corpora, and this allowed us to perform a deeper investigation on the performance of this technique. This investigation aimed at understanding how sensitive the WSD approach is w.r.t. the amount of unannotated data (i.e., raw text) used for training, model complexity, how biased the method is towards the choice of the most frequent senses (MFS), and identifying limitations that cannot be overcome with bigger unannotated datasets.

The contribution of this paper is thus two-fold: On the one hand, we present a reproduction study whose results are publicly available and hence can be freely used by the community. Notice that the lack of available models has been explicitly mentioned, in a recent work, as the cause for the missing comparison of this technique with other competitors (Raganato et al., 2017b, footnote 10). On the other hand, we present other experiments to shed more light on the value of this and similar methods.

We anticipate some conclusions. First, a positive result is that we were able to reproduce the method from Yuan et al. (2016b) and obtain similar results to the ones originally published. However, to our surprise, these results were obtained using a much smaller corpus of 1.8 billion tokens (Gigaword), which is less than 2% of the data used in the original study. In addition, we observe that the amount of unannotated data is important, but that the relationship between its size and the improvement is not linear, meaning that exponentially more unannotated data is needed in order to improve the performance. Moreover, we show that the percentage of correct sense assignments is more balanced w.r.t sense popularity, meaning that the system has a less-strong bias towards the most-frequent sense (MFS) and is better at recognizing both popular and unpopular meanings. Finally, we show that the limited sense coverage in the annotated datasets is a major limitation, as shown by the fact that resulting model does not

have a representation for more than 30% of the meanings which should have been considered for disambiguating the test sets.

2. Background

Current WSD systems can be categorized according to two dimensions: whether they use raw text without any preassigned meaning (unannotated data henceforth), and whether they exploit the relations between synsets in WordNet (synset relations henceforth). One prominent state-of-the-art system that does not rely on unannotated data nor exploits synset relations is It Makes Sense (IMS) (Zhong and Ng, 2010; Taghipour and Ng, 2015). This system uses an SVM to train classifiers for each lemma using only annotated data as training evidence.

In contrast, graph-based WSD systems do not use (un)annotated data but rely on the synset relations. The system UKB (Agirre et al., 2014) represents WordNet as a graph where the synsets are the nodes and the relations are the edges. After the node weights have been initialized using the Personalized Page Rank algorithm, they are updated depending on context information. Then, the synset with the highest weight is chosen. Babelfy (Moro et al., 2014) and the system by Weissenborn et al. (2015) both represent the whole input document as a graph with synset relations as edges and jointly disambiguate nouns and verbs. In the case of Babelfy, a densest-subgraph heuristic is used to compute the high-coherence semantic interpretations of the text. Instead, in Weissenborn et al. (2015) a set of complementary objectives, which include sense probabilities and type classification, are combined together to perform WSD.

A number of systems make use of both unannotated data and synset relations. Both Tripodi and Pelillo (2017) and Camacho-Collados et al. (2016) make use of statistical information from unannotated data to weigh the relevance of nodes in a graph, which is then used to perform WSD. Rothe and Schütze (2017)

use word embeddings as a starting point and then rely on the formal constraints in a lexical resource to create synset embeddings.

Recently, there has been a surge in WSD approaches that use unannotated data but do not consider synset relations. One example is provided by Iacobacci et al. (2016), who investigated the role of word embeddings as features in a WSD system. Four methods (concatenation, average, fractional decay, and exponential decay) are used to extract features from the sentential context using word embeddings. The features are then added to the default feature set of IMS (Zhong and Ng, 2010). Moreover, Raganato et al. (2017b) present a number of end-to-end neural WSD architectures. The best performing one is based on a bidirectional Long Short-Term Memory (BLSTM) with attention and two auxiliary loss functions (part-of-speech and the WordNet coarse-grained semantic labels). Melamud et al. (2016) also make use of unannotated data to train a BLSTM. The work by Yuan et al. (2016b), which we consider in this paper, belongs to this last category. Different from Melamud et al. (2016), it uses significantly more unannotated data, the model contains more hidden units (2048 vs. 600), and the sense assignment is more elaborated. We describe this approach in more detail in the following section.

3. WSD with Language Models

The method proposed by Yuan et al. (2016b) performs WSD by annotating each lemma in a text with one WordNet synset that is associated with its meaning. Broadly speaking, the disambiguation is done by: 1) constructing a language model from a large unannotated dataset; 2) extracting sense embeddings from this model using a much smaller annotated dataset; 3) relying on the sense embeddings to make predictions on the lemmas in unseen sentences. Each operation is described below.

Constructing Language Models. Long Short-Term Memory

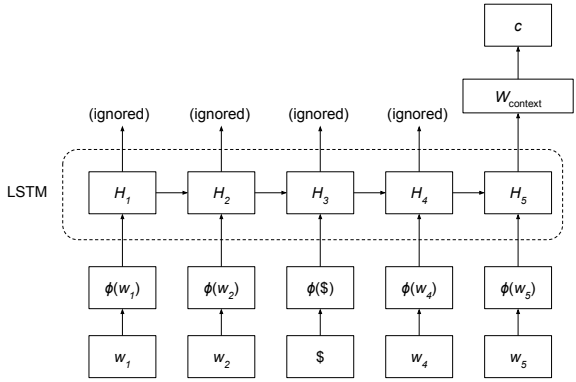


Figure 3.4: The LSTM model used to perform language modeling and compute context embeddings. At training time, a softmax layer is added, allowing it to predict the omitted word; at test time, the context embeddings are used for WSD in a nearest-neighbor or label-propagation procedure.

(LSTM) (Hochreiter and Schmidhuber, 1997) is a celebrated recurrent neural network architecture that has proven to be effective in many natural language processing tasks (Sutskever et al., 2014; Dyer et al., 2015; He et al., 2017, among others). Different from previous architectures, LSTM is equipped with trainable gates that control the flow of information, allowing the neural networks to learn both short- and long-range dependencies.

In Yuan et al. (2016b), the first operation consists of constructing an LSTM language model to capture the meaning of words in context. They use an LSTM network with a single hidden layer of h nodes. Given a sentence $s = (w_1, w_2, \ldots, w_n)$, they replace word $w_k (1 \leq k \leq n)$ by a special token \$. The model takes this new sentence as input and produces a context vector $\mathbf{c}$ of dimensionality p (see Figure 3.4).[1]

Each word w in the vocabulary $\mathcal{V}$ is associated with an embedding $\phi_o(w)$ of the same dimensionality. The model is trained

[1] As usual, vectors are indicated with boldface to distinguish them to scalar and other symbols.

to predict the omitted word, minimizing the softmax loss over a big collection $\mathscr{D}$ of sentences.

$$\ell = -\sum_{s\in\mathscr{D}}\sum_{k=1}^{|s|}\log\frac{\exp(\mathbf{c}\cdot\phi_{\mathrm{o}}(w_k))}{\sum_{w'\in\mathscr{V}}\exp(\mathbf{c}\cdot\phi_{\mathrm{o}}(w'))}$$

After the model is trained, we can use it to extract *context* embeddings, i.e., latent numerical representations of the sentence surrounding a given word.

Calculating Sense Embeddings. The model produced by the LSTM network is meant to capture the "meaning" of words in the context they are mentioned. In order to perform the sense disambiguation, we need to extract from it a suitable representation for word senses. To this purpose, the method relies on another corpus where each word is annotated with the corresponding sense.

The main intuition is that words used with the same sense are mentioned in contexts which are very similar to each other as well. This suggests a simple way to calculate sense embeddings. First, the LSTM model is invoked to compute the context vector for each occurrence of one sense in the annotated dataset. Once all context vectors are computed, the sense embedding is defined as the average of all vectors. Let us assume, for instance, that the sense $horse_n^2$ (that is, the second sense of horse as a noun) appears in the two sentences:

(1) The move of the $horse_n^2$ to the corner forced the checkmate.

(2) Karjakin makes up for his lost bishop a few moves later, trading rooks and winning black's $horse_n^2$.

In this case, the method will replace the sense by \$ in the sentences and feed them to the trained LSTM model to calculate two context vectors $\mathbf{c_1}$ and $\mathbf{c_2}$. The sense embedding $\mathbf{s_{horse_n^2}}$ is then

computed as:

$$\mathbf{s}_{\mathbf{horse}^2_\mathbf{n}} = \frac{\mathbf{c_1} + \mathbf{c_2}}{2}$$

This procedure is computed for every sense that appears in the annotated corpus.

Averaging technique to predict senses. After all sense embeddings are computed, the method is ready to disambiguate target words. This procedure proceeds as follows:

1. Given an input sentence and a target word, it replaces the occurrence of the target word by \$ and uses the LSTM model to predict a context vector $\mathbf{c_t}$.
2. The lemma of the target word is used to retrieve from WordNet the candidate synsets $s_1, \ldots, s_n$ where n is the number of synsets. Then, the procedure looks up the corresponding sense embeddings $\mathbf{s_1}, \ldots, \mathbf{s_n}$ computed in the previous step.
3. The procedure invokes a subroutine to choose one of the n senses for the context vector $\mathbf{c_t}$. It selects the sense whose vector is closest to $\mathbf{c_t}$ using cosine as the similarity function.

Label Propagation. Yuan et al. (2016b) argue that the averaging procedure is suboptimal because of two reasons. First, the distribution of occurrences of senses is unknown whereas averaging is only suitable for spherical clusters. Second, averaging reduces the representation of occurrences of each sense to a single vector and therefore ignores sense prior. For this reason, they propose to use label propagation for inference as an alternative to averaging. Label propagation (Zhu and Ghahramani, 2002) is a classic semi-supervised algorithm that has been employed in WSD (Niu et al., 2005) and other NLP tasks (Chen et al., 2006; Zhou, 2011). The procedure involves predicting senses for not only the target cases but also for unannotated words queried from a corpus. It represents both the target cases and unannotated

words as points in a vector space and iteratively propagates classification labels from the target classes to the words. In this way, it can be used to construct non-spherical clusters and to give more influence to frequent senses.

Overall algorithm. The overall disambiguation procedure that we implemented proceeds as follows:

1. *Monosemous:* First, the WSD algorithm checks whether the target lemma is monosemous (i.e., there is only one synset). In this case, the disambiguation is trivial.
2. *Label propagation:* If the label propagation is enabled, then it checks whether the target lemma occurs at least once in the annotated dataset and at least once in the auxiliary unannotated dataset. In this case, the procedure applies the *label propagation* technique for selecting the candidate synset.
3. *Averaging:* If the previous strategies are not applicable and there is at least one occurrence of the target lemma in the annotated dataset, then we apply the *averaging* technique for selecting the candidate synset.
4. *MFS fallback:* If the target lemma does not appear in the annotated dataset, then the system picks the most-frequent synset.[2]

4. Reproduction Study: Methodology

Before we report the results of our experiments, we describe the datasets used and give some details regarding our implementation.

Training data. The 100-billion-token corpus used in the original publication is not publicly available. Therefore, for the training of the LSTM models, we used the English Gigaword Fifth Edition (Linguistic Data Consortium (LDC) catalog number

[2]Notice that Yuan et al. (2016b) did not report if they use an MFS-fallback strategy or simply returned no answer.

LDC2011T07). The corpus consists of 1.8 billion tokens in 4.1 million documents, originated from four major news agencies. We leave the study of bigger corpora for future work.

For the training of the sense embeddings, we use the same two corpora used by Yuan et al. (2016b):

1. *SemCor* (Miller et al., 1993a) is a corpus containing approximately 240,000 sense annotated words. The tagged documents originate from the Brown corpus (Francis and Kucera, 1979) and cover various genres.
2. *OMSTI* (Taghipour and Ng, 2015) contains one million sense annotations automatically tagged by exploiting the English-Chinese part of the parallel MultiUN corpus (Eisele and Chen, 2010). A list of English translations were manually created for each WordNet sense. If the Chinese translation of an English word matches one of the manually curated translations for a WordNet sense, that sense is selected.

Implementation. We used the BeautifulSoup HTML parser to extract plain text from the Gigaword corpus. Then, we used the English models[3] of Spacy 1.8.2 for sentence boundary detection and tokenization. The LSTM model is implemented using TensorFlow 1.2.1 (Abadi et al., 2015). We chose TensorFlow because of its industrial-grade quality and because it can train large-scale models.

The main computational bottleneck of the entire process is the training of the LSTM model. Although we do not use a 100-billion-token corpus, training the model on Gigaword can already take years if not optimized properly. To reduce training time, we assumed that all (padded) sentences in the batch have the same length. This optimization increases the speed by 17% as measured on a smaller model ($h = 100, p = 10$). Second, following Yuan et

[3] `en_core_web_md-1.2.1`

al., we use the sampled softmax loss function (Jean et al., 2015). Third, we grouped sentences of similar length together while varying the number of sentences in a batch to fully utilize GPU RAM. Together, these heuristics increased training speed by 42 times.

Although Yuan et al. proposed to use a distributed implementation of label propagation (Ravi and Diao, 2016), we found that scikit-learn (Pedregosa et al., 2011) was fast enough for our experiments. For hyperparameter tuning, we use the annotations in OMSTI (which are not used at test time). After measuring the performance of some variations of label propagation (scikit-learn implementation: *LabelPropagation* or *LabelSpreading*; similarity measure: *inner product* or radial basis function with different values of γ), we found that the combination of *LabelSpreading* and *inner product* similarity leads to the best result which is also better than averaging on the development set.

Evaluation framework. For evaluating the WSD predictions, we selected two test sets: one from the Senseval2 (Palmer et al., 2001) competition, which tests the disambiguation of nouns, verbs, adjectives and adverbs, and one from the 2013 edition (Navigli et al., 2013b), which focuses only on nouns.

The test set from Senseval-2 is the English All-Words Task; *senseval2* henceforth. This dataset contains 2,282 annotations from three articles from the Wall Street Journal. Most of the annotations are nominal, but the competition also contains annotations for verbs, adjectives, and adverbs. In this test set, 66.8% of all target words are annotated with the most-frequent sense (MFS) of the lemma. This means that the simple strategy of always selecting the MFS would score 66.8% F_1 on this dataset.

The test set from SemEval-2013 is the one taken from task 12: Multilingual Word Sense Disambiguation; *semeval2013* henceforth. This task consists of two disambiguation tasks: Entity Linking and Word Sense Disambiguation for English, German,

French, Italian, and Spanish. This test set contains 13 articles from previous editions of the workshop on Statistical Machine Translation.[4] The articles contain 1,644 test instances in total, which are all nouns. The application of the MFS baseline on this dataset yields an F_1 score of 63.0%.

5. Results

In this section, we report our reproduction of the results of Yuan et al. (2016b) and additional experiments to gain a deeper insight into the strengths and weaknesses of the approach. These experiments focus on the performance on the most- and less-frequent senses, coverage of the annotated dataset and the consequent impact on the overall predictions, the granularity of the sense representation, and the impact of the unannotated data and model complexity on the accuracy of WSD.

Reproduction results. We trained the LSTM model with the best reported settings in Yuan et al. (2016b) (hidden layer size $h = 2048$, embedding dimensionality $p = 512$) using a machine equipped with an Intel Xeon E5-2650, 256GB of RAM, 8TB of disk space, and two nVIDIA GeForce GTX 1080 Ti GPUs. During our training, one epoch took about one day to finish with TensorFlow fully utilizing one GPU. The whole training process took four months. We tested the performance of the downstream WSD task three times during the training and observed that the best performance is obtained at the 65$^{\text{th}}$ epoch, despite a later model producing a lower negative log-likelihood. Thus, we used the model produced at the 65$^{\text{th}}$ epoch for our experiments below.

Table 3.8 (p. 91) presents the results using the test sets *senseval2* and *semeval2013*, respectively. The top part of the table presents our reproduction results, the middle part reports the results from Yuan et al. (2016b), while the bottom part reports a

[4] `http://www.statmt.org`

representative sample of the other state-of-the-art approaches.

It should be noted that with the test set *semeval2013*, all scorers use WordNet 3.0, therefore the performance of the various methods can be directly compared. However, not all answers in *senseval2* can be mapped to WN3.0 and we do not know how Yuan et al. (2016b) handled these cases. In the WSD evaluation framework (Moro et al., 2014) that we selected for evaluation, these cases were either re-annotated or removed. Thus, our F_1 on *senseval2* cannot be directly compared with the F_1 in the original paper.

From a first glance at Table 3.8 (p. 91), we observe that if we use SemCor to train the synset embeddings, then our results come close to the state-of-the-art on *senseval2* (0.720 vs. 0.733). On *semeval2013*, we achieve results comparable to other embeddings-based approaches (Raganato et al., 2017b; Iacobacci et al., 2016; Melamud et al., 2016). However, the gap with the graph-based approach of Weissenborn et al. (2015) is still significant. When we use both SemCor and OMSTI for the annotated data, our results drop 0.02 point for *senseval2*, whereas they increase by almost 0.01 for *semeval2013*. Different from Yuan et al. (2016b), we did not observe improvement by using label propagation (comparing *T: SemCor, U: OMSTI* against *T:SemCor* without propagation). However, the performance of the label propagation strategy is still competitive on both test sets.

Most- vs. less-frequent-sense instances. The original paper only analyses the performance on the whole test sets. We extend this analysis by looking at the performance for disambiguating the most frequent-sense (MFS) and less-frequent-sense (LFS) instances. The first type of instances are the ones for which the correct link is the most-frequent sense, whereas the second subset consists of the remaining ones. This analysis is important because it is well-known that the simple strategy of always choosing the MFS is a strong baseline in WSD, thus there is a tendency for

WSD systems to overfit towards the MFS (Postma et al., 2016).

Table 3.6 (p. 90) shows that the method by Yuan et al. (2016b) does not overfit towards the MFS to the same extent as other supervised systems since the recall on LFS instances is still quite high 0.41 (a lower recall on LFS instances than on MFS ones is expected due to the reduced training data for them).

On semeval13, the recall on LFS is already relatively high using only SemCor (0.33), and reaches 0.38 when using both SemCor and OMSTI. For comparison, the default system IMS (Zhong and Ng, 2010) trained on SemCor only obtains an *R_lfs* of 0.15 on semeval13 (Postma et al., 2016) and only reaches 0.33 with a large amount of annotated data.

Finally, our implementation of the label propagation does seem to slightly overfit towards the MFS. When we compare the results of the *averaging technique* using SemCor and OMSTI versus when we use *label propagation*, we notice an increase in the MFS recall (from 0.85 to 0.91), whereas the LFS recall drops from 0.40 to 0.32.

Meaning coverage in annotated datasets. The WSD procedure depends on an annotated corpus to compose its sense representations, making missing annotations an insurmountable obstacle. In fact, annotated datasets only contain annotations for a proper subset of the possible candidate synsets listed in WordNet. We analyze this phenomenon using four statistics:

1. *Candidate Coverage:* For each test set, we performed a lookup in WordNet to determine the unique candidate synsets of all target lemmas. We then determined what percentage of these candidate synsets that have *at least one* annotation in the annotated dataset.
2. *Lemma Coverage:* Given a target lemma in a test set, we performed a lookup in WordNet to determine the unique candidate synsets. If *all* candidate synsets of that target lemma have at least one annotation in the annotated dataset,

we claim that the lemma is covered. The lemma coverage is then the percentage of all covered target lemmas. A high lemma coverage indicates that annotated dataset covers most of the meanings in the test set.

3. *Gold Coverage:* We calculate the percentage of the *correct* answers in the test set that has at least one annotation in the annotated dataset.

The column "Candidate Coverage" of Table 3.9 (p. 92) shows that SemCor only contains less than 70% of all candidate synsets for *senseval2* and *semeval2013*, meaning that a model will never have a representation for more than 30% of the candidate synsets. Even with the addition of OMSTI, the coverage does not exceed 70%, meaning that we lack evidence for a significant number of potential annotations. Moreover, the column "Lemma Coverage" illustrates that we have evidence for all potential solutions for only 30% of the lemmas in both WSD competitions, meaning that in the large majority of the cases some solutions are never seen. The column "Gold coverage" measures whether the right answers are at least seen in the annotated dataset. The numbers illustrate that 20% of the solutions in the test sets do not have any annotations. With our approach, these answers can only be returned if the lemma is monosemous or by random guess otherwise.

To further investigate these issues, Table 3.10 (p. 92) reports the recall of the various disambiguation strategies which could be invoked depending on the coverage of the lemma (these can be: *monosemous*, *averaging*, *label propagation*, *MFS* – see the overall procedure reported in Section 3). We observe that the MFS fallback plays a significant role in obtaining the overall high accuracy since it is invoked many times, especially with OMSTI due to the low coverage of the dataset (in this case it is invoked in 775 cases vs. 1072 of averaging). For example, if we had not applied the MFS fallback strategy for *senseval2* using

SemCor as the annotated corpus, our performance would have dropped from 0.72 to 0.66, below the MFS baseline of 0.67 for this task.[5] Label propagation was indeed applied on half of the cases, but leads to lower results. From these results, we learn that the effectiveness of this method strongly depends on the coverage of the annotated datasets: If it is not high, as it is with OMSTI, then the performance of this method reduces to the one of choosing the MFS.

Granularity of sense representation. Rothe and Schütze (2017) provided evidence for the claim that the granularity of the sense representations has an influence on WSD performance. More in particular, their WSD system performed better when trained on sensekeys (called *lexemes* in their paper) than on synsets. Although a sensekey-based disambiguation results in less annotated data per target lemma, the sensekey representation is more precise (since it is a lemma associated with a particular meaning) than at the synset level.

The reimplementation discussed in this paper allows us to answer the question: "How will LSTM models work if we lower the disambiguation level from synset to sensekey?" Table 3.7 (p. 90) presents the results of this experiment. As we can see from the table, our method also returns better performance on both test sets. This behavior is interesting and one possible explanation is that sensekeys are more discriminative than synsets and this favors the disambiguation.

Impact of unannotated data and model size. Since unannotated data is abundant, it is tempting to use more and more data to train language models, hoping that better word embeddings would translate into improved WSD performance. The fact that Yuan et al. (2016b) used a 100-billion-token corpus only reinforces this

[5] *senseval2* contains 2,282 instances, of which the system would answer incorrectly 135 instances if the MFS fallback strategy is not used, hence dropping 0.06 in performance.

intuition. We empirically evaluate the effectiveness of unlabeled data by varying the size of the corpus used to train LSTM models and measure the corresponding WSD performance. More in particular, the size of the training data was set at 1%, 10%, 25%, and 100% of the GigaWord corpus (which contains 1.8×10^7, 1.8×10^8, 4.5×10^8 and 1.8×10^9 words, respectively).

Figure 3.5a (p. 93) shows the effect of unannotated data volume on WSD performance. The data points at 100 billion (10^{11}) tokens correspond to Yuan et al. (2016b)'s reported results. As might be expected, a bigger corpus leads to more meaningful context vectors and therefore higher performance on WSD. However, the amount of data needed for 1% of improvement in F_1 grows exponentially fast (notice that the horizontal axis is in log scale). Extrapolating from this graph, to get a performance of 0.8 F_1 by adding more unannotated data, one would need a corpus of 10^{12} tokens. This observation also applies to the balance of the sense assignment. Using only 25% of the unannotated data already yields a recall of 35% on the less-frequent senses.

In addition, one might expect to push the performance further by increasing the capacity of the LSTM models. To evaluate this possibility, we performed an experiment in which we varied the sizes of LSTM models trained on 100% of the GigaWord corpus and evaluated against *senseval2* and *semeval2013*, respectively. Figure 3.5b (p. 93) suggests that it is possible but one would need exponentially bigger models.

Finally, Reimers and Gurevych (2017) have showed that it is crucial to report the distribution of test scores instead of only one score as this practice might lead to wrong conclusions. As pointed out at the beginning of Section 5, our biggest models take months to train, making training multiple versions of them impractical. However, we trained our smallest model ($h = 100, p = 10$) ten times and our second smallest model ($h = 256, p = 64$) five times and observed that as the number of parameters increased, the stan-

dard deviation of F_1 decreased from 0.008 to 0.003. We, therefore, believe random fluctuation does not affect the interpretation of the results.

6. Conclusions

This paper reports the results of a reproduction study of the model proposed by Yuan et al. (2016b) and an additional analysis to gain a deeper understanding of the impact of various factors on its performance.

A number of interesting conclusions can be drawn from our results. First, we observed that we do not need a very large unannotated dataset to achieve state-of-the-art all-words WSD performance since we used the Gigaword corpus, which is two orders of magnitude smaller than Yuan et al. (2016b)'s proprietary corpus, and got similar performance on *senseval2* and *semeval2013*. A more detailed analysis hints that adding more unannotated data and increasing model capacity are subject to diminishing returns. Moreover, we observed that this approach has a more balanced sense assignment than other techniques, as shown by the relatively good performance on less-frequent-sense instances. In addition, we identified that the limited sense coverage in annotated dataset places a potentially upper bound for the overall performance.

The code with detailed replication instructions is available at: `https://github.com/cltl/wsd-dynamic-sense-vector` and the trained models at: `https://doi.org/10.6084/m9.figshare.6352964.v1`.

Acknowledgements

The research for this paper was supported by the Netherlands Organisation for Scientific Research (NWO) via the Spinoza-prize Vossen projects (SPI 30-673, 2014-2019). We thank the support of the NWO project scilens (`https://projects.cwi.nl/scilens`) for providing the hardware to run some of the experi-

ments. Experiments were also carried out on the Dutch national e-infrastructure with the support of SURF Cooperative. We would like to thank our friends and colleagues Antske Fokkens, Emiel van Miltenburg, Chantal van Son, Pia Sommerauer, and Roxane Segers for many useful comments and discussions.

Model	senseval2 F_1	senseval2 R_mfs (n=1524)	senseval2 R_lfs (n=758)	semeval2013 F_1	semeval2013 R_mfs (n=1035)	semeval2013 R_lfs (n=609)
Our LSTM (T: SemCor)	**0.72**	0.88	**0.41**	0.65	0.84	0.33
Our LSTM (T: OMSTI)	0.67	0.87	0.27	0.65	**0.86**	0.29
Our LSTM (T: SemCor+OMSTI)	0.70	0.85	0.40	**0.66**	0.82	**0.38**
Our LSTMLP (T: SemCor+OMSTI)	0.71	**0.91**	0.32	0.64	0.85	0.29

Table 3.6: Performance of our implementation with respect to MFS and LFS recall. *R_mfs* and *R_lfs* are the recall on the most-frequent-sense and least-frequent-sense instances respectively. *n* represents the number of considered instances.

Model	senseval2 sensekey F_1	senseval2 synset F_1	semeval2013 sensekey F_1	semeval2013 synset F_1
Our LSTM (T: SemCor)	**0.726**	**0.720**	0.661	0.647
Our LSTM (T: OMSTI)	0.688	0.675	0.655	0.651
Our LSTM (T: SemCor+OMSTI)	0.703	0.699	**0.667**	**0.656**

Table 3.7: Comparison of F_1 scores of our implementation using either synset or sensekey level to represent meanings.

Model/Method	Datasets	Scorer	senseval2 F_1	semeval2013 F_1
Our LSTM	T: SemCor	framework	**0.720**	0.647
Our LSTM	T: OMSTI	framework	0.675	0.651
Our LSTM	T: SemCor+OMSTI	framework	0.699	**0.656**
Our LSTMLP	T: SemCor, U:OMSTI	framework	0.714	0.642
Yuan et al. (2016b) LSTM	T: SemCor	mapping to WN3.0	0.736	0.670
Yuan et al. (2016b) LSTM	T: OMSTI	mapping to WN3.0	0.724	0.673
Yuan et al. (2016b) LSTMLP	T: SemCor, U: OMSTI	mapping to WN3.0	**0.739**	**0.679**
Raganato et al. (2017b)	T: SemCor	framework	0.720	0.669
Iacobacci et al. (2016) IMS+emb	T: SemCor	framework	0.710	0.673
Iacobacci et al. (2016) IMS+emb	T: SemCor+OMSTI	framework	0.708	0.663
Iacobacci et al. (2016) IMS-s+emb	T: SemCor	framework	0.722	0.659
Iacobacci et al. (2016) IMS-s+emb	T: SemCor+OMSTI	framework	**0.733**	0.667
Melamud et al. (2016)	T: SemCor	framework	0.718	0.656
Melamud et al. (2016)	T: SemCor+OMSTI	framework	0.723	0.672
Agirre et al. (2014) UKB-g*	P: SemCor	framework	0.688	0.688
Weissenborn et al. (2015)	P: SemCor	competition	-	**0.728**

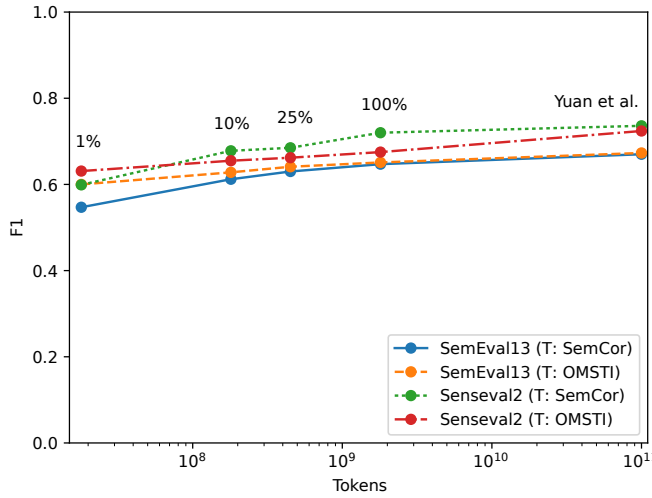
Table 3.8: Performance of our implementation compared to already published results. We report the model/method used to perform WSD, the used annotated dataset and scorer, and F_1 for each test set. In the naming of our models, *LSTM* indicates that the *averaging* technique was used for the sense assignment, while *LSTMLP* refers to the results obtained using *label propagation* (see Section 3). The datasets following *T:* indicate the annotated corpus used to represent the senses while *U:OMSTI* stands for using OMSTI as unlabeled sentences in case label propagation is used. *P: SemCor* indicates that sense distributions from SemCor are used in the system architecture. Three scorers are used: *"framework"* refers to the WSD evaluation framework from Raganato et al. (2017a); *"mapping to WN3.0"* refers to the evaluation used by Yuan et al. (2016b) while *"competition"* refers to the scorer provided by the competition itself (e.g., semeval2013).

Datasets	senseval2 Candidate Coverage	senseval2 Lemma Coverage	senseval2 Gold Coverage	semeval2013 Candidate Coverage	semeval2013 Lemma Coverage	semeval2013 Gold Coverage
SemCor	63.51%	29.72%	80.92%	66.64%	28.42%	83.08%
OMSTI	29.98%	5.9%	43.82%	31.22%	5.94%	45.23%
SemCor+OMSTI	**66.26%**	**32.25%**	**81.98%**	**69.89%**	**31.83%**	**84.76%**

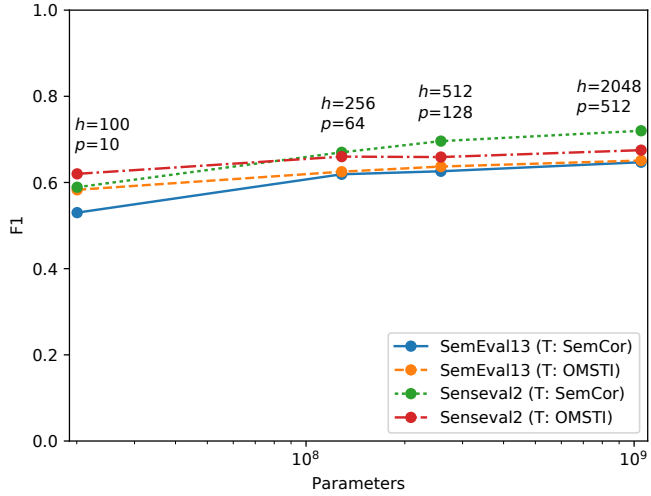
Table 3.9: Statistics about the coverage of annotated datasets in WordNet.

Competition	Model	MFS fallback	Averaging	Label propagation
senseval2	Our LSTM (T: SemCor)	0.64 (n=135)	0.66 (n=1712)	-
	Our LSTM (T: OMSTI)	0.66 (n=775)	0.56 (n=1072)	-
	Our LSTM (T: SemCor+OMSTI)	0.64 (n=135)	0.63 (n=1712)	-
	Our LSTMLP (T:SemCor, U:OMSTI)	0.64 (n=135)	0.71 (n=644)	0.61 (n=1068)
semeval2013	Our LSTM (T: SemCor)	0.39 (n=41)	0.56 (n=1262)	-
	Our LSTM (T: OMSTI)	0.58 (n=427)	0.55 (n=876)	-
	Our LSTM (T: SemCor+OMSTI)	0.40 (n=40)	0.57 (n=1263)	-
	Our LSTMLP (T:SemCor, U:OMSTI)	0.40 (n=40)	0.57 (n=397)	0.55 (n=866)

Table 3.10: The recall is shown for three WSD strategies and n indicates the number of instances it was actually used. The recall on all monosemous instances was 1.0 for both competitions and is hence not shown in the table. There are 435 monosemous instances in *senseval2* and 341 in *semeval2013*.



(a) Performance vs. unannotated corpus size



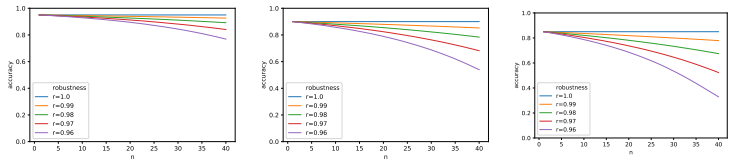
(b) Performance vs. number of parameters

Figure 3.5: The effect of (a) the size of unannotated corpus and (b) the number of parameters on WSD performance. Number of parameters includes the weights of the hidden layer, the weights of the projection layer, and the input and output embeddings. Notice that the horizontal axis is in log scale.

4. Adaptation

Consider a hypothetical NLP system that builds up its output via a sequence of n steps, each introducing a substructure (e.g. an edge in a syntactic tree or a part-of-speech label) with the accuracy γ_0. Instead of independent decisions, if each step takes previously built structures as input, the step-wise accuracy is better modeled as a function of a base accuracy and preceding errors. Assume that each such error decreases the accuracy by a small constant, we arrive at $\gamma = \max(\gamma_0 - m(1-r), 0)$ where $m \in \{0, .., n-1\}$ is the number of pre-existing errors and $r \in (0, 1)$ stands for the robustness of the system against them. This simple model allows us to simulate the effect of error propagation on accuracy (Figure 4.1). The shape of the curves resembles empirical results in transition-based dependency parsing (McDonald and Nivre, 2007). Together, they illustrate how error propagation can seriously degrade performance, and how increasing robustness can alleviate this problem in a significant way.

The paper in this chapter attempts to increase robustness by replacing the conventional supervised training regime with



(a) $\gamma_0 = 0.95$ **(b)** $\gamma_0 = 0.90$ **(c)** $\gamma_0 = 0.85$

Figure 4.1: Accuracy of the n^{th} decision simulated for some values of base accuracy rate γ_0, robustness r, and decision sequence length n.

reinforcement learning. In transition-based dependency parsing (see Section 2.5.3), the parser can be considered an agent that builds up dependency trees by executing actions sequentially, each modifying its internal state while optionally adding one more arc. To learn a policy that decides which action to take, a naïve training scheme applies a sequence of correct actions and trains the parser on predicting the next one. This means the parser is never trained on how to behave on erroneous states and therefore is more susceptible to error propagation. To expand the training set, efforts have been made in designing dynamic oracles, i.e. algorithms that compute the optimal action at any valid parsing state with respect to a reference parse tree (Goldberg and Nivre, 2012). A disadvantage of this approach is that a distinct algorithm has to be designed for each flavor of transition-based dependency parsing (Goldberg and Nivre, 2012, 2013; Goldberg et al., 2014; Gómez-Rodríguez et al., 2014; Björkelund and Nivre, 2015). Reinforcement learning offers a general and elegant solution: it trains parsers on their own actions using tree-level accuracy as reward signals. Our paper shows that this solution leads to improved performance. More importantly, we devise a procedure to quantify the effect of error propagation and show that parsers trained with reinforcement learning are indeed more robust.

Dependency parsing is a convenient test bed because solutions are small and annotations abundant. However, the algorithms developed in our paper are general enough to be applied to virtually any NLP problem. Replace parsing actions with clustering ones and we can perform coreference resolution. Plugging in labeling actions instead, we can solve named-entity resolution. Machine translation, speech recognition, and image captioning can also be tackled if we use sequence scores such as BLEU as rewards. We could also solve more than one problem at once by combining action sets. In support of this hypothesis, various flavors of reinforcement learning have been applied in all of the above-

mentioned tasks (Clark and Manning, 2016a; Lao et al., 2019; Ranzato et al., 2016; Bengio et al., 2015). Although error propagation is not measured in those papers, there is little reason to doubt a reduction similar to what we have observed.

Tackling Error Propagation through Reinforcement Learning: A Case of Greedy Dependency Parsing

Minh Le and **Antske Fokkens**
Department of Language, Literature and Communication
Vrije Universiteit Amsterdam
{m.n.le,antske.fokkens}@vu.nl

Abstract

Error propagation is a common problem in NLP. Reinforcement learning explores erroneous states during training and can therefore be more robust when mistakes are made early in a process. In this paper, we apply reinforcement learning to greedy dependency parsing which is known to suffer from error propagation. Reinforcement learning improves accuracy of both labeled and unlabeled dependencies of the Stanford Neural Dependency Parser, a high performance greedy parser, while maintaining its efficiency. We investigate the portion of errors which are the result of error propagation and confirm that reinforcement learning reduces the occurrence of error propagation.

1. Introduction

Error propagation is a common problem for many NLP tasks (Song et al., 2012; Quirk and Corston-Oliver, 2006; Han et al., 2013; Gildea and Palmer, 2002; Yang and Cardie, 2013). It can occur when NLP tools applied early on in a pipeline make mistakes that have negative impact on higher-level tasks further down the pipeline. It can also occur within the application of

a specific task, when sequential decisions are taken and errors made early in the process affect decisions made later on.

When reinforcement learning is applied, a system actively tries out different sequences of actions. Most of these sequences will contain some errors. We hypothesize that a system trained in this manner will be more robust and less susceptible to error propagation.

We test our hypothesis by applying reinforcement learning to greedy transition-based parsers (Yamada and Matsumoto, 2003; Nivre, 2004), which have been popular because of superior efficiency and accuracy nearing state-of-the-art. They are also known to suffer from error propagation. Because they work by carrying out a sequence of actions without reconsideration, an erroneous action can exert a negative effect on all subsequent decisions. By rendering correct parses unreachable or promoting incorrect features, the first error induces the second error and so on. McDonald and Nivre (2007) argue that the observed negative correlation between parsing accuracy and sentence length indicates error propagation is at work.

We compare reinforcement learning to supervised learning on Chen and Manning (2014)'s parser. This high performance parser is available as open source. It does not make use of alternative strategies for tackling error propagation and thus provides a clean experimental setup to test our hypothesis. Reinforcement learning increased both unlabeled and labeled accuracy on the Penn TreeBank and German part of SPMRL (Seddah et al., 2014). This outcome shows that reinforcement learning has a positive effect, but does not yet prove that this is indeed the result of reduced error propagation. We therefore designed an experiment which identified which errors are the result of error propagation. We found that around 50% of avoided errors were cases of error propagation in our best arc-standard system. Considering that 27% of the original errors were caused by error propagation, this

result confirms our hypothesis.

This paper provides the following contributions:

1. We introduce Approximate Policy Gradient (APG), a new algorithm that is suited for dependency parsing and other structured prediction problems.
2. We show that this algorithm improves the accuracy of a high-performance greedy parser.
3. We design an experiment for analyzing error propagation in parsing.
4. We confirm our hypothesis that reinforcement learning reduces error propagation.

To our knowledge, this paper is the first to experimentally show that reinforcement learning can reduce error propagation in NLP.

The rest of this paper is structured as follows. We discuss related work in Section 2. This is followed by a description of the parsers used in our experiments in Section 3. Section 4 outlines our experimental setup and presents our results. The error propagation experiment and its outcome are described in Section 5. Finally, we conclude and discuss future research in Section 6.

2. Related Work

In this section, we address related work on dependency parsing, including alternative approaches for reducing error propagation, and reinforcement learning.

2.1. Dependency Parsing

We use Chen and Manning (2014)'s parser as a basis for our experiments. Their parser is open-source and has served as a reference point for many recent publications (Dyer et al., 2015; Weiss et al., 2015; Alberti et al., 2015; Honnibal and Johnson, 2015, among others). They provide an efficient neural network

that learns dense vector representations of words, PoS-tags and dependency labels. This small set of features makes their parser significantly more efficient than other popular parsers, such as the Malt (Nivre et al., 2007) or MST (McDonald et al., 2005b) parser while obtaining higher accuracy. They acknowledge the error propagation problem of greedy parsers, but leave addressing this through (e.g.) beam search for future work.

Dyer et al. (2015) introduce an approach that uses Long Short-Term Memory (LSTM). Their parser still works incrementally and the number of required operations grows linearly with the length of the sentence, but it uses the complete buffer, stack and history of parsing decisions, giving the model access to global information. Weiss et al. (2015) introduce several improvements on Chen and Manning (2014)'s parser. Most importantly, they put a globally-trained perceptron layer instead of a softmax output layer. Their model uses smaller embeddings, rectified linear instead of cubic activation function, and two hidden layers instead of one. They furthermore apply an averaged stochastic gradient descent (ASGD) learning scheme. In addition, they apply beam search and increase training data by using unlabeled data through the tri-training approach introduced by Li et al. (2014b), which leads to further improvements.

Kiperwasser and Goldberg (2016b) introduce a new way to represent features using a bidirectional LSTM and improve the results of a greedy parser. Andor et al. (2016) present a mathematical proof that globally normalized models are more expressive than locally normalized counterparts and propose to use global normalization with beam search at both training and testing.

Our approach differs from all of the work mentioned above, in that it manages to improve results of Chen and Manning (2014) *without changing the architecture of the model nor the input representation*. The only substantial difference lies in the way the model is trained. In this respect, our research is most similar to

training approaches using dynamic oracles (Goldberg and Nivre, 2012). Traditional static oracles can generate only one sequence of actions per sentence. A dynamic oracle gives *all* trajectories leading to the best possible result from every valid parse configuration. They can therefore be used to generate more training sequences including those containing errors. A drawback of this approach is that dynamic oracles have to be developed specifically for individual transition systems (e.g. arc-standard, arc-eager). Therefore, a large number of dynamic oracles have been developed in recent years (Goldberg and Nivre, 2012, 2013; Goldberg et al., 2014; Gómez-Rodríguez et al., 2014; Björkelund and Nivre, 2015). In contrast, the reinforcement learning approach proposed in this paper is more general and can be applied to a variety of systems.

Zhang and Chan (2009) present the only study we are aware of that also uses reinforcement learning for dependency parsing. They compare their results to Nivre et al. (2006b) using the same features, but they also change the model and apply beam search. It is thus unclear to what extend their improvements are due to reinforcement learning.

Even though most approaches mentioned above improve the results reported by Chen and Manning (2014) and even more impressive results on dependency parsing have been achieved since (notably, Andor et al. (2016)), Chen and Manning's parser provides a better baseline for our purposes. We aim at investigating the influence of reinforcement learning on error propagation and want to test this in a clean environment, where reinforcement learning does not interfere with other methods that address the same problem.

2.2. Reinforcement Learning

Reinforcement learning has been applied to several NLP tasks with success, e.g. agenda-based parsing (Jiang et al., 2012), se-

mantic parsing (Berant and Liang, 2015) and simultaneous machine translation (Grissom II et al., 2014). To our knowledge, however, none of these studies investigated the influence of reinforcement learning on error propagation.

Learning to Search (L2S) is probably the most prominent line of research that applies reinforcement learning (more precisely, imitation learning) to NLP. Various algorithms, e.g. SEARN (Daumé III et al., 2009) and DAgger (Ross et al., 2011), have been developed sharing common high-level steps: a *roll-in* policy is executed to generate training states from which a *roll-out* policy is used to estimate the loss of certain actions. The concrete instantiation differs from one algorithm to another with choices including a referent policy (static or dynamic oracle), learned policy, or a mixture of the two. Early work in L2S focused on reducing reinforcement learning into binary classification (Daumé III et al., 2009), but newer systems favored regressors for efficiency (Chang et al., 2015, Supplementary material, Section B). Our algorithm APG is simpler than L2S in that it uses only one policy (pre-trained with standard supervised learning) and applies the existing classifier directly without reduction (the only requirement is that it is probabilistic). Nevertheless, our results demonstrate its effectiveness.

APG belongs to the family of policy gradient algorithms (Sutton et al., 1999), i.e. it maximizes the expected reward directly by following its gradient w.r.t. the parameters. The advantage of using a policy gradient algorithm in NLP is that gradient-based optimization is already widely used. REINFORCE (Williams, 1992; Ranzato et al., 2016) is a widely-used policy gradient algorithm but it is also well-known for suffering from high variance (Sutton et al., 1999).

We directly compare our approach to REINFORCE, whereas we leave a direct comparison to L2S for future work. Our experiments show that our algorithm results in lower variance and

achieves better performance than REINFORCE.

Recent work addresses the approximation of reinforcement learning gradient in the context of machine translation. Shen et al. (2016)'s algorithm is roughly equivalent to the combination of an oracle and random sampling. Their approach differs from ours, because it does not retain memory across iteration as in our best-performing model (see Section 4).

2.3. Reinforcement and error propagation

As mentioned above, previous work that applied reinforcement learning to NLP has, to our knowledge, not shown that it improved results by reducing error propagation.

Work on identifying the impact of error propagation in parsing is rare, Ng and Curran (2015b) being a notable exception. They provide a detailed error analysis for parsing and classify which kind of parsing errors are involved with error propagation. There are four main differences between their approaches and ours. First, Ng and Curran correct arcs in the tree and our algorithm corrects decisions of the parsing algorithm. Second, our approach distinguishes between cases where one erroneous action deterministically leads to multiple erroneous arcs and cases where an erroneous action leads to conditions that indirectly result in further errors (see Section 1 for a detailed explanation). Third, Ng and Curran's algorithm corrects all erroneous arcs that are the same type of parsing error and point out that they cannot examine the interaction between multiple errors of the same type in a sentence. Our algorithm corrects errors incrementally and therefore avoids this issue. Finally, the classification and analysis presented in Ng and Curran (2015b) are more extensive and detailed than ours. Our algorithm can, however, easily be extended to perform similar analysis. Overall, Ng and Curran's approach for error analysis and ours are complementary. Combining them and applying them to various systems would form an interesting

direction for future work.

3. A Reinforced Greedy Parser

This section describes the systems used in our experiments. We first describe the arc-standard algorithm, because familiarity with it helps to understand our error propagation analysis. Next, we briefly point out the main differences between the arc-standard algorithm and the alternative algorithms we experimented with (arc-eager and swap-standard). We then outline the traditional and our novel machine learning approaches. The features we used are identical to those described in Chen and Manning (2014). We are not aware of research identifying the best feature for a neural parser with arc-eager or swap-standard so we use the same features for all transition systems.

3.1. Transition-Based Dependency Parsing

In an arc-standard system (Nivre, 2004), a parsing configuration consists of a triple $\langle \Sigma, \beta, A \rangle$, where Σ is a stack, β is a buffer containing the remaining input tokens and A are the dependency arcs that are created during parsing process. At initiation, the stack contains only the root symbol (Σ = [ROOT]), the buffer contains the tokens of the sentence ($\beta = [w_1, ..., w_n]$) and the set of arcs is empty ($A = \emptyset$).

The arc-standard system supports three transitions. When σ_1 is the top element and σ_2 the second element on the stack, and β_1 the first element of the buffer:[1]

LEFT$_l$ adds an arc $\sigma_1 \xrightarrow{l} \sigma_2$ to A and removes σ_2 from the stack.

RIGHT$_l$ adds an arc $\sigma_2 \xrightarrow{l} \sigma_1$ to A and removes σ_1 from the stack.

SHIFT moves β_1 to the stack.

[1] Naturally, the transitions LEFT$_l$ and RIGHT$_l$ can only take place if the stack contains at least two elements and SHIFT can only occur when there is at least one element on the buffer.

Step	Transition	Stack	Buffer	Arcs
0		<ROOT>	*waves hit ... Big Board*	$\emptyset$
1	SHIFT	<ROOT> *waves*	*hit stocks ... Big Board*	$\emptyset$
2	SHIFT	<ROOT> *waves hit*	*stocks themselves ... Big Board*	$\emptyset$
3	$\text{LEFT}_{\text{nsubj}}$	<ROOT> *hit*	*stocks themselves ... Big Board*	$A_1 = \{ hit \xrightarrow{\text{nsubj}} waves \}$
4	SHIFT	<ROOT> *hit stocks*	*themselves on the Big Board*	A_1
5	SHIFT	<ROOT> *hit stocks themselves*	*on the Big Board*	A_1
6	$\text{RIGHT}_{\text{dep}}$	<ROOT> *hit stocks*	*on the Big Board*	$A_2 = A_1 \cup \{ stock \xrightarrow{\text{dep}} themselves \}$
7	$\text{RIGHT}_{\text{dobj}}$	<ROOT> *hit*	*on the Big Board*	$A_3 = A_2 \cup \{ hit \xrightarrow{\text{dobj}} stock \}$

Table 4.1: Parsing oracle walk-through

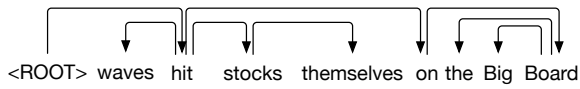


Figure 4.2: Correct dependencies for a simplified example from Penn TreeBank

When the buffer is empty, the stack contains only the root symbol and A contains a parse tree, the configuration is completed. For a sentence of N_w tokens, a full parse takes $2N_w + 1$ transitions to complete (including the initiation). Figure 4.2 provides the gold parse tree for a (simplified) example from the Penn Treebank. The steps taken to create the dependencies between the sentence's head word *hit* and its subject and direct object are provided in Table 4.1 (p. 106).

To demonstrate that reinforcement learning can train different systems, we also carried out experiments with arc-eager (Nivre, 2003) and swap-standard (Nivre, 2009). Arc-eager is designed for incremental parsing and included in the popular MaltParser (Nivre et al., 2006a). Swap-standard is a simple and effective solution to unprojective dependency trees. Because arc-eager does not guarantee complete parse trees, we used a variation that employs an action called UNSHIFT to resume processing of tokens that would otherwise not be attached to a head (Nivre and Fernández-González, 2014).

3.2. Training with a Static Oracle

In transition-based dependency parsing, it is common to convert a dependency treebank $\mathscr{D} \ni (x, y)$ into a collection of input features $s \in \mathscr{S}$ and corresponding gold-standard actions $a \in \mathscr{A}$ for training, using a static oracle $\mathscr{O}$. In Chen and Manning (2014), a neural network works as a function mapping input features to probabilities of actions: $f_{NN} : \mathscr{S} \times \mathscr{A} \rightarrow [0, 1]$. The neural network is trained to minimize negative log-likelihood loss on the

converted treebank:

$$\mathscr{L} = \sum_{(x,y)\in\mathscr{D}} \sum_{(s,a)\in\mathscr{O}(x,y)} -\log f_{NN}(s,a;\theta) \tag{4.1}$$

3.3. Reinforcement Learning

Following Maes et al. (2009), we view transition-based dependency parsing as a deterministic Markov Decision Process. The problem is summarized by a tuple $\langle \mathscr{S}, \mathscr{A}, \mathscr{T}, r \rangle$ where $\mathscr{S}$ is the set of all possible states, $\mathscr{A}$ contains all possible actions, $\mathscr{T}$ is a mapping $\mathscr{S} \times \mathscr{A} \to \mathscr{S}$ called *transition function* and $r : \mathscr{S} \times \mathscr{A} \to \mathbb{R}$ is a reward function.

A state corresponds to a configuration and is summarized into input features. Possible actions are defined for each transition system described in Section 1. We keep the training approach simple by using only one reward $r(\bar{y})$ at the end of each parse.

Given this framework, a stochastic policy guides our parser by mapping each state to a probabilistic distribution of actions. During training, we use function f_{NN} described in Section 2 as a stochastic policy. At test time, actions are chosen in a greedy fashion following existing literature. We aim at finding the policy that maximizes the expected reward (or, equivalently, minimizes the expected loss) on the training dataset:

$$\text{maximize } \eta = \sum_{(x,y)\in\mathscr{D}} \sum_{a_{1:m}\sim f} r(\bar{y}) \prod_{i=1}^{m} f_{NN}(s_i, a_i; \theta) \tag{4.2}$$

where $a_{1:m}$ is a sequence of actions obtained by following policy f_{NN} until termination and $s_{1:m}$ are corresponding states (with s_{m+1} being the termination state).

3.4. Approximate Policy Gradient

Gradient ascent can be used to maximize the expected reward in Equation 4.2. The gradient of expected reward w.r.t. parameters

is (note that $\mathrm{d}z = z\mathrm{d}(\log z)$):

$$\frac{\partial \eta}{\partial \theta} = \sum_{(x,y)\in\mathscr{D}} \sum_{a_{1:m}\sim f_{NN}} r(\bar{y}) \prod_{i=1}^{m} f_{NN}(s_i, a_i) \sum_{i=1}^{m} \frac{\partial}{\partial \theta} \log f_{NN}(s_i, a_i; \theta) \tag{4.3}$$

Because of the exponential number of possible trajectories, calculating the gradient exactly is not possible. We propose to replace it by an approximation (hence the name *Approximate* Policy Gradient) by summing over a small subset U of trajectories. Following common practice, we also use a baseline $b(y)$ that only depends on the correct dependency tree. The parameter is then updated by following the approximate gradient:

$$\Delta\theta \propto \sum_{(x,y)\in\mathscr{D}} \sum_{a_{1:m}\in U} (r(\bar{y}) - b(y)) \prod_{i=1}^{m} f_{NN}(s_i, a_i) \sum_{i=1}^{m} \frac{\partial}{\partial \theta} \log f_{NN}(s_i, a_i; \theta) \tag{4.4}$$

Instead of sampling one trajectory at a time as in REINFORCE, Equation 4.4 has the advantage that sampling over multiple trajectories could lead to more stable training and higher performance. To achieve that goal, the choice of U is critical. We empirically evaluate three strategies:

RL-ORACLE: only includes the oracle transition sequence.

RL-RANDOM: randomly samples k distinct trajectories at each iteration. Every action is sampled according to f_{NN}, i.e. preferring trajectories for which the current policy assigns higher probability.

RL-MEMORY: samples randomly as the previous method but retains k trajectories with highest rewards across iterations in a separate memory. Trajectories are "forgotten" (re-

moved) randomly with probability ρ before each iteration.[2]

Intuitively, trajectories that are more likely and produce higher rewards are better training examples. It follows from Equation 4.3 that they also bear bigger weight on the true gradient. This is the rationale behind RL-RANDOM and RL-ORACLE. For a suboptimal parser, however, these objectives sometimes work against each other. RL-MEMORY was designed to find the right balance between them. It is furthermore important that the parser is pretrained to ensure good samples. Algorithm 1 (p. 118) illustrates the procedure of training a dependency parser using the proposed algorithms.

4. Reinforcement Learning Experiments

We first train a parser using a supervised learning procedure and then improve its performance using APG. We empirically tested that training a second time with supervised learning has little to no effect on performance.

4.1. Experimental Setup

We use PENN Treebank 3 with standard split (training, development and test set) for our experiments with arg-standard and arg-eager. Because the swap-standard parser is mainly suited for non-projective structures, which are rare in the PENN Treebank, we evaluate this parser on the German section of the SPMRL dataset. For PENN Treebank, we follow Chen and Manning's preprocessing steps. We also use their pretrained model[3] for arc-standard and train our own models in similar settings for other transition systems.

For reinforcement learning, we use AdaGrad for optimization.

[2] We assign a random number (drawn uniformly from $[0, 1]$) to each trajectory in memory and remove those assigned a number less than ρ.

[3] We use `PTB_Stanford_params.txt.gz` downloaded from `http://nlp.stanford.edu/software/nndep.shtml` on December 30th, 2015.

We do not use dropout because we observed that it destablized the training process. The reward $r(\bar{y})$ is the number of correct labeled arcs (i.e. LAS multiplied by number of tokens).[4] The baseline is fixed to half the number of tokens (corresponding to a 0.5 LAS score). As training takes a lot of time, we tried only few values of hyperparameters on the development set and picked $k = 8$ and $\rho = 0.01$. 1,000 updates were performed (except for REINFORCE which was trained for 8,000 updates) with each training batch contains 512 randomly selected sentences. The Stanford dependency scorer[5] was used for evaluation.

4.2. Effectiveness of Reinforcement Learning

Table 4.3 (p. 120) displays the performance of different approaches to training dependency parsers. Although we used Chen and Manning (2014)'s pretrained model and Stanford open-source software, the results of our baseline are slightly worse than what is reported in their paper. This could be due to minor differences in settings and does not affect our conclusions.

Across transition systems and two languages, APG outperforms supervised learning, verifying our hypothesis. Moreover, it is not simply because the learners are exposed to more examples than their supervised counterparts. RL-ORACLE is trained on exactly the same examples as the standard supervised learning system (SL), yet it is consistently superior. This can only be explained by the superiority of the reinforcement learning objective function compared to negative log-likelihood.

The results support our hypothesis that APG is better than REINFORCE (abbreviated as RE in Table 4.3, p. 120) as RL-MEMORY always outperforms the classical algorithm and the other two heuristics do in two out of three cases. The usefulness of

[4] Punctuation is not taken into account, following Chen and Manning (2014).

[5] Downloaded from `http://nlp.stanford.edu/software/lex-parser.shtml`.

training examples that contain errors is evident through the better performance of RL-RANDOM and RL-MEMORY in comparison to RL-ORACLE.

Table 4.4 (p. 120) shows the importance of samples for RL-RANDOM. The algorithm hurts performance when only one sample is used whereas training with two or more samples improves the results. The difference cannot be explained by the total number of observed samples because one-sample training is still worse after 8,000 iterations compared to a sample size of 8 after 1,000 iterations. The benefit of added samples is twofold: increased performance and decreased variance. Because these benefits saturate quickly, we did not test sample sizes beyond 32.

5. Error Propagation Experiment

We hypothesized that reinforcement learning avoids error propagation. In this section, we describe our algorithm and the experiment that identifies error propagation in the arc-standard parsers.

5.1. Error Propagation

Section 1 explained that a transition-based parser goes through the sentence incrementally and must select a transition from [SHIFT, LEFT$_l$, RIGHT$_l$] at each step. We use the term *arc error* to refer to an erroneous arc in the resulting tree. The term *decision error* refers to a transition that leads to a loss in parsing accuracy. Decision errors in the parsing process lead to one or more arc errors in the resulting tree. There are two ways in which a single decision error may lead to multiple arc errors. First, the decision can deterministically lead to more than one arc error, because (e.g.) an erroneously formed arc also blocks other correct arcs. Second, an erroneous parse decision changes some of the features that the model uses for future decisions and these changes can lead to further (decision) errors down the road.

We illustrate both cases using two incorrect derivations pre-

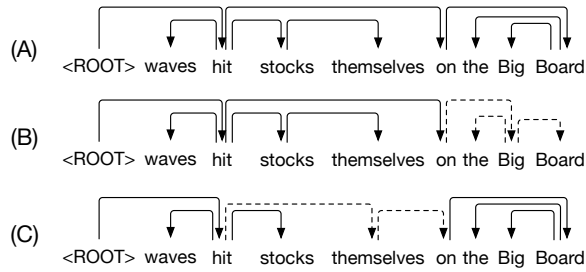


Figure 4.3: Three dependency graphs: gold (A), arc errors caused by one decision error (B) and arc errors caused by multiple decision errors (C).

sented in Figure 4.3. The original gold tree is repeated in (A). The dependency graph in Figure 4.3 (B) contains three erroneous dependency arcs (indicated by dashed arrows). The first error must have occurred when the parser executed $\text{RIGHT}_{\text{amod}}$ creating the arc *Big* $\rightarrow$ *Board*. After this error, it is impossible to create the correct relations *on* $\rightarrow$ *Board* and *Board* $\rightarrow$ *the*. The wrong arcs *Big* $\rightarrow$ *the* and *on* $\rightarrow$ *Big* are thus all the result of a single decision error.

Figure 4.3 (C) represents the dependency graph that is actually produced by our parser.[6] It contains two erroneous arcs: *hit* $\rightarrow$ *themselves* and *themselves* $\rightarrow$ *on*. Table 4.2 (p. 119) provides a possible sequence of steps that led to this derivation, starting from the moment *stocks* was added to the stack (Step 4). The first error is introduced in Step 5', where *hit* combines with *stocks* before *stocks* has picked up its dependent *themselves*. At that point, *themselves* can no longer be combined with the right head. The proposition *on*, on the other hand, can still be combined with the correct head. This error is introduced in Step 7', where the parser moves *on* to the stack rather than creating an arc from *hit*

[6]The example is a fragment of a more complex sentence consisting of 33 tokens. The parser does provide the correct output when is analyzes this sequence in isolation.

to *themselves*.[7] There are thus two decision errors that lead to the arc errors in Figure 4.3 (C). The second decision error can, however, be caused indirectly by the first error. If a decision error causes additional decision errors later in the parsing process, we talk of error propagation. This cannot be known just by looking at the derivation.

5.2. Examining the impact of decision errors

We examine the impact of individual decision errors on the overall parse results in our test set by combining a dynamic oracle and a recursive function. We use a dynamic oracle based on Goldberg et al. (2014) which gives us the overall loss at any point during the derivation. The loss is equal to the minimal number of arc errors that will have been made once the parse is complete. We can thus deduce how many arc errors are deterministically caused by a given decision error.

The propagation of decision errors cannot be determined by simply examining the increase in loss during the parsing process. We use a recursive function to identify whether a particular parse suffered from this. While parsing the sentence, we register which decisions lead to an increase in loss. We then recursively reparse the sentence correcting one additional decision error during each run until the parser produces the gold. If each erroneous decision has to be corrected in order to arrive at the gold, we assume the decision errors are independent of each other. If, on the other hand, the correction of a specific decision also fixes other decisions down the road, the original parse suffers from error propagation.

The results are presented in Table 4.5 (p. 120). *Total Loss* indicates the number of arc errors in the corpus, *Dec. Errors* the number of decision errors and *Err. Prop.* the number of decision

[7] Note that technically, *on* can still become a dependent of *hit*, but this can only happen if *on* becomes the head of *themselves* which would also be an error.

errors that were the result of error propagation. This number was obtained by comparing the number of decision errors in the original parse to the number of decision errors that needed to be fixed to obtain the gold parse. If less errors had to be fixed than originally present, we counted the difference as error propagation. Note that fixing errors sometimes leads to new decision errors during the derivation. We also counted the cases where more decision errors needed to be fixed than were originally present and report them in Table 4.5 (p. 120).[8]

On average, decision errors deterministically lead to more than one arc error in the resulting parse tree. This remains stable across systems (around 1.4 arc errors per decision error). We furthermore observe that the proportion of decision errors that are the result of error propagation has indeed reduced for all reinforcement learning models. Among the errors avoided by APG, 35.9% were propagated errors for RL-ORACLE, 48.9% for RL-RANDOM, and 51.9% for RL-MEMORY. These percentages are all higher than the proportion of propagated errors occurring in the corpus parsed by SL (27%). This outcome confirms our hypothesis that reinforcement learning is indeed more robust for making decisions in imperfect environments and therefore reduces error propagation.

6. Conclusion

This paper introduced Approximate Policy Gradient (APG), an efficient reinforcement learning algorithm for NLP, and applied it to a high-performance greedy dependency parser. We hypothesized that reinforcement learning would be more robust against error propagation and would hence improve parsing accuracy.

[8] We ran an alternative analysis where we counted all cases where fixing one decision error in the derivation reduced the overall number of decision errors in the parse by more than one. Under this alternative analysis, similar reductions in the proportion of error propagation were observed for reinforcement learning.

To verify our hypothesis, we ran experiments applying APG to three transition systems and two languages. We furthermore introduced an experiment to investigate which portion of errors were the result of error propagation and compared the output of standard supervised machine learning to reinforcement learning. Our results showed that: (a) reinforcement learning indeed improved parsing accuracy and (b) propagated errors were over-represented in the set of avoided errors, confirming our hypothesis.

To our knowledge, this paper is the first to show experimentally that reinforcement learning can reduce error propagation in an NLP task. This result was obtained by a straight-forward implementation of reinforcement learning. Furthermore, we only applied reinforcement learning in the training phase, leaving the original efficiency of the model intact. Overall, we see the outcome of our experiments as an important first step in exploring the possibilities of reinforcement learning for tackling error propagation.

Recent research on parsing has seen impressive improvement during the last year achieving UAS around 94% (Andor et al., 2016). This improvement is partially due to other approaches that, at least in theory, address error propagation, such as beam search. Both the reinforcement learning algorithm and the error propagation study we developed can be applied to other parsing approaches. There are two (related) main questions to be addressed in future work in the domain of parsing. The first addresses whether our method is complementary to alternative approaches and could also improve the current state-of-the-art. The second question would address the impact of various approaches on error propagation and the kind of errors they manage to avoid (following Ng and Curran (2015b)).

APG is general enough for other structured prediction problems. We therefore plan to investigate whether we can apply our approach to other NLP tasks such as coreference resolution or

semantic role labeling and investigate if it can also reduce error propagation for these tasks.

The source code of all experiments is publicly available at `https://bitbucket.org/cltl/redep-java`.

Acknowledgments

The research for this paper was supported by the Netherlands Organisation for Scientific Research (NWO) via the Spinoza-prize Vossen projects (SPI 30-673, 2014-2019) and the VENI project *Reading between the lines* (VENI 275-89-029). Experiments were carried out on the Dutch national e-infrastructure with the support of SURF Cooperative. We would like to thank our friends and colleagues Piek Vossen, Roser Morante, Tommaso Caselli, Emiel van Miltenburg, and Ngoc Do for many useful comments and discussions. We would like to extend our thanks the anonymous reviewers for their feedback which helped improving this paper. All remaining errors are our own.

```
MemorySeqs ← ∅;
foreach training batch b do
    foreach sentence s ∈ b do
        OracleSeq ← Oracle(s);
        SystemSeqs ← (sample k parsing transition sequences
          for s);
        if RL-Oracle then
            ComputeGradients(OracleSeq);
        else if RL-Random then
            ComputeGradients(SystemSeqs);
        else if RL-Memory then
            m ← MemorySeqs[s];
            foreach q ∈ m do
                if RandomNumber() < ρ then
                    Remove q from m;
                end
            end
            foreach q ∈ SystemSeqs do
                if |m| < k then
                    Insert q into m;
                else
                    p ← (sequence with smallest reward in m);
                    if reward(q) > reward(p) then
                        Replace p by q in m;
                    end
                end
            end
            ComputeGradients(m);
    end
    Perform one gradient descent step;
end
```

Algorithm 1: Training a dependency parser with approximate policy gradient.

Step	Transition	Stack	Buffer	Arcs
4	SHIFT	<ROOT> *hit stocks*	*themselves on the Big Board*	A_1
5'	$\text{RIGHT}_{\text{dobj}}$	<ROOT> *hit*	*themselves on the Big Board*	$A_2 = A_1 \cup \{hit \xrightarrow{\text{dobj}} stock\}$
6'	SHIFT	<ROOT> *hit themselves*	*on the Big Board*	A_2
7'	SHIFT	<ROOT> *hit themselves on*	*the Big Board*	A_2
...				
10'	SHIFT	<ROOT> *hit themselves on the Big Board*		A_2
11'	LEFT_{nn}	<ROOT> *hit themselves on the Board*		$A_3 = A_2 \cup \{Board \xrightarrow{\text{nn}} Big\}$
12'	LEFT_{det}	<ROOT> *hit themselves on Board*		$A_4 = A_3 \cup \{Board \xrightarrow{\text{det}} the\}$
13'	$\text{RIGHT}_{\text{pobj}}$	<ROOT> *hit themselves on*		$A_5 = A_4 \cup \{on \xrightarrow{\text{pobj}} Board\}$
14'	$\text{RIGHT}_{\text{dep}}$	<ROOT> *hit themselves*		$A_6 = A_5 \cup \{themselves \xrightarrow{\text{dep}} on\}$

Table 4.2: Possible parsing walk-through with error

	Arc-standard		Arc-eager		Swap-standard	
	UAS	**LAS**	**UAS**	**LAS**	**UAS**	**LAS**
SL	91.3	89.4	88.3	85.8	84.3	81.3
RE	91.9	90.2	89.7	87.2	87.5	84.4
RL-O	91.8	90.2	88.9	86.5	86.8	83.9
RL-R	**92.2**	**90.6**	89.4	87.0	87.5	84.5
RL-M	**92.2**	**90.6**	**89.8**	**87.4**	**87.6**	**84.6**

Table 4.3: Comparing training methods on PENN Treebank (arc-standard and arc-eager) and German part of SPMRL-2014 (swap-standard).

	Dev		Test		Test std.	
	UAS	**LAS**	**UAS**	**LAS**	**UAS**	**LAS**
SL	91.5	89.6	91.3	89.4	-	-
RE	92.1*	90.4*	91.9*	90.2*	0.04	0.05
1	91.2*	89.1*	91.0*	88.9*	0.12	0.15
2	91.8*	90.0*	91.6*	89.9*	0.09	0.09
4	92.2*	90.5*	92.0*	90.4*	0.09	0.08
8	**92.4***	**90.8***	92.2*	**90.6***	0.03	0.05
16	**92.4**	**90.8**	92.2	**90.6**	-	-
32	**92.4**	**90.8**	**92.3**	**90.6**	-	-

Table 4.4: Parsing accuracy of RL-RANDOM (arc-standard) with different sample sizes compared to supervised learning (SL) and REINFORCE (RE). *: significantly different from SL with $p < 0.001$

	SL	**RL-O**	**RL-R**	**RL-M**
Total Loss	7069	6227	6042	6144
Dec. Errors	5177	4410	4345	4476
Err. Prop.	1399	1124	992	1035
New errors	411	432	403	400
Loss/error	1.37	1.41	1.39	1.37
Err. Prop. (%)	27.0	25.5	22.8	23.1

Table 4.5: Overview of average impact of decision errors

5. Correction

A quick glance over the state-of-the-art in NLP reveals that many tasks have an error rate of 10-20% or higher.[1] In Chapter 1, I have argued that errors and error propagation are inevitable. Although the challenge they pose is daunting, I believe that it can be overcome.

Humans have overcome errors with their ingenuity before. Early telecommunication is marred by noise caused by electrical interference along transmission lines. That changed with Shannon's discovery of information theory (Shannon, 1948). This groundbreaking work gave us a way to measure informational content and transmit such information reliably. In Shannon (1948), we can find a mathematical proof that messages can be encoded (and decoded at the receiving end) such that they are transmitted with an arbitrarily small error.

There is certainly a parallel between digital and natural communication. In both cases, there is a message that one wants to get from A to B through an imperfect (noisy) channel. The concept of a channel can be physically realized in arbitrarily complex ways. An instant message, for example, might be transformed through various TCP/IP layers before hitting the wire, hops around the Internet, goes through TCP/IP layers again, this time in reverse, before being reconstructed on the recipient's screen. In the case of natural language, the message is what the speaker intends to convey (not the utterance but its intended effect on the hearer)

[1] I used the listing at `https://paperswithcode.com/area/natural-language-processing` combined with my personal collection of papers.

and the channel includes the speaker's language production faculty, any medium between them, and the hearer's comprehension faculty. The development of error-correcting codes has enabled us today to enjoy reliable communication across the globe and beyond (Calderbank, 1998). It is possible that error correction could one day allow us to solve not just error propagation but natural language understanding as a whole.

Recall that we want to get a message from A to B through a noisy channel. The message can be characters you see on your screen or thoughts you have in mind. Error-correcting codes are protocols use at the transmission source to calculate content (also called redundancy) that bares some information about this message. At the receiving end, the message and the redundancy are cross-checked and any detected errors are corrected. The simplest code that allows error detection is called *parity bit* which works as follows: the number of 1-bits in the message is counted; if it is an even number, a zero is written to the left of the message, otherwise, a one is prepended.[2] Obviously, the receiver will need to know how the parity bit algorithm works. Once she masters the code, she will be able to separate the redundancy from the message, check if they agree, and tell if one bit has been randomly flipped during the transmission. The parity bit code does not permit the detection of more than one error; nor does it allow error correction. However, other algorithms built on its concept can. Generally speaking, the more redundancy we add, the more errors we can fix and, therefore, tolerate.

Redundancy emerges naturally when we speak. Interpreters of verb-final languages such as German and Japanese are skilled at predicting the intended message because they cannot wait until the main verb is uttered to start interpreting. Grissom II et al.

[2]To be precise, there are two variations of parity bits: even parity makes the combination of the message and the redundancy always even, and odd parity makes it always odd. The given example is of the even variety.

(2016) show that fluent speakers of Japanese can predict the final verb significantly better than chance. In the first paper in this chapter, we will also see how human annotators can guess the content of documents that are heavily masked, i.e. many words are replaced with blanks, to perform coreference resolution.

Perhaps the most striking demonstrations of redundancy can be found in modern language models. Using state-of-the-art techniques, Khandelwal et al. (2020) report a per-word perplexity of roughly 16 on English Wikipedia. That is to say that their model finds predicting the next token in a sentence, picking from a vocabulary of millions, as hard as answering a multiple-choice question with 16 possible answers. On the other hand, NLP has so far failed to tap into this large redundancy for error correction. We have not figured out the code yet.

What does an error correcting code for NLP look like? It might take researchers many more years to find out. My best guess is that it involves calculating the compatibility of different views on the message just as the parity bit algorithm checks the compatibility of the message and the redundancy. In Section 2.4, I have also argued that world knowledge is likely to play an important role. The two papers in this chapter aim at shedding more light on this important question.

In the first paper, my coauthors and I show that, at least on OntoNotes (Weischedel et al., 2013a), many coreference resolution systems do not make appropriate use of context. Their decisions are almost entirely dependent on the expressions whose coreference status they are trying to determine. This is analogous to using only the message while ignoring the redundancy. In contrast, the annotators in our study show a drop in performance when context is removed. More importantly, via auxiliary tasks and interviews, we show that they solve difficult cases by combining their world knowledge with complex reasoning. We argue that to resolve coreference resolution in a robust way, it is essential to

incorporate a similar process.

The results are weakened somewhat by the fact that OntoNotes does not mark singletons (things that are mentioned only once and therefore do not belong to any coreference chain) which makes the task easier than it truly is. However, given the complexity of reasoning based on context, it would be very surprising if models gain this ability merely through training on some negative examples.

The second paper is an attempt to capture redundancy in a way that might facilitate error correction. Semantic role labeling (SRL), a task that attempts to identify *who does what to whom*, is often performed in a strangely incoherent way: the *who*, the *what*, and the *whom* are resolved without any semantic dependency (e.g. He et al. 2018). Sometimes constraints are applied to enforce a set of manually defined rules (Täckström et al., 2015) or to weed out duplicate and overlapping roles (Larionov et al., 2019; Li et al., 2020a). The few papers that explore the modeling of semantic dependency between arguments (Haghighi et al., 2005; Toutanova et al., 2008) found improvement in performance. Similarly, implicit SRL is often performed independently of explicit SRL (see Section 2.5.5 for an overview of these tasks). We attempt to model the joint distribution of roles using a simple neural network architecture to improve iSRL performance. While we found that our models reproduce Haghighi et al. (2005); Toutanova et al. (2008)'s results in improving SRL, we did not find improvement associated with the incorporation of role-role dependency on iSRL.

The unfortunate results might have been caused by two factors: train-test discrepancy and a lack of adaptation. Our first experiment is carried out on SemEval-2010 (Ruppenhofer et al., 2009). The corpus is a rare specimen of novels in NLP: predicates and roles are annotated on chapters of Sherlock Holmes. However, the proposed models are trained on OntoNotes which

contains various genres but all are quite different from Victorian novels. ON5V (Moor et al., 2013b) is used in the second experiment. The corpus is derived from OntoNotes but the distribution of predicates is radically different from the original dataset. In hindsight, I could have used adaptation (Ruder et al., 2019), i.e. further training on the test domain, to improve performance. Furthermore, the learned knowledge about role-role interaction could have been integrated more closely with the baseline models to eliminate confounding factors.

It does not help that SemEval-2010 is very small and neural networks are sensitive to random initialization. The two factors worked together to decrease the reliability (as opposed to validity) of the evaluation. In one run, we observed that the proposed model gets an F-score of 7% below the mean (-31% relative terms). For the future development of iSRL, it is important that bigger and broader test sets are developed.

We know that error correction is possible because humans do it all the time. Neuroscience research has shown that during language comprehension the brain continuously checks interpretations against what it already knows (Nieuwland and van Berkum, 2006). This test is so reliable that when no interpretation passes, the input itself is thrown into doubt. Consider the following excerpt:[3]

(1) BTW, when you said "Xander's love and desire for him near palatable", did you mean "palpable"? Palatable is kind of funny in context, but I didn't think it was what you meant.

This type of correcting questions is familiar to all language users. Although it is still far beyond the reach of NLP systems, I believe that we will get there someday.

[3]Taken from the English Web 2015 corpus (Jakubíček et al., 2013)

An Input Ablation Analysis of Coreference Resolution

Minh Le, Antske Fokkens, and **Piek Vossen**
Department of Language, Literature and Communication
Vrije Universiteit Amsterdam
{m.n.le,antske.fokkens,piek.vossen}@vu.nl

Abstract

We introduce a method for systematically analyzing what information coreference resolution systems use and what additional information may be valuable. We mask specific types of information in input documents and let humans and a broad set of automatic systems annotate the modified documents. Equipped with this method, we study the coreference linking step (i.e. gold mentions are given) on OntoNotes. Our analysis reveals insights into the studied systems and the task itself. First, given gold-standard mentions, all studied systems make their linking decisions almost exclusively based on mentions. Second, the neural end-to-end system we investigate does not make use of instance-specific knowledge. Third, people make use of world and script knowledge when resolving challenging cases, suggesting that modeling these phenomena is a fruitful research direction.

1. Introduction

In coreference resolution research, a lot of effort has been devoted to feature engineering. Many features were proposed, most of them aim at matching mentions (Soon et al., 2001; Lee et al., 2011; Culotta et al., 2007, among others), while a few try to

capture textual context (Yarowsky, 1994) and link the text with real-world entities (Rahman and Ng, 2012; Ponzetto and Strube, 2006). As new features are increasingly harder to come by, researchers have shifted their effort to algorithmic improvement (e.g. Ng and Cardie (2002a); Luo et al. (2004); Poon and Domingos (2008)). Recently, the incorporation of contextualized language models trained on large amounts of data has resulted in strong boosts in performance (Joshi et al., 2020; Wu et al., 2019b). There remains, however, a large gap between machine and human performance.Answering the two following questions can provide essential insights into narrowing this gap: *What information is being captured by existing systems?* and, more importantly, *What information is (not) needed to improve their performance?* This paper introduces a system-independent method to address these questions.

In language modeling, information use has been studied by perturbing the textual input (Khandelwal et al., 2018). We apply this technique to coreference resolution by defining seven types of transformations and studying their effect on systems' and human annotators' performance. We call this technique **input ablation analysis** for its resemblance to the popular feature ablation analysis.

Previous work has observed that some coreference resolution systems rely too much on mentions (Emami et al., 2019). To study this effect directly and at scale, we investigate performance of systems on text where we either **mask** mentions or their surrounding context or **map** mentions to alternative tokens that contain only a part of the original information. The proposed transformations allow us to compare multiple systems from various perspectives beyond overall performance. Moreover, the transformed texts can be worked on by human annotators. Understanding how people solve them brings valuable insights into how to improve systems.

Using these automatic and manual annotations combined with

interviews and ad-hoc analyses, we aim to provide insights to researchers for choosing worthy research directions in coreference resolution. Constructed corpora are released so they can be used as test suites for future approaches and annotations are available for further analysis.[1] We show that on OntoNotes and with respect to the linking step of coreference resolution:

1. All studied systems exclusively rely on mentions while ignoring the surrounding context.
2. Humans can also perform well while seeing mentions only. However, the way they achieve this is via a deep understanding of the remaining text.
3. We find evidence that the studied neural end-to-end system (Lee et al., 2017) does not capture instance-specific knowledge from names.
4. Instance-level knowledge, such as enabled by joint processing of coreference and entity linking, provides modest benefit.

The method itself forms the main contribution of this paper. It can be adapted to other NLP tasks and applied to further analyze results of any fully replicable experiment.

This paper is structured as follows: we first describe input ablation analysis and the transformations used in our research in more detail (Section 2) and then provide the settings chosen for our experiments (Section 3). Given this foundation, we present the results of our experiments in Section 4. Finally, related work is discussed in Section 5 and conclusions offered in Section 6.

2. Input Ablation Analysis

Feature ablation analysis is a popular method for analyzing NLP systems. When applying it, researchers remove features one-by-one and observe the change in the performance. If *gender*

[1] `https://bitbucket.org/cltl/even`

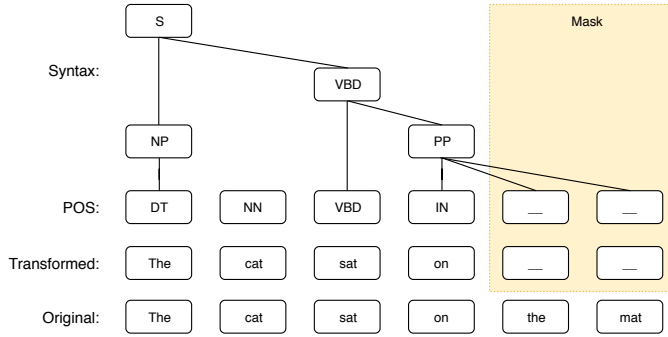


Figure 5.1: An altered text and parse tree for the sentence "The cat sat on the mat". Not only a chunk of text is masked, but also the corresponding PoS tags and the syntactic subtree that would have covered them (an NP in this case) is removed.

matching, e.g. a popular feature, is taken away, a system might mistakenly annotate *Hillary Clinton* and *he* as coreferential.

In a similar vein, we take away parts of the text or layers of information via two operations: masking and mapping. To mask a certain text span, we replace each word with <MASKED> (represented here as a blank for readability). Some coreference resolution systems start with layers of interpretation (syntax, semantics, etc.) already given. Because the robustness of, for example, syntactic and semantic parsers against corrupted texts is an orthogonal research topic, we do not re-parse documents. Instead, we apply transformations directly on gold part-of-speech (PoS) tags, named-entity tags, syntax trees and predicate-argument structures as illustrated in Figure 5.1. Systems are evaluated on both the original and derived texts and difference in performance is measured. Naturally, the higher the number of masked tokens is, the bigger the difference is expected.

Mapping is a generalization of masking, where the original text span is replaced by a new one that may maintain some information from the original span. The precise operations employed in this research will be defined in the rest of this section.

The proposed approach is applicable to any analysis for which the target sites of transformation can be reliably determined. It is not suitable to study, for example, the impact of context in mention boundary detection because mentions and context are not defined until the task is done. For this reason, we have excluded mention boundary detection from our analysis and rely instead on gold mention boundaries. This paper focuses on perturbing CoNLL-2012 (Pradhan et al., 2012) texts which have served as a standard benchmark in coreference resolution in recent years. CoNLL-2012 is attractive because it comes with gold-standard and automatic annotations of multiple layers of interpretation. This helps us focus the comparison on coreference resolution only and identify targets of transformations (e.g. names) with confidence.

2.1. Masking Mentions and Context

Several researchers have used textual context to improve coreference (Versley et al., 2008; Peters et al., 2018; Rahman and Ng, 2012; Peng and Roth, 2016, e.g.) However, it is not clear how much context contributes to their performance. Emami et al. (2019) create a dataset to test if a system ignores the predicate-argument structure preceding a mention and observe that existing systems do so in 22-100% of the cases. It is also not clear how much we can improve performance by employing context and what kind of context understanding is needed to reach human-level performance.

To address these questions, we conduct experiments on automatic systems where we mask either mentions or context. In CoNLL-2012, mentions include noun phrases, pronouns, but also verbs that refer to a person, object or event. Notably, singletons are not included therefore any mention corefers with at least one other mention. The majority of mentions are short, containing on average 2.4 tokens (std=3.3, variations between genres are negli-

gible). In the context of the current paper, we define context as any text passages other than mentions. To illustrate the proposed transformations, consider the following passage from document bn/cnn/00/cnn_0030 in CoNLL-2012 training set:

(2) [Secretary of State Madeleine Albright]$_1$ has accepted an invitation to visit [North Korea]$_2$ and meet with [leader Kim Jong-il]$_3$. [She]$_1$ made the unexpected announcement at a [dinner]$_4$ last night in [Washington]$_5$.

The following two versions of the excerpt have their mentions or context masked:

(3) [_ _ _ _ _]$_1$ has accepted an invitation to visit [_ _]$_2$ and meet with [_ _ _]$_3$. [_]$_1$ made the unexpected announcement at a [_]$_4$ last night in [_]$_5$.

(4) [Secretary of State Madeleine Albright]$_1$ _ _ _ _ _ _ [North Korea]$_2$ _ _ _ [leader Kim Jong-il]$_3$. [She]$_1$ _ _ _ _ _ _ [dinner]$_4$ _ _ _ [Washington]$_5$.

We ask human annotators to solve a subset of the documents to gauge the possible usefulness of each type of information. To deepen the understanding of the processing of mentions and context, we also annotate documents where $k\%$ of either mentions or context is removed, with $k \in \{20, 40, 60, 80\}$. For a given type of transformation, the sets of masked tokens M_k are selected randomly according to a uniform distribution, satisfying the constraint: $M_p \subset M_q, \forall p < q$. In words, each version is derived from the previous one by masking some additional tokens.

2.2. Mapping Proper Names

Proper names deserve special attention because of their variety compared to other mentions. Measured on the CoNLL-2012 development set, they account for 23% of tokens within mention boundaries and 36% of types. We study the impact of names

by using an one-to-one mapping to randomly generated tokens, keeping their most basic property while hiding gender, type, and the entity, among other information. For Excerpt (2), this transformation returns:

(5) [Secretary of State *_ROO12_ _PI3_*]$_1$ has accepted an invitation to visit [*_PHY9_ _JOOL_*]$_2$ and meet with [leader *_GER16_ _XUN_*]$_3$. [She]$_1$ made the unexpected announcement at a [dinner]$_4$ last night in [*_FIM32_*]$_5$.

To deal with names, traditional models employ a variety of string-matching features such as exact string match (Soon et al., 2001), substring match (Ng and Cardie, 2002b), and "nationality matching" (Ng, 2007). Interestingly, these features are missing in newer end-to-end models (Lee et al., 2018, 2017). We hypothesize that this omission leads to either memorization, where prevalent names are assigned unwarranted meanings; or negligence, where names are ignored during the formation of coreferential clusters. A certain level of memorization of proper names is beneficial because some names, such as *United States*, *Japan*, *Disneyland*, are prevalent and have stable and unambiguous meaning. However, excessive memorization is not desirable because of two reasons. First, corpora are often constructed from a set of similar documents from the same time period and social context, resulting in a bigger train-test overlap than what can be expected in a natural setting. Second, even when a system encounters names it was trained on, their referent and attributes such as type and gender might differ. A mention of *Clinton* is, for instance, more likely to refer to *Hillary Clinton* in 2016 and to *Bill Clinton* in the 1990s.

To detect memorization and negligence, we propose several transformations of names:

ORIGINAL The original documents are returned intact.

-EXTERNAL Mapping a name into a randomly-generated to-

ken as in Example (5), therefore making it impossible to connect to external sources of knowledge such as word embeddings, gazetteers, and ontologies. The same dictionary is used for the training, development, and test sets.

-INTERNAL Mapping in the same way as -EXTERNAL, but using different dictionaries in training, development, and test sets, thus preventing exact matching againts memorized names.

-MATCH Occurrences of the same name are mapped into different tokens even within a document.

MASKED Names are replaced by a special token.

If a system ignores names, it will perform at similar levels on -ORIGINAL and -EXTERNAL. Otherwise, we expect to see a drop in performance associated with -EXTERNAL. If a model memorizes names, we expect a further decrease of performance on -INTERNAL when task-internal knowledge is removed.

For human annotators, we merge -EXTERNAL and -INTERNAL into one condition named MAPPED and eliminate -MATCH to avoid confusion. We attempt to disentangle instance-specific knowledge and class-level knowledge by asking annotators to predict the true identity and properties of name-mapped entities. If an annotator can guess some properties of an entity but cannot pinpoint it to the exact referent, related errors can be attributed to a mixture of missing instance-specific knowledge and other factors such as heightened cognitive load. On the other hand, if annotators can correctly guess the hidden entity, they have access to the instance-specific knowledge about it and the first cause can be eliminated.

3. Experimental Settings

In this section, we describe the settings we used to evaluate automatic systems and human annotators.

3.1. Evaluation

Coreference resolution can be solved in two steps: mention detection and mention linking (Bengtson and Roth, 2008). Because the proposed transformations reveal mention boundaries as a side effect, we evaluate the second step only, i.e. all models are instructed to use gold-standard mention boundaries and mentions are clearly delimited in the annotation interface.

Following common practice (Pradhan et al., 2012), we use average F_1 to measure performance. It should be noted that our manual annotation is different from the conventional setting. Instead of building a gold standard, we are interested in how well a human can still perform the task while some information is removed. Therefore, instead of calculating inter-annotator agreement, we compare annotations done on edited documents with the existing gold standard and measure average F_1 score.

3.2. Dataset

CoNLL-2012 (Pradhan et al., 2012) contains 2,385 annotated English documents, totaling at 1.6M words. The documents come from various genres such as newswire, weblogs, and telephone conversations. The dataset is distributed with a stable train/development/test division. We will use the training set for training while merging development and test set for annotation and evaluation. Generally speaking, one annotator cannot work on two transformations of the same document, we thus need more documents to obtain statistically meaningful results.

3.3. Systems and Baselines

To demonstrate the value of our analysis, we will evaluate high-performance open source systems that represent different paradigms:

SIEVE (Lee et al., 2013) is a rule-based system composed of multiple stages from the most precise to the highest recall, tied together by hard-coded constraints.

CORT-MP (Martschat and Strube, 2015) is an implementation of the simple-yet-competitive mention-pair approach powered by a simple linear model (perceptron) and a rich set of features.

CORT-MR is an implementation of mention ranking from the same library. This approach represents competitions between possible antecedents and has been shown to improve performance over the pair-wise approach (Denis and Baldridge, 2008).

CORT-EM realizes the entity-mention approach. It goes further than mention-ranking by encoding entities as a tree of corefering mentions connected by anaphor-antecedent relations.

DEEP-COREF (Clark and Manning, 2016b) is an neural entity-ranking model. It encodes entity-level information by pooling the distributed representation of all mention pairs that connect two entities. It also reduces the number of hand-crafted features, replacing some of them with distributed features learned by a neural network.

E2E (Lee et al., 2017) replaces all hand-crafted features with BiLSTM-based contextualized representation of words. The decoding scheme is mention ranking with pair-wise scores computed by a two-layer feed-forward neural network.

To ensure that the difference in performance is not due to out-of-vocabulary tokens, we retrain models on the transformed training set for each version of CONLL-2012.[2] To put the results in perspective, we also include two baselines:

[2]The most recent models (Joshi et al., 2020, 2019) are not easily retrainable, which is why we leave their analysis to future work.

RANDOM iterates a randomly-ordered list of mentions and picks antecedents randomly with an uniform distribution from the pool of previous mentions plus a special mention ε (meaning to create a new cluster).

HEAD-MATCH links all mentions that have the same head word together. Head words are found by syntactic rules defined in the Cort library (Martschat and Strube, 2015).

3.4. Manual Annotation

To investigate what level of performance is possible without certain types of information, we ask human annotators to perform the same task systems do, i.e. linking given mentions into coreferring clusters. Annotations were performed by three master's students, one native and two near-native speakers of English, with a background in NLP. Details about the annotation process and annotation tool can be found in Section 1 and Section 2 of Supplementary Material.

It is important to note that we do not aim to investigate general properties of language processing in humans. Instead, we are interested in studying the contribution of certain features to the performance of powerful and interpretable systems and evidence from a handful of human annotators can be sufficient to gain such insights.

As an artefact of the annotation process, the distribution of genres differs across annotators and types of transformation. We counter this by resampling 25 times according to the original distribution and present here the mean and standard deviation of performance on samples. For a fair comparison, experiments on automatic systems are also performed on 25 resampled versions.

4. Results and Discussions

In this section, we present the results of our experiments. Each of the following subsections will discuss one question concerning

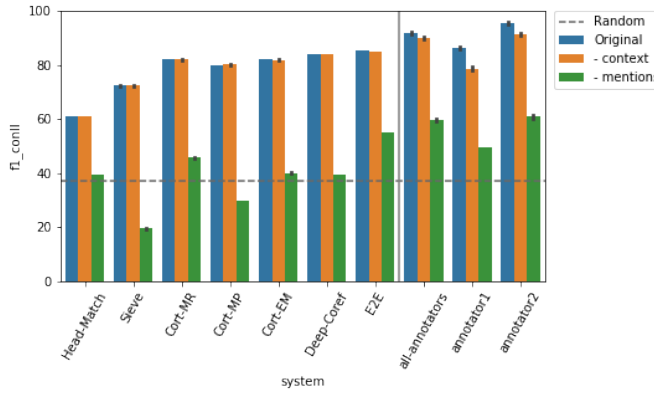


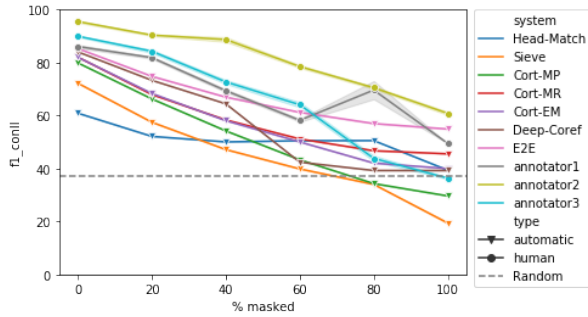
Figure 5.2: Performance of automatic systems and annotators when either context or mentions are removed. The horizontal dashed line is the random baseline. During data processing, we discovered that the third annotator only worked on one uncommon genre in their -CONTEXT set. For this reason, we drop the annotator's results from this figure.

either automatic or human performance.

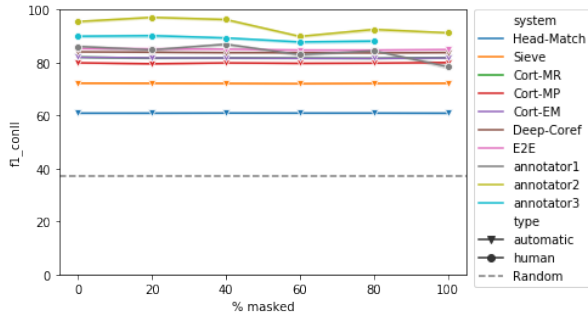
4.1. Mentions and Context for Systems

The results in Figure 5.2 (left part) and Figure 5.3 support the hypothesis that algorithms rely almost exclusively on the content within mention boundaries. When context is masked, all models show an unchanged level of performance whereas when mentions are masked, most systems decay linearly and rest at or below chance level.

Although it is still possible that a model trained on ORIG learns how to make use of context, the results show that the contribution of this information is insignificant or negative. For SIEVE and CORT, this is not surprising given that their feature sets contain no information from the surrounding context. DEEP-COREF, on the other hand, does include preceding and following words to a mention, all words in the sentence and document, but their contribution is minimal (0.28% in absolute F_1). E2E composes span representation from BiLSTM embeddings which is com-



(a) Masking mentions



(b) Masking context

Figure 5.3: Performance of automatic systems and annotators when more and more mention or context tokens are masked. Because the third annotator happened to sees only one uncommon genre in their set of 100% context-masked documents, we have dropped the corresponding point.

puted over whole sentences. Theoretically, this design allows the model to make use of the context. In practice, it provides only insignificant benefit over what was already achieved by exploiting mentions, showing a difference of only 0.36% absolute F_1. Nevertheless, the representations it computes for the remaining context prove useful as it outperforms other systems in -MENTION setting.

In the -MENTION setting, the HEAD-MATCH baseline produces a single cluster per document. This strategy gets very low score on CEAF (Luo, 2005): 6.4%F_1, but achieves 23.84%F_1

on B^3 (Bagga and Baldwin, 1998) with a highly skewed precision/recall trade-off: P=13.65%, R=100%. It also gets a high score of 87.46%F_1 on MUC (Vilain et al., 1995), which is known to favor fewer big clusters (Luo, 2005). Therefore, the overall performance (which is the average of the three) is rather high at 39.23%.

4.2. Estimating the Value of Mentions and Context

Because humans are the gold-standard natural language processors, analyzing their performance can provide insights into what kind of information is (not) useful for improving the performance of automatic systems. Let $\Delta_{\mathscr{I}}$ be the drop of performance on documents with the information $\mathscr{I}$ removed compared to original documents, if $\Delta_{\mathscr{I}_1} > \Delta_{\mathscr{I}_2}$ then we can reason that $\mathscr{I}_1$ is more useful than $\mathscr{I}_2$.

The right part of Figure 5.2 shows the performance of our annotators on perturbed documents. It is immediately clear from the plot that $\Delta_{\text{mention}} \gg \Delta_{\text{context}}$. The effect of context masking, as measured by change in average F_1, is smaller than one might expect: $\Delta = 5.54\%$ for Annotator 1, $\Delta = 3.48\%$ for Annotator 2. This result is consistent across metrics: when we pool all manual annotations together (called *all-annotators* in Figure 5.2), we found $\Delta = 2.26$ for B^3, $\Delta = 1.49$ for CEAF, and $\Delta = 1.88$ for MUC. Moreover, the line of performance vs. context masking in Figure 5.3b indicates that much of the effect might be because of random fluctuation. We asked the annotator to rate their confidence in a 0-5 scale and the recorded levels show only a slight decrease as more and more context is masked.

These results demonstrate that it is possible to solve CoNLL-2012 with high performance without making use of context. While it is still possible that context processing is needed to reach human-level performance, research on the effect of context on coreference resolution should consider evaluating on other

datasets, for example Zhao et al. (2018) and Emami et al. (2019), for higher sensitivity.

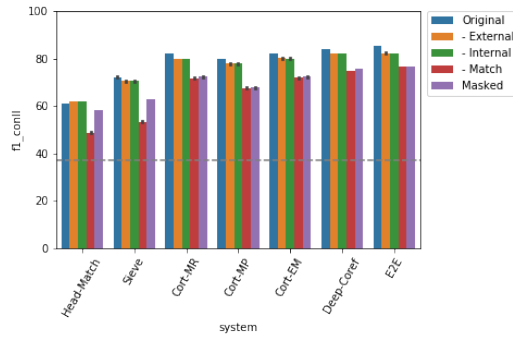
4.3. The Role of Comprehension

The results discussed above seem to suggest that CoNLL-2012 can largely be solved through mention-matching only. We further explore how annotators perform the task to verify this outcome.

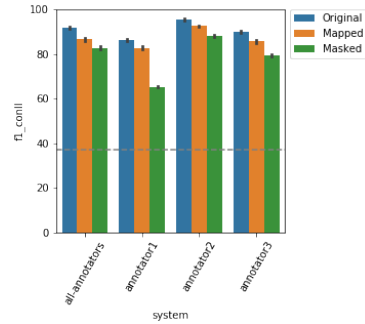
We asked annotators to summarize a random selection of documents they annotated. They were instructed to capture as much of what is described in a document as possible using no more than 3 sentences (for details, see Section 3 in Supplementary Material). A total of 210 summaries were gathered. We found that, although it is true that masking severely affects comprehension, our annotators can still come up with 3 summarizing sentences for 17 out of 63 documents that are masked by 80% or more. The average length of those sentences are 10.9 tokens (excluding punctuations). Table 5.1 presents some randomly chosen samples of the summaries.

Sometimes CoNLL-2012 mentions include rich information that can be considered a kind of embedded context. Only 3.6% of mentions in the dataset contain 10 or more tokens so this type of mentions must be rare. However, humans are good at using very little information to construct stories. These stories are not always accurate but provide a foundation for annotators to reason about coreference. This interpretation is supported by our interviews with annotators (see Section 4 in Supplementary Material).

The results in this section suggest that although both humans and machines are robust to context removal, they do so for different reasons. If the aim is to create coreference resolution systems that truly understand coreference, and therefore generalize better, we might need to work on deeper processing of all that is available in the text instead of ignoring a big chunk of it. If the aim is to reach human-level performance on CoNLL-2012, the way



(a) Systems



(b) Humans

Figure 5.4: Performance of systems and human annotators against different methods of name manipulation

annotators solve hard cases should also provide a hint for how to overcome the last 10%.

4.4. Names for Systems

Figure 5.4a shows that names are indeed an important source of information to coreference resolution systems. Comparing MASKED and ORIGINAL, we can see that the benefit of name processing in both rule-based and machine learning systems exceeds that of simple head matching and explains between 6% and 12% of absolute F_1 score.

The results show that machine learning-based systems are able to ignore names when they are misleading: their performance

is almost the same for -MATCH and MASKED conditions. In contrast, because HEAD-MATCH and SIEVE lack this ability, their performance drops significantly.

It is clear that no system ignores names in -EXTERNAL and -INTERNAL conditions because the performance stays significantly higher than that of -MATCH and MASKED. E2E did not memorize names either as evident in the identical performance on -EXTERNAL and -INTERNAL despite substantial overlap of between training and evaluation sets that it can use in the -INTERNAL condition but not in -EXTERNAL.

This leaves open the question of how E2E makes use of names. One possibility is that it learns what names look like and puts similar ones into the same cluster or, to the same effect, avoids putting different-looking names in the same cluster. To test this hypothesis, we put a model trained on -INTERNAL through evaluation on ORIGINAL and vice versa. This leads to a loss of -4.85% and -5.65% in absolute F_1 compared to models trained and tested on the same variant of names. Together, the results in this section suggest that E2E does not link names to knowledge about individual entities, including those that are unambiguous and beneficial, and might not be able to differentiate similar-looking names.

4.5. Estimating the Value of Instance-level Knowledge

The performance drop associated with mapped or masked names is the total effect of a few factors: the loss of identifiers (same tokens often mean same entity), the loss of type information (such as an entity being a person or a country), the loss of instance-level information (such as encyclopedic knowledge about Barack Obama), and the effect of heightened cognitive load (the increase in error rate caused by working with randomly generated tokens). This observation can be summarized as:

$$\Delta_{\text{mapped}} = \Delta_{\text{instance}} + \Delta_{\text{type}} + \varepsilon \tag{5.1}$$

$$\Delta_{\text{masked}} = \Delta_{\text{instance}} + \Delta_{\text{type}} + \varepsilon + \Delta_{\text{identifiers}} = \Delta_{\text{mapped}} + \Delta_{\text{identifiers}} \tag{5.2}$$

where ε stands for fatigue-induced errors.

The drop caused by name-mapping in human annotators is between 2.9% and 4.3% absolute F_1. To differentiate between type knowledge and instance-level knowledge, we asked annotators to guess the name or some properties of masked entities (see Table 5.2 for some samples). The annotators are highly proficient at inferring properties of hidden entities from context: they attempted to guess either the name or properties in 95% of 294 masked named-entities, with a precision of 86%. In 74% of guesses, the annotator answer the type of a masked name (e.g. that it is a place, a company, a location, or a governmental body) but cannot pinpoint the hidden entity. Assuming that fatigue-induced error rate is negligible, a big part of Δ_{mapped} can be attributed to the loss of instance-specific knowledge. Compared to $\Delta_{\text{identifiers}}$ which ranges from 4.6% to 17.3%, the results indicate that harvesting instance-level knowledge, such as via joint coreference resolution and entity linking, can bring improvement but only modestly.

5. Related Work

Following the rise of neural networks in natural language processing, analysis methods have gained interest (Belinkov and Glass, 2019), including work that modulates input to gain insight into the inner workings of models. Li et al. (2016a) devised algorithms to find the minimum set of words to erase to change any prediction. On the flip side, Feng et al. (2018) study the maximum set of words that one can erase without changing a prediction. Wu et al. (2019a) build a framework to create, manage, and execute

rewriting rules which can be used to test hypotheses about a system. Different from our work, these approaches help interpret a system but do not allow comparison between different systems and between systems and human annotators. The research by Khandelwal et al. (2018) is closer to ours in this regard. The authors examine different ways to alter the context preceding a word to reveal how language models make use of context. Sankar et al. (2019) do the same for neural dialog systems.

In coreference resolution research, efforts have been put into analyzing CoNLL-2012 (Moosavi and Strube, 2017) and building challenge sets that are superior in specific properties: smaller lexical overlap (Ghaddar and Langlais, 2016), avoiding gender bias (Zhao et al., 2018; Rudinger et al., 2018; Webster et al., 2018) and testing for common-sense reasoning (Emami et al., 2019). While these corpora can demonstrate the shortcomings of current systems, they have little to say about how much their resolution might help to improve performance.

6. Conclusions

We analyze the behavior of systems while performing the linking step of coreference resolution on CoNLL-2012 data. By analyzing the response of systems to changes in input, we demonstrate that a broad sample of methods is exclusively dependent on mention-bound features, including methods that compute contextualized embeddings over whole sentences. This outcome supports and extends earlier findings by Emami et al. (2019). We also show evidence that the neural end-to-end system of Lee et al. (2017) might not capture instance-specific knowledge of names, including unambiguous and beneficial ones.

Human annotators in this research project can perform coreference linking on CoNLL-2012 to a high F_1 while using hardly any context, seemingly supporting its omission in many systems. Analysis of manual annotations, surveys, and interviews showed,

however, that they achieve this via a deep understanding of mentions and the embedded context. Either by context processing or enhanced mention processing, a deeper understanding of the text is likely needed to bring the studied systems to human-level performance.

The next step in our research is to apply our method and investigate if Transformer-based models indeed make better use of context. In addition, the research in this paper can be extended in two directions: similar experiments can be done on other datasets and a larger study in which people solve cases where systems make errors would further our understanding of human-level coreference resolution.

7. Supplementary Material

7.1. Annotation Procedure

Before annotating target documents, we required annotators to review relevant sections in OntoNotes guidelines (Weischedel et al., 2013a) and annotate 15 unmodified documents to at least 90% F_1. Throughout the project, we also include original documents as a means of quality checking and discard batches that have performance on control documents lower than 80%.

Because one of the goals of our research is to uncover information that might improve the performance of automatic systems, we ask annotators not to use heuristics but make decisions based on their understanding of the text. There are three mechanisms through which we encourage this behavior. First, when someone starts annotating, only the first few sentences are shown and new ones are revealed after all mentions on screen are resolved. This is to avoid skimming and easy-first, string-matching heuristics. Second, we randomly ask for confidence judgment in the scale of 0 (not confident) to 5 (absolutely certain). Third, we hold weekly meetings in which annotators take turn to annotate while

"thinking out loud" or discuss difficult cases. To avoid information leakage, an annotator is not allowed to work on two incompatible versions of the same document.[3]

7.2. Annotation Interface

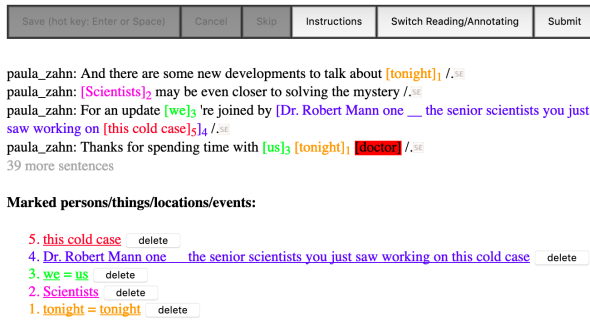


Figure 5.5: The user interface of our crowd-based annotation tool. Annotators start with a given set of mentions marked in red. They put mentions into "groups" (i.e. clusters or entities) by clicking on them. They can also mark singleton groups, remove a mention from a group or break a group into stand-alone mentions to start over.

To facilitate annotation, we aim at making an intuitive and distraction-free user interface. This translates into the following design choices:

1. The interface should allow the user to record multiple positive coreference decisions after reading one passage of text as opposed to the one-question-per-text approach of Phrase Detectives (Poesio et al., 2013).
2. Similar to GATE (Cunningham et al., 2002) but different from BRAT (Stenetorp et al., 2012) and MMAX2 (Kopeć, 2014), it should represent group membership by colors and annotations instead of arrows. This is to avoid displacing pieces of the text and cluttering the space with too many

[3] One way that two versions become incompatible is that they belong to different types of manipulations, e.g. context masking and name-mapping. If they are of the same type, we allow a version with less masking to follow one with more masking but not the other way around.

intersecting arrows.

3. It should only require clicking on the text itself most of the time to minimize eye movement. Different from the QuizBowl annotation app (Guha et al., 2015), When grouping two mentions together, we require annotators to click on both to lower the chance of mistake.

These considerations crystallize into the interface depicted in Figure 5.5. At the end of the annotation project, we completed 1,100 documents in a total of 34.5 person-days. Excluding the initial training phase, we annotated at a speed of roughly 15 minutes/document.

7.3. Summary Experiment

We asked annotators to summarize a random selection of documents they annotated. They were instructed to summarize each document in 3 sentences, where 1 or 2 sentences were also allowed if their understanding did not allow for three sentences. They were not allowed to use generic expressions (e.g. *someone*, *something*). For documents that mainly contain conversations, we requested reports about what was talked about rather than the conversations themselves.

A total of 210 summaries were gathered. Figure 5.6 shows the statistics of summaries as more and more content is masked.

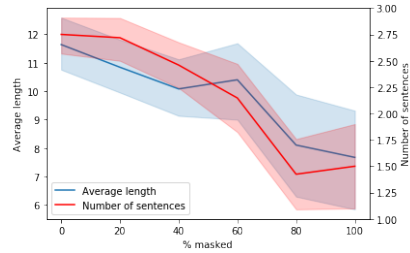


Figure 5.6: Number of sentences and average sentence length given by annotators as more and more context is removed.

7.4. Interviews with Annotators

We conducted 3 interviews with annotators at different points in the project. Each interview lasts between 23 minutes and 1 hour in which annotators annotate a few documents while sharing their screen with everyone else. Annotators are asked to "think out loud" and the experimenter can ask questions to help them elaborate their thinking process. All interviews are recorded.

To see what thinking process is imployed to solve hard documents, consider the following excerpt from a 100% context-masked document:[4]

(6) _ _ _
_ [Taiwan] _ _ _ _ _ _ _ _ _ [Ruan] _ _ _ _ [the mainland] _ _
[They] _ _ _ _ _ _ _ [the election] _ _ _ _ _ _ _ [the election] _ _ [they] _ _ _ _ _ _ _ _ [We] _ _ _ _ _ [they] _ _ _ _ _
_ _ _ [them] _ _ [Ruan 's] _ _ _ _ _ _
[Ruan] _ _ _ _ _ _ _ _ _ _ _ _ _ _ [Chen Shui - bian] _ _ _ _ _ _ _ _ _
[He]$_9$ _ [Chen] _ [the staunchly anti-communist Richard Nixon, who was able to normalize ties with [the Chinese communists] precisely because [he]$_{10}$ was trusted by the American people]

The reasoning line that an annotator used to resolve [he]$_{10}$ is as follows:

> Annotator 3: Uhm … I'm not sure this *he* refers to which, *Nixon* or to *Chen*. But, honestly, from my world knowledge I would say he [pointing to *Nixon*] because I just assume not many people in the 60's from the US knew any Chinese people and trusted them, so …

In a telephone conversation, after 80% of context is masked,

[4]Document: dev/mz_sinorama_10_ectb_1020 (part 001)

we get:[5]

(7) B: __ __ [Cocoa] [she] [got] __
A: [She] [married] [that guy]
B: __ __ __ __ __ __ __ guy __ __ [you] __ kidding __ little __ __ __ __
[. . .]
B: __
B: yeah
B: __ __ very __
B: __ [they] [got] __

To solve the mentions *they* and *got* in the last line, an annotator used common sense knowledge about marriage:

> Annotator 2: about *they* . . . I assume that's Cocoa and this guy [. . .] I'll probably say that this [pointing to the last *got*] goes back to *got* here [pointing to the first one] which now I think this is still referring to the marriage and I'm gonna just group these two [cluster of *got* and cluster of *married*] together.

Although we did find documents with repetitive pronouns that can be solved without understanding the storyline, they are a minority and cluster in certain genres:

> Annotator 1: . . . the one before this which is a biblical text [. . .] the format is the same. Like, recently I was doing one and it was about 90 lines and I thought, wow, this is gonna take a long time but it actually ends up very quickly because . . . the lines are shorter, or repetitive sometimes, maybe.
>
> Annotator 3: [. . .] biblical text, they follow certain pattern [. . .] it's different from the forum part, those online things, where people just blurb out what they think.

[5] Document: dev/tc_ch_00_ch_0020 (part 001)

Document: test/mz-sinorama-10-ectb-1049 (part 004), masked context: 100% **Excerpt:** [The weeping river] $ __ __ __ __ [the weeping junks] __ $ __ [several of these boats] __ $ __ __ __ __ __ __ __ __ [the sunken junks] __ __ __ __$ **Summary:** The government is planning to build a riverside highway to Tanshui. Another Tanshui project involves building high-rise apartments. Some residents of Tanshui are unhappy with these projects.
Document: test/nw-wsj-23-wsj-2344 (part 000), masked context: 100% **Excerpt:** [Gulf Resources & Chemical Corp.] __ [it] __ __ [pay] __ __ __ __ __ __ [an accord with [the Environmental Protection Agency] regarding an environmental cleanup of [a defunct smelter [the company] formerly operated in Idaho]] __$ **Summary:** Gulf Resources & Chemical Corp. made an agreement with the EPA regarding the cleanup of one of their defunct facilities. The cleanup is to be funded by the federal Superfund program. In the meantime, the company is going through a reorganization process.
Document: test/bc-msnbc-00-msnbc-0007 (part 004), masked context: 80% **Excerpt:** Tim Russert: __ __ [the latest study from [the Institute for Science and International Security]] __ $ [They] __ __ __ __ [North Korea] __ __ __ __ __ develop __ __ __ __ __ __ __ nuclear __ __ __ __ __ __ __ __ two weapons in __ __ __ [George bush] __ __ __ $ **Summary:** According to a new study, North Korea could be developing nuclear weapons. Nicholas Burns believes that the U.S. still has an enormous influence on North Korea. The U.S. policy is to denuclearize North Korea.

Table 5.1: Sample of heavily masked documents and their summarizations. The symbol $ denotes the end of a sentence and square brackets denote mention boundaries.

Query	Excerpt	Guess	Key
DIM68, _NIE_ _ME_	In _DIM68_, _CA_ _PO_ _NIE_ _ME_ was back on television this morning trying to shore up public support for his legal cause.	location, politician	Washington, Al Gore
CEEN33	With regard to cross-strait relations, _HEE38_ made his famous "testicles analogy." _CEEN33_ is a part of China, he stated, just as testicles are a part of a man's body.	Taiwan	Taiwan
_ZE83_ _LEE_, _PHE_	_ZE83_ _LEE_ received a very warm welcome in _PHE_, but the local press still mentioned "checkbook diplomacy" in reporting on the visit.	president, a developing country	President Chen, Nicaragua

Table 5.2: Samples of annotators' guesses about the identity or characteristics of hidden named-entities.

Neural Models of Selectional Preferences for Implicit Semantic Role Labeling

Minh Le and **Antske Fokkens**
Department of Language, Literature and Communication
Vrije Universiteit Amsterdam
{m.n.le,antske.fokkens}@vu.nl

Abstract

Implicit Semantic Role Labeling is a challenging task: it requires high-level understanding of the text while annotated data is very limited. Due to the lack of training data, most researches either resort to simplistic machine learning methods or focus on automatically acquiring training data. In this paper, we explore the possibilities of using more complex and expressive machine learning models trained on a large amount of explicit roles. In addition, we compare the impact of one-way and multi-way selectional preference with the hypothesis that the added information in multi-way models are beneficial. Although our models surpass a baseline that uses prototypical vectors for SemEval-2010, we otherwise face mostly negative results. Selectional preference models perform lower than the baseline on ON5V, a dataset of five ambiguous and frequent verbs. They are also outperformed by the Naïve Bayes model of Feizabadi and Padó (2015) on both datasets. We conclude that, even though multi-way selectional preference improves results for predicting explicit semantic roles compared to one-way selectional preference, it harms performance for implicit roles. We release our source code, including the reimplementation

of two previously unavailable systems to enable further experimentation.

Keywords: neural network, implicit semantic role labeling, selectional preferences

1. Introduction

Defined as the recovery of semantic roles beyond immediate syntactic structure, implicit Semantic Roles Labeling (iSRL) can contribute valuable information for obtaining complete semantic interpretations of text. Yet, it has been elusive since its first shared task eight years ago (Ruppenhofer et al., 2009).

The main difficulty faced by researchers is the small size of training data. Compared to traditional SRL datasets, SemEval-2010 is hundreds-fold smaller, containing only slightly more than a hundred of training examples (Table 5.3). Early work applying traditional semantic role labeling (SRL) techniques to iSRL was met with deflating results. Therefore, researchers limited themselves to simplistic machine learning models such as Naïve Bayes (Feizabadi and Padó, 2015, among others) or abandoned machine learning altogether (Laparra and Rigau, 2013). Several studies were devoted to the automatic expansion of training data (see Section for an overview).

This paper presents an attempt to recover implicit semantic roles using neural networks. We take advantage of the fact that OntoNotes contains a vast amount of manually annotated explicit semantic roles from which we can learn the selectional preference of frames (e.g. *look.01* prefers animate fillers for role *A0* (looker)). A neural network is used to capture complex interactions between a predicate, a target role and its co-occurring roles. In addition, we compare the impact of one-way selectional preference, taking only the selectional preference of the predicate for the target role into account, to multi-way selectional preference, which uses

	SemEval		OntoNotes
	Train	Test	
Words	8K	9K	1,700K
Frames	344	371	7,007
Predicates	811	1,008	324,996
Predicates with DNI	102	118	0

Table 5.3: Statistics of an iSRL dataset (SemEval-2010, PropBank version) and a traditional SRL dataset (OntoNotes).

information from all semantic roles related to the predicate.

The contribution of this paper is twofold: First, we experimented with a class of simple neural models for iSRL and two types of selection preference. While the results are mostly negative, they highlight the importance of discourse information (see Section and) and suggest future directions that should (not) be taken.

The second contribution lies in addressing the challenges we met in carrying out this research and interpreting our results. The nature of these challenges lies in the fact that (1) all resources for implicit Semantic Role Labeling are small, (2) previous approaches differ in the dataset and the metrics they use for evaluation, and (3) to our knowledge, none of the existing systems is available as open source code. This has led to a situation that is typical for challenging tasks using small datasets: it is almost impossible to determine what the state-of-the-art approach is and how new work relates to this. Even results from papers that are evaluated on the same dataset are difficult to compare, because differences in results can be due to the difference in features, machine learning algorithm, method of extending data, heuristics or (as pointed out in Fokkens et al. (2013b)) choices in preprocessing and data preparation. As part of this research, we built an experimental platform for iSRL. This platform provides open source implementations for the experiments reported in this paper, for the system described in Schenk and Chiarcos (2016)

which inspired our own approach and for Feizabadi and Padó (2015)'s approach which provided state-of-the-art performance on SemEval-2010.

The rest of the paper is organized as follows: Section summarizes the foundation of iSRL and related work. In Section , we outline our models of selectional preference. Section quantifies the effectiveness of selectional preference with regard to iSRL. Section concludes the work and outlines future directions of research.

2. Background and Related Work

In this section, we explain what implicit Semantic Role Labeling entails. This is followed by an overview of previous work on this task. Next, we address related work that uses selectional preferences. Consider the following sentence from SemEval-2010 training set:

(8) Apparently [the tenants]$_{\text{A0}}$ had [brought]$_{\text{bring.01}}$ [little or nothing]$_{\text{A1}}$ with them, and all the furniture down to the smallest details had been taken over with [the house]$_{\text{A2}}$.

The roles *A0* and *A1* of the predicate *bring.01* can be filled with phrases in the immediate syntactic structure while the filler of *A2* falls into a separate clause. Typically, a SRL system would annotate the fillers for *A0* and *A1* and ignore *A2*. It is therefore called a Null Instantiation (NI).

Null-instantiations can be indefinite (INI) and definite (DNI). To reuse examples from Ruppenhofer et al. (2009), in the blog headline *More babbling about what it means to know*, the subject of knowing is not expected to be instantiated within the discourse. In contrast, in the sentence *Don't tell me you didn't know!*, the hearer expects a concrete filler for the role of what (s)he should know and it can be expected to be present in previous context.

The first example is a case of INI while the second is a DNI.

2.1. Previous work on iSRL

Traditional SRL techniques led to very low results for iSRL due to data sparseness (Chen et al., 2010; Tonelli and Delmonte, 2010). Researchers therefore explored simpler alternatives such as BayesNet (Silberer and Frank, 2012; Roth and Frank, 2013, 2015), Naïve Bayes (Feizabadi and Padó, 2015), and memory-based learning (Schenk et al., 2015). Others proposed non-parametric approaches such as observed frequency (Laparra and Rigau, 2012), prototypical vectors (Schenk and Chiarcos, 2016) and other heuristics (Laparra and Rigau, 2013; Gorinski et al., 2013).

In addition to methods of machine learning and heuristics, previous work investigated the possibilities of increasing training data. Feizabadi and Padó (2015) combine multiple corpora and apply domain adaptation methods to deal with the difference in genre. They demonstrated that combining two iSRL corpora led to improved performance. Silberer and Frank (2012) and Roth and Frank (2015) used heuristics to generate iSRL training examples from manually and automatically annotated SRL corpora. This work differs from these approaches, because their research focused on creating iSRL training examples of reasonable quality rather than using a SRL resource directly.

2.2. Selectional preferences

Selectional preference has a long research tradition (Katz and Fodor, 1963) and has been applied in various tasks such as syntactic parsing (Zhou et al., 2011), textual inference (Ritter et al., 2010), and semantic role labeling (Zapirain et al., 2013). The idea is simple: a role is filled with some words more frequently than others. For example, *the man* is much more likely a filler for the role *A0* (leader) of the predicate *lead.01* than e.g. *the bottle*

(an inanimate object) although one can construct a grammatically and semantically correct example for each filler.

Next to the role's semantics, co-occurring roles also have an influence. For example, if *lead.01*'s role *A4* (goal) is filled by *the guest house*, *the nation* is an implausible filler for *A1* (thing led), while it is perfectly plausible had we not known what fills *A4*. This is known as *multi-way selectional preference* (Van de Cruys, 2014).

One-way selectional preferences have been applied to implicit semantic role labeling before. Silberer and Frank (2012)'s system include a feature calculated using weighted similarity to head words that are observed to fill a role. The selectional preference model itself is described in (Erk, 2007) and (Resnik, 1996). A simpler model that uses unweighted similarity is used by Schenk and Chiarcos (2016).

Our results show that adding multi-way selectional preference improves results on explicit semantic roles, but not for iSRL. Recently work by Do et al. (2017) is closest to our work but they apply their methods on nominal data and did not compare one-way and multi-way selectional preference.

2.3. Neural networks

Recent years have witnessed a surge of research interest in neural networks for natural language processing (Goldberg, 2016). Plenty of models have been proposed for various tasks (Godbole et al., 2015; Zhou and Xu, 2015; Andor et al., 2016, among many others). Apart from Do et al. (2017) who uses a different architecture for a different version of the task, we are not aware of work that applies neural networks to iSRL.

3. Models

In this study, we focus exclusively on *DNI resolution*, the last and hardest step in iSRL. For each test case, we assume that the

predicate p is already identified and disambiguated, the target role r^* is given, and overt roles are coupled with their fillers $\{(r_j, g_j)|j = 1..m\}$. The goal is to rank the correct filler highest among the candidates $\{c_i|i = 1..n\}$. To test a simple multi-way selectional preference model, we use the following formula to assign a score for each candidate:

$$s(c_i) = \bigoplus_{j=1}^{m} f(p, r^*, r_j, g_j, c_i) \tag{5.3}$$

where m is the number of explicit roles known to the system, $\bigoplus$ is a aggregation function (e.g. *sum* or *max*). In the case of *one-way* selectional preference, the formula degenerates into:

$$s'(c_i) = f'(p, r^*, c_i) \tag{5.4}$$

f and f' are neural networks that have the same architecture, except the number of inputs. The precise form of the neural networks and the aggregation function is determined via a hyper-parameter search (see Section).

Role fillers $\{g_j\}$ and candidates $\{c_i\}$ can be transformed into features by extracting the head word, but other features can also be used. Together with predicate and role names, they are embedded into a vector space. The embedding matrix is trainable and can be initialized with pretrained word vectors for better performance.

Compared to a model that computes prototypical vectors as the average of observed vectors in the fashion of Schenk and Chiarcos (2016), our neural models have (at least) two advantages:

- Distributed representation is used to represent predicates and roles, not only fillers, allowing the model to work in cases of unseen predicates or predicate-role combinations.

- The representation enables the sharing of statistical strength between predicates, i.e. rare predicates can get more accurate predictions by means of resemblance to frequent predicates.

An additional motivation is that multi-way preference can offer a solution to the context-dependent nature of semantic role labeling. Our current results, however, do not provide sufficient evidence to support such a claim.

4. Experiments

We evaluate our models for DNI resolution by comparing our model to a baseline and to Feizabadi and Padó (2015). In addition, we perform an ablation analysis to find out which components of the model are useful.

4.1. Data

We train our selectional preference model on OntoNotes (Weischedel et al., 2013b), a balanced 1.7M words corpus with over 320K manually annotated predicates and their explicit arguments.

SemEval-2010 (Ruppenhofer et al., 2010) is a standard dataset to evaluate iSRL systems. It contains chapters of Sherlock Holmes, one for training and two for testing, annotated with both implicit and explicit semantic roles. The organizers provide two versions of the same dataset: one annotated with FrameNet roles and the other PropBank. Because OntoNotes was compiled using PropBank, we also use the PropBank version of SemEval-2010.

Note that OntoNotes differs from SemEval-2010 in task (explicit versus implicit SRL), genres (news, weblogs and conversations versus novel) and time period (20th century versus 19th century). Training on OntoNotes SRL and testing on SemEval-2010 iSRL can be seen as a form of domain adaptation and requires powerful generalization.

We also test our models on ON5V (Moor et al., 2013a) which

Predicate	Role	Filler head	Filler full
pay.01	A2	refugees	the refugees
pay.01	A2	U.S.	U.S.
pay.01	A2	families	the victims' families
pay.01	A2	trust	this trust
pay.01	A2	Warner	AOL Time Warner
pay.01	A2	they	they
pay.01	A2	lenders	lenders
pay.01	A2	lawyers	lawyers
pay.01	A2	one	one

Table 5.4: Examples from ON5V showing the diversity of fillers in terms of semantic types, part-of-speech, and topics.

poses a different challenge. Implicit semantic roles were manually annotated on top of explicit semantic roles and other linguistic information on a selection of OntoNotes documents. The authors chose five highly frequent verbs to annotate in order to create "high-volume of annotations for specific verb predicates".[1] As a result, the words and phrases that fill each role are very diverse, as illustrated by the examples in Table 5.4. To achieve high performance on this dataset, a model needs to be selective yet general enough to encompass different types of fillers.

4.2. Baseline

Our baseline is inspired by Schenk and Chiarcos (2016). For every <predicate, role> pair found in our training set, it computes a prototypical vector and, at test time, returns the candidate that is closest to the prototypical vector. Following their best model, we use the pretrained embeddings from Collobert et al. (2011).

Due to some differences between research questions and experimental setup, the results cannot be compared to Schenk and Chiarcos's algorithm directly. Firstly, for a fair comparison with selectional preference-based models, we use only the head word of each candidate (whereas they average all words in the phrase).

[1] They are: *pay*, *give*, *bring*, *leave*, *put*.

Secondly, we train and evaluate on PropBank-style datasets while they use FrameNet-style data.

4.3. Experimental Setup

We use the baseline described in the previous section and a Naïve Bayes model trained on SemEval-2010 data (Feizabadi and Padó, 2015) to compare to our model's performance. To quantify the effect of different aspects of the model, we investigate the following variants:

ONEWAY captures one-way selectional preference and represents fillers by their syntactic head.

MULTIWAY captures multi-way selectional preference and represents fillers by their syntactic head.

SYNSEM uses richer features for fillers rather than selectional preferences. We use five syntactic and semantic features from Feizabadi and Padó (2015), namely, *Expected roles*, *Semantic Type*, *Word Frequency*, *POS*, and *Constituent type*.[2]

SYNSEM+ONEWAY combines richer features with one-way selectional preference.

SYNSEM+MULTIWAY takes into account co-occurring roles to capture multi-way selectional preference.

We construct one training example for each argument found in OntoNotes and split the data into 90% for training and 10% for development. Models are trained to choose the right filler for each target role with as input: the predicate, the role and, if applicable, other explicit arguments. We evaluate on the NI-only test set from SemEval-2010 using the standard evaluation script (Ruppenhofer et al., 2009).

We initiated the embedding matrix with 27K vectors from the pretrained embeddings of Collobert et al. (2011). We use AdaGrad (Duchi et al., 2010) for optimization; the initial learning

[2]See Table 2 in their paper. We did not use their discourse features because they require iSRL annotations which is not available in OntoNotes.

	Train (%)	Validation (%)
ONEWAY	59.52	46.61±0.23
MULTIWAY	58.52	47.64±0.19

Table 5.5: Accuracy of selectional preference models on OntoNotes (for validation set we report mean and standard deviation over 5 runs).

rate was customized for each model to avoid gradient explosion. All models were trained until no improvement was observed on the development set (but not more than 1,000 epochs, for practical reasons). To account for random initialization in neural networks, we run each model 15 times and average the results. An arbitrary but fixed random seed was used for each run to ensure reproducibility.

All hyperparameters were tuned on OntoNotes development set. We tested sum and max for aggregation function; sigmoid, tanh, and cube for activation function (Chen and Manning, 2014); different strength of dropout (Hinton et al., 2012), regularization, and learning rate. Because of limited computational resource, we performed a random hyperparameter search to find the best setting. As discussed in Section , fillers can be represented in different ways. We observed that using both the head word and the closest coreferent non-pronoun head word is better than using the head word only on our development set. Notice that gold coreference chains are assumed to be available at test time and were used in previous work (Silberer and Frank, 2012) as well as the system we compare to (Feizabadi and Padó, 2015).

The source code of all experiments, including random seeds and replication instructions, is publicly available at: `bitbucket.org/cltl/isrl-sp`.

4.4. Results on SemEval-2010

Table 5.5 shows the performance of selectional preference models with regard to resolving the explicit roles of OntoNotes. Selec-

tional preference alone (without the help of syntactic structures) can find the correct filler in more than 47% of the cases. We observe a small but statistically significant ($p < 0.05$) improvement on the validation set by adding multiway selectional preference.

The results in Table 5.6 show that our models significantly increase the F_1 score above the baseline on SemEval-2010 dataset. Both neural models show significant improvement in precision and an even bigger improvement in recall. This can be attributed to their ability to generalize to unseen predicate-role combinations and abstract away from observed ones in their hidden layer. Contrary to our expectation, MULTIWAY is inferior to ONEWAY ($p < 0.05$).

ONEWAY and MULTIWAY do not outperform F&P which is simpler in terms of machine learning architecture, but is trained on in-domain, iSRL data, and uses more features. To bridge the gap between the models, we also integrate Feizabadi and Pado's syntactic and semantic features into our neural models but they do not lead to improved performance.

Table 5.5 and Table 5.6 reveal an increase in random fluctuation when moving from OntoNotes to SemEval-2010, probably because of a difference of some orders of magnitude in size. Moreover, SYNSEM+ONEWAY gets an F_1-score of 16.54% for one of its runs (lower than the mean of all other neural models) and 29.43% for another (higher than all means). These observations stress the importance of running experiments multiple times when random factors (such as parameter initialization and the order of training examples) are involved. Based on a single run, a model might be heavily over- or underrated.

4.5. Results on ON5V

In Table 5.7, we report the results of models on ON5V (Moor et al., 2013b). Again, the Naïve Bayes model using both local and discourse information proposed by Feizabadi and Padó (2015)

	P	R	F_1
Baseline	26.85	21.80	24.07
F&P	35.04	30.83	32.80
ONEWAY	28.80	28.67	28.74 $\pm$ 1.63
MULTIWAY	27.02	26.82	26.92 $\pm$ 1.42
SYNSEM	16.63	16.54	16.58 $\pm$ 1.51
ONEWAY+SYNSEM	24.05	23.91	23.98 $\pm$ 5.05
MULTIWAY+SYNSEM	17.29	17.29	17.29 $\pm$ 0.00

Table 5.6: Results on SemEval-2010. Results of neural models are averaged over 15 runs. F_1 scores are reported with mean and standard deviation when possible.

	P	R	F_1
Baseline	13.00	13.00	13.00
F&P	16.72	15.19	15.90
ONEWAY	10.64	10.64	10.64 $\pm$ 1.44
MULTIWAY	9.14	9.14	9.14 $\pm$ 1.46
SYNSEM	5.92	5.92	5.92 $\pm$ 2.10
ONEWAY+SYNSEM	10.37	10.37	10.37 $\pm$ 5.50
MULTIWAY+SYNSEM	1.24	1.24	1.24 $\pm$ 0.00

Table 5.7: Results on ON5V

clearly provides the best performance, whereas neural models do not show improvement over the baseline ($p < 0.05$).

The disappointing performance points to its inherent limitation: it expects one prototypical filler per $\langle$predicate, role$\rangle$ pair. As shown in Table 5.4, this assumption breaks in ON5V, resulting in a lower mean and higher variance.

We expected that MULTIWAY would alleviate this problem by varying the predicted vector based on surrounding roles. While it achieves that for explicit SRL on OntoNotes (Table 5.5), the result does not carry over to ON5V.

Local syntactic and semantic information do not improve results for SemEval. This applies even more strongly to ON5V. SYNSEM leads to very low results when standing alone and does not improve performance when combined with ONEWAY or

MULTIWAY (the difference between ONEWAY and its combination with SYNSEM is not statistically significant). In comparison with F&P, this result emphasizes the importance of discourse information in the task.

5. Conclusions

In this paper, we investigated the use of more expressive machine learning models for implicit Semantic Role Labeling. We proposed novel neural models that use selectional preference and applied them to iSRL. Our empirical results show that neural models are only better than a lookup table of prototypical vectors in a natural setting such as SemEval-2010 while underperforming for highly frequent and ambiguous words in ON5V. Furthermore, the added expressive power does not help neural models to overcome a simpler model trained on in-domain data and equipped with discourse features (though it should be noted that we tested only a small family of simple architectures, cf. Do et al. (2017)). Multi-way preference is found to be helpful in the case of (explicit) semantic role labeling but not for iSRL. Although the results are mostly negative, our research provides hard-earned insights into this challenging task which we believe will be useful for researchers.

We release all of our models and the implementation of Schenk and Chiarcos (2016) and Feizabadi and Padó (2015)'s models as open-source software. We also report the fluctuation of results which stresses the importance of measuring a model multiple times when stochastic factors are involved.

Overall, this paper provides a solid basis for further research. Our observations on fluctuation and significance suggest more evaluation data may be needed to identify the true impact of specific models and features.

Acknowledgements

The work presented in this paper was funded by the Netherlands Organization for Scientific Research (NWO) via the Spinoza grant, awarded to Piek Vossen and via VENI grant 275-89-029 awarded to Antske Fokkens. Experiments were carried out on the Dutch national e-infrastructure with the support of SURF Cooperative.

6. Conclusions

> "I'll try if I know all the things I used to know. Let me see: four times five is twelve, and four times six is thirteen, and four times seven is–oh dear! I shall never get to twenty at that rate! However, the Multiplication Table doesn't signify: let's try Geography. London is the capital of Paris, and Paris is the capital of Rome, and Rome–no, that's all wrong, I'm certain!"
>
> *Alice in Wonderland (Lewis Carroll)*

This thesis has taken us to various places in NLP-land. It should be clear by now that error propagation occurs in every task and every model architecture. In fact, wherever errors occur there is a chance of error propagation and in NLP we have never run out of errors. From this perspective, I have set up for myself an impossible task. Nevertheless, it is my hope that this thesis has offered useful insights into this important phenomenon. To wrap up, this last chapter will take you through a quick tour of what we have learned so far (Section 6.1) including summaries of our five papers cast as case studies of and solutions to error propagation (Section 6.1.2 and 6.1.4). From this vantage point aiming at a future where NLP reaches human-level performance, I will offer my perspective on the remaining challenges in Section 6.2 and promising research directions in Section 6.3.

6.1 What We Have Learned

Error propagation can be informally defined as when an error begets another. A review of the literature presented in this thesis shows that the problem is known to occur in various tasks from low- to high-level: chunking (Song et al., 2012), syntactic parsing (e.g. Kong and Smith 2014), semantic role labeling (e.g. He et al. 2017), coreference resolution (e.g. Peng et al. 2015), sentiment analysis (Gómez-Rodríguez et al., 2017), opinion mining (Yang and Cardie, 2013), timeline extraction (Caselli et al., 2015), and machine translation (e.g. Han et al. 2013). The degradation of performance caused by error propagation is significant, sometimes as big as reported incremental improvements in a task (Dridan and Oepen, 2013). The prospect of error propagation might even preclude the application of a task if the state-of-the-art performance is deemed not high enough. To better understand this vexing phenomenon, I have examined it in different angles and studied possible solutions.

6.1.1 Types of Error Propagation

Error propagation can be classified along at least three dimensions: within- vs across-task, hard (deterministic) vs soft (probabilistic), and discrete vs continuous.

Error propagation occurs consistently in traditional pipelines when a module passes its errors to the next (e.g. Dridan and Oepen 2013). It can also occur when a single-task system performs consecutive actions with some prominent examples being transition-based parsing (McDonald and Nivre, 2007; Kummerfeld et al., 2012) and coreference resolution (Clark, 2015). In the first example, an incorrect parsing decision at the beginning of a sentence might render correct syntactic structures in the later part of the sentence inaccessible (among other consequences). In the second example, an incorrect cluster formed by multiple

entities ensures that later mentions of such entities are clustered incorrectly in one way or another.

The two examples above also exemplify hard error propagation, i.e. when an error deterministically causes another error. Soft (or probabilistic) error propagation can also occur in each of the two cases. A strength of transition-based systems is rich non-local features based on partially-build structures. However, this is also their Achilles' heel when it comes to error propagation since errors can create bad features that increase the chance of further errors. Similarly, in coreference resolution, a bad cluster might encourage further incorrect clustering decisions because of misleading features.

As the output of NLP systems are mainly discrete structures, it is natural that the majority of the literature so far has been concerned with discrete errors. However, with the rise of deep learning, more and more NLP systems make use of word embeddings and neural networks. More importantly, in those systems, embeddings and internal distributed representations precede and directly cause discrete actions. Talking about discrete errors without their continuous counterpart is like talking about positions in a swimming competition without mentioning swimming techniques.

It is hard to identify errors in real-value vectors because we do not possess a gold-standard set of vectors to compare them against. Instead, this thesis has taken a property-based approach: we can define expected properties that a system might satisfy to a certain degree. The stability of the internal representations of deep neural networks was briefly examined using adversarial examples in computer vision (object recognition). I have shown that a small perturbation of the input is propagated and exacerbated along the layers of a deep neural network until the final prediction is changed. Adversarial examples are found in virtually any application of deep neural networks, including NLP (e.g. Ribeiro

et al. 2018). Moreover, aberration might arise naturally in the first layers of the network because word embeddings have their own kinds of errors.

6.1.2 Case Studies: Errors in Word Embeddings

Two papers presented in this thesis have to do with the study of errors in word embeddings. The first paper, **Taxonomy Beats Corpus in Similarity Identification, but Does It Matter?** shows that word vector models often perform worse than taxonomy-based models on similarity judgements. Building on an evaluation method proposed in the paper, in this thesis I went beyond summary measurements at a model level to identify problematic vectors in Google's word2vec (Mikolov et al., 2013a): those that seem to encode an uncommon sense, have incoherent neighbors, or are assigned into the wrong cluster. An estimated 19% of words in the dataset I studied have one of those issues, indicating that word embeddings models might contain a substantial amount of errors.

The second paper, **A Deep Dive into Word Sense Disambiguation with LSTM**, attempts to measure the extent to which dynamic embeddings models can capture word senses. Word sense disambiguation (WSD) is a standard task in NLP but in this thesis, I treat it as a probing task (Conneau et al., 2018) for contextualized embeddings models. If a model is insensitive to context, the highest score it can get in WSD is equal to the score of the most-frequent-sense baseline (MFS) whereas a perfect model would get 100%. Our experiments show that the studied LSTM models perform very close to MFS (between 1% and 8% relative improvement) and peak at $F_1 = 72\%$. Moreover, we show that the amount of data or parameters needed for an increase of 1% of F_1 is exponentially large.

6.1.3 Existing Remedies

To systematize approaches to reducing error propagation, I proposed a taxonomy of five strategies which can be further grouped into three main ideas (mitigation, adaptation, and correction):

S1: By reducing the amount of errors at the first steps of a sequence of decisions, we can prevent some error propagation from happening. Approaches in this strategy include work on static and dynamic embeddings as well as improvements on other foundational tasks such as sentence boundary detection and part-of-speech tagging.

S2: To reduce the chance of error propagation, we can also reorder decisions such that easier ones are executed first. This is the idea behind easy-first approaches (e.g. Goldberg and Elhadad 2010). When a joint-processing approach is applied to a transition-based system, decisions are also reordered. For example, in Constant and Nivre (2016), syntactic parsing and multi-word expression recognition decisions, which are usually executed by separate modules, become interleaved.

S3: The length of the chain of decisions directly influences the chance of error propagation. By making the chain shorter, we can reduce or, if all decisions are independent, eliminate error propagation. Approaches that could achieve this feat include graph-based joint processing (e.g. Denis and Baldridge 2009) and global inference (e.g. Andor et al. 2016). End-to-end learning, either single-task (e.g. Lee et al. 2017) or multi-task (e.g. Collobert and Weston 2008), is often used to remove preprocessing steps that might lead to error propagation. In the context of non-projective transition-based dependency parsing, Nivre et al. (2009) report improvement by training a model to make less transition steps.

S4: Instead of reducing the amount of errors that precede a de-

cision, adaptation is the idea that decisions can be made more robust to such errors. A very simple approach to improve robustness is to train a system on predicted, as opposed to gold standard, annotations (e.g. Che et al. 2012). This minimizes the difference between training and testing therefore cuts down propagated errors and increases performance. Another method is to accept multiple outputs from upstream tasks (e.g. Henestroza Anguiano and Candito 2012). Finkel et al. (2006) take this approach to extremes by proposing that whole pipelines can be modeled as Bayesian networks.

S5: By comparison to humans, some errors will always happen and error propagation will not be solved completely until we can detect and correct errors. Relatively little work has been done in this area, focusing on syntax (e.g. Delecraz et al. 2017). Reranking, which aims to reduce errors statistically without detecting errors explicitly, is studied for syntactic and semantic parsing (e.g. Charniak and Johnson 2005; Ge and Mooney 2006).

6.1.4 Proposed Solutions

There are at least two reasons to believe errors are inevitable and, therefore, mitigation cannot be the full solution. First, humans consistently make mistakes during language comprehension (Rayner and Frazier, 1982). Second, an inherent tension exists in any system that builds its output incrementally: the first decisions are the most important to get right yet are based on the smallest amount of information.

Together with my colleagues, I have explored adaptation and (the preconditions of) error correction. In the paper **Tackling Error Propagation through Reinforcement Learning: A Case of Greedy Dependency Parsing**, we show that reinforcement learning can improve the robustness of a syntactic parser against

error propagation (Strategy S4) and, at the same time, increase the overall performance. Reinforcement learning incorporates a system's own actions and, therefore, errors into the learning process. This exposure means that the system is more robust when errors inevitably arise at run time.

To detect and correct errors (Strategy S5), I have argued that we need to integrate information from multiple views of the input and check for inconsistency, analogous to the way error correcting codes work in telecommunication (Calderbank, 1998). In the paper **An Input Ablation Analysis of Coreference Resolution**, we show that many high-performing systems of coreference resolution base their decisions on a single source: the very expressions that they are clustering. Replacing the surrounding tokens with blanks has almost no effect on their performance. In **Neural Models of Selectional Preferences for Implicit Semantic Role Labeling**, we developed a simple model to capture the interdependence of semantic roles as a first step towards a model of information integration. The model improves upon the baseline when tested on predicting explicit role fillers (the same task it was trained on). However, when applying the model to implicit semantic role labeling, we obtained largely negative results.

6.2 Challenges

An implicit goal of our research into error propagation is to improve the performance of NLP systems. Otherwise, this thesis would not have been necessary since global inference systems already achieved zero error propagation by optimizing and predicting their whole output in one go. Optimizing such involved structures, however, puts a strain on computation. As a result, global inference systems are limited to simple features which in turn place a cap to achievable performance. At the other extreme, transition-based systems enable rich features at the risk of severe

error propagation.

Deep learning systems are successful partly because they avoid this trade-off to some extent. Distributed representations allow for rich features without committing to any concrete interpretation of the input. This allows deep neural networks to perform some of the functions of NLP pipelines (Tenney et al., 2019) within their layers while reducing error propagation. Nevertheless error propagation persists in its continuous form within neural layers and in discrete form when the system perform dependent sequential actions. From this point of view, two big challenges facing researchers are: how to make neural models more robust to continuous error propagation and how to reinforce the sequential discrete decisions that cannot be avoided.

Revolving around the first challenge, adversarial examples have recently become a vibrant research area in NLP (Zhang et al., 2020). It was demonstrated in various settings that we can change a small part of a text (a few characters or a few words) in a way that preserves meaning while changing the prediction of a neural model. As a concrete example, Liang et al. (2018) reported fooling the deep convolutional text classification model of Zhang et al. (2015) to change its prediction by introducing a common misspelling in a four-sentence passage. As argued earlier in this thesis (see Section 2.1.3), such hypersensitivity is likely caused by continuous error propagated and exacerbated through layers until it is large enough to flip the prediction. Making neural networks more robust to perturbation is a hard problem and continues to attract research interest in not only NLP but also machine learning research.

Regarding the second challenge, as language is inherently linear and coherent, it is hard to imagine a language processing system that can achieve human-level performance without sequential, dependent decisions. At a certain level, language comprehension is like scientific research: we need hypotheses to base

our reasoning on while being aware that they might be wrong and ready to correct them upon new evidence. Research around garden-path sentences shows that humans can do this (Rayner and Frazier, 1982). On what basis do we decide something is incorrect? In this thesis, I have advanced the argument that we need world knowledge for this type of decisions (see Section 2.4). Representing and extracting world knowledge is a long-standing challenge in artificial intelligence research.

Robust and knowledge-rich language processing systems are challenging goals that might take many more decades to realize. However, as the applications of NLP reach more and more aspects of everyday life, including critical areas such healthcare, the need to address them is more urgent than ever. A precautionary case of how NLP without knowledge can go wrong is IBM Watson: after winning *Jeopardy!* against human champions, the language processing wizard is deployed in various projects in diagnostics, clinical-decision support, and personalized medicine. After many years and billions of dollar spent on development, the results were overwhelmingly negative (Strickland, 2019). In one case, Watson was unable to update its beliefs based on the latest paper because it relied on the statistics on thousands of others; in another case, it recommended surveillance to some patients with metastatic cancer (Strickland, 2019). Not every case of NLP absurdity is obscure or limited to specialized settings. Facebook's machine translation once translated "good morning" in Arabic into "attack them" in Hebrew, resulting in a brief arrest of its user.[1] To avoid such clashes between human expectation and system capability, I believe that research into world knowledge should precede or at least parallel the real-world deployment of NLP systems.

[1] `https://www.bbc.com/news/world-middle-east-41714152`, accessed on 31 August 2020.

6.3 Future Research

How to capture world knowledge is a hard question but one clue can be extracted from its definition: world knowledge is what we assume the listener to know and therefore left unspoken. Recently, Bender and Koller (2020) have argued that language models that are trained on form only cannot learn meaning. Throughout the course of my PhD study, my experience in extracting knowledge from text falls along the same line: distributional lexical models, both static and dynamic, were found to contain capricious distortion and my efforts in extracting typical semantic relations and event chains resulted in vague or meaningless patterns. For these reasons, I believe that to find world knowledge, it is necessary to escape from the boundaries of NLP.

Visual concepts are different from all other types of concepts NLP has access to so far. Embeddings are abundant in modern NLP but are structurally poor: they carry little information beyond similarity and relatedness and support few operations beyond analogical reasoning (in the sense that a model makes predictions based on superficial similarity to observations in the training data). Structured knowledge of the types captured by WordNet, CYC and modern ontologies can be arbitrarily rich but, because they are built by experts, the supply has always been limited. The largest ontologies have been harvested from textual sources and contain mostly specific entities rather than semantic knowledge (Färber et al., 2015).

If properly represented and harvested, visual concepts can be both rich and abundant. By analyzing first images and then videos, we can learn not only visual characteristics but also intuitive physics, common physical mechanisms and typical social interactions. Visual concepts also serve as the basis of conceptual metaphors that help us understand abstract ones (Lakoff and Johnson, 1980). Without them, much of linguistic usage would

become arbitrary and require memorization.

None of the above assumes that extracting visual concepts is easy or applying them would solve all issues in NLP. However, in my opinion, those are necessary parts of the final solution as opposed to, for example, applying existing architectures on ever larger amounts of data. To paraphrase Bender and Koller (2020), I believe capturing world knowledge via visual concepts is the right hill to climb.

Bibliography

Martin Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Man, Rajat Monga, Sherry Moore, Derek Murray, Jon Shlens, Benoit Steiner, Ilya Sutskever, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Oriol Vinyals, Pete Warden, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. 2015. TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems .

Eneko Agirre, Enrique Alfonseca, Keith Hall, Jana Kravalova, Marius Paşca, and Aitor Soroa. 2009. A study on similarity and relatedness using distributional and WordNet-based approaches. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics, Boulder, Colorado, pages 19–27.

Eneko Agirre, Oier López de Lacalle, and Aitor Soroa. 2014. Random walks for knowledge-based word sense disambiguation. *Computational Linguistics* 40(1):57–84. https://doi.org/10.1162/COLI_a_00164.

Naveed Akhtar and Ajmal Mian. 2018. Threat of Adversarial Attacks on Deep Learning in Computer Vision: A Survey. *IEEE Access* 6:14410–14430. https://doi.org/10.1109/ACCESS.2018.2807385.

Chris Alberti, Daniel Andor, Ivan Bogatyy, Michael Collins, Dan Gillick, Lingpeng Kong, Terry Koo, Ji Ma, Mark Omernick, Slav Petrov, and Others. 2017. SyntaxNet models for the CoNLL 2017 shared task .

Chris Alberti, David Weiss, Greg Coppola, and Slav Petrov. 2015. Improved transition-based parsing and tagging with neural networks. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Lisbon, Portugal, pages 1354–1359. https://doi.org/10.18653/v1/D15-1159.

James Allen. 1995. *Natural Language Understanding (2nd Ed.)*. Benjamin-Cummings Publishing Co., Inc., USA.

Bharat Ram Ambati, Rahul Agarwal, Mridul Gupta, Samar Husain, and Dipti Misra Sharma. 2011. Error detection for treebank validation. In *Proceedings of the 9th Workshop on Asian Language Resources*. Asian Federation of Natural Language Processing, Chiang Mai, Thailand, pages 23–30.

Daniel Andor, Chris Alberti, David Weiss, Aliaksei Severyn, Alessandro Presta, Kuzman Ganchev, Slav Petrov, and Michael Collins. 2016. Globally normalized transition-based neural networks. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, Berlin, Germany, pages 2442–2452. https://doi.org/10.18653/v1/P16-1231.

Giuseppe Attardi and Massimiliano Ciaramita. 2007. Tree revision learning for dependency parsing. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Proceedings of the Main Conference*. Association for Computational Linguistics, Rochester, New York, pages 388–395.

Ricardo Baeza-Yates. 2004. Challenges in the interaction of information retrieval and natural language processing. In *Proceedings of 5th international conference on Computational Linguistics and Intelligent Text Processing (CICLing)*. pages 445–456.

Amit Bagga and Breck Baldwin. 1998. Algorithms for Scoring Coreference Chains Shortcomings of the MUC-6 Algorithm. In *Proceedings of the 1st International Conference on Language Resources and Evaluation*. pages 563–566.

Collin F. Baker, Charles J. Fillmore, and John B. Lowe. 1998. The Berkeley FrameNet project. In *COLING 1998 Volume 1: The 17th International Conference on Computational Linguistics*.

Rajendra Banjade, Nabin Maharjan, NobalB. Niraula, Vasile Rus, and Dipesh Gautam. 2015. Lemon and Tea Are Not Similar: Measuring Word-to-Word Similarity by Combining Different Methods. *Computational Linguistics and Intelligent Text Processing* 9041:335–346. https://doi.org/10.1007/978-3-319-18111-0_25.

Mohit Bansal, Kevin Gimpel, and Karen Livescu. 2014. Tailoring continuous word representations for dependency parsing. In *Proceedings of the 52nd*

Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers). Association for Computational Linguistics, Baltimore, Maryland, pages 809–815. https://doi.org/10.3115/v1/P14-2131.

Marco Baroni, Georgiana Dinu, and Germán Kruszewski. 2014. Don't count, predict! a systematic comparison of context-counting vs. context-predicting semantic vectors. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, Baltimore, Maryland, pages 238–247. https://doi.org/10.3115/v1/P14-1023.

Hannah Béchara, Y Ma, and J Genabith. 2011. Statistical Post-Editing for a Statistical MT System.

Yonatan Belinkov and James Glass. 2019. Analysis methods in neural language processing: A survey. *Transactions of the Association for Computational Linguistics* 7:49–72. https://doi.org/10.1162/tacl_a_00254.

Emily M. Bender and Alexander Koller. 2020. Climbing towards NLU: On meaning, form, and understanding in the age of data. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, Online, pages 5185–5198. https://doi.org/10.18653/v1/2020.acl-main.463.

Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. 2015. Scheduled sampling for sequence prediction with recurrent neural networks. In Corinna Cortes, Neil D. Lawrence, Daniel D. Lee, Masashi Sugiyama, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*. pages 1171–1179.

Eric Bengtson and Dan Roth. 2008. Understanding the value of features for coreference resolution. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Honolulu, Hawaii, pages 294–303.

Jonathan Berant and Percy Liang. 2015. Imitation learning of agenda-based semantic parsers. *Transactions of the Association for Computational Linguistics* 3:545–558. https://doi.org/10.1162/tacl_a_00157.

Or Biran, Samuel Brody, and Noémie Elhadad. 2011. Putting it simply: a context-aware approach to lexical simplification. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics:*

Human Language Technologies: short papers-Volume 2. Association for Computational Linguistics, pages 496–501.

Anders Björkelund and Joakim Nivre. 2015. Non-deterministic oracles for unrestricted non-projective transition-based dependency parsing. In *Proceedings of the 14th International Conference on Parsing Technologies*. Association for Computational Linguistics, Bilbao, Spain, pages 76–86. https://doi.org/10.18653/v1/W15-2210.

BNC. 2007. The British National Corpus, version 3 (BNC XML Edition). Technical report, Oxford University Computing Services on behalf of the BNC Consortium. http://www.natcorp.ox.ac.uk/.

Bernd Bohnet and Joakim Nivre. 2012. A transition-based system for joint part-of-speech tagging and labeled non-projective dependency parsing. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*. Association for Computational Linguistics, Jeju Island, Korea, pages 1455–1465.

Marisa Ferrara Boston, John Hale, and Reinhold Kliegl. 2008. Parsing costs as predictors of reading difficulty: An evaluation using the Potsdam Sentence Corpus. *Parsing costs as predictors of reading difficulty: An evaluation using the Potsdam Sentence Corpus* 2(1):1–12. https://doi.org/10.16910/jemr.2.1.1.

Chloé Braud and Pascal Denis. 2015. Comparing word representations for implicit discourse relation classification. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Lisbon, Portugal, pages 2201–2211. https://doi.org/10.18653/v1/D15-1262.

Elia Bruni, Gemma Boleda, Marco Baroni, and Nam-Khanh Tran. 2012. Distributional semantics in technicolor. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*. Association for Computational Linguistics, pages 136–145.

Razvan Bunescu. 2008. Learning with probabilistic features for improved pipeline models. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Honolulu, Hawaii, pages 670–679.

A. Robert Calderbank. 1998. The art of signaling: Fifty years of coding theory. *IEEE Transactions on Information Theory* 44(6):2561–2595.

Jose Camacho-Collados and Mohammad Taher Pilehvar. 2018. From word to sense embeddings: A survey on vector representations of meaning. *Journal of Artificial Intelligence Research* 63:743–788. https://doi.org/10.1613/jair.1.11259.

José Camacho-Collados, Mohammad Taher Pilehvar, and Roberto Navigli. 2016. Nasari: Integrating explicit knowledge and corpus statistics for a multilingual representation of concepts and entities. *Artificial Intelligence* 240:36–64.

Erik Cambria and Bebo White. 2014. Jumping NLP curves: A review of natural language processing research. *IEEE Computational Intelligence Magazine* 9(2):48–57. https://doi.org/10.1109/MCI.2014.2307227.

Nicholas Carlini and David Wagner. 2017. Towards Evaluating the Robustness of Neural Networks. *Proceedings - IEEE Symposium on Security and Privacy* pages 39–57. https://doi.org/10.1109/SP.2017.49.

Xavier Carreras and Lluís Màrquez. 2004. Introduction to the CoNLL-2004 shared task: Semantic role labeling. In *Proceedings of the Eighth Conference on Computational Natural Language Learning (CoNLL-2004) at HLT-NAACL 2004*. Association for Computational Linguistics, Boston, Massachusetts, USA, pages 89–97.

Tommaso Caselli, Piek Vossen, Marieke van Erp, Antske Fokkens, Filip Ilievski, Rubén Izquierdo, Minh Le, Roser Morante, and Marten Postma. 2015. When it's all piling up: investigating error propagation in an NLP pipeline. In *Proceedings of the Workshop on NLP Applications: Completing the Puzzle, WNACP 2015, co-located with the 20th International Conference on Applications of Natural Language to Information Systems (NLDB 2015)*. Passau, Germany.

Kai-Wei Chang, Akshay Krishnamurthy, Alekh Agarwal, Hal Daumé III, and John Langford. 2015. Learning to search better than your teacher. In Francis R. Bach and David M. Blei, editors, *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*. JMLR.org, volume 37 of *JMLR Workshop and Conference Proceedings*, pages 2058–2066.

Eugene Charniak, Glenn Carroll, John Adcock, Anthony Cassandra, Yoshihiko Gotoh, Jeremy Katz, Michael Littman, and John McCann. 1996. Taggers for parsers. *Artificial Intelligence* 85(1–2):45–57. https://doi.org/10.1016/0004-3702(95)00108-5.

Eugene Charniak and Mark Johnson. 2005. Coarse-to-fine n-best parsing and MaxEnt discriminative reranking. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*. Association for Computational Linguistics, Ann Arbor, Michigan, pages 173–180. https://doi.org/10.3115/1219840.1219862.

Wanxiang Che and Ting Liu. 2010. Jointly modeling WSD and SRL with Markov Logic. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*. Coling 2010 Organizing Committee, Beijing, China, pages 161–169.

Wanxiang Che, Valentin Spitkovsky, and Ting Liu. 2012. A comparison of Chinese parsers for Stanford dependencies. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. Association for Computational Linguistics, Jeju Island, Korea, pages 11–16.

Danqi Chen and Christopher Manning. 2014. A fast and accurate dependency parser using neural networks. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, Doha, Qatar, pages 740–750. https://doi.org/10.3115/v1/D14-1082.

Desai Chen, Nathan Schneider, Dipanjan Das, and Noah A. Smith. 2010. SEMAFOR: Frame argument resolution with log-linear models. In *Proceedings of the 5th International Workshop on Semantic Evaluation*. Association for Computational Linguistics, Uppsala, Sweden, pages 264–267.

Jinxiu Chen, Donghong Ji, Chew Lim Tan, and Zhengyu Niu. 2006. Relation extraction using label propagation based semi-supervised learning. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, Sydney, Australia, pages 129–136. https://doi.org/10.3115/1220175.1220192.

Colin Cherry and Hongyu Guo. 2015. The unreasonable effectiveness of word representations for Twitter named entity recognition. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, Denver, Colorado, pages 735–745. https://doi.org/10.3115/v1/N15-1075.

Jennifer Chu-Carroll, Krzysztof Czuba, John Prager, and Abraham Ittycheriah. 2003. In question answering, two heads are better than one. In *Proceedings of the 2003 Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*. pages 24–31.

Moustapha Cissé, Piotr Bojanowski, Edouard Grave, Yann N. Dauphin, and Nicolas Usunier. 2017. Parseval networks: Improving robustness to adversarial examples. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*. PMLR, volume 70 of *Proceedings of Machine Learning Research*, pages 854–863.

Kevin Clark. 2015. Neural Coreference Resolution. Technical report.

Kevin Clark and Christopher D. Manning. 2016a. Deep reinforcement learning for mention-ranking coreference models. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Austin, Texas, pages 2256–2262. https://doi.org/10.18653/v1/D16-1245.

Kevin Clark and Christopher D. Manning. 2016b. Improving coreference resolution by learning entity-level distributed representations. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, Berlin, Germany, pages 643–653. https://doi.org/10.18653/v1/P16-1061.

William W. Cohen and Vitor R. Carvalho. 2005. Stacked sequential learning. *IJCAI International Joint Conference on Artificial Intelligence* pages 671–676.

Ronan Collobert and Jason Weston. 2008. A unified architecture for natural language processing: deep neural networks with multitask learning. In William W. Cohen, Andrew McCallum, and Sam T. Roweis, editors, *Machine Learning, Proceedings of the Twenty-Fifth International Conference (ICML 2008), Helsinki, Finland, June 5-9, 2008*. ACM, volume 307 of *ACM International Conference Proceeding Series*, pages 160–167. https://doi.org/10.1145/1390156.1390177.

Ronan Collobert, Jason Weston, L{\'e}on Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *Journal of Machine Learning Research* 12:2493–2537.

Alexis Conneau, German Kruszewski, Guillaume Lample, Loïc Barrault, and Marco Baroni. 2018. What you can cram into a single $&!#* vector: Probing

sentence embeddings for linguistic properties. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, Melbourne, Australia, pages 2126–2136. https://doi.org/10.18653/v1/P18-1198.

Matthieu Constant and Joakim Nivre. 2016. A transition-based system for joint lexical and syntactic analysis. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, Berlin, Germany, pages 161–171. https://doi.org/10.18653/v1/P16-1016.

Aron Culotta, Michael Wick, and Andrew McCallum. 2007. First-order probabilistic models for coreference resolution. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Proceedings of the Main Conference*. Association for Computational Linguistics, Rochester, New York, pages 81–88.

Hamish Cunningham, Diana Maynard, Kalina Bontcheva, and Valentin Tablan. 2002. GATE: A Framework and Graphical Development Environment for Robust NLP Tools and Applications. In *Proceedings of the 40th Anniversary Meeting of the Association for Computational Linguistics (ACL'02)*.

Hal Daumé III, John Langford, and Daniel Marcu. 2009. Search-based Structured Prediction. *Machine Learning* 75(3):297–325. https://doi.org/10.1007/s10994-009-5106-x.

Sebastien Delecraz, Alexis Nasr, Frederic Bechet, and Benoit Favre. 2017. Correcting prepositional phrase attachments using multimodal corpora. In *Proceedings of the 15th International Conference on Parsing Technologies*. Association for Computational Linguistics, Pisa, Italy, pages 72–77.

Felice Dell'Orletta. 2009. Ensemble system for Part-of-Speech tagging. *Proceedings of EVALITA 2009* .

Pascal Denis and Jason Baldridge. 2008. Specialized Models and Reranking for Coreference Resolution. *Proceedings of the Conference on Empirical Methods in Natural Language Processing* pages 660–669. https://doi.org/10.3115/1613715.1613797.

Pascal Denis and Jason Baldridge. 2009. Global joint models for coreference resolution and named entity classification. *Procesamiento del Lenguaje Natural* 42(1):87–96.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Association for Computational Linguistics, Minneapolis, Minnesota, pages 4171–4186. https://doi.org/10.18653/v1/N19-1423.

Markus Dickinson. 2010. Detecting errors in automatically-parsed dependency relations. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, Uppsala, Sweden, pages 729–738.

Markus Dickinson and Amber Smith. 2011. Detecting dependency parse errors with minimal resources. In *Proceedings of the 12th International Conference on Parsing Technologies*. Association for Computational Linguistics, Dublin, Ireland, pages 241–252.

Quynh Ngoc Thi Do, Steven Bethard, and Marie-Francine Moens. 2017. Improving implicit semantic role labeling by predicting semantic frame arguments. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. Asian Federation of Natural Language Processing, Taipei, Taiwan, pages 90–99.

Xin Dong, Shangyu Chen, and Sinno Jialin Pan. 2017. Learning to prune deep neural networks via layer-wise optimal brain surgeon. In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*. pages 4857–4867.

Rebecca Dridan and Stephan Oepen. 2013. Document parsing: Towards realistic syntactic analysis. In *Proceedings of the 13th International Conference on Parsing Technologies (IWPT 2013)*. Assocation for Computational Linguistics, Nara, Japan, pages 127–133.

Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. 2019. DROP: A reading comprehension benchmark requiring discrete reasoning over paragraphs. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long*

and Short Papers). Association for Computational Linguistics, Minneapolis, Minnesota, pages 2368–2378. https://doi.org/10.18653/v1/N19-1246.

John C. Duchi, Elad Hazan, and Yoram Singer. 2010. Adaptive subgradient methods for online learning and stochastic optimization. In Adam Tauman Kalai and Mehryar Mohri, editors, *COLT 2010 - The 23rd Conference on Learning Theory, Haifa, Israel, June 27-29, 2010*. Omnipress, pages 257–269.

Greg Durrett and Dan Klein. 2013. Easy victories and uphill battles in coreference resolution. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Seattle, Washington, USA, pages 1971–1982.

Chris Dyer, Miguel Ballesteros, Wang Ling, Austin Matthews, and Noah A. Smith. 2015. Transition-based dependency parsing with stack long short-term memory. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. Association for Computational Linguistics, Beijing, China, pages 334–343. https://doi.org/10.3115/v1/P15-1033.

Andreas Eisele and Yu Chen. 2010. MultiUN: A multilingual corpus from united nation documents. In *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10)*. European Language Resources Association (ELRA), Valletta, Malta.

Jason M. Eisner. 1996. Three new probabilistic models for dependency parsing. In *Proceedings of the 16th International Conference on Computational Linguistics (COLING-96)*. https://doi.org/10.3115/992628.992688.

Ali Emami, Paul Trichelair, Adam Trischler, Kaheer Suleman, Hannes Schulz, and Jackie Chi Kit Cheung. 2019. The KnowRef coreference corpus: Removing gender and number cues for difficult pronominal anaphora resolution. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, Florence, Italy, pages 3952–3961. https://doi.org/10.18653/v1/P19-1386.

Katrin Erk. 2007. A simple, similarity-based model for selectional preferences. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*. Association for Computational Linguistics, Prague, Czech Republic, pages 216–223.

Kevin Eykholt, Ivan Evtimov, Earlence Fernandes, Bo Li, Amir Rahmati, Chaowei Xiao, Atul Prakash, Tadayoshi Kohno, and Dawn Song. 2018. Robust physical-world attacks on deep learning visual classification. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*. IEEE Computer Society, pages 1625–1634. https://doi.org/10.1109/CVPR.2018.00175.

Michael Färber, Basil Ell, Carsten Menne, and Achim Rettinger. 2015. A Comparative Survey of DBpedia, Freebase, OpenCyc, Wikidata, and YAGO. *Semantic Web* pages 1–5. http://arxiv.org/abs/1901.11373.

Benoit Favre, Bernd Bohnet, and Dilek Hakkani-Tur. 2010. Evaluation of semantic role labeling and dependency parsing of automatic speech recognition output. *2010 IEEE International Conference on Acoustics, Speech and Signal Processing* pages 5342–5345. https://doi.org/10.1109/ICASSP.2010.5494946.

Hongliang Fei, Xu Li, Dingcheng Li, and Ping Li. 2019. End-to-end deep reinforcement learning based coreference resolution. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, Florence, Italy, pages 660–665. https://doi.org/10.18653/v1/P19-1064.

Parvin Sadat Feizabadi and Sebastian Padó. 2015. Combining seemingly incompatible corpora for implicit semantic role labeling. In *Proceedings of the Fourth Joint Conference on Lexical and Computational Semantics*. Association for Computational Linguistics, Denver, Colorado, pages 40–50. https://doi.org/10.18653/v1/S15-1005.

Christiane Fellbaum, editor. 1998. *WordNet: An Electronic Lexical Database*. MIT Press, Cambridge, MA.

Shi Feng, Eric Wallace, Alvin Grissom II, Mohit Iyyer, Pedro Rodriguez, and Jordan Boyd-Graber. 2018. Pathologies of neural models make interpretations difficult. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Brussels, Belgium, pages 3719–3728. https://doi.org/10.18653/v1/D18-1407.

Charles J. Fillmore. 1982. Frame semantics. In *Linguistics in the morning calm*. Hanshin Publishing Co., Seoul, pages 111–137.

Charles J Fillmore, Christopher R Johnson, and Miriam R L Petruck. 2003. Background to framenet. *International journal of lexicography* 16(3):235–250.

Jenny Rose Finkel. 2010. *Holistic language processing: Joint models of linguistic structure*. Ph.D. thesis.

Jenny Rose Finkel, Christopher D. Manning, and Andrew Y. Ng. 2006. Solving the problem of cascading errors: Approximate Bayesian inference for linguistic annotation pipelines. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Sydney, Australia, pages 618–626.

Lev Finkelstein, Evgeniy Gabrilovich, Yossi Matias, Ehud Rivlin, Zach Solan, Gadi Wolfman, and Eytan Ruppin. 2001. Placing search in context: The concept revisited. In *Proceedings of the 10th international conference on World Wide Web*. ACM, pages 406–414.

Antske Fokkens, Marieke Erp, Marten Postma, Ted Pedersen, Piek Vossen, and Nuno Freire. 2013a. Offspring from reproduction problems: What replication failure teaches us. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, pages 1691–1701.

Antske Fokkens, Marieke van Erp, Marten Postma, Ted Pedersen, Piek Vossen, and Nuno Freire. 2013b. Offspring from reproduction problems: What replication failure teaches us. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, Sofia, Bulgaria, pages 1691–1701.

William Foland and James H. Martin. 2016. CU-NLP at SemEval-2016 task 8: AMR parsing using LSTM-based recurrent neural networks. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*. Association for Computational Linguistics, San Diego, California, pages 1197–1201. https://doi.org/10.18653/v1/S16-1185.

Martin Forst and Ronald M. Kaplan. 2006. The importance of precise tokenizing for deep grammars. In *Proceedings of the Fifth International Conference on Language Resources and Evaluation (LREC'06)*. European Language Resources Association (ELRA), Genoa, Italy.

Jennifer Foster. 2010. "cba to check the spelling": Investigating parser performance on discussion forum posts. In *Human Language Technologies: The*

2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics. Association for Computational Linguistics, Los Angeles, California, pages 381–384.

Jennifer Foster, Özlem Çetinoğlu, Joachim Wagner, Joseph Le Roux, Joakim Nivre, Deirdre Hogan, and Josef van Genabith. 2011. From news to comment: Resources and benchmarks for parsing the language of web 2.0. In *Proceedings of 5th International Joint Conference on Natural Language Processing*. Asian Federation of Natural Language Processing, Chiang Mai, Thailand, pages 893–901.

Winthrop Nelson Francis and Henry Kucera. 1979. Brown corpus manual. *Brown University* .

Ruifang Ge and Raymond J. Mooney. 2006. Discriminative reranking for semantic parsing. In *Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions*. Association for Computational Linguistics, Sydney, Australia, pages 263–270.

Jeremy Getman, Joe Ellis, Stephanie Strassel, Zhiyi Song, and Jennifer Tracey. 2018. Laying the groundwork for knowledge base population: Nine years of linguistic resources for TAC KBP. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*. European Language Resources Association (ELRA), Miyazaki, Japan.

Abbas Ghaddar and Phillippe Langlais. 2016. Coreference in Wikipedia: Main concept resolution. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*. Association for Computational Linguistics, Berlin, Germany, pages 229–238. https://doi.org/10.18653/v1/K16-1023.

Amin Ghiasi, Ali Shafahi, and Tom Goldstein. 2020. Breaking certified defenses: Semantic adversarial examples with spoofed robustness certificates. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Daniel Gildea and Martha Palmer. 2002. The necessity of parsing for predicate argument recognition. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, Philadelphia, Pennsylvania, USA, pages 239–246. https://doi.org/10.3115/1073083.1073124.

Varun Godbole, Wei Liu, and Roberto Togneri. 2015. An Investigation of Neural Embeddings for Coreference Resolution. *Computational Linguistics*

and Intelligent Text Processing 9041:241–251. https://doi.org/10.1007/978-3-319-18111-0.

Yoav Goldberg. 2016. A Primer on Neural Network Models for Natural Language Processing. *Journal of Artificial Intelligence Research* 57:345–420.

Yoav Goldberg and Michael Elhadad. 2010. An efficient algorithm for easy-first non-directional dependency parsing. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics, Los Angeles, California, pages 742–750.

Yoav Goldberg and Joakim Nivre. 2012. A dynamic oracle for arc-eager dependency parsing. In *Proceedings of COLING 2012*. The COLING 2012 Organizing Committee, Mumbai, India, pages 959–976.

Yoav Goldberg and Joakim Nivre. 2013. Training deterministic parsers with non-deterministic oracles. *Transactions of the Association for Computational Linguistics* 1:403–414. https://doi.org/10.1162/tacl_a_00237.

Yoav Goldberg, Francesco Sartorio, and Giorgio Satta. 2014. A tabular method for dynamic oracles in transition-based parsing. *Transactions of the Association for Computational Linguistics* 2:119–130. https://doi.org/10.1162/tacl_a_00170.

Carlos Gómez-Rodríguez, Iago Alonso-Alonso, and David Vilares. 2017. How Important is Syntactic Parsing Accuracy? An Empirical Evaluation on Sentiment Analysis. *Artificial Intelligence Review* pages 1–17. https://doi.org/10.1007/s10462-017-9584-0.

Carlos Gómez-Rodríguez, Francesco Sartorio, and Giorgio Satta. 2014. A polynomial-time dynamic oracle for non-projective dependency parsing. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, Doha, Qatar, pages 917–927. https://doi.org/10.3115/v1/D14-1099.

Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. 2015. Explaining and harnessing adversarial examples. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Philip Gorinski, Josef Ruppenhofer, and Caroline Sporleder. 2013. Towards weakly supervised resolution of null instantiations. In *Proceedings of the 10th International Conference on Computational Semantics (IWCS 2013) –*

Long Papers. Association for Computational Linguistics, Potsdam, Germany, pages 119–130.

Alvin Grissom II, He He, Jordan Boyd-Graber, John Morgan, and Hal Daumé III. 2014. Don't until the final verb wait: Reinforcement learning for simultaneous machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, Doha, Qatar, pages 1342–1352. https://doi.org/10.3115/v1/D14-1140.

Alvin Grissom II, Naho Orita, and Jordan Boyd-Graber. 2016. Incremental prediction of sentence-final verbs: Humans versus machines. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*. Association for Computational Linguistics, Berlin, Germany, pages 95–104. https://doi.org/10.18653/v1/K16-1010.

Anupam Guha, Mohit Iyyer, Danny Bouman, and Jordan Boyd-Graber. 2015. Removing the training wheels: A coreference dataset that entertains humans and challenges computers. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, Denver, Colorado, pages 1108–1118. https://doi.org/10.3115/v1/N15-1117.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. 2017. On calibration of modern neural networks. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*. PMLR, volume 70 of *Proceedings of Machine Learning Research*, pages 1321–1330.

Kelvin Guu, John Miller, and Percy Liang. 2015. Traversing knowledge graphs in vector space. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Lisbon, Portugal, pages 318–327. https://doi.org/10.18653/v1/D15-1038.

Aria Haghighi, Kristina Toutanova, and Christopher Manning. 2005. A joint model for semantic role labeling. In *Proceedings of the Ninth Conference on Computational Natural Language Learning (CoNLL-2005)*. Association for Computational Linguistics, Ann Arbor, Michigan, pages 173–176.

Peter Hagoort, Lea Hald, Marcel Bastiaansen, and Karl Magnus Petersson. 2004. Integration of word meaning and world knowledge in language comprehension. *science* 304(5669):438–441.

Jan Hajič, Massimiliano Ciaramita, Richard Johansson, Daisuke Kawahara, Maria Antònia Martí, Lluís Màrquez, Adam Meyers, Joakim Nivre, Sebastian Padó, Jan Štěpánek, Pavel Straňák, Mihai Surdeanu, Nianwen Xue, and Yi Zhang. 2009. The CoNLL-2009 shared task: Syntactic and semantic dependencies in multiple languages. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL 2009): Shared Task*. Association for Computational Linguistics, Boulder, Colorado, pages 1–18.

Hannaneh Hajishirzi, Leila Zilles, Daniel S. Weld, and Luke Zettlemoyer. 2013. Joint coreference resolution and named-entity linking with multi-pass sieves. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Seattle, Washington, USA, pages 289–299.

Keith Hall and Václav Novák. 2005. Corrective modeling for non-projective dependency parsing. In *Proceedings of the Ninth International Workshop on Parsing Technology*. Association for Computational Linguistics, Vancouver, British Columbia, pages 42–52.

Keith Hall and Václav Novák. 2011. Corrective Dependency Parsing. In *Trends in Parsing Technologies*, pages 151–167. https://doi.org/10.1007/978-90-481-9352-3_9.

Dan Han, Pascual Martínez-Gómez, Yusuke Miyao, Katsuhito Sudoh, and Masaaki Nagata. 2013. Effects of parsing errors on pre-reordering performance for Chinese-to-Japanese SMT. In *Proceedings of the 27th Pacific Asia Conference on Language, Information, and Computation (PACLIC 27)*. Department of English, National Chengchi University, Taipei, Taiwan, pages 267–276.

Dan Han, Katsuhito Sudoh, Xianchao Wu, Kevin Duh, Hajime Tsukada, and Masaaki Nagata. 2012. Head finalization reordering for Chinese-to-Japanese machine translation. In *Proceedings of the Sixth Workshop on Syntax, Semantics and Structure in Statistical Translation*. Association for Computational Linguistics, Jeju, Republic of Korea, pages 57–66.

Laura Hasler, Constantin Orasan, and Karin Naumann. 2006. NPs for events: Experiments in coreference annotation. In *Proceedings of the Fifth International Conference on Language Resources and Evaluation (LREC'06)*. European Language Resources Association (ELRA), Genoa, Italy.

Luheng He, Kenton Lee, Omer Levy, and Luke Zettlemoyer. 2018. Jointly predicting predicates and arguments in neural semantic role labeling. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. Association for Computational Linguistics, Melbourne, Australia, pages 364–369. https://doi.org/10.18653/v1/P18-2058.

Luheng He, Kenton Lee, Mike Lewis, and Luke Zettlemoyer. 2017. Deep semantic role labeling: What works and what's next. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, Vancouver, Canada, pages 473–483. https://doi.org/10.18653/v1/P17-1044.

Enrique Henestroza Anguiano and Marie Candito. 2012. Probabilistic lexical generalization for French dependency parsing. In *Proceedings of the ACL 2012 Joint Workshop on Statistical Parsing and Semantic Processing of Morphologically Rich Languages*. Association for Computational Linguistics, Jeju, Republic of Korea, pages 1–11.

Aurélie Herbelot and Marco Baroni. 2017. High-risk learning: acquiring new word vectors from tiny data. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Copenhagen, Denmark, pages 304–309. https://doi.org/10.18653/v1/D17-1030.

Felix Hill, KyungHyun Cho, Sebastien Jean, Coline Devin, and Yoshua Bengio. 2014a. Not All Neural Embeddings are Born Equal. *NIPS 2014 Workshop on Learning Semantics* .

Felix Hill, Roi Reichart, and Anna Korhonen. 2014b. SimLex-999: Evaluating Semantic Models with (Genuine) Similarity Estimation. *ArXiv e-prints* .

Felix Hill, Roi Reichart, and Anna Korhonen. 2015. SimLex-999: Evaluating semantic models with (genuine) similarity estimation. *Computational Linguistics* 41(4):665–695. https://doi.org/10.1162/COLI_a_00237.

Geoffrey E. Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R. Salakhutdinov. 2012. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580* pages 1–18. https://doi.org/arXiv:1207.0580.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9(8):1735–1780.

Johannes Hoffart, Stephan Seufert, Dat Ba Nguyen, Martin Theobald, and Gerhard Weikum. 2012. KORE: keyphrase overlap relatedness for entity disambiguation. In Xue-wen Chen, Guy Lebanon, Haixun Wang, and Mohammed J. Zaki, editors, *21st ACM International Conference on Information and Knowledge Management, CIKM'12, Maui, HI, USA, October 29 - November 02, 2012*. ACM, pages 545–554. https://doi.org/10.1145/2396761.2396832.

Johannes Hoffart, Mohamed Amir Yosef, Ilaria Bordino, Hagen Fürstenau, Manfred Pinkal, Marc Spaniol, Bilyana Taneva, Stefan Thater, and Gerhard Weikum. 2011. Robust disambiguation of named entities in text. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Edinburgh, Scotland, UK., pages 782–792.

Kristy Hollingshead and Brian Roark. 2007. Pipeline iteration. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*. Association for Computational Linguistics, Prague, Czech Republic, pages 952–959.

Matthew Honnibal and Mark Johnson. 2015. An improved non-monotonic transition system for dependency parsing. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Lisbon, Portugal, pages 1373–1378. https://doi.org/10.18653/v1/D15-1162.

Eduard Hovy, Mitchell Marcus, Martha Palmer, Lance Ramshaw, and Ralph Weischedel. 2006. OntoNotes: The 90% solution. In *Proceedings of the Human Language Technology Conference of the NAACL, Companion Volume: Short Papers*. Association for Computational Linguistics, New York City, USA, pages 57–60.

Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, Melbourne, Australia, pages 328–339. https://doi.org/10.18653/v1/P18-1031.

Ignacio Iacobacci, Mohammad Taher Pilehvar, and Roberto Navigli. 2016. Embeddings for word sense disambiguation: An evaluation study. In *Proceedings of the 54th Annual Meeting of the Association for Computational*

Linguistics (Volume 1: Long Papers). Association for Computational Linguistics, Berlin, Germany, pages 897–907. https://doi.org/10.18653/v1/P16-1085.

Miloš Jakubíček, Adam Kilgarriff, Vojtěch Kovář, Pavel Rychlý, and Vít Suchomel. 2013. The Tenten Corpus Family. In *7th International Corpus Linguistics Conference CL*. pages 125–127.

Sébastien Jean, Kyunghyun Cho, Roland Memisevic, and Yoshua Bengio. 2015. On using very large target vocabulary for neural machine translation. In *ACL-IJCNLP 2015*. pages 1–10. https://doi.org/10.3115/v1/P15-1001.

Jay J Jiang and David W Conrath. 1997. Semantic similarity based on corpus statistics and lexical taxonomy. In *Proceedings of the 10th Research on Computational Linguistics International Conference*. The Association for Computational Linguistics and Chinese Language Processing (ACLCLP), pages 19–33.

Jiarong Jiang, Adam R. Teichert, Hal Daumé III, and Jason Eisner. 2012. Learned prioritization for trading off accuracy and speed. In Peter L. Bartlett, Fernando C. N. Pereira, Christopher J. C. Burges, Léon Bottou, and Kilian Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25: 26th Annual Conference on Neural Information Processing Systems 2012. Proceedings of a meeting held December 3-6, 2012, Lake Tahoe, Nevada, United States*. pages 1340–1348.

Jing Jiang. 2008. Domain Adaptation in Natural Language Processing. *PhD Thesis* http://www.mysmu.edu/faculty/jingjiang/.

Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S. Weld, Luke Zettlemoyer, and Omer Levy. 2020. SpanBERT: Improving pre-training by representing and predicting spans. *Transactions of the Association for Computational Linguistics* 8:64–77. https://doi.org/10.1162/tacl_a_00300.

Mandar Joshi, Omer Levy, Luke Zettlemoyer, and Daniel Weld. 2019. BERT for coreference resolution: Baselines and analysis. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Association for Computational Linguistics, Hong Kong, China, pages 5803–5808. https://doi.org/10.18653/v1/D19-1588.

Jerrold J. Katz and Jerry A. Fodor. 1963. The structure of a semantic theory. *Language* 39(2):170–210.

Urvashi Khandelwal, He He, Peng Qi, and Dan Jurafsky. 2018. Sharp nearby, fuzzy far away: How neural language models use context. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, Melbourne, Australia, pages 284–294. https://doi.org/10.18653/v1/P18-1027.

Urvashi Khandelwal, Omer Levy, Dan Jurafsky, Luke Zettlemoyer, and Mike Lewis. 2020. Generalization through memorization: Nearest neighbor language models. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Douwe Kiela and Stephen Clark. 2014. A Systematic Study of Semantic Vector Space Model Parameters. In *Proceedings of EACL 2014, Workshop on Continuous Vector Space Models and their Compositionality (CVSC)*. Association for Computational Linguistics, pages 21–30.

Eliyahu Kiperwasser and Yoav Goldberg. 2016a. Easy-first dependency parsing with hierarchical tree LSTMs. *Transactions of the Association for Computational Linguistics* 4:445–461. https://doi.org/10.1162/tacl_a_00110.

Eliyahu Kiperwasser and Yoav Goldberg. 2016b. Simple and accurate dependency parsing using bidirectional LSTM feature representations. *Transactions of the Association for Computational Linguistics* 4:313–327. https://doi.org/10.1162/tacl_a_00101.

Lingpeng Kong and Noah A. Smith. 2014. An Empirical Comparison of Parsing Methods for Stanford Dependencies. *arXiv preprint arXiv:1404.4314* .

Mateusz Kopeć. 2014. MMAX2 for coreference annotation. In *Proceedings of the Demonstrations at the 14th Conference of the European Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics, Gothenburg, Sweden, pages 93–96. https://doi.org/10.3115/v1/E14-2024.

Alex Krizhevsky. 2009. Learning Multiple Layers of Features from Tiny Images .

Henry Kučera and Winthrop Nelson Francis. 1967. *Computational analysis of present-day American English*. Dartmouth Publishing Group.

Sayali Kulkarni, Amit Singh, Ganesh Ramakrishnan, and Soumen Chakrabarti. 2009. Collective annotation of Wikipedia entities in web

text. *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '09* page 457. https://doi.org/10.1145/1557019.1557073.

Niraj Kumar, Kannan Srinathan, and Vasudeva Varma. 2015. Semantic Role Labeling of Speech Transcripts. In *CICLing 2015*. pages 583–595. https://doi.org/10.1007/978-3-642-28601-8.

Jonathan K. Kummerfeld, David Hall, James R. Curran, and Dan Klein. 2012. Parser showdown at the Wall Street corral: An empirical investigation of error types in parser output. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*. Association for Computational Linguistics, Jeju Island, Korea, pages 1048–1059.

Alexey Kurakin, Ian Goodfellow, and Samy Bengio. 2017. Adversarial examples in the physical world. *ICLR 2017* .

Marta Kutas and Kara D Federmeier. 2011. Thirty years and counting: finding meaning in the N400 component of the event-related brain potential (ERP). *Annual review of psychology* 62:621–47. https://doi.org/10.1146/annurev.psych.093008.131123.

George Lakoff and Mark Johnson. 1980. Metaphors we live by. *Chicago/London* .

Yadi Lao, Jun Xu, Sheng Gao, Jun Guo, and Jirong Wen. 2019. Name entity recognition with policy-value networks. In Benjamin Piwowarski, Max Chevalier, Éric Gaussier, Yoelle Maarek, Jian-Yun Nie, and Falk Scholer, editors, *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2019, Paris, France, July 21-25, 2019*. ACM, pages 1245–1248. https://doi.org/10.1145/3331184.3331349.

Egoitz Laparra and German Rigau. 2012. Exploiting Explicit Annotations and Semantic Types for Implicit Argument Resolution. In *Semantic Computing (ICSC), 2012 IEEE Sixth International Conference on*. pages 75–78. https://doi.org/10.1109/ICSC.2012.47.

Egoitz Laparra and German Rigau. 2013. ImpAr: A deterministic algorithm for implicit semantic role labelling. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, Sofia, Bulgaria, pages 1180–1189.

Daniil Larionov, Artem Shelmanov, Elena Chistova, and Ivan Smirnov. 2019. Semantic role labeling with pretrained language models for known and unknown predicates. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2019)*. INCOMA Ltd., Varna, Bulgaria, pages 619–628. https://doi.org/10.26615/978-954-452-056-4_073.

Claudia Leacock and Martin Chodorow. 1998. Combining local context and WordNet similarity for word sense identification. *WordNet: An electronic lexical database* 49(2):265–283.

Matthew Lease and Eugene Charniak. 2005. Parsing biomedical literature. In *Second International Joint Conference on Natural Language Processing: Full Papers*. https://doi.org/10.1007/11562214_6.

Heeyoung Lee, Angel Chang, Yves Peirsman, Nathanael Chambers, Mihai Surdeanu, and Dan Jurafsky. 2013. Deterministic coreference resolution based on entity-centric, precision-ranked rules. *Computational Linguistics* 39(4):885–916. https://doi.org/10.1162/COLI_a_00152.

Heeyoung Lee, Yves Peirsman, Angel Chang, Nathanael Chambers, Mihai Surdeanu, and Dan Jurafsky. 2011. Stanford's multi-pass sieve coreference resolution system at the CoNLL-2011 shared task. In *Proceedings of the Fifteenth Conference on Computational Natural Language Learning: Shared Task*. Association for Computational Linguistics, Portland, Oregon, USA, pages 28–34.

Kenton Lee, Luheng He, Mike Lewis, and Luke Zettlemoyer. 2017. End-to-end neural coreference resolution. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Copenhagen, Denmark, pages 188–197. https://doi.org/10.18653/v1/D17-1018.

Kenton Lee, Luheng He, and Luke Zettlemoyer. 2018. Higher-order coreference resolution with coarse-to-fine inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*. Association for Computational Linguistics, New Orleans, Louisiana, pages 687–692. https://doi.org/10.18653/v1/N18-2108.

Hector J Levesque, Ernest Davis, and Leora Morgenstern. 2011. The Winograd schema challenge. In *AAAI Spring Symposium: Logical Formalizations of Commonsense Reasoning*.

Omer Levy and Yoav Goldberg. 2014a. Dependency-based word embeddings. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. Association for Computational Linguistics, Baltimore, Maryland, pages 302–308.

Omer Levy and Yoav Goldberg. 2014b. Neural word embedding as implicit matrix factorization. In Z. Ghahramani, M. Welling, C. Cortes, N.D. Lawrence, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, Curran Associates, Inc., pages 2177–2185.

Jiwei Li, Will Monroe, and Dan Jurafsky. 2016a. Understanding Neural Networks through Representation Erasure. *arXiv preprint* https://arxiv.org/abs/1612.08220.

Tao Li, Parth Anand Jawale, Martha Palmer, and Vivek Srikumar. 2020a. Structured tuning for semantic role labeling. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, Online, pages 8402–8412. https://doi.org/10.18653/v1/2020.acl-main.744.

Xiaoya Li, Jingrong Feng, Yuxian Meng, Qinghong Han, Fei Wu, and Jiwei Li. 2020b. A unified MRC framework for named entity recognition. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, Online, pages 5849–5859. https://doi.org/10.18653/v1/2020.acl-main.519.

Yujia Li, Daniel Tarlow, Marc Brockschmidt, and Richard S. Zemel. 2016b. Gated graph sequence neural networks. In Yoshua Bengio and Yann LeCun, editors, *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*.

Zhenghua Li, Min Zhang, Wanxiang Che, Ting Liu, and Wenliang Chen. 2014a. Joint Optimization for Chinese POS Tagging and Dependency Parsing. *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 22(1):274–286. https://doi.org/10.1109/TASLP.2013.2288081.

Zhenghua Li, Min Zhang, and Wenliang Chen. 2014b. Ambiguity-aware Ensemble Training for Semi-supervised Dependency Parsing. In *ACL 2014*. pages 457–467.

Zuchao Li, Jiaxun Cai, and Hai Zhao. 2019. Effective Representation for Easy-First Dependency Parsing. In Abhaya C Nayak and Alok Sharma, editors, *PRICAI 2019: Trends in Artificial Intelligence*. Springer International Publishing, Cham, pages 351–363.

Bin Liang, Hongcheng Li, Miaoqiang Su, Pan Bian, Xirong Li, and Wenchang Shi. 2018. Deep text classification can be fooled. In Jérôme Lang, editor, *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13-19, 2018, Stockholm, Sweden*. ijcai.org, pages 4208–4215. https://doi.org/10.24963/ijcai.2018/585.

Xiaoqiang Luo. 2005. On coreference resolution performance metrics. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Vancouver, British Columbia, Canada, pages 25–32.

Xiaoqiang Luo, Abe Ittycheriah, Hongyan Jing, Nanda Kambhatla, and Salim Roukos. 2004. A mention-synchronous coreference resolution algorithm based on the bell tree. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL-04)*. Barcelona, Spain, pages 135–142. https://doi.org/10.3115/1218955.1218973.

Ji Ma, Tong Xiao, Jingbo Zhu, and Feiliang Ren. 2012. Easy-first Chinese POS tagging and dependency parsing. In *Proceedings of COLING 2012*. The COLING 2012 Organizing Committee, Mumbai, India, pages 1731–1746.

Ji Ma, Jingbo Zhu, Tong Xiao, and Nan Yang. 2013. Easy-first POS tagging and dependency parsing with beam search. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. Association for Computational Linguistics, Sofia, Bulgaria, pages 110–114.

Xuezhe Ma and Eduard Hovy. 2016. End-to-end sequence labeling via bi-directional LSTM-CNNs-CRF. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, Berlin, Germany, pages 1064–1074. https://doi.org/10.18653/v1/P16-1101.

Francis Maes, Ludovic Denoyer, and Patrick Gallinari. 2009. Structured prediction with reinforcement learning. *Machine Learning* (77):271–301. https://doi.org/10.1007/s10994-009-5140-8.

Diego Marcheggiani, Anton Frolov, and Ivan Titov. 2017. A simple and accurate syntax-agnostic neural model for dependency-based semantic role labeling. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*. Association for Computational Linguistics, Vancouver, Canada, pages 411–420. https://doi.org/10.18653/v1/K17-1041.

Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics* 19(2):313–330.

Katja Markert and Malvina Nissim. 2005. Comparing knowledge sources for nominal anaphora resolution. *Computational Linguistics* 31(3):367–402. https://doi.org/10.1162/089120105774321064.

Sebastian Martschat and Michael Strube. 2015. Latent structures for coreference resolution. *Transactions of the Association for Computational Linguistics* 3:405–418. https://doi.org/10.1162/tacl_a_00147.

Andrew McCallum. 2009. Joint inference for natural language processing. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL-2009)*. Association for Computational Linguistics, Boulder, Colorado, page 1.

Diana McCarthy and Roberto Navigli. 2009. The english lexical substitution task. *Language Resources and Evaluation* 43(2):139–159. https://doi.org/10.1007/s10579-009-9084-1.

Ryan McDonald, Koby Crammer, and Fernando Pereira. 2005a. Online large-margin training of dependency parsers. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*. Association for Computational Linguistics, Ann Arbor, Michigan, pages 91–98. https://doi.org/10.3115/1219840.1219852.

Ryan McDonald and Joakim Nivre. 2007. Characterizing the errors of data-driven dependency parsing models. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*. Association for Computational Linguistics, Prague, Czech Republic, pages 122–131.

Ryan McDonald, Fernando Pereira, Kiril Ribarov, and Jan Hajič. 2005b. Non-projective dependency parsing using spanning tree algorithms. In *HLT-EMNLP 2005*. Association for Computational Linguistics, pages 523–530.

Oren Melamud, Jacob Goldberger, and Ido Dagan. 2016. Context2vec: Learning generic context embedding with bidirectional LSTM. In *CoNLL*. pages 51–61.

Adam Meyers, Ruth Reeves, Catherine Macleod, Rachel Szekely, Veronika Zielinska, Brian Young, and Ralph Grishman. 2004. The NomBank project: An interim report. In *Proceedings of the Workshop Frontiers in Corpus*

Annotation at HLT-NAACL 2004. Association for Computational Linguistics, Boston, Massachusetts, USA, pages 24–31.

Tomas Mikolov, Greg Corrado, Kai Chen, and Jeffrey Dean. 2013a. Efficient Estimation of Word Representations in Vector Space. *ICLR 2013* pages 1–12. https://doi.org/10.1162/153244303322533223.

Tomas Mikolov, Greg Corrado, Kai Chen, and Jeffrey Dean. 2013b. Efficient Estimation of Word Representations in Vector Space. *ICLR Workshop* .

George Miller, Claudia Leacock, Randee Tengi, and Ross T. Bunker. 1993a. A semantic concordance. In *Human language technology: Proceedings of a Workshop Held at Plainsboro, New Jersey, March 21-24, 1993*.

George A. Miller, Claudia Leacock, Randee Tengi, and Ross T. Bunker. 1993b. A semantic concordance. In *Human Language Technology: Proceedings of a Workshop Held at Plainsboro, New Jersey, March 21-24, 1993*.

Tatjana Moor, Michael Roth, and Anette Frank. 2013a. The ontonotes 5 verbs non-local role linking data set (on5v) pages 369–375.

Tatjana Moor, Michael Roth, and Anette Frank. 2013b. Predicate-specific annotations for implicit role binding: Corpus annotation, data analysis and evaluation experiments. In *Proceedings of the 10th International Conference on Computational Semantics (IWCS 2013) – Short Papers*. Association for Computational Linguistics, Potsdam, Germany, pages 369–375.

Nafise Sadat Moosavi and Michael Strube. 2017. Lexical features in coreference resolution: To be used with caution. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. Association for Computational Linguistics, Vancouver, Canada, pages 14–19. https://doi.org/10.18653/v1/P17-2003.

Andrea Moro, Alessandro Raganato, and Roberto Navigli. 2014. Entity linking meets word sense disambiguation: a unified approach. *TACL 2014* 2:231–244. http://www.aclweb.org/anthology/Q14-1019.

Roberto Navigli. 2009. Word sense disambiguation: A survey. *ACM Computing Surveys (CSUR)* 41(2):10.

Roberto Navigli, David Jurgens, and Daniele Vannella. 2013a. SemEval-2013 task 12: Multilingual word sense disambiguation. In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation*

(SemEval 2013). Association for Computational Linguistics, Atlanta, Georgia, USA, pages 222–231.

Roberto Navigli, David Jurgens, and Daniele Vannella. 2013b. SemEval-2013 task 12: Multilingual Word Sense Disambiguation. In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*. Association for Computational Linguistics, Atlanta, Georgia, USA, pages 222–231.

Roberto Navigli, Kenneth C. Litkowski, and Orin Hargraves. 2007. Semeval-2007 task 07: Coarse-grained english all-words task. In *Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval-2007)*. Association for Computational Linguistics, Prague, Czech Republic, pages 30–35. http://www.aclweb.org/anthology/S/S07/S07-1006.

Douglas L. Nelson, Cathy L. McEvoy, and Thomas A. Schreiber. 2004. The university of south florida free association, rhyme, and word fragment norms. *Behavior Research Methods, Instruments, & Computers* 36(3):402–407.

Dominick Ng and James R. Curran. 2015a. Identifying cascading errors using constraints in dependency parsing. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. Association for Computational Linguistics, Beijing, China, pages 1148–1158. https://doi.org/10.3115/v1/P15-1111.

Dominick Ng and James R Curran. 2015b. Identifying Cascading Errors using Constraints in Dependency Parsing. In *ACL-IJCNLP*. ACL, Beijing, pages 1148–1158.

Hwee Tou Ng, Siew Mei Wu, Yuanbin Wu, Christian Hadiwinoto, and Joel Tetreault. 2013. The CoNLL-2013 shared task on grammatical error correction. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning: Shared Task*. Association for Computational Linguistics, Sofia, Bulgaria, pages 1–12.

Vincent Ng. 2007. Shallow Semantics for Coreference Resolution. In *Proceedings of the Twentieth International Joint Conference on Artificial Intelligence (IJCAI)*. Hyderabad, India, pages 1689–1694.

Vincent Ng and Claire Cardie. 2002a. Identifying anaphoric and non-anaphoric noun phrases to improve coreference resolution. In *COLING 2002: The 19th International Conference on Computational Linguistics*.

Vincent Ng and Claire Cardie. 2002b. Improving machine learning approaches to coreference resolution. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, Philadelphia, Pennsylvania, USA, pages 104–111. https://doi.org/10.3115/1073083.1073102.

Khanh Nguyen and Brendan O'Connor. 2015. Posterior calibration and exploratory analysis for natural language processing models. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Lisbon, Portugal, pages 1587–1598. https://doi.org/10.18653/v1/D15-1182.

Mante S. Nieuwland and Jos J. A. van Berkum. 2006. When Peanuts Fall in Love. *Journal of Cognitive Neuroscience* 18(7):1098–1111. https://doi.org/10.1162/jocn.2006.18.7.1098.

Zheng-Yu Niu, Dong-Hong Ji, and Chew Lim Tan. 2005. Word sense disambiguation using label propagation based semi-supervised learning. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*. Association for Computational Linguistics, Ann Arbor, Michigan, pages 395–402. https://doi.org/10.3115/1219840.1219889.

Joakim Nivre. 2003. An efficient algorithm for projective dependency parsing. In *Proceedings of the Eighth International Conference on Parsing Technologies*. Nancy, France, pages 149–160.

Joakim Nivre. 2004. Incrementality in deterministic dependency parsing. In *Proceedings of the Workshop on Incremental Parsing: Bringing Engineering and Cognition Together*. Association for Computational Linguistics, Barcelona, Spain, pages 50–57.

Joakim Nivre. 2009. Non-projective dependency parsing in expected linear time. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*. Association for Computational Linguistics, Suntec, Singapore, pages 351–359.

Joakim Nivre and Daniel Fernández-González. 2014. Arc-eager Parsing with the Tree Constraint. *Computational Linguistics* 40(2):259–267.

Joakim Nivre, Johan Hall, and Jens Nilsson. 2006a. MaltParser: A data-driven parser-generator for dependency parsing. In *Proceedings of the Fifth International Conference on Language Resources and Evaluation (LREC'06)*. European Language Resources Association (ELRA), Genoa, Italy.

Joakim Nivre, Johan Hall, Jens Nilsson, Atanas Chanev, Eryiğit Gülşen, Sandra Kübler, Svetoslav Marinov, and Erwin Marsi. 2007. MaltParser: A language-independent system for data-driven dependency parsing. *Natural Language Engineering* 13(02):95–135.

Joakim Nivre, Johan Hall, Jens Nilsson, Gülşen Eryiğit, and Svetoslav Marinov. 2006b. Labeled pseudo-projective dependency parsing with support vector machines. In *CoNLL 2006*. ACL, pages 221–225.

Joakim Nivre, Marco Kuhlmann, and Johan Hall. 2009. An improved oracle for dependency parsing with online reordering. In *Proceedings of the 11th International Conference on Parsing Technologies (IWPT'09)*. Association for Computational Linguistics, Paris, France, pages 73–76.

Simon Ostermann, Michael Roth, Ashutosh Modi, Stefan Thater, and Manfred Pinkal. 2018. SemEval-2018 task 11: Machine comprehension using commonsense knowledge. In *Proceedings of The 12th International Workshop on Semantic Evaluation*. Association for Computational Linguistics, New Orleans, Louisiana, pages 747–757. https://doi.org/10.18653/v1/S18-1119.

Sebastian Padó and Mirella Lapata. 2007. Dependency-based construction of semantic space models. *Computational Linguistics* 33(2):161–199.

Santanu Pal, Sudip Kumar Naskar, Mihaela Vela, Qun Liu, and Josef van Genabith. 2017. Neural automatic post-editing using prior alignment and reranking. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*. Association for Computational Linguistics, Valencia, Spain, pages 349–355.

Martha Palmer, Hoa Trang Dang, and Christiane Fellbaum. 2007. Making fine-grained and coarse-grained sense distinctions, both manually and automatically. *Natural Language Engineering* 13(02):137–163.

Martha Palmer, Christiane Fellbaum, Scott Cotton, Lauren Delfs, and Hoa Trang Dang. 2001. English tasks: All-words and verb lexical sample. In *Proceedings of the Second International Workshop on Evaluating Word Sense Disambiguation Systems (SENSEVAL-2)*. Association for Computational Linguistics, Toulouse, France, pages 21–24.

Martha Palmer, Daniel Gildea, and Paul Kingsbury. 2005. The Proposition Bank: An annotated corpus of semantic roles. *Computational Linguistics* 31(1):71–106. https://doi.org/10.1162/0891201053630264.

Silvia Pareti. 2011. Annotating attribution relations and their features. In *Proceedings of the fourth workshop on Exploiting Semantic Annotations in Information Retrieval*. ACM, pages 19–20.

Ted Pedersen. 2010. Information content measures of semantic similarity perform better without sense-tagged text. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics, pages 329–332.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Édouard Duchesnay. 2011. Scikit-learn: Machine Learning in {P}ython. *Journal of Machine Learning Research* 12:2825–2830.

Wenzhe Pei, Tao Ge, and Baobao Chang. 2015. An effective neural network model for graph-based dependency parsing. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. Association for Computational Linguistics, Beijing, China, pages 313–322. https://doi.org/10.3115/v1/P15-1031.

Haoruo Peng, Kai-Wei Chang, and Dan Roth. 2015. A joint framework for coreference resolution and mention head detection. In *Proceedings of the Nineteenth Conference on Computational Natural Language Learning*. Association for Computational Linguistics, Beijing, China, pages 12–21. https://doi.org/10.18653/v1/K15-1002.

Haoruo Peng and Dan Roth. 2016. Two discourse driven language models for semantics. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, Berlin, Germany, pages 290–300. https://doi.org/10.18653/v1/P16-1028.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. Association for Computational Linguistics, New Orleans, Louisiana, pages 2227–2237. https://doi.org/10.18653/v1/N18-1202.

Mohammad Taher Pilehvar, David Jurgens, and Roberto Navigli. 2013. Align, disambiguate and walk: A unified approach for measuring semantic similarity. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, Sofia, Bulgaria, pages 1341–1351.

Barbara Plank. 2016. What to do about non-standard (or non-canonical) language in NLP. *arXiv preprint arXiv:1608.07836* .

Massimo Poesio, Jon Chamberlain, Udo Kruschwitz, Livio Robaldo, and Luca Ducceschi. 2013. Phrase Detectives: Utilizing Collective Intelligence for Internet-scale Language Resource Creation. *ACM Trans. Interact. Intell. Syst.* 3(1):4202–4206. https://doi.org/10.1145/2448116.2448119.

Simone Paolo Ponzetto and Michael Strube. 2006. Exploiting semantic role labeling, WordNet and Wikipedia for coreference resolution. In *Proceedings of the Human Language Technology Conference of the NAACL, Main Conference*. Association for Computational Linguistics, New York City, USA, pages 192–199.

Hoifung Poon and Pedro Domingos. 2007. Joint inference in information extraction. *Proceedings of the National Conference on Artificial Intelligence* 1:913–918.

Hoifung Poon and Pedro Domingos. 2008. Joint unsupervised coreference resolution with Markov Logic. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Honolulu, Hawaii, pages 650–659.

Marten Postma, Ruben Izquierdo Bevia, and Piek Vossen. 2016. More is not always better: balancing sense distributions for all-words word sense disambiguation. In *COLING 2016: Technical Papers*. pages 3496–3506.

Sameer Pradhan, Edward Loper, Dmitriy Dligach, and Martha Palmer. 2007. SemEval-2007 task-17: English lexical sample, SRL and all words. In *Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval-2007)*. Association for Computational Linguistics, Prague, Czech Republic, pages 87–92.

Sameer Pradhan, Alessandro Moschitti, Nianwen Xue, Olga Uryupina, and Yuchen Zhang. 2012. CoNLL-2012 shared task: Modeling multilingual unrestricted coreference in OntoNotes. In *Joint Conference on EMNLP and CoNLL - Shared Task*. Association for Computational Linguistics, Jeju Island, Korea, pages 1–40.

Sameer Pradhan, Wayne Ward, Kadri Hacioglu, James Martin, and Daniel Jurafsky. 2005. Semantic role labeling using different syntactic views. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*. Association for Computational Linguistics, Ann Arbor, Michigan, pages 581–588. https://doi.org/10.3115/1219840.1219912.

Chris Quirk and Simon Corston-Oliver. 2006. The impact of parse quality on syntactically-informed statistical machine translation. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Sydney, Australia, pages 62–69.

Roy Rada, Hafedh Mili, Ellen Bicknell, and Maria Blettner. 1989. Development and application of a metric on semantic nets. *Systems, Man and Cybernetics, IEEE Transactions on* 19(1):17–30. https://doi.org/10.1109/21.24528.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language Models are Unsupervised Multitask Learners.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2019. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. *arXiv e-prints* .

Alessandro Raganato, Jose Camacho-Collados, and Roberto Navigli. 2017a. Word sense disambiguation: A unified evaluation framework and empirical comparison. In *EACL 2017*. Association for Computational Linguistics, pages 99–110. http://aclweb.org/anthology/E17-1010.

Alessandro Raganato, Claudio Delli Bovi, and Roberto Navigli. 2017b. Neural sequence learning models for word sense disambiguation. In *EMNLP 2017*. Association for Computational Linguistics, pages 1156–1167. http://aclweb.org/anthology/D17-1120.

Karthik Raghunathan, Heeyoung Lee, Sudarshan Rangarajan, Nathanael Chambers, Mihai Surdeanu, Dan Jurafsky, and Christopher Manning. 2010. A multi-pass sieve for coreference resolution. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Cambridge, MA, pages 492–501.

Altaf Rahman and Vincent Ng. 2012. Resolving complex cases of definite pronouns: The Winograd schema challenge. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*. Association for Computational Linguistics, Jeju Island, Korea, pages 777–789.

Karthik Raman, Adith Swaminathan, Johannes Gehrke, and Thorsten Joachims. 2013. Beyond myopic inference in big data pipelines. In Inderjit S. Dhillon, Yehuda Koren, Rayid Ghani, Ted E. Senator, Paul Bradley, Rajesh Parekh, Jingrui He, Robert L. Grossman, and Ramasamy Uthurusamy, editors, *The 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2013, Chicago, IL, USA, August 11-14, 2013*. ACM, pages 86–94. https://doi.org/10.1145/2487575.2487588.

Marc'Aurelio Ranzato, Sumit Chopra, Michael Auli, and Wojciech Zaremba. 2016. Sequence level training with recurrent neural networks. In Yoshua Bengio and Yann LeCun, editors, *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*.

Sujith Ravi and Qiming Diao. 2016. Large scale distributed semi-supervised learning using streaming approximation. In Arthur Gretton and Christian C. Robert, editors, *Proceedings of the 19th International Conference on Artificial Intelligence and Statistics, AISTATS 2016, Cadiz, Spain, May 9-11, 2016*. JMLR.org, volume 51 of *JMLR Workshop and Conference Proceedings*, pages 519–528.

Keith Rayner and Lyn Frazier. 1982. Making and correcting errors during sentence comprehension: Eye movements in the analysis of structurally ambiguous sentences. *Cognitive Psychology* 14(2):178–210. https://doi.org/10.1016/0010-0285(82)90008-1.

Nils Reimers and Iryna Gurevych. 2017. Reporting score distributions makes a difference: Performance study of LSTM-networks for sequence tagging. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Copenhagen, Denmark, pages 338–348. https://doi.org/10.18653/v1/D17-1035.

Philip Resnik. 1995. Using information content to evaluate semantic similarity in a taxonomy. In *Proceedings of the 14th international joint conference on Artificial intelligence-Volume 1*. Morgan Kaufmann Publishers Inc., pages 448–453.

Philip Resnik. 1996. Selectional constraints: an information-tehoretic model and its computational realization. *Cognition* 61(508):127–159. https://doi.org/10.1016/S0010-0277(96)00722-6.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2018. Semantically equivalent adversarial rules for debugging NLP models. In *Proceedings of*

the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). Association for Computational Linguistics, Melbourne, Australia, pages 856–865. https://doi.org/10.18653/v1/P18-1079.

Alan Ritter, Mausam, and Oren Etzioni. 2010. A Latent Dirichlet Allocation method for selectional preferences. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, Uppsala, Sweden, pages 424–434.

Michael Röder, Ricardo Usbeck, Sebastian Hellmann, Daniel Gerber, and Andreas Both. 2014. N$ˆ3$-A Collection of Datasets for Named Entity Recognition and Disambiguation in the NLP Interchange Format. In *LREC*. pages 3529–3533.

Stephane Ross, Geoffrey J Gordon, and J Andrew Bagnell. 2011. A Reduction of Imitation Learning and Structured Prediction to No-Regret Online Learning. *AISTATS* 15:627–635.

Michael Roth and Anette Frank. 2013. Automatically identifying implicit arguments to improve argument linking and coherence modeling. In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 1: Proceedings of the Main Conference and the Shared Task: Semantic Textual Similarity*. Association for Computational Linguistics, Atlanta, Georgia, USA, pages 306–316.

Michael Roth and Anette Frank. 2015. Inducing implicit arguments from comparable texts: A framework and its applications. *Computational Linguistics* 41(4):625–664. https://doi.org/10.1162/COLI_a_00236.

Sascha Rothe and Hinrich Schütze. 2017. Autoextend: Combining word embeddings with semantic resources. *Computational Linguistics* 43(3):593–617. https://doi.org/10.1162/COLI_a_00294.

Herbert Rubenstein and John B Goodenough. 1965. Contextual correlates of synonymy. *Communications of the ACM* 8(10):627–633.

Sebastian Ruder, Matthew E. Peters, Swabha Swayamdipta, and Thomas Wolf. 2019. Transfer learning in natural language processing. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Tutorials*. Association for Computational Linguistics, Minneapolis, Minnesota, pages 15–18. https://doi.org/10.18653/v1/N19-5004.

Rachel Rudinger, Jason Naradowsky, Brian Leonard, and Benjamin Van Durme. 2018. Gender bias in coreference resolution. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*. Association for Computational Linguistics, New Orleans, Louisiana, pages 8–14. https://doi.org/10.18653/v1/N18-2002.

Josef Ruppenhofer, Caroline Sporleder, Roser Morante, Collin Baker, and Martha Palmer. 2009. SemEval-2010 task 10: Linking events and their participants in discourse. In *Proceedings of the Workshop on Semantic Evaluations: Recent Achievements and Future Directions (SEW-2009)*. Association for Computational Linguistics, Boulder, Colorado, pages 106–111.

Joseph Ruppenhofer, Caroline Sporleder, Roser Morante, Colin Baker, and Martha Palmer. 2010. SemEval-2010 Task 10: Linking Events and Their Participants in Discourse pages 45–50.

Kenji Sagae. 2010. Self-training without reranking for parser domain adaptation and its impact on semantic role labeling. In *Proceedings of the 2010 Workshop on Domain Adaptation for Natural Language Processing*. Association for Computational Linguistics, Uppsala, Sweden, pages 37–44.

Magnus Sahlgren. 2006. *The Word-Space Model: Using distributional analysis to represent syntagmatic and paradigmatic relations between words in high-dimensional vector spaces*. Ph.D. thesis, Stockholm University.

Chinnadhurai Sankar, Sandeep Subramanian, Chris Pal, Sarath Chandar, and Yoshua Bengio. 2019. Do neural dialog systems use the conversation history effectively? an empirical study. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, Florence, Italy, pages 32–37. https://doi.org/10.18653/v1/P19-1004.

Niko Schenk and Christian Chiarcos. 2016. Unsupervised learning of prototypical fillers for implicit semantic role labeling. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, San Diego, California, pages 1473–1479. https://doi.org/10.18653/v1/N16-1173.

Niko Schenk, Christian Chiarcos, and Maria Sukhareva. 2015. Towards the unsupervised acquisition of implicit semantic roles. In *Proceedings of the*

International Conference Recent Advances in Natural Language Processing. INCOMA Ltd. Shoumen, BULGARIA, Hissar, Bulgaria, pages 570–578.

Djamé Seddah, Sandra Kübler, and Reut Tsarfaty. 2014. Introducing the SPMRL 2014 Shared Task on Parsing Morphologically-Rich Languages. In *Proceedings of the First Joint Workshop on Statistical Parsing of Morphologically Rich Languages and Syntactic Analysis of Non-Canonical Languages*. pages 103–109.

Claude Elwood Shannon. 1948. A mathematical theory of communication. *Bell System Technical Journal* 27(3):379–423. https://doi.org/doi:10.1002/j.1538-7305.1948.tb01338.x.

Shiqi Shen, Yong Cheng, Zhongjun He, Wei He, Hua Wu, Maosong Sun, and Yang Liu. 2016. Minimum risk training for neural machine translation. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, Berlin, Germany, pages 1683–1692. https://doi.org/10.18653/v1/P16-1159.

Carina Silberer and Anette Frank. 2012. Casting implicit role linking as an anaphora resolution task. In **SEM 2012: The First Joint Conference on Lexical and Computational Semantics – Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation (SemEval 2012)*. Association for Computational Linguistics, Montréal, Canada, pages 1–10.

Michel Simard, Cyril Goutte, and Pierre Isabelle. 2007. Statistical phrase-based post-editing. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Proceedings of the Main Conference*. Association for Computational Linguistics, Rochester, New York, pages 508–515.

Osvaldo Simeone. 2018. A Very Brief Introduction to Machine Learning with Applications to Communication Systems. *IEEE Transactions on Cognitive Communications and Networking* 4(4):648–664. https://doi.org/10.1109/TCCN.2018.2881442.

Karen Simonyan and Andrew Zisserman. 2015. Very deep convolutional networks for large-scale image recognition. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Sameer Singh, Karl Schultz, and Andrew McCallum. 2009. Bi-directional joint inference for entity resolution and segmentation using imperatively-defined factor graphs. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 5782 LNAI(PART 2):414–429. https://doi.org/10.1007/978-3-642-04174-7_27.

Hyun-Je Song, Jeong-Woo Son, Tae-Gil Noh, Seong-Bae Park, and Sang-Jo Lee. 2012. A cost sensitive part-of-speech tagging: Differentiating serious errors from minor errors. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, Jeju Island, Korea, pages 1025–1034.

Wee Meng Soon, Hwee Tou Ng, and Daniel Chung Yong Lim. 2001. A machine learning approach to coreference resolution of noun phrases. *Computational Linguistics* 27(4):521–544. https://doi.org/10.1162/089120101753342653.

Robyn Speer and Catherine Havasi. 2012. Representing general relational knowledge in ConceptNet 5. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*. European Language Resources Association (ELRA), Istanbul, Turkey, pages 3679–3686.

Pontus Stenetorp, Sampo Pyysalo, Goran Topić, Tomoko Ohta, Sophia Ananiadou, and Jun'ichi Tsujii. 2012. brat: a web-based tool for NLP-assisted text annotation. In *Proceedings of the Demonstrations at the 13th Conference of the European Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics, Avignon, France, pages 102–107.

Veselin Stoyanov and Jason Eisner. 2012. Easy-first coreference resolution. In *Proceedings of COLING 2012*. The COLING 2012 Organizing Committee, Mumbai, India, pages 2519–2534.

Carolyn Strange, Daniel Mcnamara, Josh Wodak, and Ian Wood. 2014. Mining for the Meanings of a Murder : The Impact of OCR Quality on the Use of Digitized Historical Newspapers. *Digital Humanities Quarterly* 8(1):1–17.

E. Strickland. 2019. Ibm watson, heal thyself: How ibm overpromised and underdelivered on ai health care. *IEEE Spectrum* 56(4):24–31. https://doi.org/10.1109/MSPEC.2019.8678513.

Mihai Surdeanu, Richard Johansson, Adam Meyers, Lluís Màrquez, and Joakim Nivre. 2008. The CoNLL 2008 shared task on joint parsing of syntactic

and semantic dependencies. In *CoNLL 2008: Proceedings of the Twelfth Conference on Computational Natural Language Learning*. Coling 2008 Organizing Committee, Manchester, England, pages 159–177.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. In Zoubin Ghahramani, Max Welling, Corinna Cortes, Neil D. Lawrence, and Kilian Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*. pages 3104–3112.

Richard S. Sutton, David Mcallester, Satinder Singh, and Yishay Mansour. 1999. Policy Gradient Methods for Reinforcement Learning with Function Approximation. In *NIPS 1999*. pages 1057–1063. https://doi.org/10.1.1.37.9714.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian J. Goodfellow, and Rob Fergus. 2014. Intriguing properties of neural networks. In Yoshua Bengio and Yann LeCun, editors, *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*.

Oscar Täckström, Kuzman Ganchev, and Dipanjan Das. 2015. Efficient inference and structured learning for semantic role labeling. *Transactions of the Association for Computational Linguistics* 3:29–41. https://doi.org/10.1162/tacl_a_00120.

Kaveh Taghipour and Hwee Tou Ng. 2015. One million sense-tagged instances for word sense disambiguation and induction. In *Proceedings of the Nineteenth Conference on Computational Natural Language Learning*. Association for Computational Linguistics, pages 338–344. https://doi.org/10.18653/v1/K15-1037.

Thomas Tanay and Lewis Griffin. 2016. A Boundary Tilting Persepective on the Phenomenon of Adversarial Examples pages 1–20. http://arxiv.org/abs/1608.07690.

Ian Tenney, Dipanjan Das, and Ellie Pavlick. 2019. BERT rediscovers the classical NLP pipeline. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, Florence, Italy, pages 4593–4601. https://doi.org/10.18653/v1/P19-1452.

Erik F. Tjong Kim Sang. 2002. Introduction to the CoNLL-2002 shared task: Language-independent named entity recognition. In *COLING-02: The 6th Conference on Natural Language Learning 2002 (CoNLL-2002)*.

Sara Tonelli and Rodolfo Delmonte. 2010. VENSES++: Adapting a deep semantic processing system to the identification of null instantiations. In *Proceedings of the 5th International Workshop on Semantic Evaluation*. Association for Computational Linguistics, Uppsala, Sweden, pages 296–299.

Kristina Toutanova, Aria Haghighi, and Christopher D. Manning. 2008. A global joint model for semantic role labeling. *Computational Linguistics* 34(2):161–191. https://doi.org/10.1162/coli.2008.34.2.161.

Stephen Tratz and Eduard Hovy. 2011. A fast, accurate, non-projective, semantically-enriched parser. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Edinburgh, Scotland, UK., pages 1257–1268.

Rocco Tripodi and Marcello Pelillo. 2017. A game-theoretic approach to word sense disambiguation. *Computational Linguistics* 43(1):31–70. https://doi.org/10.1162/COLI_a_00242.

Tim Van de Cruys. 2014. A neural network approach to selectional preference acquisition. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, Doha, Qatar, pages 26–35. https://doi.org/10.3115/v1/D14-1004.

Teun Adrianus Van Dijk, Walter Kintsch, and Teun Adrianus Van Dijk. 1983. Strategies of discourse comprehension .

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*. pages 5998–6008.

Ashish Venugopal, Andreas Zollmann, Noah A Smith, and Stephan Vogel. 2008. Wider pipelines: N-best alignments and parses in MT training. In *Proceedings of AMTA*. Citeseer, pages 192–201.

Yannick Versley, Alessandro Moschitti, Massimo Poesio, and Xiaofeng Yang. 2008. Coreference systems based on kernels methods. In *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008)*. Coling 2008 Organizing Committee, Manchester, UK, pages 961–968.

Marc Vilain, John Burger, John Aberdeen, Dennis Connolly, and Lynette Hirschman. 1995. A model-theoretic coreference scoring scheme. In *Sixth Message Understanding Conference (MUC-6): Proceedings of a Conference Held in Columbia, Maryland, November 6-8, 1995*.

Daniel Walker, William B. Lund, and Eric K. Ringger. 2010. Evaluating models of latent document semantics in the presence of OCR errors. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Cambridge, MA, pages 240–250.

Peiqi Wang, Xinfeng Xie, Lei Deng, Guoqi Li, Dongsheng Wang, and Yuan Xie. 2018. Hitnet: Hybrid ternary recurrent neural network. In Samy Bengio, Hanna M. Wallach, Hugo Larochelle, Kristen Grauman, Nicolò Cesa-Bianchi, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*. pages 602–612.

Rebecca Watson. 2006. Part-of-speech Tagging Models for Parsing. *Proceedings of the Computaional Linguistics in the UK Conference (CLUK-06)* .

Kellie Webster, Marta Recasens, Vera Axelrod, and Jason Baldridge. 2018. Mind the GAP: A balanced corpus of gendered ambiguous pronouns. *Transactions of the Association for Computational Linguistics* 6:605–617. https://doi.org/10.1162/tacl_a_00240.

Ralph Weischedel and Ada Brunstein. 2005. BBN pronoun coreference and entity type corpus. *Linguistic Data Consortium, Philadelphia* 112.

Ralph Weischedel, Martha Palmer, Mitchell Marcus, Eduard Hovy, Sameer Pradhan, Lance Ramshaw, Nianwen Xue, Ann Taylor, Jeff Kaufman, Michelle Franchini, Mohammed El-Bachouti, Robert Belvin, and Ann Houston. 2013a. OntoNotes Release 5.0 LDC2013T19. *Linguistic Data Consortium* .

Ralph Weischedel, Martha Palmer, Mitchell Marcus, Eduard Hovy, Sameer Pradhan, Lance Ramshaw, Nianwen Xue, Ann Taylor, Jeff Kaufman,

Michelle Franchini, Mohammed El-Bachouti, Robert Belvin, and Ann Houston. 2013b. OntoNotes Release 5.0 LDC2013T19 .

David Weiss, Chris Alberti, Michael Collins, and Slav Petrov. 2015. Structured training for neural network transition-based parsing. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. Association for Computational Linguistics, Beijing, China, pages 323–333. https://doi.org/10.3115/v1/P15-1032.

Dirk Weissenborn, Leonhard Hennig, Feiyu Xu, and Hans Uszkoreit. 2015. Multi-objective optimization for the joint disambiguation of nouns and named entities. In *ACL 2015*. Association for Computational Linguistics, pages 596–605. https://doi.org/10.3115/v1/P15-1058.

Ronald J. Williams. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning* 8(3-4):229–256. https://doi.org/10.1007/BF00992696.

Tongshuang Wu, Marco Tulio Ribeiro, Jeffrey Heer, and Daniel Weld. 2019a. Errudite: Scalable, reproducible, and testable error analysis. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, Florence, Italy, pages 747–763. https://doi.org/10.18653/v1/P19-1073.

Wei Wu, Fei Wang, Arianna Yuan, Fei Wu, and Jiwei Li. 2019b. Coreference Resolution as Query-based Span Prediction. *arXiv preprint* http://arxiv.org/abs/1911.01746.

Zhibiao Wu and Martha Palmer. 1994. Verbs semantics and lexical selection. In *Proceedings of the 32nd annual meeting on Association for Computational Linguistics*. Association for Computational Linguistics, pages 133–138.

Jun Xie, Chao Ma, Janardhan Rao Doppa, Prashanth Mannem, Xiaoli Z. Fern, Thomas G. Dietterich, and Prasad Tadepalli. 2015. Learning greedy policies for the easy-first framework. In Blai Bonet and Sven Koenig, editors, *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, January 25-30, 2015, Austin, Texas, USA*. AAAI Press, pages 2339–2345.

Hiroyasu Yamada and Yuji Matsumoto. 2003. Statistical dependency analysis with support vector machines. In *Proceedings of the Eighth International Conference on Parsing Technologies*. Nancy, France, pages 195–206.

Bishan Yang and Claire Cardie. 2013. Joint inference for fine-grained opinion extraction. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, Sofia, Bulgaria, pages 1640–1649.

David Yarowsky. 1994. Decision lists for lexical ambiguity resolution: Application to accent restoration in Spanish and French. In *32nd Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, Las Cruces, New Mexico, USA, pages 88–95. https://doi.org/10.3115/981732.981745.

Wen-tau Yih and Vahed Qazvinian. 2012. Measuring word relatedness using heterogeneous vector space models. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, pages 616–620.

Kazuhiro Yoshida, Yoshimasa Tsuruoka, Yusuke Miyao, and Jun'ichi Tsujii. 2007. Ambiguous part-of-speech tagging for improving accuracy and domain portability of syntactic parsers. *IJCAI International Joint Conference on Artificial Intelligence* pages 1783–1788.

Dayu Yuan, Ryan Doherty, Julian Richardson, Colin Evans, and Eric Altendorf. 2016a. Semi-supervised Word Sense Disambiguation with Neural Language Models. In *COLING 2016: Technical Papers*. The COLING 2016 Organizing Committee, Osaka, Japan, pages 1374–1385.

Dayu Yuan, Julian Richardson, Ryan Doherty, Colin Evans, and Eric Altendorf. 2016b. Semi-supervised word sense disambiguation with neural models. In *COLING 2016: Technical Papers*. The COLING 2016 Organizing Committee, pages 1374–1385.

Beñat Zapirain, Eneko Agirre, Lluís Màrquez, and Mihai Surdeanu. 2013. Selectional preferences for semantic role classification. *Computational Linguistics* 39(3):631–663. https://doi.org/10.1162/COLI_a_00145.

Lidan Zhang and Kwok Ping Chan. 2009. Dependency parsing with energy-based reinforcement learning. In *Proceedings of the 11th International Conference on Parsing Technologies (IWPT'09)*. Association for Computational Linguistics, Paris, France, pages 234–237.

Wei Emma Zhang, Quan Z. Sheng, Ahoud Alhazmi, and Chenliang Li. 2020. Adversarial Attacks on Deep-learning Models in Natural Language Process-

ing. *ACM Transactions on Intelligent Systems and Technology* 11(3):1–41. https://doi.org/10.1145/3374217.

Xiang Zhang, Junbo Jake Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In Corinna Cortes, Neil D. Lawrence, Daniel D. Lee, Masashi Sugiyama, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*. pages 649–657.

Yue Zhang and Stephen Clark. 2008. A Tale of Two Parsers : investigating and combining graph-based and transition-based dependency parsing using beam-search. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. October, pages 562–571.

Yue Zhang and Joakim Nivre. 2011. Transition-based dependency parsing with rich non-local features. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, Portland, Oregon, USA, pages 188–193.

Yue Zhang and Joakim Nivre. 2012. Analyzing the effect of global learning and beam-search on transition-based dependency parsing. In *Proceedings of COLING 2012: Posters*. The COLING 2012 Organizing Committee, Mumbai, India, pages 1391–1400.

Jieyu Zhao, Tianlu Wang, Mark Yatskar, Vicente Ordonez, and Kai-Wei Chang. 2018. Gender bias in coreference resolution: Evaluation and debiasing methods. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*. Association for Computational Linguistics, New Orleans, Louisiana, pages 15–20. https://doi.org/10.18653/v1/N18-2003.

Valentin Zhikov, Georgi Georgiev, Kiril Simov, and Petya Osenova. 2013. Combining POS tagging, dependency parsing and coreferential resolution for Bulgarian. In *Proceedings of the International Conference Recent Advances in Natural Language Processing RANLP 2013*. INCOMA Ltd. Shoumen, BULGARIA, Hissar, Bulgaria, pages 755–762.

Zhi Zhong and Hwee Tou Ng. 2010. It Makes Sense: A wide-coverage word sense disambiguation system for free text. In *ACL 2010: System*

Demonstrations. Association for Computational Linguistics, pages 78–83. http://www.aclweb.org/anthology/P10-4014.

Guangyou Zhou, Jun Zhao, Kang Liu, and Li Cai. 2011. Exploiting web-derived selectional preference to improve statistical dependency parsing. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, Portland, Oregon, USA, pages 1556–1565.

Guo-Dong Zhou. 2011. Learning noun phrase anaphoricity in coreference resolution via label propagation. *Journal of Computer Science and Technology* 26(1):34–44.

Jie Zhou and Wei Xu. 2015. End-to-end learning of semantic role labeling using recurrent neural networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. Association for Computational Linguistics, Beijing, China, pages 1127–1137. https://doi.org/10.3115/v1/P15-1109.

Xiaojin Zhu and Z Ghahramani. 2002. Learning from labeled and unlabeled data with label propagation. *CMU CALD tech report* (CMU-CALD-02-107). https://doi.org/10.1109/IJCNN.2002.1007592.

Index

Accumulations of errors, 14
Adaptation, 20
Adversarial example, 18
Alleviation strategy, 20

Cascading errors, 14
Compounding errors, 14
Coreference resolution, 32
Correction, 21

Determinstic error propagation, 16

Easy-first, 23
End-to-end learning, 27
Error propagation, 14

Filler, 39
Frame, 39

Global inference, 27
Graph-based processing, 24

Hard error propagation, 16

Implicit semantic role labeling, 39

Jackknifing, 29
Joint inference, 23
Joint processing, 23

Meaning conflation deficiency, 22
Mention, 32
Mitigation, 20

Ontonotes, 159
Ordering accuracy, 48

Post-editing, 30
Predicate, 39
Probabilistic error propagation, 16

Reranking, 31
Role, 39

Selectional preference, 40
Semantic role labeling, 39
Soft error propagation, 16
Stacked sequential learning, 29

Training/test mismatch, 29
Transition-based parsing, 37
Transition-based processing, 25

Word embeddings, 21
Word sense disambiguation, 36
Word2vec, 21
World knowledge, 32