

Robust Part-of-Speech Tagging of Social Media Text

Von der Fakultät für Ingenieurwissenschaften,
Abteilung Informatik und Angewandte Kognitionswissenschaft
der Universität Duisburg-Essen
zur Erlangung des akademischen Grades

Doktor der Ingenieurwissenschaften

genehmigte Dissertation

von

Tobias HORSMANN
aus
Köln

1. Gutachter: Prof. Dr. Torsten Zesch
 2. Gutachter: Prof. Dr. Barbara Plank
- Tag der mündlichen Prüfung: 17. April 2018

Abstract

Part-of-speech (PoS) taggers are an important processing component in many Natural Language Processing (NLP) applications, which led to a variety of taggers for tackling this task. Recent work in this field showed that tagging accuracy on informal text domains is poor in comparison to formal text domains. In particular, social media text, which is inherently different from formal standard text, leads to a drastically increased error rate. These arising challenges originate in a lack of robustness of taggers towards domain transfers. This increased error rate has an impact on NLP applications that depend on PoS information.

The main contribution of this thesis is the exploration of the concept of robustness under the following three aspects: (i) domain robustness, (ii) language robustness and (iii) long tail robustness. Regarding (i), we start with an analysis of the phenomena found in informal text that make tagging this kind of text challenging. Furthermore, we conduct a comprehensive robustness comparison of many commonly used taggers for English and German by evaluating them on the text of several text domains. We find that the tagging of informal text is poorly supported by available taggers. A review and analysis of currently used methods to adapt taggers to informal text showed that these methods improve tagging accuracy but offer no satisfactory solution. We propose an alternative tagging approach that reaches an increased multi-domain tagging robustness. This approach is based on tagging in two steps. The first step tags on a coarse-grained level and the second step refines the tags to the fine-grained tags. Regarding (ii), we investigate whether each language requires a language-tailored PoS tagger or if the construction of a competitive language independent tagger is feasible. We explore the technical details that contribute to a tagger's language robustness by comparing taggers based on different algorithms to learn models of 21 languages. We find that language robustness is a less severe issue and that the impact of the tagger choice depends more on the granularity of the tagset that shall be learned than on the language. Regarding (iii), we investigate methods to improve tagging of infrequent phenomena of which no sufficient amount of annotated training data is available, which is a common challenge in the social media domain. We propose a new method to overcome this lack of data that offers an inexpensive way of producing more training data. In a field study, we show that the quality of the produced data suffices to train tagger models that can recognize these under-represented phenomena.

Furthermore, we present two software tools, FlexTag and DeepTC, which we developed in the course of this thesis. These tools provide the necessary flexibility for conducting all the experiments in this thesis and ensure their reproducibility.

Zusammenfassung

Part-of-Speech (PoS) tagging (Wortklassenerkennung) ist ein wichtiger Verarbeitungsschritt in vielen sprachverarbeitenden Anwendungen. Heute gibt es daher viele PoS Tagger, die diese wichtige Aufgabe automatisiert erledigen. Es hat sich gezeigt, dass PoS tagging auf informellen Texten oft nur mit unzureichender Genauigkeit möglich ist. Insbesondere Texte aus sozialen Medien sind eine große Herausforderung. Die erhöhte Fehlerrate, welche auf mangelnde Robustheit zurückgeführt werden kann, hat schwere Folgen für Anwendungen die auf PoS Informationen angewiesen sind.

Diese Arbeit untersucht daher Tagger-Robustheit unter den drei Gesichtspunkten der (i) Domänenrobustheit, (ii) Sprachrobustheit und (iii) Robustheit gegenüber seltenen linguistischen Phänomene. Für (i) beginnen wir mit einer Analyse der Phänomene, die in informellen Texten häufig anzutreffen sind, aber in formalen Texten nur selten bis gar keine Verwendung finden. Damit schaffen wir einen Überblick über die Art der Phänomene die das Tagging von informellen Texten so schwierig machen. Wir evaluieren viele der üblicherweise benutzen Tagger für die englische und deutsche Sprache auf Texten aus verschiedenen Domänen, um einen umfassenden Überblick über die derzeitige Robustheit der verfügbaren Tagger zu bieten. Die Untersuchung ergab im Wesentlichen, dass alle Tagger auf informellen Texten große Schwächen zeigen. Methoden, um die Robustheit für domänenübergreifendes Tagging zu verbessern, sind prinzipiell hilfreich, lösen aber das grundlegende Robustheitsproblem nicht. Als neuen Lösungsansatz stellen wir Tagging in zwei Schritten vor, welches eine erhöhte Robustheit gegenüber domänenübergreifenden Tagging bietet. Im ersten Schritt wird nur grob-granular getaggt und im zweiten Schritt wird dieses Tagging dann auf das fein-granulare Level verfeinert. Für (ii) untersuchen wir Sprachrobustheit und ob jede Sprache einen zugeschnittenen Tagger benötigt, oder ob es möglich ist einen sprach-unabhängigen Tagger zu konstruieren, der für mehrere Sprachen funktioniert. Dazu vergleichen wir Tagger basierend auf verschiedenen Algorithmen auf 21 Sprachen und analysieren die notwendigen technischen Eigenschaften für einen Tagger, der auf mehreren Sprachen akkurate Modelle lernen kann. Die Untersuchung ergibt, dass Sprachrobustheit an für sich kein schwerwiegendes Problem ist und, dass die Tagsetgröße des Trainingskorpus ein wesentlich stärkerer Einflussfaktor für die Eignung eines Taggers ist als die Zugehörigkeit zu einer gewissen Sprache. Bezuglich (iii) untersuchen wir, wie man mit seltenen Phänomenen umgehen kann, für die nicht genug Trainingsdaten verfügbar sind. Dazu stellen wir eine neue kostengünstige Methode vor, die nur einen minimalen Aufwand an manueller Annotation erwartet, um zusätzliche Daten für solche seltenen Phänomene zu produzieren. Ein Feldversuch hat gezeigt, dass die produzierten Daten ausreichen um das Tagging von seltenen Phänomenen deutlich zu verbessern.

Abschließend präsentieren wir zwei Software-Werkzeuge, FlexTag und DeepTC, die wir im Rahmen dieser Arbeit entwickelt haben. Diese Werkzeuge bieten die notwendige Flexibilität und Reproduzierbarkeit für die Experimente in dieser Arbeit.

Contents

Abstract	iii
Zusammenfassung	v
1 Introduction	1
1.1 Main Contributions	3
1.2 Organization of Thesis	4
2 Theoretical Background	7
2.1 Part-of-Speech Tagging	7
2.2 Tagsets	22
2.3 Evaluation Metrics	25
2.4 Robustness	26
2.5 Chapter Conclusion	32
3 Domain Robustness - Challenges	33
3.1 The Social Media Domain	33
3.2 Theoretical Challenges	34
3.3 Practical Challenges	39
3.4 Chapter Conclusion	48
4 Domain Robustness - Existing Approaches	51
4.1 More Data	54
4.2 More Knowledge	57
4.3 Combining Approaches	61
4.4 Transferability to Other Languages	63
4.5 Chapter Conclusion	72
5 Domain Robustness - Two-Step Tagging	73
5.1 Potential of Coarse-grained Tagging	74
5.2 Coarse-grained Cross-Domain Robustness	76
5.3 Tagging in Two-steps	81
5.4 Chapter Conclusion	86
6 Language Robustness	89
6.1 Corpora	91
6.2 Tagger Implementations	92

6.3	Direct Comparison of Taggers	97
6.4	Comparison to State-of-the-art Taggers	97
6.5	Chapter Conclusion	99
7	Long Tail Robustness	101
7.1	Fitting Towards a Phenomenon	102
7.2	Generalization of the Approach	110
7.3	Chapter Conclusion	117
8	Technical Prerequisites	119
8.1	FlexTag – A Flexible PoS Tagger	119
8.2	Extending DKPro Text Classification	123
8.3	Chapter Conclusion	129
9	Conclusion	131
9.1	Summary	131
9.2	Limitations and Outlook	133
9.3	Closing Remarks	134
A	Language Robustness - Results per Corpus	135

List of Figures

1.1 Example of a PoS tagged sentence	1
1.2 Social media posting with non-standard language use	2
1.3 Robustness scenarios	3
2.1 Example of a Part-of-Speech tagged sentence	7
2.2 PoS tagger model training process with prediction	9
2.3 Example of generalizing, over- and underfitting classifier	10
2.4 Categorization of PoS tagging related concepts	11
2.5 Support Vector Machine example	16
2.6 Trigram Hidden Markov Model as directed graph	17
2.7 Conditional Random Field as undirected graph	18
2.8 Basic neural network with one hidden layer	19
2.9 Recurrent neural network	20
2.10 Structure of a plain LSTM cell	21
2.11 Example of words in embedding space	22
2.12 Example of mapping language dependent tags to the Universal tagset	25
3.1 Tagging a non-standard Twitter message with the Stanford tagger . .	35
3.2 Comparison of PoS distribution between text domains	38
3.3 Type/token ratio for the written, social and the spoken domain . . .	39
3.4 Domain robustness	40
3.5 Overall macro-averaged results for each English and German model .	45
3.6 Results of the English models per text domain	46
3.7 Results of the German models per text domain	47
4.1 PoS tagging by normalization versus domain adaptation	52
4.2 Overview of domain adaptation approaches	53
4.3 Re-training learning curve	55
4.4 Results of mixed re-training	56
4.5 Results of oversampling	56
4.6 Voting vs. mixed-retraining	58
4.7 Results of using PoS dictionaries	59
4.8 Enhancing tagging by cluster knowledge	59
4.9 Improvements by using Brown and LDA clusters	60
4.10 Results for each approach on the English dataset	62
4.11 Results for each approach on the German dataset	66

4.12 Results for each approach on the Italian dataset	70
5.1 PoS tagging in two steps	74
5.2 Learning curves for the <i>News</i> and the <i>Twitter</i> dataset	77
5.3 Cross-domain learning curves for <i>News</i> dataset vs. <i>Twitter</i> dataset . .	78
5.4 Implementation of two-step tagging	81
5.5 Two stacked CRF taggers as baseline	82
5.6 Decreasing fine-grained accuracy by error propagation	84
6.1 Evaluation for language robustness	89
6.2 LSTM architectures in our replication setup	92
6.3 Results of LSTM architectures per language group	93
6.4 Averaged results of CRF feature set parametrizations	95
6.5 Effect of tagset size on accuracy for a HMM, CRF and LSTM tagger .	96
6.6 Results of a HMM, CRF and LSTM tagger on multilingual corpora .	97
6.7 Reproduction of best results for selected languages	98
7.1 Long tail robustness	101
7.2 Producing more training data of under-resourced phenomena	104
7.3 Results of tagging OOV instances of VVPPER	107
7.4 Evaluation of tagging VVPPER instances in plain text	110
7.5 Evaluation of tagging ADVART instances in plain text	113
7.6 Evaluation of tagging APPRART instances in plain text	116
8.1 Flexibility levels of PoS taggers	121
8.2 Feature extractor that detects user mentions in Twitter	122
8.3 Processing schema for experiments in DKPro TC	125
8.4 Vectorization N-to-1, N-to-M and N-to-N	127
A.1 Results of CRF feature set parameterizations on multi-lingual corpora	136
A.2 Results of LSTM architectures on multilingual corpora	137
A.3 Results of a HMM, CRF and LSTM tagger on multilingual corpora .	139

List of Tables

2.1	Confusion matrix example	26
2.2	Calculation example of Precision, Recall and F-Score	26
3.1	Examples of non-standard language use in social media	36
3.2	English and German tagger models that are evaluated for robustness .	42
3.3	Corpora selection to evaluate domain transfer robustness of models .	44
3.4	Accuracy and execution time per text domain of the English models .	47
3.5	Accuracy and execution time per text domain of the German models .	47
3.6	Examples of the English and German spoken corpora	48
4.2	Frequency of the newly introduced tags in the German dataset	64
4.3	Results on German test data subsets per approach	67
4.4	Official results of the German Empirikom shared task	67
4.5	Results on the German test data per tag for each approach	68
4.6	Results on the Italian test data per tag for each approach	71
4.7	Official results of the Italian shared task	72
5.1	Tagging accuracy for fine-mapped and coarse-grained tagging	76
5.2	Cross-domain tagging accuracy of fine- and coarse-grained models .	80
5.3	F-Score for coarse-grained CRF taggers	81
5.4	Results of tagging in two steps	83
5.5	F-Score of fine-grained tags for two-step tagging under oracle condition	85
6.1	The replication setup	90
6.2	Multilingual corpora collection for comparing PoS taggers robustness .	91
7.1	Example of full-verb with personal pronoun contractions (VVPPER) .	103
7.2	F-Score results on VVPPER instances	106
7.3	Improvements of the contextualised two-step tagging on VVPPER .	108
7.4	Twitter postings that have been tagged as VVPPER	111
7.5	Example of adverb with article contractions (ADVART)	112
7.6	F-Score results on ADVART instances	112
7.7	Twitter postings that have been tagged as ADVART	114
7.8	Example of preposition with article contractions (APPRART)	115
7.9	F-Score results on APPRART instances	115
7.10	Twitter postings that have been tagged as APPRART	117

A.1 Accuracy of CRF tagger configurations on multi-lingual corpora	135
A.2 Accuracy of LSTM taggers on multi-lingual corpora	138

Chapter 1

Introduction

The increasing amount of digitalization in daily life has led to a ubiquitous exposure to information. A large part of this information is consumed as text, e.g. we read for instance news articles and product reviews, and communicate with friends on social media websites. Humans are intuitively able to find information on involved actors and their actions. The sentence construction and the way in which nouns, verbs, adverbs or adjectives occur allow humans to understand a text. The word classes or the parts of speech (PoS) are thus an information of essential importance to text comprehension.

PoS taggers are tools that belong to the group of Natural Language Processing (NLP) applications and automatically perform the task of PoS recognition. Taggers are used standalone, for instance by linguists, but also as component in more complex NLP setups. In many processing setups, knowing the word's PoS is similarly valuable as to humans. An example of a PoS tagged sentence is shown in Figure 1.1. Many words can occur with more than just one PoS. The example contains, for instance, an ambiguity of the word "can" that was successfully resolved, the first time as a *verb* and the second time as a *noun*. Resolving such ambiguities requires analyzing the surrounding word context; for instance, the article *the* before the second "can" is a strong evidence that this occurrence of "can" is a noun.

The usefulness of PoS information depends on its quality, i.e. correct tagging. In the worst case, poor tagging results degenerates to a point at which the application is rendered inoperable. Tagging results on English and German usually reach an accuracy of 97% (Manning, 2011; Giesbrecht and Evert, 2009), which creates the impression that tagging performs satisfactorily. However, such accurate tagging results are only achievable on highly formal text such as newswire. When tagging less formal text domains, for instance social media, the same taggers reach only between 70%

Can	you	open	the	can	for	me	?
VERB	PRON	VERB	DET	NOUN	PREP	PRON	PUNCT

FIGURE 1.1: Example of a PoS tagged sentence in which a label is assigned to each word that specifies the syntactical function of the word in the sentence



FIGURE 1.2: Social media posting with non-standard language use

to 80% accuracy (Beißwenger et al., 2016; Ritter et al., 2011). Such a severe drop in accuracy is certainly surprising but also indicates a lack of robustness of these taggers against text domain shifts.

To improve the understanding of the peculiarities of social media, Figure 1.2 shows an example message taken from the social media platform Twitter. The key differences to formal text are for instance word contractions, repetition of characters, slang expressions, abbreviations and hashtags that can be composed of multiple words (Eisenstein, 2013; Baldwin et al., 2013). Such phenomena are not part of the standard language and usually do not occur in formal text. However, formal text, i.e. newswire text, is most commonly used as training data for these taggers. Thus, taggers trained on formal text perform well on formal text but not on less formal text. The preferred use of formal text for model training might explain this lacking robustness. Solving this robustness challenge by simply training a new model on social media data is not possible, either. There are only a few PoS annotated social media corpora available that are too small in size for model training.

Hence, in this thesis, we investigate the robustness of PoS taggers. In Figure 1.3, we show the three robustness scenarios on which we focus. *Domain robustness* investigates the robustness problem that we described above. We evaluate English and German taggers on same domain text as the training data and on foreign domain text. In a same domain evaluation, a tagger model trained on newswire text is applied to text coming from the same domain as the model’s training data, for instance another newswire corpus. In this case, the model will barely encounter any unknown linguistic phenomena because the model has been trained on this domain. Thus, an accurate tagging should be possible. In a foreign domain evaluation, the model is applied to text from a different text domain, for instance social media. The model will encounter linguistic phenomena that did not occur in the newswire training data. Depending on the number and frequency of unknown linguistic phenomena, the model will be challenged to assign the correct tag. This provides a more complete picture of the

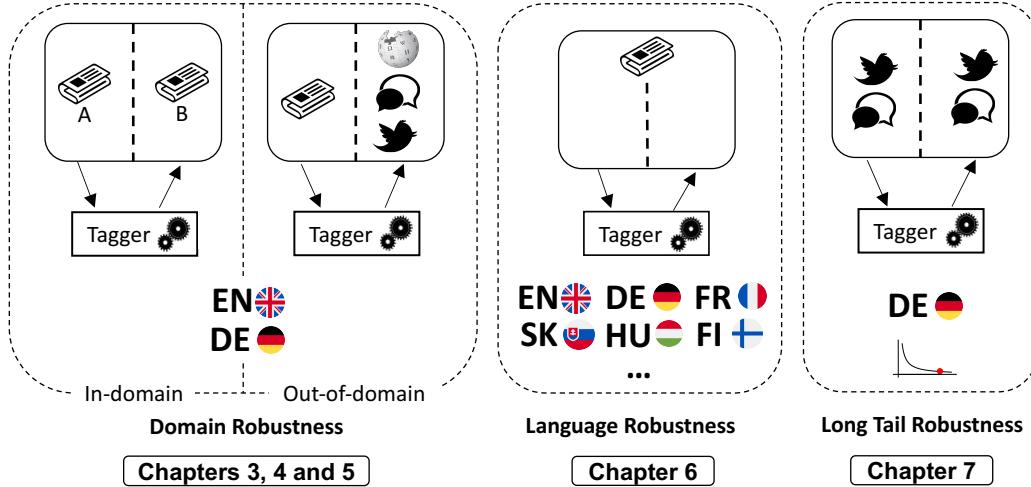


FIGURE 1.3: Robustness scenarios

robustness currently available taggers offer. Furthermore, we explore common strategies to improve robustness of models, in particular for the social media domain, and additionally propose a new tagging method to increase general cross-domain tagging robustness. *Language robustness* investigates if tagger implementations are able to learn and predict the PoS for more than just one language. More specifically, we investigate if each language requires a language-tailored tagger implementation or if it is possible to construct a tagger that works equally well for a variety of languages. *Long tail robustness* focuses on dealing with sparseness of training samples when training a model. On low-resourced domains, such as the social media domain, many phenomena are often under-represented in the nevertheless small training datasets. This prevents robust tagging of these phenomena due to their infrequency. We investigate methods of dealing with these rare phenomena.

1.1 Main Contributions

The main contributions of this thesis are summarized as follows:

Domain Robustness: We provide a comprehensive overview over the offered robustness of currently available off-the-shelf tagger models by evaluating them on three different text domains. We discuss existing strategies to improve tagger model robustness and analyze them in detail to learn which word classes (or tags) they improve. We analyze the effect of each strategy alone and in combination and confirm their effectiveness on three languages, namely English, German and Italian. Furthermore, we introduce a new tagging approach that is based on tagging in two steps that achieves a better multi-domain robustness than the baseline taggers.

Language Robustness: We investigate the requirements for constructing a language-independent tagger. We experiment with different tagger algorithms and architectures to find a setup that can learn accurate models of 21 languages. We also investigate the accuracy trade-off between language-independent and language-fitted taggers by comparing the results to reference values that we find in the literature.

Long Tail Robustness: We propose a new method to fit a PoS tagger towards phenomena that occur only infrequently in the training data. This method requires only a minimal amount of manual annotation and allows an inexpensive production of additional annotated training instances for such infrequent phenomena. We confirm the usefulness of this method by adapting taggers to three infrequent phenomena.

Software Tools: We developed two software tools in the course of this thesis on which we based the conducted experiments. The first one is FlexTag, which is a highly flexible PoS tagger that grants researchers a high flexibility in experimenting with feature configurations. The second one is DeepTC, which provides reproducibility of deep learning experiments by embedding them into an end-to-end shareable environment. Both tools have been publicly released to the researcher community.

1.2 Organization of Thesis

In the remainder of this chapter, we provide an overview of the thesis and the publications in which the chapter content was originally published:

Chapter 2 In this chapter, we introduce the basic concepts of PoS tagging that we will use in this thesis. Furthermore, we discuss the notion of *robustness* for classification tasks in general.

Chapter 3 This chapter deals with the challenges of domain robustness with respect to PoS tagging. Using the social media domain as an example of text domains inherently different from standard text, we start with a theoretical analysis of social media corpora. This provides an overview of distinctive properties found in informal text domains and clarifies the differences to standard language. Furthermore, in an empirical evaluation of many commonly used taggers, we analyze the practical impact of these differences by comparing tagging results within and outside the text domain on which the tagger models have been trained originally.

Published:

Tobias Horsmann, Nicolai Erbs, and Torsten Zesch (2015). Fast or Accurate? - A Comparative Evaluation of PoS Tagging Models. In: *Proceedings of the International Conference of the German Society for Computational Linguistics and Language Technology (GSCL)*. Essen, Germany, pp. 22 – 30.

Chapter 4 In this chapter, we analyze methods that have been proposed in literature to construct a PoS tagger that tags social media text more robustly and more accurately. Each method is evaluated alone and in combination to find the most promising combination. The general applicability of the approaches is confirmed by constructing social media taggers for English, German and Italian.

Published:

Tobias Horsmann and Torsten Zesch (2015). Effectiveness of Domain Adaptation Approaches for Social Media PoS Tagging. In: *Proceeding of the Italian Conference on Computational Linguistics*. Trento, Italy: Academia University Press, pp. 166 – 170.

Tobias Horsmann and Torsten Zesch (2016b). Building a Social Media Adapted PoS Tagger Using FlexTag – A Case Study on Italian Tweets. In: *Fifth Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (EVALITA)*. Naples, Italy, pp. 95 – 98.

Tobias Horsmann and Torsten Zesch (2016c). LTL-UDE @ EmpiriST 2015: Tokenization and PoS Tagging of Social Media Text. In: *Proceedings of the Web as Corpus Workshop (WAC-X) and the EmpiriST Shared Task*. Berlin, Germany, ACL, pp. 120 – 126.

Chapter 5 In this chapter, we introduce a new approach to PoS tagging that is more robust against domain transfers by tagging in two steps. This approach breaks the tagging task down into two smaller sub-problems. In the first step, a coarse-grained tagging is applied that is refined in the second step to fine-grained tags. We conduct an extensive evaluation of this two-step tagging and experiment with different algorithm for its implementation.

Published:

Tobias Horsmann and Torsten Zesch (2016a). Assigning Fine-grained PoS Tags based on High-precision Coarse-grained Tagging. In: *Proceedings of International Conference on Computational Linguistics: Technical Papers (COLING)*. Osaka, Japan: Association for Computational Linguistics, pp. 328 – 336.

Chapter 6 In this chapter, we investigate the multilingual robustness of taggers. We analyze the technical requirements for constructing multilingual taggers using commonly used sequence tagging approaches. We evaluate different tagger configurations on 21 languages and determine technical requirements for implementing a multilingually robust PoS tagger.

Published:

Tobias Horsmann and Torsten Zesch (2017a). Do LSTMs really work

so well for PoS tagging? – A replication study. In: *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Copenhagen, Denmark, pp. 738 – 747.

Chapter 7 In this chapter, we discuss a method to improve the robustness of tagging infrequent phenomena. In the few annotated social media datasets, many interesting phenomena are extremely rare. Models trained on these corpora perform only poorly, especially on PoS tags that occur rarely in the training data. To avoid poor scaling and cost-intensive manual annotation of more data, we present and evaluate an inexpensive and well-scaling method to produce more training instances of a particular linguistic phenomenon.

Published:

Tobias Horsmann, Michael Beißwenger and Torsten Zesch (2017b). Reliable Part-of-Speech Tagging of Low-frequency Phenomena in the Social Media Domain. In: *Proceedings of the Conference on CMC and Social Media Corpora for the Humanities*. Bozen, Italy, pp. 39 – 43.

Michael Beißwenger, **Tobias Horsmann** and Torsten Zesch (2017c). Part-of-Speech Tagging for Corpora of Computer-mediated Communication: A Case Study on Finding Rare Phenomena. In: *Investigating computer-mediated communication: Corpus-based approaches to language in the digital world*. ISBN 978-961-237-950-6. Ljubljana, Slovenia, Ljubljana University Press, pp. 192 – 215.

Chapter 8 In this chapter, we present the two software projects that have been developed in the course of this thesis, FlexTag and DeepTC. The experiments conducted in the previous chapters are based on these two software projects and, hence, are essential tools for the completion of this work. FlexTag is a highly flexible PoS tagger that allows quick and easy experimentation with various feature sets. DeepTC is a Deep Learning extension of DKPro Text Classification (Daxenberger et al., 2014) that improves the reproducibility of Deep Learning experiments.

Published:

Torsten Zesch and **Tobias Horsmann** (2016). FlexTag: A Highly Flexible PoS Tagging Framework. In: *Proceedings of the International Conference on Language Resources and Evaluation (LREC)*. Portorož, Slovenia, ELRA, pp. 4259 – 4263.

Tobias Horsmann and Torsten Zesch (2018). DeepTC – An Extension of DKPro Text Classification for Fostering Reproducibility of Deep Learning Experiments. In: *Proceedings of the International Conference on Language Resources and Evaluation (LREC)*. Miyazaki, Japan, ELRA, pp. (publication pending)

Chapter 2

Theoretical Background

In this chapter, we introduce the theoretical background necessary for understanding the following chapters. Furthermore, we will provide an overview of work related to Part-of-Speech (PoS) tagging.

2.1 Part-of-Speech Tagging

PoS tagging is the process of determining a word's syntactical category, e.g. if a word is a noun, adposition, pronoun, etc. Figure 2.1 shows an example sentence with the part of speeches of the occurring words. The words are annotated with a label, the PoS tag, which describes the syntactic function of the word in the sentence. The assignment of PoS tags is either done manually by humans or in an automatized fashion by tools. To avoid labor- and time-intensive manual annotation, PoS taggers are frequently used to automatically annotate large collections of plain text.

Automatized assignment of PoS tags, i.e. tagging, is a difficult task as language is ambiguous, for instance in Figure 2.1 the word "back" occurs two times: The first time it is an adverb, the second time it is a noun. Language ambiguity is a frequent phenomenon and there are many words that can take on different syntactic functions within in a sentence. Only when considering the surrounding words is a disambiguation possible. In this case, the word *my* that occurs right before the second "back" allows us to disambiguate this word's tag to be a noun.

PoS tags are used by linguists but also for building applications. Linguists might, for instance, use PoS sequences to find instances of interesting syntactic patterns they want to study. Natural Language Processing (NLP) applications that use PoS are, for instance, sentiment classification (Agarwal et al., 2011), machine translation (Ueffing and Ney, 2003) or authorship attribution (Stamatatos, 2009). If automatically-assigned PoS information is used, already small tagging errors might have a huge

I	went	back	because	my	back	hurts	.
PRON	VERB	ADV	PREP	PRON	NOUN	VERB	PUNCT

FIGURE 2.1: Example of a Part-of-Speech tagged sentence

impact on such applications. The accuracy of tagging varies strongly by the kind of text that is being tagged. For instance, the Stanford tagger (Toutanova et al., 2003) reaches 97% accuracy on a newswire corpus (Manning, 2011) but drops to 80% on a Twitter corpus (Ritter et al., 2011). This tagging error propagates through the following processing steps in the NLP application and harm its performance. Thus, due to its fundamental importance, even minor PoS tagging improvements will benefit the overall performance of an application.

In this chapter, we will introduce the concepts used to perform automatic PoS tagging and provide an overview about related work in this field. The existing approaches are most easily distinguished based on the requirement for annotated training data. Approaches that require annotated training data belong to the *supervised approaches*; if they learn from plain text they belong to the *unsupervised approaches*. We will mostly focus on supervised tagging. Therefore, we will discuss first the general procedure of training a supervised model and using it to make predictions before we discuss (mostly supervised) approaches for PoS tagging.

2.1.1 Supervised Machine Learning

In this section, we briefly discuss the procedure of supervised machine learning for the application of training a PoS tagging model and using this model for predicting tags. A PoS tagger compose of two essential components, the tagger itself i.e. the software implementation, and the model which stores the model parametrization i.e. feature weights (more to the weights below). The model is the result of the training process and is used for making PoS tag predictions. Thus, when training a tagger, one actually trains a model for this tagger. Some taggers, for instance the TreeTagger (Schmid, 1994b), offer more than just one model. Offering more than one model provides the flexibility to pick a more suited model for a certain task. Models can differ, for instance, by the used features or training data (more on that further below). It is therefore misleading to say a tagger performs poorly. It would be more accurate to say that a tagger with a certain model performs poorly.

Model Training and Prediction of Tags PoS tagging distinguishes between two phases, training and prediction. In Figure 2.2, we show the basic process of training a supervised PoS tagger model and using it for predicting the tags on plain text. Text annotated with PoS tags is provided to the tagger as training data. This annotated training data is usually created by human expert-annotators, which makes annotated data an extremely expensive resource. This text runs through a *feature extraction* step which extracts an input representation that is provided to the *machine learning algorithm*. The feature extractors are defined by humans and extract properties relevant to solving a particular classification task. The algorithm learns a function that maps certain feature values to a PoS tag. During model training feature weights are learned, i.e. features that receive a high weight are highly discriminative

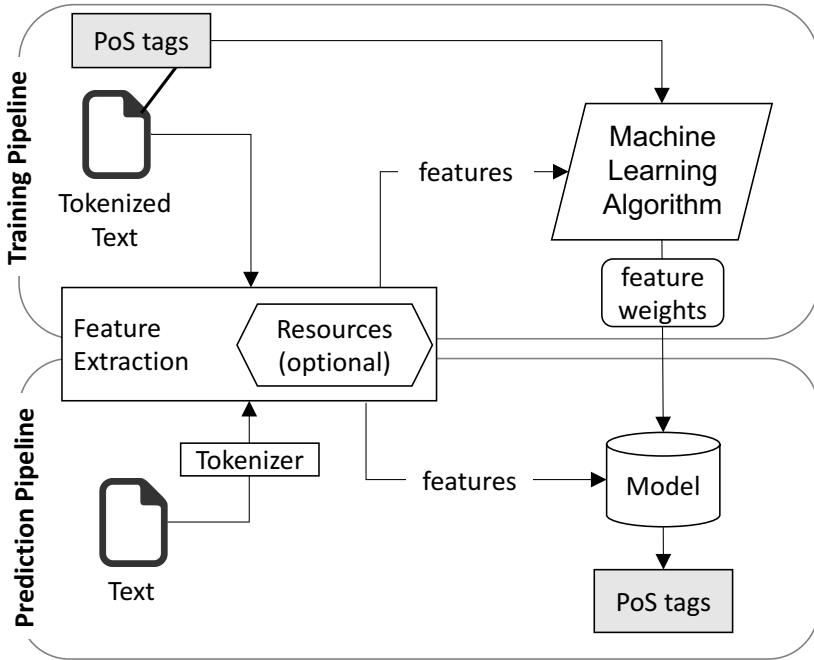


FIGURE 2.2: PoS tagger model training process with prediction

for distinguishing the PoS tags of the training data (if the training data are representative for the problem, the same feature should also be discriminative for unseen data). These learned feature weights are stored in the *model*. The feature extraction step can be enriched by providing additional knowledge from external *resources* that helps to create distinctive features. During a prediction step, assuming plain text as input, the text is first tokenized and then passed through the feature extraction step. It is crucial that the feature extraction step between model training and prediction is exactly the same. Only if the feature values encountered during training are also found in the prediction phase, is a successful tagging possible. The machine learning algorithm uses the trained feature weights in the model and the extracted feature values to predict the PoS tags. If during training a resource was used, that same resource has to be used during prediction as well. In case of Deep Learning neural networks (which we discuss in more detail further below), this feature extraction step is done implicitly by the network itself and requires no human-defined features.

Generalization A key requirement for a trained classifier is its ability to generalize to unseen samples and avoid over- and underfitting. The differences are shown in Figure 2.3, which we briefly discuss. Generalization finds a reasonable decision boundary to distinguish the two classes even if this means allowing some errors. In this case, a good generalization would allow a miss-classified sample (circle). A classifier that *overfits* to the training data might be able to accurately recognize the values in the training data but fitted unreasonably strong to this information. In this case, the classifier overfitted to the triangle samples, this separation achieves a perfect separation of the training samples but will certainly lead to an increased number of errors when

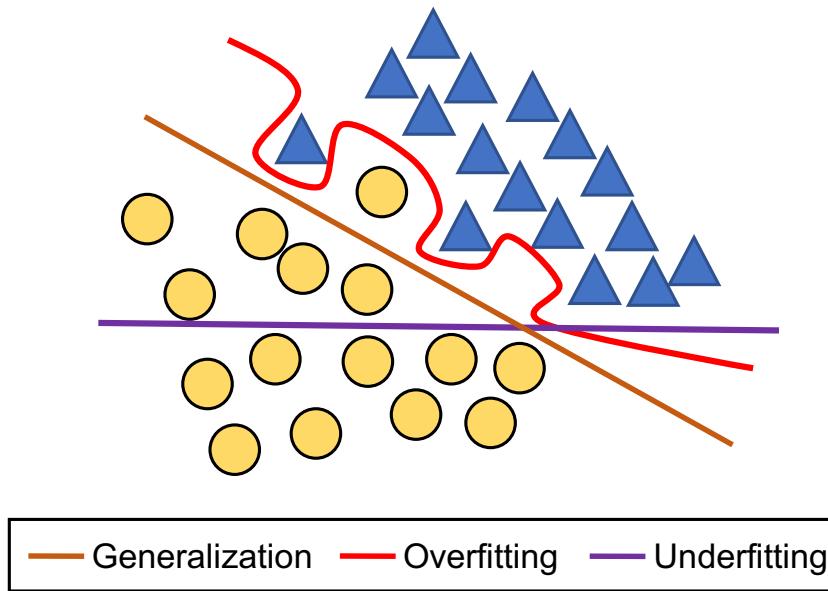


FIGURE 2.3: Example of a generalizing, overfitting and underfitting classifier. The overfitting led to a decision boundary that exactly models the training data distribution, while underfitting does not respect the distribution enough. A generalizing classifier learns a decision boundary that is generally valid even if it has to allow some classification error in the training data (circle) (figure adapted from (Patterson and Gibson, p.27, Figure 1.7))

unseen data are provided to the classifier. A classifier that *underfits* to the training data fails to learn how to distinguish the training data. In practice, overfitting is the more severe problem.

Significance Tests Significance tests are a tool for investigating if the superior performance of one classifier over another one is coincidental or is substantial, i.e. statistically significant. When comparing the raw score between two classifiers, for instance accuracy (discussed further below), one can easily answer the question which of the two classifiers works better. However, the question one tries to answer is actually a different one. The objective lies in answering if the classifier can be considered to be in general superior or not.

A significance test is based on the null hypothesis that two classifiers perform *not differently* or *equivalently*. A statistical test is not concerned with notions of superiority or inferiority but simply measures the *difference* between two results and estimates whether a found difference is due to chance or not. If the difference between two classification results are larger than chance, the null hypothesis is rejected and one can conclude that there is a high likelihood that the difference (i.e. the improvement) can also be found on other datasets with the same properties (Berg-Kirkpatrick et al., 2012). Alternatively, one can say that the test tries to answer the question if the results of both classifiers are drawn from two differ populations (the null hypothesis tests if they are from the same population). If both classifier drew their results from the same population they perform equivalent, even if raw scores might differ to some

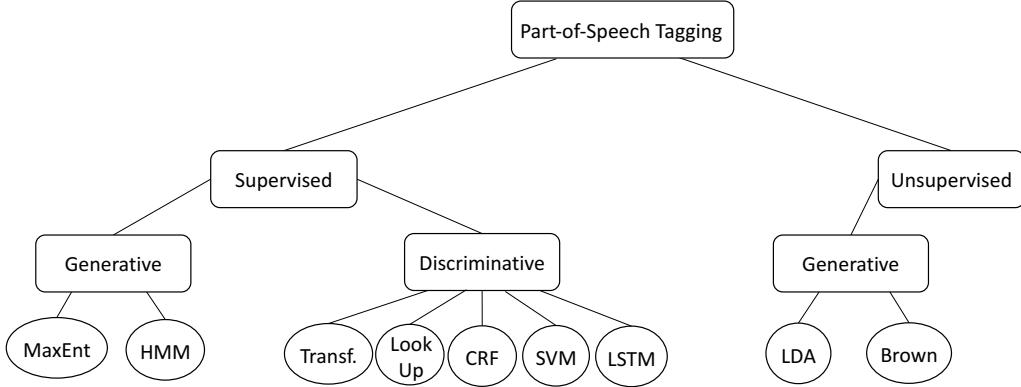


FIGURE 2.4: Categorization of PoS tagging related concepts that are relevant for this thesis into supervised and unsupervised approaches and whether they use a generative or discriminative model. MaxEnt=Maximum Entropy, HMM=Hidden Markov Model, Transf.=Transformation Rule, Lookup=PoS dictionary, CRF=Conditional Random Fields, LSTM=Long-Short-Term-Memory neural network, LDA=Latent Dirichlet Allocation, Brown=Brown Clustering

extent. If a significance is found, the classifiers drew from different populations and, thus, the better performance (in raw score) of one classifier can be expected to hold also on other datasets.

We will use in this thesis the McNemar (McNemar, 1947) test when comparing tagger results for statistically significant improvements. A McNemar test assumes as null hypothesis that two classifiers have the same error rate (Japkowicz and Shah, 2011, pp. 226–228). The test considers only error cases in which either of the two classifiers made a classification error, let A_E be classification errors of classifier A and B_E classification errors of classifier B. The difference between classification results is significant if the obtained X^2 equals or exceeds the values in a chi-square distribution.

$$X^2 = \frac{(|A_E - B_E| - 1)^2}{A_E + B_E} \quad (2.1)$$

An important value when working with significance tests is the p -value. The p -value is the probability for the null hypothesis to be true. Thus, the probability of getting a better result with one classifier without being significantly better. A p -value of 0.05 would mean that 95% of the results are significantly better, i.e. there is only a chance of 5% that classifier A performs like classifier B (null hypothesis), in the remaining cases classifier A is (statistically significantly) better than B.

2.1.2 Generative and Discriminative Models

We categorize the PoS tagging approaches as shown in Figure 2.4. For its mostly statistical nature, PoS tagging is sometimes categorized as using *generative* or *discriminative* model (Bishop, 2011, pp. 196–217). The tagging problem is formulated that for a sequence of words $x = (x_1, \dots, x_n)$ a fitting sequence of PoS tags $y = (y_1, \dots, y_n)$

has to be determined. This is done by finding a function $f : x \rightarrow y$ that maps a word to its tag.

Generative models aim at learning an approximation of the joint probability $p(x, y)$ from the training data. A direct prediction of a tag for a word is not possible. By applying Bayes' rules, the conditional or posterior probability is derived for making a prediction or classifying an instance of x with the most probable tag y :

$$f(x) = \operatorname{argmax}_{y \in Y} p(y|x) = \operatorname{argmax}_{y \in Y} \frac{p(y)p(x|y)}{p(x)} \quad (2.2)$$

with $p(x|y)$ being the probability of generating x given y as label and $p(y)$ being the prior probabilities over labels y . The denominator $p(x)$ can be omitted as constant value which simplifies the equation to:

$$f(x) = \operatorname{argmax}_{y \in Y} p(y)p(x|y) \quad (2.3)$$

The usage of generative models extends beyond the use case of prediction as they try to approximate the *true* label distribution. The modeling of the joint probability allows the model to also *generate* or simulate new data samples, which gives the model its name. In our use case the generative property is irrelevant as we are interested in prediction in this thesis. Thus, the generative model essentially tries to solve a more general problem by starting to estimate the joint distribution and computes the conditional probability based on this approximation (Sutton and McCallum, 2012).

Discriminative models do not try to model the joint distribution $p(x, y)$. Instead, they learn the decision boundary between label classes enabling a direct computation of $p(y|x)$ (Ng and Jordan, 2002). This direct computation of the discriminative model limits its use to prediction tasks i.e. discrimination between label classes. A further difference between the two models is that learning the *generative* model requires necessarily large amounts of training data in order to approximate the *true* distribution $p(x, y)$. The *discriminative* model is less demanding as the focus lies on learning label boundaries and does not try to approximate $p(x, y)$.

Both kinds of models, generative and discriminative, are found in supervised and unsupervised PoS tagging.

2.1.3 Unsupervised Tagging

In unsupervised PoS tagging (Brown et al., 1992; Biemann, 2006; Goldwater and Griffiths, 2007; Christodoulopoulos et al., 2010; Das and Petrov, 2011) no human annotated training data is required. Instead of annotated training data, a similarity function is used that groups words together that are similar according to this similarity function. The underlying idea is to group words together when they tend to occur in similar word contexts. For PoS tagging, the implication is that similar words also share similar syntactical properties i.e. have the same word class. The advantage is the low threshold for using these taggers as plain text is often more easily available than

annotated data. The drawback is that the word groups, i.e. clusters, that are created in the clustering process are difficult to interpret. The word clusters are usually identified by randomly assigned identification numbers. Furthermore, the number of created clusters is often much higher than the number of classes in a human defined tagset (discussed in Section 2.2) that are used in supervised tagging.

A common strategy to include *unsupervised* knowledge in *supervised* tagging is to provide the cluster id as feature information into the model training process (Ritter et al., 2011; Owoputi et al., 2013). In the following paragraphs, we introduce Brown clustering (Brown et al., 1992) and Latent Dirichlet Allocation (LDA) (Blei et al., 2003), which we will use in this thesis in the just described fashion.

Brown Clustering Brown clustering is a hierarchical agglomerative hard clustering algorithm. As a result, words are associated with exactly one cluster. For utilizing Brown clustering for PoS tagging it is important to note again that language is ambiguous and a single word might have more than just one possible PoS. Brown clustering is, hence, not able to account for this ambiguity of words we find in language.

The clustering approach starts by assigning a separate cluster to each word in a corpus. Usually a frequency cut-off is used for reasons of computational efficiency to consider only frequent words. At the beginning, there are as many words as there are clusters. The words in the individual clusters are then iteratively merged so that a function F is learned, which assigns the vocabulary V into k classes, i.e. $F : V \rightarrow \{1, 2, \dots, k\}$. The number of k classes that are being created is a parameter that is set before execution to achieve a runtime behavior that is practically feasible. Finding a suited k for a task requires experimenting (Derczynski et al., 2015).

The vocabulary of the corpus is then merged into the k clusters by computing the similarity function SF also known as mutual information metric (Liang, 2005):

$$SF = \sum_{c,c'} p(c, c') \frac{p(c, c')}{p(c)p(c')} + \sum_w p(w) \log(p(w)) \quad (2.4)$$

with c and c' being clusters and w a word belonging to a cluster. This results in the vocabulary being categorized in the k clusters. Ideally, words that are semantically similar to a human intuition are merged into the same cluster i.e. the words *dog*, *cat*, *mouse* would be expected to be placed into the same cluster, which are all nouns. Finally, after all vocabulary has been assigned to one of the clusters, $k - 1$ merges are performed that create a hierarchy of similarity between clusters. This hierarchy is encoded as bit string that expresses the degree in similarity between the words in the k clusters. For using a bi-gram language model in the similarity function, we consider Brown clustering to be a generative model.

Latent Dirichlet Allocation (LDA) Clustering LDA clustering (Blei et al., 2003; Chrupala, 2011) is a soft clustering algorithm. Unlike Brown clustering this

allows a single word to be member of several clusters. LDA clustering computes for a word the probability of belonging to a certain cluster. This accounts for the language ambiguity with respect to PoS tagging (Rehbein, 2013), i.e. a word has not necessarily a single pre-determined PoS tag. However, no information about cluster similarity is provided as in Brown clustering.

LDA was originally introduced for topic modeling to find related topics within document collections. The assumption is that each document contains latent topics of which each has a typical word distribution that characterizes this topic (Blei et al., 2003). To utilize LDA for PoS tagging, a word form is the equivalent to a document, the words are context features and the topic is the cluster in which we are interested (Chrupala, 2011). As a result, one retrieves a probability distribution of a word form belonging to a certain cluster. For a discussion of the mathematical details of LDA see Blei et al. (2003). LDA belongs to the generative models.

2.1.4 Supervised Tagging

In supervised PoS tagging, a machine learning algorithm learns from a PoS annotated training data set to assign a word its PoS. The possible PoS are defined in the tagset that has been used to annotate the training data. These tagsets are human-defined and are discussed further below in detail. The most salient difference to unsupervised tagging is the requirement for a gold standard, i.e. annotated training data, from which the tagger can learn from. Furthermore, the number of possible PoS tags is fixed by the tagset and correspond to a human-intuition of PoS categories. PoS tagging is sometimes also called a sequence classification task. This name originates from the circumstance that determining a word's PoS depends on the surrounding words. Sequential patterns in the occurrence can be easily exploited for making more informed label predictions. For instance, some words occur usually in a close proximity to each other when they have a certain PoS and thus, have a certain correlation to occur together.

There is a rich variety of approaches that have been used for PoS tagging. We will subsequently discuss machine learning algorithms used for tackling PoS tagging.

Lookup (Baseline) Tagger The most simplistic form of a supervised tagger composes of a simple lookup mechanism which stores the most frequent tag for a word in a dictionary. This approach belongs to the *discriminative* models as the decision boundary between tags is approximated by relying on a frequency bias learned from the corpus. This rather basic approach might be additionally improved by using morphological clues such as typical pre-, in- or suffixes for certain word classes. Furthermore, a default tag is assigned in case a word is not contained in the dictionary.

Transformation Rule Taggers This approach also assigns the most frequent tag from a PoS dictionary and applies afterwards a set of correction rules. Assigning the most frequent tag from a dictionary will be correct in many cases but will still

make many errors. Taggers based on transformation rules apply a set of correction rules learned from the training corpus after assigning the most frequent tag. Possible correction rules are tested during rule learning and permanently stored if conditions are found in which assigning a different tag than the most frequent one accounts for more improvements than newly introduced errors. The taggers by Brill (1992) and by Hepple (2000) follow this approach.

Support Vector Machine (SVM) SVMs (Vapnik and Lerner, 1963) also belong to the *discriminative* models. SVMs distinguishes between two classes by focusing on learning a decision boundary between these classes, i.e. a hyperplane. How to apply a two-class classifier to PoS tagging is discussed further below. This separating hyperplane maximizes the distance to the vectors at the decision boundary, which results in only a few data points being located at the decision boundary, i.e. the name providing support vectors as shown in Figure 2.5. This makes SVMs a highly robust method when the amount of available training data is limited as the SVM focuses on learning this hyperplane, which depends on only few data points. The optimization problem that is solved to determine this hyperplane is (Bishop, 2011, pp. 326–331):

$$\min_w \frac{1}{2} \|w\|^2 \quad (2.5)$$

$$\text{s.t. } \forall_i y_i(w^T x_i + b) \geq 1 ; y_i \in \{1, -1\} \quad (2.6)$$

With w being the normal vector to the hyperplane, $y_i(\cdot)$ describing the margin and b being a bias term. SVMs can also be trained to allow for a certain margin of error by introducing a slack variable which changes the formula to:

$$\min_w \frac{1}{2} \|w\|^2 + C \left(\sum_i \xi_i \right)^k \quad (2.7)$$

$$\text{s.t. } \forall_i y_i(w^T x_i + b) \geq 1 - \xi_i ; y_i \in \{1, -1\} \quad (2.8)$$

The newly introduced variable ξ allows for a classification error for data points that violate the margin. C is a regularization term that can be used to additionally control for over-/ underfitting of the SVM. The SVM using a slack variable is sometimes called a soft margin SVM compared to the one without that is called a hard margin SVM.

Most classification problems (including PoS tagging) require distinctions of more than two classes, which goes beyond the number of classes a SVM can distinguish. Several strategies exist to tackle classification problems with $K > 2$ classes (Bishop, 2011, 338–339). For instance, training several SVMs and applying them after each other, i.e. *one-versus-the-rest* classification. Alternatively, one can use a voting approach in which $K(K - 1)/2$ SVMs are trained. The prediction of the SVMs are considered as votes to assign a sample to a certain class, also called *one-versus-one* classification. A commonly used concept used with SVMs is the Kernel trick (Goodfellow et al., 2016, pp. 137–139). By applying a kernel-based transformation, the

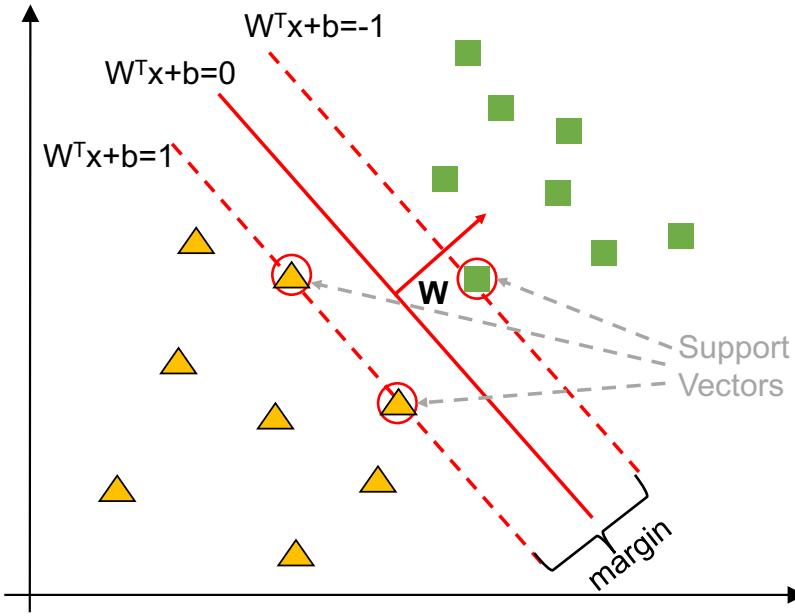


FIGURE 2.5: Support Vector Machine that separates two data point categories and highlights the support vectors that define the boundaries of the margin (figure adapted from (Flach, 2012, p.212, Figure 7.7))

data vectors are transformed into a higher dimensional space. This enables a linear separation of problems that are in their original form not linearly separable by using convex optimization techniques.

SVMs are no sequence classifier and do not make use of the word order in a sentence. This makes SVMs an untypical choice for tackling sequence classification problems. SVMs have yet been successfully applied to PoS tagging (Giménez and Màrquez, 2004). Today, SVMs are mostly used for PoS tagging when dealing with under-resourced languages where only small annotated corpora are available (Ekbal and Bandyopadhyay, 2008; Das et al., 2015; Behera et al., 2015). When a large amount of training data is available, approaches based on a HMM, CRF or LSTM neural network (all subsequently discussed) are the preferred choice.

Hidden Markov Model (HMM) A prominent example for PoS tagging of the *generative* models is the Hidden Markov Model. In a HMM, the words in a sequence are considered as observations. Each observation relates to a hidden *event*, which is in the case of PoS tagging the PoS tag associated with the word in this sequence. A HMM is formally described by a five-tupel of $\{Q, O, \Pi, A, B\}$. $Q = q_1, \dots, q_n$ is a set of N states, a set of observations in the sequences $O = o_1, \dots, o_n$ which are part of a vocabulary V i.e. $o_n \in v_1, \dots, v_n$, Π are probabilities of the initial states, a matrix A holding transition probabilities of one state moving into another state and emission probabilities $B = b_i(o_n)$ that an observation o_n is generated from state i (Jurafsky and Martin, 2009, p. 179).

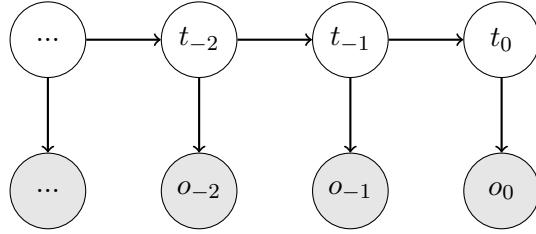


FIGURE 2.6: Trigram Hidden Markov Model as directed graph in which prediction of t_0 considers the two previously predicted tags t_{-1} and t_{-2}

An assumption of the basic HMM, the Markov assumption, is that the probability to reach a certain state is only depending on the current state. This most basic HMM is also called a first order HMM. For PoS tagging this means that the word appearance depends only on its tag which ignores surrounding words. To utilize also the sequence information, higher orders HMMs that consider the previous n predicted tags in their decision process are used for PoS tagging. The probability function of a third order HMM model is:

$$t = \underset{t^n}{\operatorname{argmax}} = \prod_i^n p(w_i|t_i)p(t_i|t_{i-1}, t_{i-2}) \quad (2.9)$$

This is also shown in Figure 2.6 where the prediction of y_0 considers the two preceding predictions. Well-known HMM PoS tagger implementations based on such a third-order HMM are the taggers by Brants (2000) and by Halász et al. (2007). While higher order HMMs are basically possible they also complicate the calculation of probabilities as parameters in this model grow exponentially with the order. Furthermore, the higher the order of an HMM is, the stronger the model is challenged with data sparsity as longer sequences will inevitably also contain more observations that did not occur in the training data. Third order HMMs are thus found to be a reasonable trade-off between contextual information won by considering sequential information and increasing data sparsity.

The HMM can be easily represented as a directed graph. During prediction, the most probable sequence of states, which is the most probable path in the graph, has to be determined to arrive at the most probable tagging for a sequence. Finding this most probable path in such a graph is frequently done using the greedy Viterbi (Viterbi, 1967) algorithm to tackle the computational complexity of this task. This algorithm is also known as max-sum algorithm (Bishop, 2011, pp. 411–416).

Maximum Entropy (MaxEnt) A further kind of generative, statistical model is the maximum entropy model (Ratnaparkhi, 1996; Toutanova and Manning, 2000; Dandapat et al., 2007). The general MaxEnt model for PoS tagging is described as follows (Ratnaparkhi, 1996):

$$p(h, t) = \pi \mu \prod_j^k a_j^{f_j(h, t)} \quad (2.10)$$

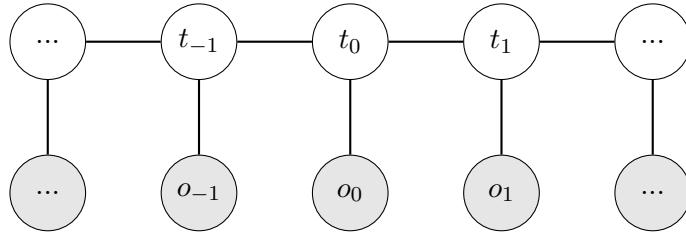


FIGURE 2.7: Conditional Random Field as undirected graph in which the prediction of t_0 is conditioned on the entire sequence

with π being a normalizing constant value, $\{\mu, a_1, \dots, a_n\}$ being model parameters and $f_j(\cdot)$ being a feature function of features that evaluate to a boolean $\{0, 1\}$. Given the words $\{w_1, \dots, w_n\}$ from the training data with the labels $\{l_1, \dots, l_n\}$ one tries to find the model p that maximizes the likelihood of the model for the data by solving:

$$p = \prod_{i=1}^n \pi \mu \prod_j^k a_j^{f_j(h_i, t_i)} \quad (2.11)$$

Active features i.e. $f_j(h_i, t_i) = 1$ contribute to the joint probability $p(h_i, t_i)$ while constraining t_i to be limited to certain tags.

Conditional Random Fields (CRF) Conditional Random Fields (Lafferty et al., 2001) are *descriptive* models and, unlike the HMM which model $p(x, y)$, CRFs model the conditional probability $p(y|x)$ directly. While HMMs are based on the Markov assumption, CRFs can operate on all states in a sequence instead of only the preceding N states as in HMMs. If one assumes a graphical representation, CRFs are modeled as undirected graph, as shown in Figure 2.7, in contrast to the directed graph in a HMM representation. The formula to predict the most likely tag is:

$$p(y|x; w) = \frac{\exp(\sum_i \sum_j w_j f_j(y_{i-1}, y_i, x, i))}{\sum_{y \in Y} \exp(\sum_i \sum_j w_j f_j(y_{i-1}, y_i, x, i))} \quad (2.12)$$

The most distinctive difference to the HMMs is a weighted (w) feature function (f) that operates on all observations of a sequence. Several CRF based PoS tagger implementations are available (Schmid and Laws, 2008) but are also frequently used in other areas, for instance named entity recognition (McCallum and Li, 2003; Ritter et al., 2011), syntactical parsing (Sha and Pereira, 2003) or relation extraction from text (Bundschatz et al., 2008).

Neural Networks A more recent approach to PoS tagging is using neural networks (Collobert et al., 2011; Goodfellow et al., 2016), which are discriminative models.

A fundamental neural network composes of at least three layers, an input layer, a hidden layer and an output layer. The number of hidden layer might vary, for this introduction we assume a single hidden layer with four neurons as shown in Figure 2.8. The neurons of each layer are connected to neurons in preceding or following layers. The input variables represent an information useful to determine the output. During

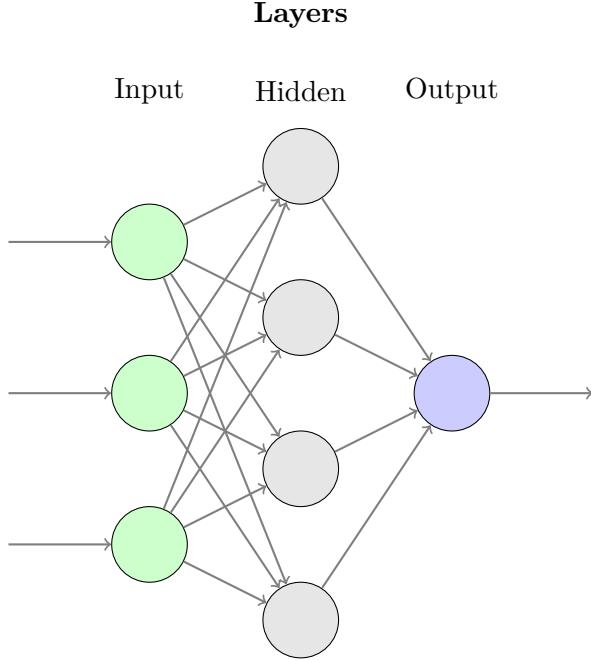


FIGURE 2.8: Basic neural network with one hidden layer

network training, the neurons in a layer learn weights for each feature to predict the correct output. The higher the weight, the more the feature contributes in the prediction. The neurons located in the hidden and output layer has an activation function. This activation function maps the weighted output of a neuron into a fixed numeric range depending on the used activation function. Common activations are Sigmoid ($\text{sig}(x) = \frac{e^x}{e^x + 1}$) that map values in a range of $[0, 1]$ or the hyperbolic tangent ($\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$) that maps values to $[-1, 1]$. The resulting value of the activation function is passed to the next layer as input.

Network training composes of a forward and a backward step. In the forward step, the current network weights are used to predict the output, i.e. the gold labels. Based on the occurring error the gap towards the gold labels is computed. The backward step passes the error in the reversed direction through the network and updates the weights. The most commonly used method is the back-propagation algorithm that uses gradient decent to perform this weight adjustment. During back-propagation, the gradient is computed and the weights are in- or decreased according to the gradient. The intuition is that when all weights are updated according to the gradient, the network approximates the point of “no error”. During training, the weights in the layers adjust and eventually determine a function that is able to predict the output value for a given input.

There are no best practices for how many network layers and how many hidden neurons are advisable for a certain task. The example shown in Figure 2.8 assumes a single output neuron which is suited for binary predictions. If more than two output states shall be distinguished an accordingly higher number of output neurons is necessary. For computing the most likely label, algorithms such as Softmax are

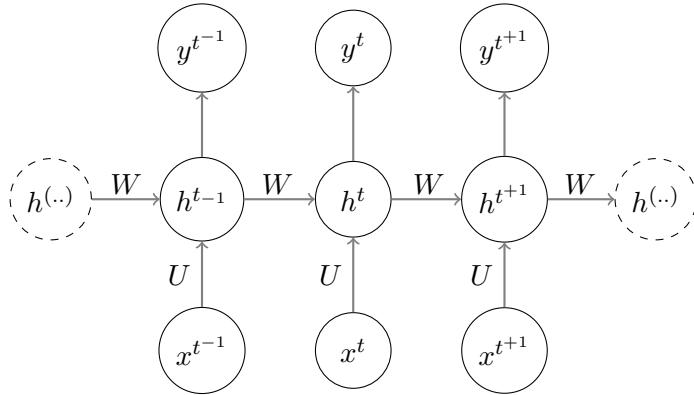


FIGURE 2.9: A recurrent neural network shown as time-unfolded graph with three time steps t . A weight matrix W is passed between the hidden states h to carry sequential knowledge from previous time steps into the next step. U is a weight matrix of input-to-hidden weights. x are training samples and y the corresponding gold labels (figure adapted and simplified from (Goodfellow et al., 2016, p.369, Figure 10.3))

applied that transform a multi-dimensional vector to a real valued vector in the range of $[0, 1]$. This allows determining the most likely label in classification tasks with more than two labels.

Recurrent Neural Networks (RNN) RNNs are a kind of neural network suited for sequence labeling tasks. In a non-sequential neural network it is assumed that the inputs are independent of each other. Thus, the basic difference is the idea of sharing information between the neurons. We show in Figure 2.9 such a RNN as time-unfolded graph. The weight matrix W represents the shared information between time steps. This allows a time step t to also consider information of earlier time steps t^{-n} . Hence, RNNs hold a memory of previously processed inputs (or words in our case). The formula for a hidden state at the time step h_t for a sample x_t is:

$$h_t = (W_{h_{-1}}x_t + Ux_t) \quad (2.13)$$

The general issue with RNNs is the challenge to model long-range dependencies that might or might not be of relevance. The weight matrix W is not able to account for a varying importance of earlier information. The larger the distance in time steps between two neurons, the weaker the information of earlier time steps becomes. This is also known as the vanishing or exploding gradient problem (Hochreiter, 1998), respectively, because all sequential information is carried by a single weight matrix.

To account for this problem, the concept of gated cells has been introduced that allow preserving such long-distance information.

Long Short Term Memory (LSTM) Neural Networks LSTMs (Hochreiter and Schmidhuber, 1997) are a kind of recurrent neural network that tackles the vanishing or exploding gradient problem by using a gated cell mechanism. Distinctive difference to normal RNNs is a self-loop within an LSTM cell that adds an additional

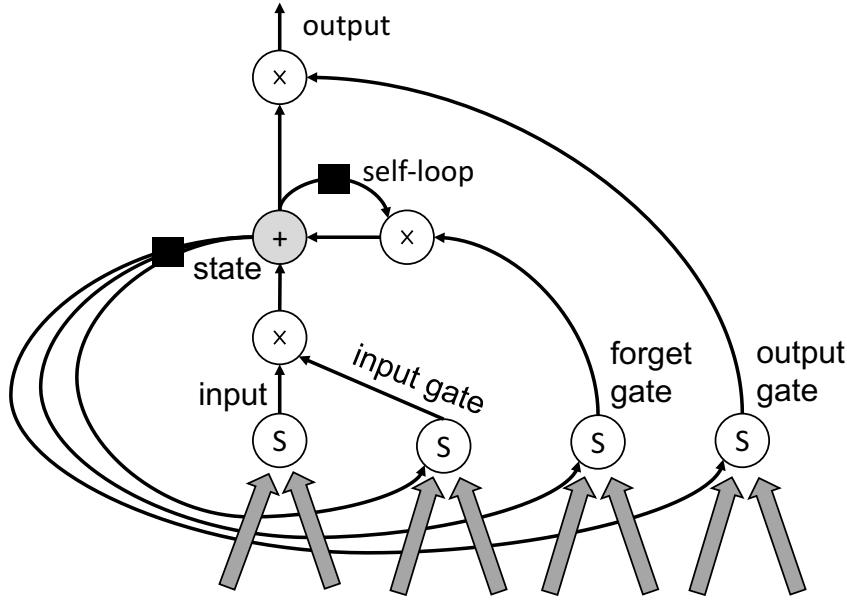


FIGURE 2.10: Structure of a plain LSTM cell with three additional sigmoidal (S) gates and a state with a weighted self-loop controlled by the forget gate, black square indicates a time step delay (from (Goodfellow et al., 2016, p.398 Figure 10.16))

“internal” recurrence to a cell on top of the “outer” network recurrence (Goodfellow et al., 2016, p. 399). A plain LSTM cell has three gates: input, forget and output gate that contribute to the hidden state at a time step as shown in Figure 2.10. The contribution of the gates is determined by a function such as sigmoid or the hyperbolic tangent to compute the current cell state. The computation of the hidden state in an LSTM at a given time step is as follows (\circ is an element-wise multiplication):

$$(\text{Forget Gate}) \quad f_t = \sigma(W_f x_t + U_f h_{h-1} + b_f) \quad (2.14)$$

$$(\text{Input Gate}) \quad i_t = \sigma(W_i x_t + U_i h_{h-1} + b_i) \quad (2.15)$$

$$(\text{Output Gate}) \quad o_t = \tanh(W_o x_t + U_o h_{h-1} + b_o) \quad (2.16)$$

$$(\text{Cell State}) \quad c_t = f_t \circ c_{t-1} + i_t \circ \tanh(W_c x_t + U_c h_{h-1} + b_c) \quad (2.17)$$

$$(\text{Hidden State}) \quad h_t = o_t \circ \tanh(c_t) \quad (2.18)$$

This more complex cell management lead to situations in which a cell is not updating at all and, thus, allowing to carrying unaltered information to future time-steps.

LSTMs have been frequently used for a variety of sequence labeling tasks including PoS tagging (Plank et al., 2016; Cimino and Dell’Orletta, 2016; Yasunaga et al., 2017). While the idea of using neural networks for PoS tagging is not new (Schmid, 1994a), recent advances led to a resurfacing of neural networks for a variety of NLP tasks. There are variations of LSTMs such as the peephole LSTMs (Gers and Schmidhuber, 2000) or other gating-related units (Cho et al., 2014). With respect to PoS tagging, the plain LSTM variant is most commonly used. To leverage all sequential information contained in sentences, it is wide-spread to use bi-directional LSTMs (Graves et al.,

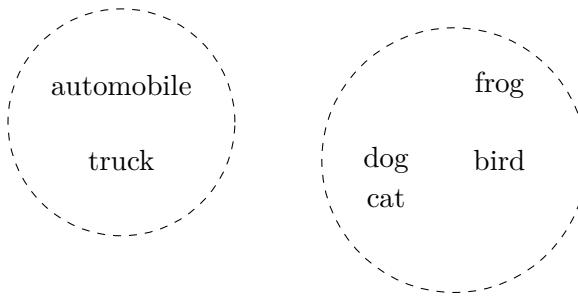


FIGURE 2.11: Example of words in embedding space with semantically similar words being located more closely to each other such as animals or vehicles (adapted from (Socher et al., 2013) Figure 2)

2005) that use an additional LSTM to process the sequence or input sentence in reverted order. This allows using sequential information of a right-to-left and left-to-right pass over a sequence. The output of both LSTMs is then concatenated for the following processing steps.

Word Embeddings Neural networks are frequently initialized with distributional word knowledge that is provided by word embeddings. This is also sometimes called unsupervised pre-training. In a word embedding, each word is represented by an N -dimensional dense vector. These dense vectors inform the neural network about higher or lower (semantic) similarity between the words (Mikolov et al., 2013; Pennington et al., 2014) by lower or higher proximity of the words in this embedding space (Goodfellow et al., 2016, p. 452). Figure 2.11 shows an example in which semantically related words are more closely located to each other in the embedding space. This word embedding initialization provides an informed network state which usually leads to better results than a pure random initialization (Erhan et al., 2010).

Word embeddings are trained over large collections of plain text. For its ease of access, the most available embeddings today are created from web-published resources such as Wikipedia text or social media websites (Al-Rfou et al., 2013; Pennington et al., 2014). It is quite common to train word embeddings with a neural-network-based language model (Bengio et al., 2006; Collobert and Weston, 2008; Mikolov et al., 2013) or factorization of co-occurrences as in Pennington et al. (2014). While the early word embeddings focused only on word-level information more recent work also incorporates sub-word information that also capture morphological information in the embeddings (Bojanowski et al., 2017).

2.2 Tagsets

Tagsets define a number of PoS tags that are used to annotate a corpus. Supervised PoS taggers are trained on data that are annotated with such a human-defined PoS tagset. Tagsets find their origin in linguistic theory and vary in granularity. Most tagsets are fine-grained and tailored to a particular language. For a language, several competing tagsets might be available.

English Language For English, the following tagsets exist: C5 (Leech et al., 1994) and C7 (Garside et al., 1997) tagset with 61 and 137 tags, the Brown (Nelson Francis and Kuçera, 1964) tagset with 87¹ tags and the Penn Treebank (PTB) (Marcus et al., 1993) tagset with 45 tags. The number of tagsets is a result of experimenting with different granularities and methods. For instance, the Brown and C5/C7 tagsets are considerably more fine-grained than the PTB tagset. In the Brown tagset one finds tags that are unique to certain lexical items, which results in many sparse tags. Furthermore, some cases allow a combination of tags, for instance a non-English foreign word that functions as preposition would receive a combined tag marking it as both, a foreign word but also a preposition. This can lead to a drastically increasing number of tags that are extremely sparse and extend far beyond the 87 base tags. In the C5/C7 tagset a running number is added to the tag of a group of words when they serve the same grammatical purpose, for instance for prepositions composing of several words such as *in terms of* (Jurafsky and Martin, 2017). This numbering adds an additional layer of annotation and is not purely syntactical anymore and extends beyond the scope of a pure PoS tagset.

The PTB tagset originated from concerns with the Brown and C5/C7 tagset and aimed at reducing the tagset sizes drastically. Lexical and syntactical redundancies have been removed but also the problem of infrequency of tags in the Brown tagset has been tackled (Taylor et al., 2003, pp. 6–7). Some of these reductions have been undone in a modified PTB tagset used for instance in Zeldes (2016) re-introducing the dedicated tags for the verb forms *have*, *be* and *do*, and their inflection forms. These tags existed in the Brown and C5/C7 tagset but have been collapsed to fewer tags in the PTB tagset.

The usefulness of a tagset depends on the task. The many fine-grained distinctions in the Brown and C5/C7 are useful for a detailed linguistically analysis of text. To enable a comparison between languages, considerably less fine-grained and language independent tagsets surfaced, for instance the Universal tagset (Petrov et al., 2012) (12 tags) or the Universal Dependency tagset (17 tags) (Nivre et al., 2016) (which we discuss further below separately). Today, the PTB tagset is the de-facto standard for English that is occasionally used in a refined version for annotation of corpora (Zeldes, 2016) but also for PoS taggers (Schmid, 1995).

Morphologically Fine-grained Tagsets Morphologically complex languages require often tagsets that are much more fine-grained than the tagsets that we just discussed for English. Morphologically fine-grained tagsets exist in particular for Slavic languages such as Czech or Slovene (Erjavec and Krek, 2008; Jakubíček et al., 2011). On top of the basic syntactical category, these tagsets also distinguish the grammatical case of a word (for instance accusative, dative, genitive), the grammatical gender (for instance masculine, feminine, neuter) or the grammatical person (for

¹without combined or negated tags

instance first, second, third person). If for each morpho-syntactical combination a dedicated tag is created, tagset sizes of thousand and more tags are easily reached.

Proper Nouns Many PoS tagsets contain additionally to tags for (common) nouns also tags for proper nouns. However, this frequently found distinction between nouns and proper nouns is strictly speaking not a syntactical distinction. A side effect of this non-syntactical distinction is a high number of PoS tagging errors, which are confusions of nouns with proper nouns and the other way around. Finding proper nouns is a rather straight-forward task in standard English as proper nouns are all capitalized. Such a simple distinction by just looking at the first letter is often not possible for other languages. For instance, in German, all nouns - not just proper nouns - are written with a capital letter. Thus, for focusing on a strictly syntactical task it is reasonable to treat proper nouns as common nouns. Furthermore, determining proper nouns is tackled as an own NLP task, named entity recognition (Cucerzan and Yarowsky, 1999; Tjong Kim Sang and De Meulder, 2003).

Language Independence Recently, language independent tagsets surfaced, which are considerably more coarse-grained than the language-dependent ones we discussed so far. Most prominent examples are the Universal PoS tagset with 12 tags and the Universal Dependencies tagset with 17 tags. While language dependent tagsets usually distinguish between common inflection forms found in a language, the coarse-grained language independent tagsets do not. In the Universal PoS tagset or the Universal Dependencies tagset nouns, verbs, adjectives, etc. are all represented by a single PoS tag. Figure 2.12 shows an example of how fine-grained tags of the English PTB tagset are mapped to the coarse-grained Universal PoS tagset. These more coarse-grained tagsets surfaced to enable cross-language comparison between languages by making the PoS annotation comparable. Similar to the tagsets in English that we discussed above, determining a suited granularity for a coarse-grained tagset is a time-consuming task. The 17 tag Universal Dependencies tagset resulted from the lessons learned with the 12 tag Universal PoS tagset. The latter one did allow for cross-lingual PoS comparisons but was too coarse-grained for constructing dependency trees of multiple languages. This led to the 17-tag Universal Dependencies tagset, which introduced additional tags for instance for proper nouns or subordinating and coordinating conjunctions.

Consequently, the choice of the tagset depends on the task. Tasks such as linguistic analyses that compare sentence compositions and alike usually require fine-grained PoS distinctions to study these properties. When morphological details are not of primary importance and comparisons across languages are needed, coarse-grained tagsets are sufficient.



FIGURE 2.12: Example of mapping fine-grained tags of the English Penn Treebank tagset (bottom row) to the corresponding single tag of the Universal tagset

2.3 Evaluation Metrics

Evaluation of PoS tagging results is usually done using either *Accuracy* or *F-Score* as metric, which we introduce here.

2.3.1 Accuracy

This metric is computed over all tokens. It computes the number of correctly classified token instances in comparison to all token instances. The result is a single value that expresses the quality of the tagging computed over all PoS tags. Drawback of this metric is that the frequency of PoS tags is not equally distributed. A high correctness on a frequent tag might conceal that infrequent tags are often tagged wrongly. For instance, assuming we have two tags, one occurs 98% of the time and the other one only 2% of the time. If one evaluates a tagger that always assigns the former tag, accuracy would report a stunningly high result of 98% correct tagging. The zero performance on the second tag might not be noticed at all. Nonetheless, this metric is often used to express the overall quality of a tagger.

Out-of-vocabulary Words An important criterion that distinguishes the overall quality of a tagger is the performance on words that are out-of-vocabulary (OOV). PoS taggers tend to perform well on words that occurred in the training data while performance drops, sometimes considerably, when OOV words have to be tagged. It is, hence, common to compare the overall accuracy to the accuracy on OOV words to learn how well the tagger generalizes to unseen words.

Micro vs. Macro Average: Accuracy can be computed as micro or macro average over (i) all tags within a corpus or (ii) as averaged value over several tagged corpora. *Micro* takes the frequency of a tag or the size of a corpus into consideration. Thus, tags that occur only infrequently or corpora that are small have only a small impact on the overall accuracy while frequent tags or large corpora have a large impact on the accuracy. *Macro* ignores the frequency or size of a corpus and weights all tags and corpora equally important.

		Gold	
		A	B
Prediction	A	45	25
	B	10	20
Accuracy	65%		

TABLE 2.1: Confusion matrix with fictive results for a two class classification example

		Label Class	
		A	B
Counts	TP	45	20
	FP	10	25
	FN	25	10
Precision		$\frac{45}{45+10} = .82$	$\frac{20}{20+25} = .44$
Recall		$\frac{45}{45+25} = .64$	$\frac{20}{20+10} = .67$
F-Score (F_1)		$\frac{2 \cdot .82 \cdot .64}{.82 + .64} = .72$	$\frac{2 \cdot .44 \cdot .67}{.44 + .67} = .53$
F_1	Micro	$\frac{.72 \cdot .70}{100} + \frac{.53 \cdot .30}{100} = .66$	
	Macro		$\frac{.72 + .53}{2} = .63$

TABLE 2.2: Calculation example of Precision, Recall and F-Score for the results in the confusion matrix shown in Table 2.1

2.3.2 F-Score

This metric computes a score with respect to a single tag instead of using all tags as *accuracy*. F-Score ($F_1 = 2 \cdot \frac{P \cdot R}{P+R}$) is computed from two other metrics, namely *Precision* ($P = \frac{TP}{TP+FP}$) and *Recall* ($R = \frac{TP}{TP+FN}$) which are in turn computed from counts how often a tag was tagged correctly and confused with other tags. *TP=True Positives* are words for which the tag in focus was predicted correctly, *FN=False Negatives* are words for which the tag in focus was erroneously not assigned and *FP=False Positives* are words for which the tag in focus was erroneously assigned but should have been tagged with another tag. The F-Score allows more detailed insights on assigning a certain tag and is helpful to determine strengths and weaknesses of a tagger with respect to that tag.

2.3.3 Calculation Example

In Table 2.2, we show a calculation example of computing accuracy and F-Score for the fictive example shown in Table 2.1 as confusion matrix. By looking separately on the F-Score for each class, we gain additional insights that were not visible by looking only at the accuracy metrics. The tagging performance for label B is considerably lower than for label A. For applications in which certain tags are of an increased importance, computing F-Score offers valuable information that are concealed in less sophisticated metrics such as accuracy.

2.4 Robustness

While controlled conditions in experimental setups often allow to achieve encouraging results, the robustness for usage in a real-world scenario is often not in the focus of experiments. As soon as method leaves the laboratory environment, a large variety of additional factors come into play, which adds to the difficulty of solving a task.

This is a rather general problem for all kind of classification tasks which have to be able to deal with many conditions, i.e. require a certain robustness.

Definition When we talk about robustness, we use the following definition (Huber and Ronchetti, 2009, pp. 1–2): Robustness is the insensitivity of a model to deviations from an assumption. With the three requirements that (a) the model should be efficiently able to represent the assumption, (b) small deviations of the assumption should lead to small impairment of the performance and (c) large deviations should not lead to a model breakdown, i.e. catastrophic results (Huber and Ronchetti, 2009, p. 5). Similar definitions describe robustness as a property which maintains a service under conditions unforeseeable during development that are harmful for achieving the objective (Kitano, 2004; Li et al., 2016; Vacavant, 2017).

In the context of supervised machine learning, the *assumption* is essentially the set of labeled data in the training dataset. It is assumed that these data fully represent a domain or the problem that should be solved by training a classifier. *Insensitivity to deviations* is the delta of new unlabeled data that shall be classified based on the used training data. One can think of the training data coming from a distribution A while the new data are being drawn from a different distribution B . A robust, insensitive classifier is supposed to deal with the samples from B equally well as those from A , despite of some differences between the distributions. The criterions (b) and (c) are certainly difficult to quantify and are usually task dependent. If one thinks of a safety critical task, for instance autonomously driving vehicles, an increased error rate of a few percent points in the image processing might be fatal and be considered as a model break down. In other tasks, for instance automatic quality assessment of manufactured parts, an increased error of a few points due to poor lighting conditions might still be considered as acceptable.

Hereinafter, we discuss robustness challenges and counter measures to learn more about the challenge of robustness for classification tasks in general.

Image Classification Image classification is the task of recognizing if an object of a pre-defined category is present in an image. In this field, deep neural networks have reached stunning results with a high accuracy in determining the correct category of images (He et al., 2015). The challenge of robustness when working with images has a variety of dimensions. The training data of an image processing classifier are images of a certain size and resolution. This classifier is expected to perform equally well even under conditions such as changes in illumination, orientation, scaling, etc. In other words, an image classifier should be able to classify an image correctly even if it has half the resolution of the original training data, is upside-down and taken in a (slightly) dimmed room. Further robustness challenges arise from natural noise and the technical limitations of the recording camera or compression artifacts that add a layer of noise to each photo. Robustness is only achievable with a sufficient amount

of such noisy data samples for model training to inform the classifier about these phenomena (Vacavant, 2017). Consequently, strategies for improving robustness entail duplicating samples from the training data and add variations of the image containing such minor variations (Zheng et al., 2016; Uličný et al., 2016). Alternatively, generative adversarial networks (Goodfellow et al., 2014) are used for de-noising images (Yang et al., 2017) or to generate corrupted versions of *clean* training samples (Denton et al., 2015) to provide noisy variants of images from which the model learns to cope with such variations.

The textual domain in which we are working here is quite different from images. One cannot easily create text samples that are identical to human perception and only differ in an amount of (non-visible) noise. Li et al. (2017) experimented with different ways to adapt the idea of adding noise to textual data by adding syntactical or semantical noise without changing the data too much.

Automatic Speech Recognition Automatic speech recognition (ASR) is the task of converting an audio signal of human speech into a sequence of words. In ASR, deep neural network approaches have been applied successfully, too (Deng et al., 2013). Similar to the image classification field, naturally occurring noise poses one of the biggest challenges to ASR. The dimensions of noise are similar to the one discussed for image processing. Most comparable to image processing is the device variations of microphones that are comparable to the noise added by cameras. Additionally, background noise such as traffic noise might occur but also variations among the speakers such as mumbling, dialects or pitches in the voice are common challenges. As the result of ASR is plain text, which we require as input for PoS tagging, it could be a preprocessing step from the viewpoint of our use case. While we assume to work with written discourse most of the time, errors of an imperfect ASR are certainly a robustness challenge to PoS tagging and other downstream applications. Similar techniques as for image processing have been proposed, i.e. modifying training data and producing noise-added versions of this clean samples (Ebrahim Kafoori and Ahadi, 2017) by using generative adversarial networks (Serdyuk et al., 2016). An alternative approach we will pick up later for PoS tagging is multi-style training by mixing up data originating from various conditions (or text domains in our case) to achieve an increased robustness.

PoS Tagging Robustness in PoS tagging has also received some attention. An issue that has some mentions in the literature, for instance Ritter et al. (2011), finds that tagger models trained on formal text domains perform only poorly on informal text. By combining knowledge obtained from clustering over Twitter plain text and using a mixture of training data from similar and foreign text domains, Ritter et al. (2011) construct a new tagger to deal with this robustness issue. Müller and Schuetze (2015) experiment with various kinds of distributional representation to increase robustness when tagging foreign text domains on a morphologically fine-grained level. Tsuruoka

et al. (2005) train a more robust tagger model suited for medical text by training a model on formal news and medical text. While the model trained on a mixture of text corpora reaches never the best performance its average accuracy across the text domains improve considerably i.e. multi-style training. Choi and Palmer (2012) tackled robustness by switching between a general and a specific model for tagging a sentence depending on the similarity of an input sentence to the training data of the two models. Generative adversarial models have also been used in PoS tagging (Gui et al., 2017; Yasunaga et al., 2017).

Lacking Data Variety Thus, there is a general gap between the laboratory conditions and the real world conditions, which shows that robustness is a general challenge for classifiers of various kind. The datasets for training a classifier are usually *clean* datasets that do not contain many of the real-world conditions that might be encountered outside the laboratory. Images are taken for instance under good lighting conditions with high quality cameras, audio data is well articulated while being recorded with fidelity equipment or textual data is often limited to be highly formal in its nature. It is, hence, not surprising that classifiers trained on such datasets do not necessarily generalize well to a hardly predictable number of conditions in the real world. Many datasets simply do not contain suited training instances to cover these conditions. Working with such *clean* datasets has the merit of eliminating variables in an experimental setup that are not necessary in order to show that a certain approach works. From a research perspective it, hence, is absolutely reasonable to work under such conditions. As consequence, classifier trained on such clean datasets often lack robustness.

We distinguish in total three research areas that all tackle the lack of sufficient domain data but still avoid manual annotation of more data: *normalization*, *domain adaptation* and *generation of annotated data*.

Normalization Normalization tries to restore a sample to its original unaltered state. When operating on images or audio samples, this original noise-free state is unknown. Such algorithms have no valid stopping criterion and de-noising is therefore also considered as an ill-posed problem (Gong et al., 2016) i.e. de-noising might remove information that is no noise. In case of text, an underlying vocabulary of standard words exist and it is a valid assumption that the most non-standard word forms do have a standard word form counterpart. Some exceptions might exist, for instance word creations that are only used within certain social peer-groups but for the most non-standard word forms, such a standard language counterpart is available. A frequent observation in social media are, for instance, shortening of words. The standard word form “tomorrow” might occur as “tumr” or “2morrow” (Ritter et al., 2011). Normalization aims at mapping such non-standard forms to their canonical standard form. The idea is to bring the text closer to the training data of the (PoS)

model. This eases the classification task for the model by normalizing the unknown domain-specific linguistic phenomena to their standard word forms before the tagging.

Normalization of non-standard text, as it is found today in social media, has a long history and started to shift into the research focus when working with SMS (short message services) text (Choudhury et al., 2007). Normalization entails two steps, first to determine that a non-standard word form is present and second, to determine the canonical word form. The most straightforward approach to normalization is to create hand-crafted rules or a dictionary that defines substitutions of non-standard to standard word forms (Sproat et al., 2001). Normalization is sometimes treated as machine translation task in which the unnormalized non-standard text is the source language that is translated into the target standard language (Aw et al., 2006; Kaufmann and Kalita, 2010). Pennell and Liu (2011) uses a character-level machine translation system to find substitution candidates and afterwards ranks them by using an ngram language model to determine the most likely substitution. Han and Baldwin (2011) uses an SVM-based approach by training on a mixture of standard and non-standard training samples. The non-standard training samples are automatically created by substituting a word in a context window with high ranked candidates. These candidates have been determined based on a character- and phone-based edit distance. Zhang et al. (2013) builds a weighted normalization graph based on an input sequence and a set normalization functions that substitute words. The normalization functions are created from annotated training data. The weights are learned during a training phase in which a token-wise edit distance metric is used to measure the similarity between input and expected output sequence. Jin (2015) uses additionally to a dictionary created from labeled data the Jaccard Index (Levandowsky and Winter, 1971) to measure the similarity between feature sets extracted over candidate words. van der Goot (2016) performs a module-wise candidate generation by creating features from several information sources such as word embeddings, a spell checker, prefix matches of a word form in a dictionary and a brute-force splitting of words to check if a word might be the result of omitting a whitespace. Generated features are then provided to a random forest classifier and the results of the trees are averaged. van der Goot et al. (2017) showed that social media PoS tagging improves by applying normalization but also finds that using a social media embeddings for a neural network PoS tagger on social media text is more effective. Training new embedding belongs to the approaches we discuss next.

Domain Adaptation Instead of bringing the text closer to the model training data (by normalization), one can also try to bring to the model closer to the text i.e. adapt the model to another text domain.

This assumes that at least some annotated training data is available from the target domain to which the model is adapted. A typical situation is the availability of larger amounts of annotated data from another text domain, for instance newswire, and only a small amount of annotated data from the target domain, for instance social

media. The first and obvious approach is to combine both datasets (Hara et al., 2005) (assuming the annotation is compatible i.e. same tagset) and train a model on both domain data. If the amount of data from foreign and target domain is very imbalanced, the larger foreign domain corpus will dominate over the actual more accurate information from the smaller target domain corpus. Daumé III (2007) proposed to apply a weighting approach that adds the smaller target domain data multiple times to artificially increase the weight of the target domain data. Jiang and Zhai (2007) re-weights training instances of a foreign domain training dataset to remove the domain bias and increase the usefulness of the learned information for the target domain. Plank et al. (2014) experimented with weighting instances without having any target domain data finding no significant improvements for PoS tagging. Blitzer et al. (2011) investigates how to deal with situations in which the target domain contains highly discriminative information that do not occur in the source data by creating a shared feature subspace between both domains. A combination of unsupervised with supervised machine learning applies word clustering to plain text and provides these cluster ids as additional features in the supervised machine learning process (Gimpel et al., 2011; Ritter et al., 2011; Owoputi et al., 2013; Rehbein, 2013). Including knowledge from external resources such as word clusters or PoS dictionaries improved handling of common spelling variations of canonical word forms. In particular, the already above discussed clustering approaches, LDA and Brown clustering, have been reported to improve tagging performance (not just on social media) on a variety of languages (Chrupala, 2011; Rehbein, 2013; Owoputi et al., 2013; Mueller et al., 2013). Yang and Eisenstein (2015) create feature embeddings with domain attributes to create more robust features in an unsupervised setting to learn from unlabeled foreign domain data. Peng and Dredze (2017) formulates the domain adaptation task as multi-task learning (Caruana, 1997) problem and uses a neural network based approach that creates a shared representation and learns task specific models from projections of the individual domains.

Generative Adversarial Networks (GANs) GANs involves two neural networks, one network is the generator (G) that produces samples and a discriminator (D) that tries to recognize if a sample was produced by the generator or is a genuine sample from the training data (Goodfellow, 2017). These two *roles* stand in an adversarial relationship and try to defeat each other. The basic idea is that the generator tries to create samples from the same distribution as the *clean* training data. During training, the generator tries to generate fake samples that are good enough to deceive the discriminator in believing it is a genuine sample of the distribution. The discriminator tries consequently to perfect its ability to recognize produced samples. This is accomplished by letting both actors use the same loss function that depends on both actors in which each actor controls only the own parameters that they try to optimize. The discriminator determines suited parameters by minimizes $L(\theta^D, \theta^G)$ with respect to θ^D and the generator with respect to θ^G . The discriminator aims

at maximizes the success rate of recognizing produced samples and the generator minimizes the chance that the discriminator recognizes the produced sample.

$$g = \arg \min_G \max_D L(D, G) \quad (2.19)$$

The objective to use GANs for creating more data is, thus, to reach a balance between the two parties where the generator manages to create instances the discriminator can no longer recognize as being produced. Thus, the generator has learned to generate new training data.

For improving robustness, this approach offers opportunities to produce inexpensively data samples that are close enough to the original distribution but are flawed to some extent. This provides variations of the *clean* data samples and enriches the training data pool. However, the created variations might not necessarily be suitable to improve robustness towards real world conditions.

With respect to PoS tagging, GANs provide only a limited amount of help to the robustness challenge we face. The generator assumes that a sufficiently large amount of data is available from which one can learn how to generate good samples. While it has been shown that on well-resourced languages and text domain this does offer benefits also for PoS tagging (Yasunaga et al., 2017) the use of GANs is questionable for low-resourced areas. The improvements achieved by GANs trained on formal text to tackle social media are only minor (Gui et al., 2017) and do not change the principle robustness problem. In order to use GANs effectively for PoS tagging, one would need sufficient training data from various text domains but this lack in data diversity is exactly what makes PoS tagging robustness a challenge in the first place.

2.5 Chapter Conclusion

In this chapter, we provided the theoretical foundations forming the basis of the remainder of this thesis. We provided an overview of the field of PoS tagging and the approaches used to tackle this task. An emphasis lay on supervised PoS tagging that is occasionally extended with knowledge obtained from the area of clustering using unsupervised approaches. We discussed the basic training mechanism of supervised PoS taggers and talked about the differences between tagset sizes that are commonly used for various languages. The metrics accuracy and F-Score have been introduced as primary evaluation tool for the quality of tagging results. Lastly, we provided an overview about the notion of robustness with respect to PoS tagging but also in the image and audio processing field. We arrived at the conclusion that robustness is a challenge that originates from working in simplified laboratory conditions that often do not have robustness as focus of research. The general solution to robustness related issues are furthermore strategies that aim at enriching the training data pool with additional samples. Preferably, this enhancement is automatized to avoid manual effort. While these approaches work reasonably well for areas such as image or audio processing, their applicability to text or specifically PoS tagging is only limited.

Chapter 3

Domain Robustness - Challenges

In this chapter, we will discuss the challenges that arise when working with several text domains that are inherently different to each other. When using a tagger, an implicit expectation is that the tagger's performance should not change much between text domains. This, however, is not necessarily true as each text domain has properties not found in other text domains. One such an example is social media text, which is considerably different to the formal newswire text that is often used to train tagger models. We find many mentions in the literature about poor tagging performance when applying models trained on formal text to social media text (Ritter et al., 2011; Foster et al., 2011; Owoputi et al., 2013; Rehbein, 2013; Neunerdt et al., 2013). Hence, the domain robustness of these models seems to be poor. In order to better understand which differences between text domains cause these challenges, we investigate these differences in more detail.

First, we conduct a series of corpus analyses on the lexical and syntactical level between text domains. We use texts of the formal, spoken and social media domain as prototypical example cases to learn more about the differences between text domains. Second, we investigate performance of available off-the-shelf taggers on these three text domains to learn how well todays tools can cope with text of different domains.

3.1 The Social Media Domain

We begin with providing an overview of the social media domain before we analyze the properties of this text domain. Social media is a general term which describes Internet services that allow users to interconnect and share information with each other (Kaplan and Haenlein, 2010). Information is exchanged in the form of text, audio, video or images. Information exchange takes place in a one-to-one fashion as in a private conversion but might also be broad- or multicasts to two or more users. Each user can respond to shared information and engage in discussions with other users. For this thesis, the textual information created in social media is in the focus of our interest.

User-generated Content An important aspect of social media is user-generated content. Before surfacing of social media, users had been in a passive role and only consumed (mostly professionally created) content provided in the Internet. Described

by the term Web 2.0, which summarizes the change of the user to a prosumer (producer and consumer), the user was offered a more active role and started to create own content, for instance web blogs (Obar and Wildman, 2015; Farzindar and Inkpen, 2015). Social network platforms, e.g. Twitter¹ or Facebook², provide an infrastructure for users to easily interconnect. This simplified sharing content about daily life experiences, opinions or alike with other users (Java et al., 2007). The strong private use of these platforms also led to rather informal written interactions between users. This social media discourse has often more similarities in common with spoken discourse than classical formally written one (Beißwenger et al., 2015; Eisenstein, 2013).

Computer Mediated Communication As communication takes place by technical means, such as smartphones or home computers, the term *Computer Mediated Communication* (CMC) is strongly related to social media. CMC describes communication between people using a technical device as medium (Herring, 1996). The term CMC originated in the 1960s and has been in use long before *social media* surfaced (Thurlow et al., 2004). As using social media requires using a technical device, it is often also CMC, i.e. both terms have a substantial overlap. In contrast, a service that allows one to chat with friends and enables invitation of additional people is not a classical social media platform but certainly is CMC. For this thesis, with the focus on non-standard text with respect to PoS tagging, we will use the term “social media” as an umbrella term that entails all kind of CMC and social media text.

Example of Tagging Twitter Figure 3.1 shows an example of a PoS-tagged Twitter posting that contains informal utterances using the Stanford (Toutanova et al., 2003) tagger with a model trained on formal text. We show the original PTB tags that have been predicted by the Stanford tagger and their mapping to the corresponding coarse-grained word class for improved readability. We see that the model assigns the tag for noun to almost all non-standard word forms. While it is hard to argue which tag would be appropriate, the tag for noun seems inappropriate. Only the two words that are part of the standard language dictionary, *miss* and *them*, have been predicted correctly. Thus, we find plenty of evidence that tagging social media can be a challenging task. In the remainder of this chapter, we explore these challenges from the theoretical and practical side to learn how well the currently available PoS taggers are equipped to deal with the challenges of informal utterances.

3.2 Theoretical Challenges

Social media text contains a rich variety of non-standard utterances that would not occur in formal text. When a PoS tagger is trained on standard text, many unknown

¹www.twitter.com, last access 30 August 2017

²www.facebook.com, last access 30 August 2017



(A) Example of a Twitter message with non-standard language

im	gona	miss	tf	outta	them	#classof2017
NOUN	NOUN	VERB	NOUN	NOUN	PRON	VERB
NN	NN	VBP	NN	NN	PRP	VBP
✗	✗	✓	✗	✗	✓	✗

(B) Manual evaluation the PoS tags assigned by the Stanford tagger

FIGURE 3.1: Example of non-standard language use on Twitter and the result of tagging this example with the Stanford tagger

phenomena of the social media domain pose serious challenges. For instance, Ritter et al. (2011) report an accuracy of 80.1% using the Stanford (Toutanova et al., 2003) tagger, which is a huge drop in accuracy to compared to 97.3% (Manning, 2011) reported on newswire text. Likewise, we tagged a German CMC dataset with the TreeTagger and reach an accuracy of 73.8% (Horsmann and Zesch, 2016b) in contrast to the 97.5% reported by Schmid (1995) on standard text. Thus, to better understand the characteristics of social media text, we will have a closer look on the challenges of this domain.

Lexical Phenomena We show in Table 3.1 an overview of frequent phenomena collected from the micro-blogging service Twitter. We also show the counterpart in the standard-language (if existent) and a brief description of the phenomenon. Each phenomenon might occur in isolation or in combination. We find a frequent use of acronyms, word contractions and (intentional) use of alternative word spellings, non-standard capitalization and use of emoticons or emojis. The use of non-standard language can have various reasons such as restrictions in message length, lack of language command, typing errors or social factors to express, for instance, the membership to a particular social group (Bartz et al., 2013; Eisenstein, 2013). Eisenstein

Phenomenon	Description
Acronym	<p>laugh out loud → lol be right back → brb are you → r u</p>
Contraction	<p>give me → gimme I am going to → imma going to → gonna</p>
Shortening	<p>tomorrow → 2mr good night → gn8 for ever → 4ever</p>
Repetition	<p>go → goooo no → nooooo haha → haaaaha</p>
Capitalization	<p>hello → heLLo, hElLo hey → HEY, hEY</p>
Markup	<p>@User #Hashtag</p>
Emoticons & Emojis	<p>:-P xD (^o^) (-_-)</p> 

TABLE 3.1: Examples of non-standard language use in social media

(2013) sees especially the social factors as reasons for using non-standard language *intentionally*, i.e. the language-use is part of the user's identity.

Syntactic Phenomena Social media postings might also have a poor syntax, i.e. intended or unintended grammatical errors. In posting such as “not going out tonight” the personal pronoun, e.g. “I am”, might be implied and textually omitted. A posting might contain a random number of emojis or emoticons that compose of punctuation marks, which also makes the full stop unreliable for finding sentence boundaries (Rudrapal et al., 2015). Thus, the syntactic structure of social media posting might also vary from formal sentences with a high syntactical quality.

3.2.1 Statistics on Text Domains

We now turn to comparing the lexical diversity and the distribution of PoS tags between formal, spoken and social media discourse to learn about the quantitative differences between these text domains.

Corpora We need reasonably large corpora of all three text domains to conduct such a comparison. For social media, no single, large, PoS annotated corpus exists. Therefore, we construct a larger social media corpus by combining three PoS annotated Twitter corpora (Gimpel et al., 2011; Ritter et al., 2011; Jørgensen et al., 2016) and an IRC chat corpus (Forsyth and Martell, 2007). The combined social media corpus has 78k tokens. We use the Wall-Street-Journal (WSJ) (Marcus et al., 1993) as formally written corpus and the Switchboard corpus for transcripts of speech. We select from the WSJ and Switchboard a random sample of sentences accounting for 78k tokens each, to reach a comparable corpus size between all corpora. The individual corpora use tagsets that are not directly comparable, which would complicate a comparison on PoS level. We, hence, harmonize the tagsets by mapping all tags to the Universal PoS tagset (Petrov et al., 2012) to obtain a comparable PoS annotation between the corpora.

PoS Distribution We start with an analysis of the PoS tag distribution between the three text domains. Social media platforms provide platform-specific features such as referencing other users or using hashtags, e.g. “@user1234 this was a punny day #Yolo #Vacation”. Furthermore, users make rather frequent use of emojis or emoticons. We annotate all emojis and emoticons in the social media corpus with the tag EMO, while the tag PLAT marks words which carry a special platform-specific meaning, i.e. user-mention, hashtag and retweets.

Figure 3.2 shows the results. The order of the domains is from formal, well-written to spoken language with the social media domain being placed in the middle as containing phenomena of written and spoken language. We see several stair-case effects in which the frequency of a PoS tag shrinks or increases between the three domains. Nouns are highly frequent in formal text but decrease in less formal text domains. At the same time, interjections that usually don’t occur at all in formal text, account for three percent points in social and even five points in spoken language. Also, pronouns and adverbs become more important in less formal domains and five percent of all tokens in social media carry a domain-specific functionality. The emoji and emoticon class is in relation rather small with barely more than 0.5 points of the total distribution mass.

Lexical Diversity Now, we turn to the variety in the used lexical forms. We compare again text of the three domains formal writing, social media and transcripts of speech to each other. Figure 3.3 shows the type/token ratio of the four major open word-classes, i.e. noun, verb, adjective and adverb, in the three domains. All

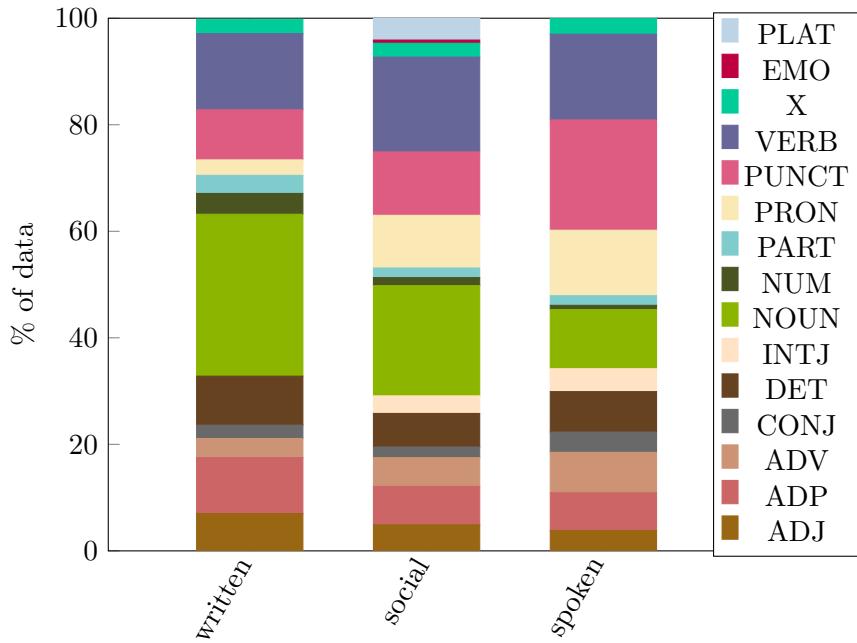


FIGURE 3.2: Comparison of PoS distribution between text domains

tokens have been lowercased for computing the type/token ratio. In social media, the variety in *nouns* is highest (high number of distinct nouns) followed by the spoken domain, and with the formal domain as having the lowest variety in nouns. This is explainable by the topical bias formal domains usually have. In our case, this bias is economical business English while the social media domain has no fixed topic. For *verbs*, we see that the domain of formal writing has more verb forms than the social media or spoken domain. While we find a higher number of verbs in Figure 3.2 for social media, we see now that those verbs are composed of few verb forms which are frequently repeated. For *adjectives and adverbs*, we see a similar distribution as for the verbs with minor differences between the domains.

Lexical Alterations When taking the standard language form of a word as reference and map all word variations to this standard form, we might have a lot less types in social media than the previous analyses allowed us to see.

We find in total 2,215 verb types (lowercased) in our social media corpus. A rather frequent phenomenon is to omit the final letter 'g' of gerund verbs e.g. "walking" becomes "walkin". When applying a simple correction rule that attaches this final letter, the number of verb types decreases to 2,132 - almost one hundred types less. We do not have such a correction rule for the other word classes. We, thus, manually screen the nouns in our social media corpus. We find many instances of the same standard-language type in which letters have been swapped, omitted or inserted. For instance, "weekend" occurs also as "weekenddd", in "line-up" the hyphen is omitted or in "guys" the last letter is substituted with the letter "z". For adjectives and adverbs, we find similar variations of the same standard language type. Thus, the number of types in social media is considerably inflated by spelling variations. When

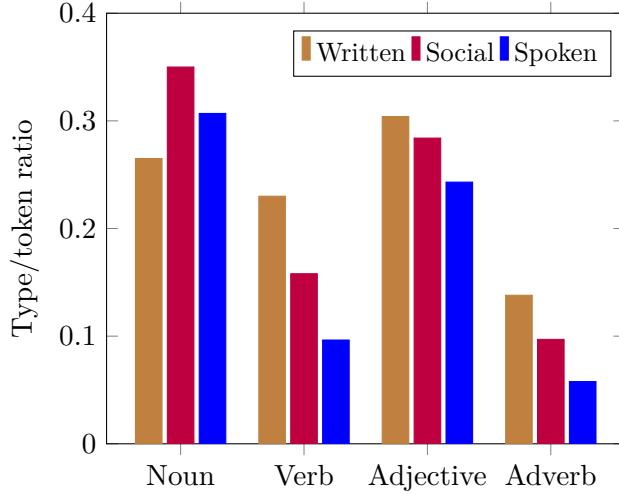


FIGURE 3.3: Type/token ratio for written, social and the spoken domain for the four major word classes

taking those variations into account, the type/token ratio of the word classes will be considerably lower than the pure quantitative analysis based on word forms in Figure 3.3 allows us to see.

3.3 Practical Challenges

We will now take a closer look on the performance of common PoS taggers on these three text domains. The decision for using a tagger or tagger model is often not just influenced by the expected tagging accuracy but also by speed. Many taggers come with several models that are optimized for different domains or offer trade-offs between accuracy and speed. Thus, instead of treating taggers as monolithic unit i.e. *Tagger X performs well*, we make a more fine-grained distinction and investigate the performance of taggers using a certain model on a certain text domain i.e. *Tagger X using model Y performs well on domain Z*.

Evaluation Domain We distinguish the two evaluation domains to evaluate domain robustness, which are shown in Figure 3.4. *In-domain* evaluation tests a model on text of the same domain as the training data of a model. *Out-of-domain* evaluation tests the domain transfer robustness by evaluating a model on foreign-domain text.

During model training, the model learns a weighted mapping from extracted feature values to a tag, i.e. a high weight means a feature value is highly discriminative for assign a certain tag. During tag prediction, the same features are extracted from the input text, the learned weighting from the model is applied and the most likely tag is predicted. The more similar the text is to the model training data, the more similar will be the extracted feature values to the values extracted during model training, which leads to a high accuracy. This also means that with growing dissimilarity of

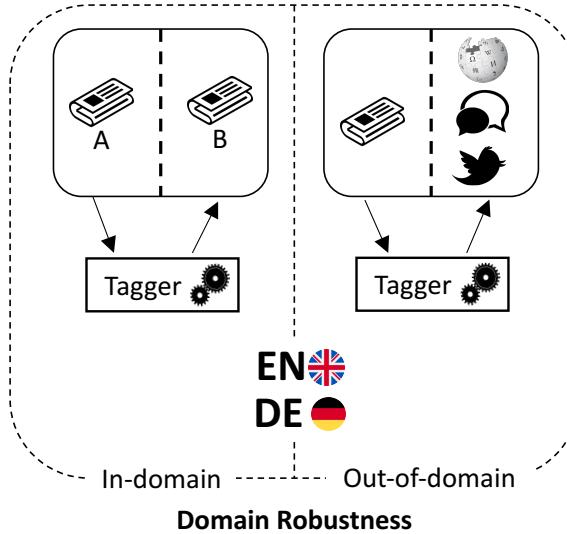


FIGURE 3.4: Evaluation for domain robustness

the input text to the training data, the larger the dissimilarity of extracted feature values will be, which challenges the model to make a correct prediction.

We use the following three text domains for this evaluation: *written*, which is orthographically correct text, *spoken transcripts*, which are conversations or monologs that might contain utterances of spontaneous speech and *social media text*, which is a mixture of text that ranges from formal to informal language use.

3.3.1 Experimental Setup

A large challenge for comparing a number of taggers is how to conduct this comparison in a feasible way. Manually installing each tagger is a time-consuming task. Furthermore, such a manual installation of each tagger places a high threshold on reproducibility. To solve this challenge, we use DKPro Core (Eckart de Castilho and Gurevych, 2014). DKPro Core is a Java-based project that uses the UIMA (Ferrucci and Lally, 2004) framework in its backend. DKPro Core provides wrappers for a wide range of taggers shielding the user from the details of installation and invocation of the taggers. The taggers are all installed as Maven artifacts, which requires no user-interaction and, hence, allows an easy replication on other computer systems. DKPro Core takes care that the taggers and their models are automatically installed on the user's computer. Furthermore, all taggers in DKPro Core are operated via a unified interface, which eases such a multi-tagger comparison.

Processing Pipeline In our setup, each corpus is read and transformed into the internal representation of DKPro Core, which is based on stand-off annotations. The wrapper transforms the internal representation of the text into the format which the tagger requires and transforms the tagged text back into the internal representation

for further processing. A final evaluation component compares the assigned tags to the gold tags from the corpus.

Directly before and after the tagger is invoked, we measure the time spent for tagging. This time measuring includes the time that the wrapper needs to provide the data to the underlying tagger implementation. In case of Java taggers, this is usually just a method call, but in case of wrapped C binaries there might be a considerable overhead. Thus, the runtime reported in this experiment might differ from running a tagger without the wrapper. A further issue that might affect the time measurement is document size. Some taggers are fastest when provided with small data chunks, while others are optimized for processing large chunks. In order to account for this difference, we run all experiments twice: (i) providing the corpus sentence-wise to the tagger, and (ii) providing all sentences at once. We then report the run that takes less time. Our interest lies in determining the relative time difference between tagger models. The absolute time differences are not of importance in this case. Thus, even when running this experiment on a faster computer, which will lead to a generally faster tagging, the relative difference between the models will still be the same. We conduct the experiments in an evaluation framework which is publicly available³.

Taggers and Models We now describe the PoS taggers and the models that we use (see Table 3.2 for an overview). If available, we provide information about the domain of the training data that is used to train the models.

Arktools (Owoputi et al., 2013). A tagger tailored to tagging social media text. Three models are available of which we use the one trained on annotated Tweets by Ritter et al. (2011), which uses an extended PTB tagset. The remaining two models are omitted as their training data are part of our evaluation set (which we discuss in the following section) i.e. a model trained on the data by Gimpel et al. (2011) and IRC chat data by Forsyth and Martell (2007);

Baseline Tagger A self-implemented baseline tagger which assigns the most frequent tag of a word according to the training corpus. Unknown word forms are tagged as noun. The English model is trained on section 0-18 of the WSJ, the German model uses the Tiger (Brants et al., 2004) corpus.

ClearNLP (Choi and Palmer, 2012). Two English models are provided. One trained on medical text and one trained on a mixture of text from various genres that are mostly news-related.

Hepple (Hepple, 2000). A rule-based tagger similar to the Brill-Tagger (Brill, 1992) for which an English model is available.

HunPos (Halácsy et al., 2007). An open-source reimplementation of the TNT tagger (Brants, 2000). Newswire models are available for English, trained on the WSJ and for German, trained on the Tiger corpus.

³<https://github.com/zesch/pos-tagger-evaluation.git>, last accessed 29 May 2017

Tagger	Lang.	Trained on	Modelname	Tagset	Domain	Abbr.
Ark	en	Ritter	ritter	PTB-RIT	social	Ark
Baseline Tagger	en de	WSJ Tiger	wsj0-18 tiger	PTB STTS	news news	Base
ClearNLP	en	Medical text OntoNotes	mayo ontonotes	PTB PTB	clinical news	C-1 C-2
Hepple	en	<i>rule-based</i>		PTB	-	Hepple
HunPos	en de	WSJ Tiger	wsj tiger	PTB STTS	news news	Hun
Mate	en de	CoNLL2009 Tiger	conll2009 tiger	PTB STTS	mixed news	Mate
LSTM	en de	WSJ Tiger	wsj0-18 -	PTB STTS	news news	LSTM
OpenNLP	en	<i>unknown</i> <i>unknown</i>	maxent perceptron	PTB PTB	<i>unknown</i> <i>unknown</i>	O-1 O-2
	de	Tiger Tiger	maxent perceptron	STTS STTS	news news	O-3 O-4
RfTagger	de	Tiger	tiger	Tiger	news	Rf
en	WSJ WSJ <i>unknown</i> WSJ	bidirectional-distsim casel.-left3w.-distsim fast wsj.-casel.-left3.-dis.	PTB PTB PTB PTB	news news <i>unknown</i> news	St-1 St-2 St-3 St-4	
Stanford	de	Negra <i>unknown</i> Negra Negra	dewac fast-caseless fast hgc	STTS STTS STTS STTS	news news news news	St-5 St-6 St-7 St-8
	en	<i>unknown</i> <i>unknown</i>	le le	PTB-TT STTS	news news	Tree

TABLE 3.2: English and German tagger models that we evaluate in our experiments for domain transfer robustness

LSTM Tagger⁴ (Plank et al., 2016). A PoS tagger based on Long-Short-Term-Memory (Hochreiter and Schmidhuber, 1997) neural networks. We train a model for English on WSJ section 0-18 and a German model on the Tiger corpus. We use the settings described in Plank et al. (2016) to train the models and use the pre-trained word embeddings by Al-Rfou et al. (2013).

Mate (Björkelund et al., 2010). Two models are provided. An English model trained on the CoNLL2009 (Hajič et al., 2009) dataset and a German model trained on the Tiger corpus.

OpenNLP. An Apache project that provides a wide range of NLP tools including a tagger.⁵ Two models for English and German are provided based on the algorithm Maximum Entropy and Perceptron. The German models are trained on the Tiger

⁴The tagger is implemented in Python and not easy to integrate into DKPro Core. Due to the good results reported for this tagger in the literature (Plank et al., 2016), we yet decided to add it to this evaluation. The time measurement is, thus, to be taken with caution as this tagger might have a speed advantage over the other taggers in our setup due to a direct execution without wrapper.

⁵<https://opennlp.apache.org>, last accessed 12 September 2017

corpus. The training data of the English models are unknown.

RFTagger (Schmid and Laws, 2008). A tagger for assigning morphologically fine-grained tags. We use the German model that is provided, which was trained on the Tiger Treebank (Brants et al., 2002).

Stanford (Toutanova et al., 2003). A popular tagger choice. Several models are provided for English and German. The models differ with respect to lowercasing of all tokens, adding distributional knowledge, or using a bidirectional model. We excluded two social media models trained by Derczynski et al. (2013)⁶ as they use training data which is part of our evaluation set.

TreeTagger (Schmid, 1994b, 1995). A tagger with a good reputation for German. Two models are provided, an English model trained on the Penn Treebank, and a German model trained on unknown proprietary training data.

Tagsets The tagsets assigned by the models and the tagsets used in the evaluation corpora are often not compatible. This mismatch will result in an artificially low accuracy. Many English models are trained on corpora annotated with the PTB tagset. Other English tagsets used by the models or the evaluation corpora are Brown (Nelson Francis and Kuçera, 1964), C5 (Leech et al., 1994) or the coarse-grained Gimpel (Gimpel et al., 2011) tagset. Furthermore, some of the PTB models use an extended PTB tagset. For instance, the English model by Schmid (1994b) assigns the inflection forms of the words *be*, *do*, *have* an own tag instead of the default verb tags. Ritter et al. (2011) added four additional tags to label the phenomena that frequently occur in Twitter messages such as hashtags or URLs, or Forsyth and Martell (2007) prefixed PTB tags with an extra character if the word-form is misspelled. In German, the *Stuttgart-Tübingen-TagSet* (Schiller et al., 1999) (STTS) with 54 tags is used for the corpora of written language. The corpora of the social and spoken domain use independent extensions of the STTS tagsets (Beißeneger et al., 2015; Rehbein, 2013) to account for domain properties not covered by the canonical STTS.

To harmonize the different tagsets, we map the fine-grained tags to the coarse-grained *universal tagset* (Petrov et al., 2012). Obviously, subtle distinctions between similar tags will be lost in the process, but for many downstream applications fine-grained distinctions between sub-tags of the same word class are not important anyway. The coarse-grained accuracy will provide a good approximation of the expected tagging quality and enable a direct comparison of the various tagger models on different corpora. Furthermore, it is more interesting to see if taggers are able to distinguish the main word classes such as adjective, adverbs, etc. on different text domains. Comparing on a coarse-grained tagset allows a comparison that has the principle distinctions between major word classes in the focus.

⁶<https://gate.ac.uk/wiki/twitter-postagger.html>, last accessed 12 September 2017

Lang.	Domain	Corpus	Tokens in (10^3)	Tagset
en	written	BNC-News	100	C5
		Brown	1,100	Brown
		GUM-News	9	PTB-TT
		GUM-Voyage	9	PTB-TT
		GUM-HowTo	13	PTB-TT
en	spoken	BNC-Conversation	100	C5
		GUM-Inverview	13	PTB-TT
		Switchboard	2,100	PTB
		Ted Talk	23	PTB
	social	Gimpel	27	Gimpel
de	written	NPS-Chat	32	PTB
		Twitter-AAVE	5	UT
	spoken	Tüba-DZ	1,500	STTS
		Hamburg-DTB	3,800	STTS
	social	Folk	100	STTS-EX-A
		EmpiriST-CMC	10	STTS-EX-B
		EmpiriST-Web	12	STTS-EX-B
		Twitter-Reh	20	STTS-EX-C
		Web-Comments	36	STTS

TABLE 3.3: Evaluation corpora of three text domains for English and German that are used in our experiments to evaluate the tagger models for domain transfer robustness

Corpora Table 3.3 gives an overview of the corpora used in our evaluation. We partitioned the evaluation corpora again into three domains: (i) formal writing, (ii) speech transcripts, and (iii) social media.

English The first set of corpora contains formal writing, e.g. news articles, travel reports and how to's. We use a subset of the newswire text from the British National Corpus (Leech et al., 1994) (BNC), the Brown corpus (Nelson Francis and Kuçera, 1964) containing American English of the 1960's and three subsections of the GUM (Zeldes, 2016) corpus. The second set contains transcripts of spoken language. We use the Switchboard (Marcus et al., 1993) corpus (telephone conversations), a subset of the British National Corpus with spoken language, one section with interviews taken from the GUM corpus and TED Talk (Neubig et al., 2014) presentation transcripts. The third set contains social media text. We use the IRC Chat corpus by Forsyth and Martell (2007), the Twitter corpus by Gimpel et al. (2011) and the African-American Vernacular English (AAVE) Twitter messages by Jørgensen et al. (2016).

In order to avoid testing on the training data, we exclude other available PoS annotated corpora such as the WSJ corpus (Marcus et al., 1993) or the Twitter corpus by Ritter et al. (2011), as many of the models have been trained using these corpora. As the provenance of some models is unknown, their results should still be treated with caution as we might accidentally be testing on training data.

German We use the STTS-annotated Tüba-DZ corpus by Telljohann et al. (2004) for the written domain based on the German newspaper *Die Tageszeitung* and the

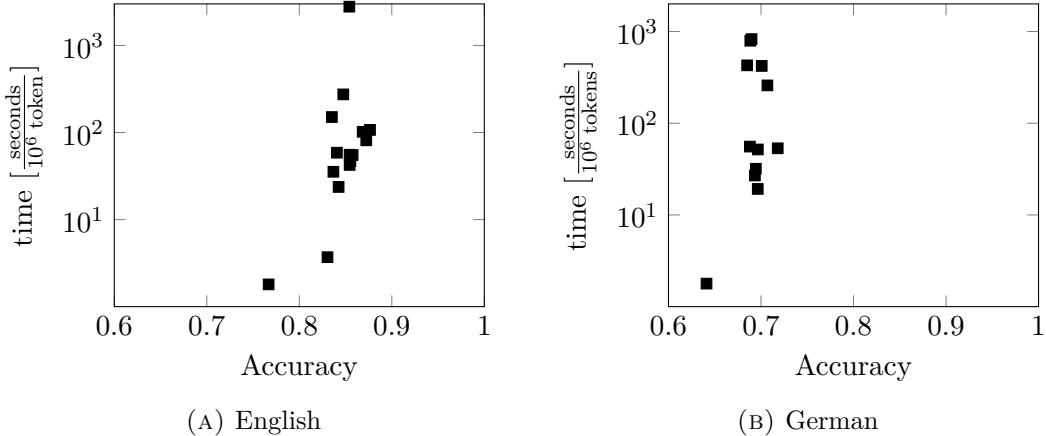


FIGURE 3.5: Macro-averaged results for each English and German model over all corpora and text domains

Hamburg Dependency Treebank by Foth et al. (2014a) with text from the technical news website *Heise.de*. For the spoken domain, we use the Folk corpus of spoken German by Westpfahl and Schmidt (2016). For the social media domain, we use the Twitter corpus Twitter-Reh by Rehbein (2013), the corpus of web comments on web articles by Neunerdt et al. (2013) and the Empiri corpus by Beißwenger et al. (2016) with a text mixture of various social media domains. We exclude the Tiger and the Negra (Skut et al., 1998) corpus as all German models are trained on one of the two.

3.3.2 Results

Figure 3.5a shows the English results over all corpora and Figure 3.5b shows the German results. The x-axis shows the macro-averaged tagging accuracy based on the coarse-grained universal tagset. The y-axis shows the normalized processing time in seconds per million tokens.

The average tagging accuracy fails to reach the high accuracy results reported in the literature of around 97% on formal corpora. In particular, the performance on the German corpora show an extremely large accuracy drop. This drop results from the spoken domain corpus which we discuss in more detail later on. A surprising finding is that the taggers and models differ more on the time-axis than on the accuracy axis. Thus, researchers need to choose according to their needs. When the focus lies only on the quality of the tagging, taking a slower model which is a bit more accurate is a reasonable decision. When working in large scale data processing setups one might choose a faster but less accurate model as trade-offs between speed and accuracy.

So far, we have only considered the macro-averaged performance over all corpora. This simulates the usage scenario in which the tagger is treated as a black-box and applied to all sorts of data without caring much about the domain. Next, we compare the tagger performance per domain.

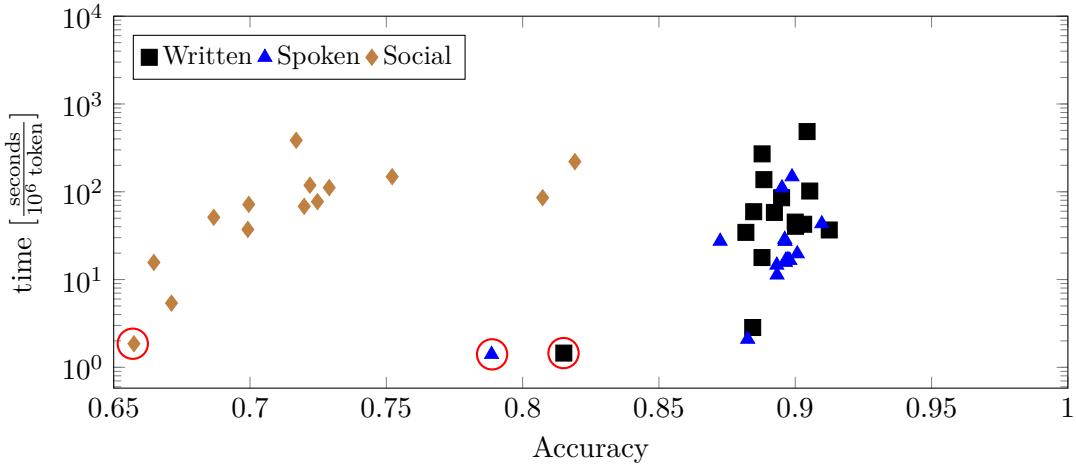


FIGURE 3.6: Results of English tagger models on formal text, transcripts of spoken language and social media text. Red circles show the *Baseline* tagger’s performance

English: Domain-specific Results Figure 3.6 shows the English evaluation results per domain, Table 3.4 shows the exact values. We see that the in-domain evaluated models perform considerably better than when evaluating on out-of-domain data. With most English models being trained on newswire data, these models perform best on formally written corpora. However, even in an in-domain evaluation no model reaches the 97% accuracy reported in the literature when testing on data from the same corpus as the training data. On social media, the tagging accuracy of some taggers drop to such an extreme extent that they degenerate almost to the baseline tagger’s performance. A surprising finding is that tagging spoken transcripts does not seem to differ much from tagging written language corpora. One would expect that transcripts of spoken language are considerably more difficult to tag than formal text. A manual screening of the corpora confirmed only a low number of informal utterances, which makes these corpora rather similar to formal text (we show examples further below when comparing the English results to the German results).

Hence, tagging formal, written discourse is supported best by tagger models. Almost all models are not equipped to deal with non-standard language, except models which have been already adapted to informal domains.

German: Domain-specific Results Figure 3.7 and Table 3.5 show the German results. The most models perform well on the written domain but have substantial loses in accuracy on other domains. All German models have been trained on formal text. Unlike for English, the German models perform rather close the 97% accuracy reported in the literature. We see that the spoken domain is by far the most challenging one in the German setup. Many taggers drop below the baseline tagger. The spoken evaluation dataset composes of a single corpus, the Folk corpus. The Folk corpus contains many colloquial and slang expressions which make this corpus particularly difficult to tag. Table 3.6 shows examples from the German Folk corpus compared to the English corpora to highlight the differences between the corpora in

	Written		Speech transcripts		Social media		Macro-average	
	accuracy ø %	time ø (seconds 10 ⁶ token)	accuracy ø %	time ø (seconds 10 ⁶ token)	accuracy ø %	time ø (seconds 10 ⁶ token)	accuracy ø %	time ø (seconds 10 ⁶ token)
Ark	89.2	58	89.6	27	80.7	86	87.2	81
Base	81.5	1	78.9	1	65.7	2	76.7	2
C-1	88.5	59	87.2	27	72.5	77	84.1	59
C-2	90.5	102	91.0	43	75.2	149	86.9	102
Hepple	88.4	3	88.3	2	67.1	5	83.0	4
Hun	88.8	18	89.3	11	69.9	37	84.2	24
LSTM	88.9	137	89.6	16	66.5	16	83.5	151
Mate	88.8	270	89.5	111	71.7	386	84.8	274
O-1	90.0	45	90.1	20	72.0	68	85.5	47
O-2	88.2	34	89.3	15	68.7	51	83.7	35
St-1	90.4	485	89.9	148	71.1	10040	85.4	2788
St-2	90.3	43	89.7	17	72.9	111	85.8	55
St-3	89.5	85	89.6	29	81.9	220	87.6	107
St-4	90.0	40	89.7	17	72.2	119	85.4	56
Tree	91.2	37	89.8	17	70.0	72	85.4	42

TABLE 3.4: Tagging accuracy and execution time of the English models as averaged values per text domain. Best accuracy values are highlighted in bold face

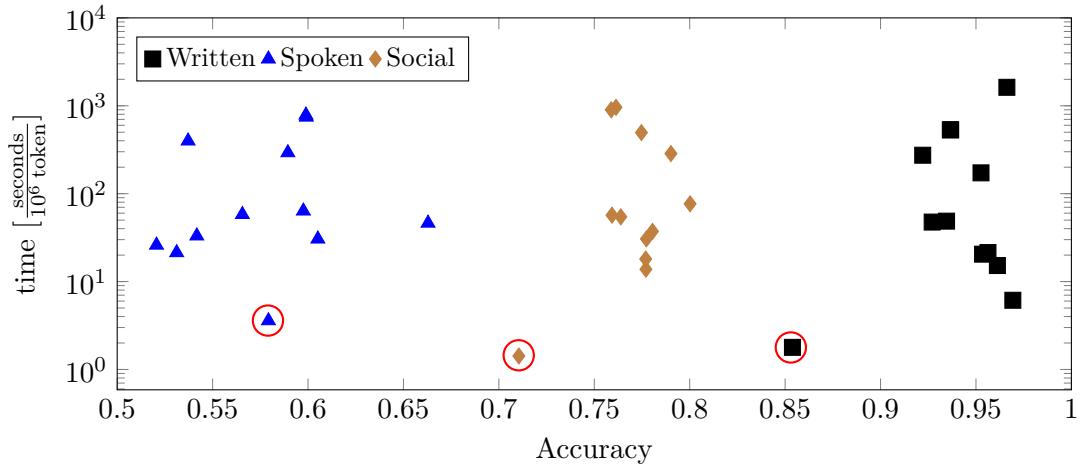


FIGURE 3.7: Results of German tagger models on formal text, transcripts of spoken language and social media text. Red circles show the *Baseline* tagger's performance

	Written		Speech transcripts		Social media		Macro-Average	
	accuracy ø %	time ø (seconds 10 ⁶ token)	accuracy ø %	time ø (seconds 10 ⁶ token)	accuracy ø %	time ø (seconds 10 ⁶ token)	accuracy ø %	time ø (seconds 10 ⁶ token)
Base	85.4	2	57.9	4	71.0	1	64.1	2
Hun	96.1	15	54.2	33	77.7	18	69.6	19
LSTM	96.6	1623	56.6	58	77.7	14	70.1	422
Mate	95.3	173	58.9	293	79.0	286	70.7	258
O-1	95.6	21	52.1	26	78.0	37	69.4	32
O-2	95.3	21	53.1	21	77.7	31	69.3	27
Rf	92.2	274	53.7	398	77.5	495	68.5	428
St-5	93.7	533	59.9	784	75.9	900	68.8	794
St-7	93.5	49	59.8	64	75.9	57	68.8	56
St-6	92.7	48	66.3	46	76.4	55	69.7	52
St-8	93.7	532	59.9	743	76.1	960	69.0	826
Tree	96.9	6	60.5	30	80.0	77	71.8	53

TABLE 3.5: Tagging accuracy and execution time of the German models as averaged values per text domains. Best accuracy values are highlighted in bold face

German examples of spoken language	
Folk	<p>nu kuck ma da hinten trinken se tee <i>Standard: nun, guck mal, da hinten trinken sie Tee</i> ENGLISH: WELL, LOOK OVER THERE, THEY ARE DRINKING TEA</p> <p>ja des geb isch ihne vor äh 500 bis 1.500 ohm <i>Standard: Ja, das gebe ich Ihnen vor, 500 bis 1.500 Ohm</i> ENGLISH: YES, I WILL PROVIDE THAT TO YOU, 500 TO 1,500 OHM</p>
SWITCHBOARD	<p>I'd be very very careful and, uh, you know, checking them out. Uh, it had to be done in hurry.</p>
TedTalk	<p>But on the other side of that, though, we are big readers in our house. Or we 'll be here all day with my childhood stories.</p>
GUM-Interview	<p>How did you come to be involved with this discovery. Are you intending to go there yourself ?</p>
BNC-Conversation	<p>Oh very nice, very nice, yes. er, anyway we 're alright now so, you know</p>

TABLE 3.6: Examples of transcripts in the German Folk corpus with many slang utterances compared to the transcripts in the English corpora with spoken language that contains mostly standard English

the German and English evaluation setup. The English corpora, despite of being transcripts of spoken language, are close to standard English and show no slang utterances as in the German corpus. Thus, informal utterances pose a huge challenge for tagging. It is, thus, not surprising to find the average tagging accuracy to be located between the spoken and formal domain corpora. Text from the social media domain contains both, informal and formal utterances.

3.4 Chapter Conclusion

In this chapter, we discussed the challenges of informal text domains for PoS tagging. We started with a discussion of the term social media and its related terms Web 2.0 and CMC that are different notions of non-professional, user-generated (textual) content. In a theoretical analysis, we investigated the difference between text of the formally written, spoken and social media domain with respect to the occurrence of

word classes and lexical diversity. We found that the domains differ considerably with social media being rich in spelling variations of individual words. In a practical analysis, we empirically evaluated 11 PoS taggers with 18 models for English and German on text of the aforementioned three text domains. We found that existing taggers are well equipped for tagging formal text but tagging accuracy drops considerably if a text contains many properties of informal, colloquial language use. In German, tagging of the Folk corpus performed even worse than tagging of the social media corpora. This observation was not possible in English for a lack of annotated corpora that contain such informal utterances.

A rather fundamental issue is also the dominance of formal newswire text for training the models. In particular, the WSJ in English and the Tiger corpus in German are the most frequently used corpora for training models. There are barely alternatives to these corpora and if they are, they are often still of a formal nature. Hence, none of these models, neither for English nor German, have encountered phenomena of informal utterances during model training. When applied to less formal text domains, these models perform only poorly.

Chapter 4

Domain Robustness - Existing Approaches

In this chapter, we investigate the effectiveness of approaches that have been proposed in the literature to improve domain robustness of tagger models. As we saw in Chapter 3, the accuracy of available taggers and their models varies with the text domain to which they are applied. While the tagger models perform decently as long the models are applied to in-domain text, they lose a considerable amount of tagging accuracy when applied to foreign text domains. In particular, the social media text turned out to be a highly challenging domain for the available taggers. This poor robustness gave rise to various approaches to improve domain robustness that we categorize into two paradigms: normalization and domain adaptation. *Normalization* removes orthographic and syntactical anomalies of a text and brings them into its standard form (Han and Baldwin, 2011; Chrupala, 2014). The text is adapted to the tagger, which enables newswire trained PoS tagger models to perform well. *Domain Adaptation* focuses on a re-training strategy and trains new models that are more similar to the social media text, i.e. the tagger is fitted to the text. We will focus on the domain adaptation approaches that we evaluate and analyze in detail to learn where and how they achieve improvements. We also investigate the validity of these approaches by constructing social media adapted models for the English, German and Italian language.

Normalization vs. Domain Adaptation Figure 4.1 shows an example for tagging social media text by using normalization versus domain adaptation. *Normalization* is probably the more challenging paradigm as it often requires a semantic interpretation and entails two tasks, first detection that a non-standard form is present (or text has been omitted) and second transforming this form into its standard form. Information such as emoji or emoticons do not have a standard form which requires additional strategies how to deal with those phenomena that cannot be substituted. Furthermore, word contractions of two or more words such as *gonna* have to be split up again into separate words and the possible misspellings of words have to be detected and corrected. A normalized social media posting allows to apply models trained on formal text without having to expect a severe drop in accuracy. This assumes

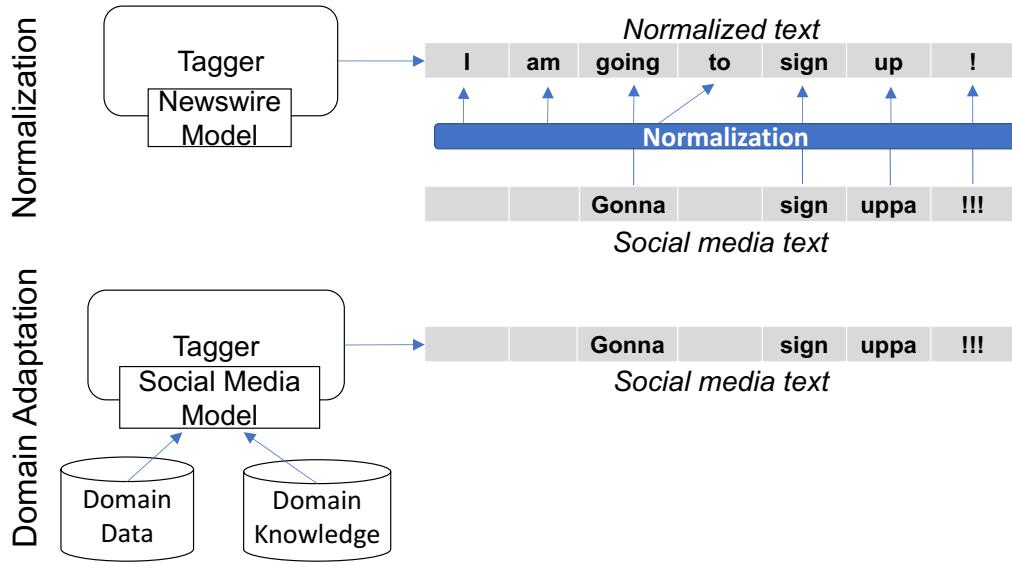


FIGURE 4.1: PoS tagging by normalization versus domain adaptation

that the normalization process provides an appropriate standard form replacement for each non-standard form. This is essentially a coverage problem of providing many mappings from non-standard word forms to a canonical standard form. Normalization has been used for improving PoS tagging accuracy (not just on social media data) (Li and Liu, 2015; Yang and Eisenstein, 2016; van der Goot et al., 2017). Li and Liu (2015) takes a joint approach and combines the normalization step with PoS tagging, instead of first normalizing and then applying the tagging as in a pipeline architecture. They obtain normalization candidates from several normalization models and determine the most likely tag for a word by applying a Viterbi (Viterbi, 1967) decoding that also considers a normalization of the word with one of its candidates. They use datasets that are annotated with PoS tags and normalized word forms. Yang and Eisenstein (2016) works on historical English and uses spelling normalization to achieve improvements on early modern English text. van der Goot et al. (2017) also investigates normalization for PoS tagging. They consider words as normalization candidates that are close in the embedding space to each other (including the canonical word form). A random forest classifier ranks each normalization candidate. They also experiment with normalizing only unknown words (from the training data) or all words. They find their normalization method to reach substantial improvements for PoS tagging and that normalizing all words is of advantage.

Domain Adaptation works directly on the social media text. Instead of modifying the text, the model of the PoS tagger is adapted to the social media domain. Hence, domain adaptation sustains all the phenomena of the social media domain. Ideally, one has a large annotated social media corpus and can directly train a new model on (social media) domain text. At the moment, no such large corpus is available which complicates the task to train an adapted model. In case of social media, the

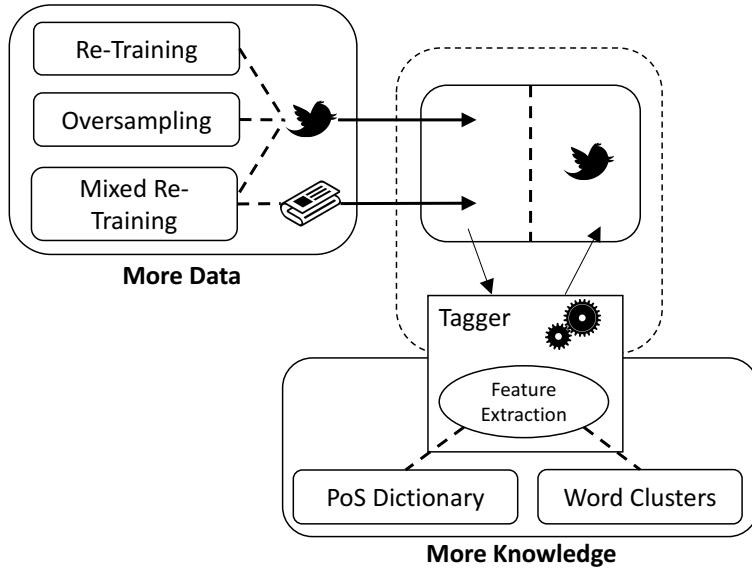


FIGURE 4.2: Overview of domain adaptation approaches

number and size of annotated datasets are considerably smaller than for other text domains. Existing datasets have usually a size of few thousand tokens, which is not sufficient for training models. We will, thus, focus hereinafter on the domain adaptation paradigm and how to tackle the lack of training data.

Domain Adaptation Approaches Two domain adaptation strategies surfaced to deal with the lack of training data. These strategies are shown in Figure 4.2, i.e. adding *more data* and adding *more knowledge*. *More data* tackles the lack of training data by adding foreign or machine-generated data (Ritter et al., 2011; Derczynski et al., 2013). Foreign domain data might be only of limited use as foreign domain text provides no new information for tagging social media phenomena. Automatically producing new social media data is inexpensive but the automatized production process is flawed to some extent, which makes it difficult to estimate the number and the impact of incorrect examples. *More knowledge* adds information obtained from resources, which usually cannot be directly used as training data (Ritter et al., 2011; Owoputi et al., 2013). The available training instances are enriched during model training by injecting knowledge from those resources into the machine learning process. The first strategy affects from *which* data is learned, the second one *what* is learned.

More Data With only little annotated data from the social media domain, directly *re-training* a model is not effective. Also, due to the high variance of word forms in this domain, annotating more data is less effective than in the news domain. More social media data would certainly be helpful. However, this would not solve the general problem of having to deal with many spelling variations in this domain. An obvious approach to improve performance is to add annotated training data from another domain (usually newswire text) what we call *mixed re-training*. However, as

the much bigger foreign-domain data easily dominates the tiny amount of annotated social media data, *oversampling* (Daumé III, 2007; Neunerdt et al., 2014) might be used to adjust for the difference in size by adding the social media data multiple times. *Voting* (Goldman and Zhou, 2000; Derczynski et al., 2013) is another approach that provides more social media training data relying on multiple already existing PoS taggers. If all taggers assign the same PoS tags to all tokens of a sentence, it is added to the training data. The assumption is that the mutual agreement between several (foreign domain) tagger models compensates for the increased error-rate of a single tagger on social media data.

More Knowledge Instead of adding more training data, one can also try to inject knowledge from outside. A *PoS dictionary* (Rehbein, 2013) provides information about the most frequent PoS tags of a word. The PoS distributions are compiled from an external knowledge source, i.e. large corpora that are either manually annotated or machine-tagged. A further approach to get more knowledge about the social media domain is *clustering* of social media data. Clustering tends to place spelling variations of the same word into the same cluster (e.g. tomorrow, tmr, 2mr, tmrrow, etc.), which improves handling of unknown word forms or spelling variations of words (Chrupala, 2011; Ritter et al., 2011; Owoputi et al., 2013).

Experimental Setup We conduct our experiments with the tagger FlexTag (Zesch and Horsmann, 2016) using Conditional Random Fields (Lafferty et al., 2001), we use an Averaged Perceptron (Collins, 2002) as training algorithm. As feature set that is common to all subsequently described evaluations, we use a tri-gram word context, the 750 most frequent character bi-grams, tri-grams and four-grams, the length of the token and features that check if a token is an emoticon, smiley, a number, a user-mention or a hashtag. As social media corpus, we use the TWITTER corpus by Ritter et al. (2011) with 15k tokens. We also report results on the coarse-grained Universal PoS (Petrov et al., 2012) tagset by mapping the results of the fine-grained PTB tagset. With several approaches to our disposal, we will evaluate the effectiveness of each approach in isolation and then compare them in combination.

4.1 More Data

We now describe and evaluate the strategies that rely on changing the training data of the tagger: re-training, oversampling, and voting.

Re-training In our first experiment, we investigate the effect of re-training. Due to the limited amount of social media data we have, we use 10fold cross-validation for evaluation and report averaged results. On the fine-grained PTB tagset, we reach an accuracy of 79.8%, on the coarse-grained tagset 86.0%. The huge difference between

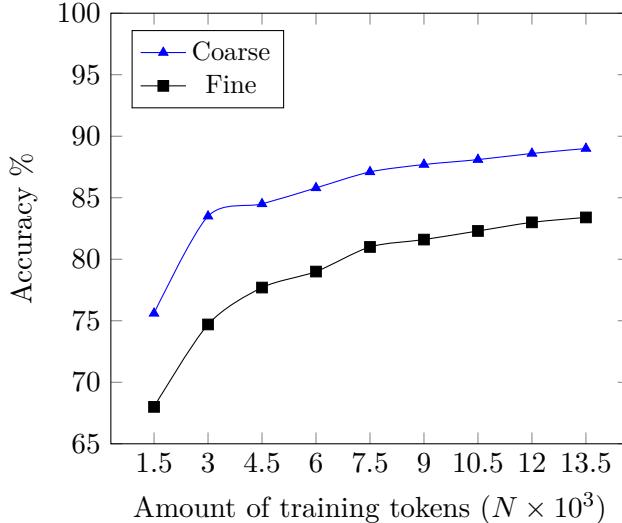


FIGURE 4.3: Re-training learning curve (10fold CV)

fine- and coarse-grained shows that a substantial amount of the errors must be confusions between the fine-grained tags belonging to the same (coarse-grained) word class, e.g., confusions of a present tense verb from with a past tense verb form. Obviously, more annotated social media training data should further improve the results.

Figure 4.3 shows a learning curve to estimate the potential of providing more data. We compute the learning curve by splitting TWITTER into ten data chunks and evaluate against a holdout chunk. We then add one additional data chunk to the training dataset in each iteration to measure the improvements of adding more data. We repeat this whole process ten times (we 10fold cross-validate ten times) to ensure that each data chunk has been in the test set once and report averaged result. The curve confirms our assumption that more data would lead to further improvements. However, the expenses of annotating more data are often too high to follow this approach. Thus, we analyze the effectiveness of less expensive but more sophisticated domain adaptation strategies.

Mixed Re-Training A quite obvious approach is training on a mixture of formal text and TWITTER, which we call *mixed re-training*. We experiment with adding newswire text from the Wall-Street-Journal (WSJ), which has a compatible tagset to the one used in TWITTER. We provide an increasing number of tokens to observe the effect of foreign domain data on accuracy. As we only want to test on TWITTER, we use a specialized version of cross validation where we cross-validate only over TWITTER but add newswire data each time. The results are shown in Figure 4.4.

After adding of 300k additional tokens, the fine- and coarse-grained tagging accuracy has improved by about two percent points. This result clearly show that the tagger benefits from more language knowledge, even if it is from a foreign text domain. However, we also see that using considerably less data, 100k, reaches an almost identical result by a learning curve that flattens out early. Since the tagger

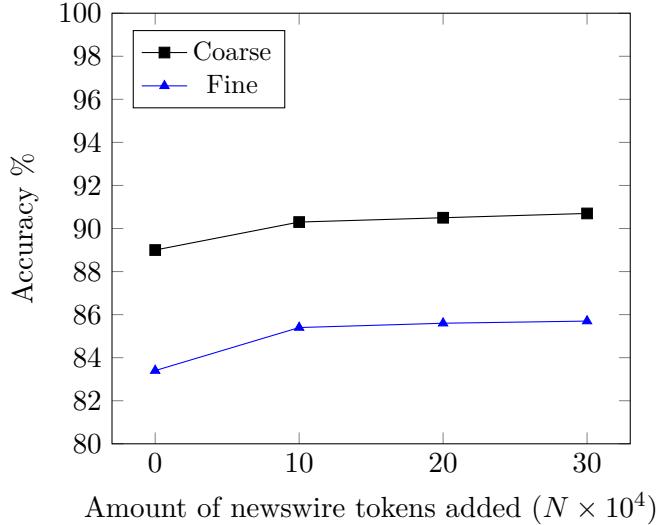


FIGURE 4.4: Results of mixed re-training (10fold CV)

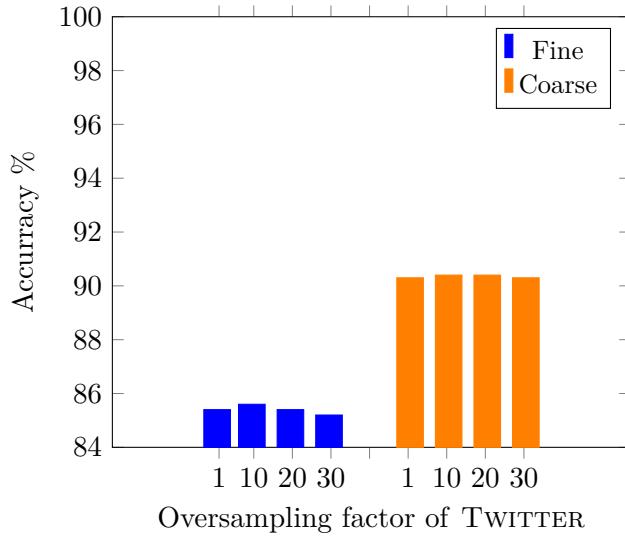


FIGURE 4.5: Results of oversampling (10fold CV)

cannot learn any social media phenomena from the newswire text we added, the improvements we see must be accounted to an improved lexical knowledge about the language. Thus, mixed re-training is quite effective to improve performance at least for such small datasets as the TWITTER dataset.

Mixed Re-Training With Oversampling A possible problem with mixed re-training is the large differences between the dataset size of the foreign domain corpus to the small social media corpus. The syntactical and orthographic properties of the foreign-domain corpus outweigh the actually more accurate knowledge by the social media data. To overcome this imbalance, *oversampling* has been proposed (Daumé III, 2007; Neunerdt et al., 2014). The idea is to boost the importance of the social media training data by adding it multiple times to the training dataset to artificially increase the weight of the more accurate information during model

training. A parameter of this approach is the factor of how often the social media data is added additionally. We base the mixed re-training experiment on 100k foreign domain data from the WSJ and evaluate again by 10fold cross-validation only over the TWITTER corpus.

Figure 4.5 shows the results for different oversampling rates in comparison to mixed re-training without oversampling and the re-training baseline. We see no improvements by oversampling the TWITTER corpus.

Voting Voting aims at producing more annotated training data by using several PoS taggers. Postings where all tagger agree are added to the training dataset. This is similar to the approach by Zhou and Li (2005), they accepted automatically labeled training samples for a third classifier as training data when two other classifiers agreed on the labeling of a data sample. The idea behind this procedure is that (newswire-trained) PoS taggers still tag a large proportion of the tokens correctly. If all taggers assign the same label sequence (i.e. all *voted* the same) the sequence is added to the training set. Thus, the lack of confidence of a (newswire-trained) tagger is overcome by using several ones. We use three existing taggers with models that use the PTB tag set: ClearNLP with the model *ontonotes*, OpenNLP with the model *maxent* and the Stanford tagger with the model *caseless-left3words-distsim*. We manually set the PoS tags for Twitter phenomena such as user-mention, hashtags, urls, etc. in a post-processing step by regular expressions (Ritter et al., 2011). We 10fold cross-validate again over TWITTER and add an increasing amount of *voted* sequences to the respective training dataset. We compare the effect of *voted* data to adding the same amount of human-annotated newswire-data from the WSJ.

The results are shown in Figure 4.6. We see small improvements when adding voted data. After adding of 70k tokens of produced data, we reach an improvement by about two percent points. Providing even more data is unlikely to improve results even further. Using hand-annotated newswire data performs similar to adding *voted* social media data. This might be surprising at first but is actually rather reasonable because voting is projecting models that have been trained on newswire on the social media data. The sequence where all newswire-trained models agree will inevitably be extremely similar to newswire text. Sequences with phenomena typical for the social media domain, which make this domain challenging, are also the postings where the taggers disagree.

4.2 More Knowledge

PoS Dictionaries We use dictionaries that store the PoS distribution for each word form as it occurs in a corpus. The underlying corpus can either be manually annotated or machine-tagged. Gimpel et al. (2011) combined the WSJ and the Brown corpus (Nelson Francis and Kuçera, 1964), Plank et al. (2014) used a crowd-sourced online dictionary, and Rehbein (2013) used the machine tagged WaCky corpus (Baroni

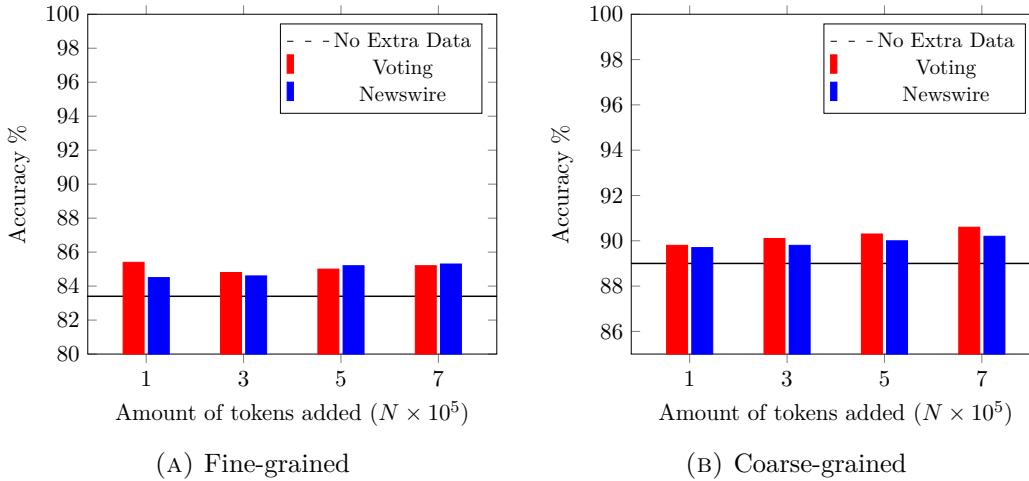


FIGURE 4.6: Voting versus mixed-retraining (10fold CV)

et al., 2009). We use two dictionaries in our experiments: HumanDict¹, created from the human-annotated Brown corpus, and MachineDict², created from the first 100 million tokens of the machine-tagged English WaCky corpus (Baroni et al., 2009). The WaCky corpus contains text crawled from the Internet such as forums, which makes the text of this domain more similar to TWITTER than the Brown corpus with its formal nature. HumanDict has a vocabulary coverage³ of 74.3% of the tokens in TWITTER and MachineDict has 85.1%. The dictionaries store for each word the three most frequent PoS tag according to the source corpus. We provide this information by introducing three additional features for each dictionary entry. If a word has less than three PoS candidate tags, a constant value is provided.

The results are shown in Figure 4.7. Surprisingly, both dictionaries improve the performance almost equally. The machine dictionary offers a small advantage, which we account to the better coverage of the domain vocabulary. We assume that the lower quality of the annotation in MachineDict eventually leads to a comparable performance between both dictionaries.

Clustering Providing knowledge from word clusters has been reported to account for substantial improvements (Ritter et al., 2011; Owoputi et al., 2013; Rehbein, 2013). The clusters are created by grouping words according to their word distributional similarity. The main idea is that words that are placed into the same cluster, i.e. occur in a similar word context, also share the same word class. This cluster label is provided for each word during model training. In a prediction task, finding a spelling variation of a word in the same cluster as a word form encountered during model training provides a bias to assign the spelling variation the same tag as the known word form. This assumes that the unknown word was seen during clustering, i.e. clusters are only helpful if a spelling variation occurred during word clustering. This

¹contains 54k entries

²contains 1,000k entries

³all words in the dictionary and corpus have been lower-cased before computing the coverage

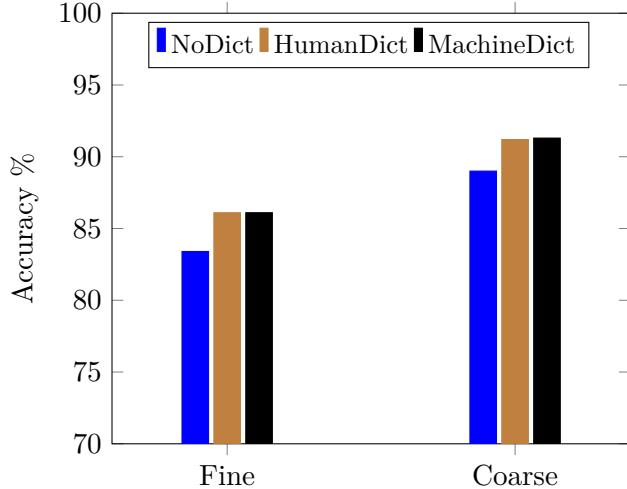


FIGURE 4.7: Results of using PoS dictionaries (10fold CV)

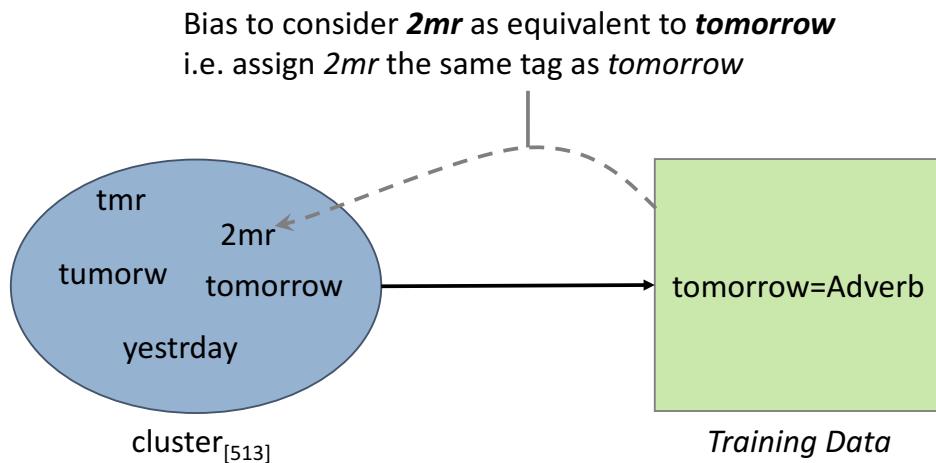


FIGURE 4.8: Example how the knowledge obtained from word clusters provide a bias to deal with unknown word forms that did not occur in the training data

is shown in Figure 4.8 which assumes that only the standard English form *tomorrow* is known from the model training phase. The cluster knowledge provides a bias to treat the out-of-vocabulary word *2mr* the same way as *tomorrow* because both words have been placed into the same cluster i.e. share a highly similar word context. Clusters created over social media text, e.g. Twitter messages, provide a high coverage of many spelling variations of the same word and, hence, offer improvements for tagging social media text. Hence, using clusters is well suited for tackling the problem of facing many spelling variations of words.

We experiment with two clustering algorithms: LDA⁴ (Blei et al., 2003; Chrupala, 2011) and hierarchical Brown clustering⁵ (Brown et al., 1992). Brown clustering has been used by Owoputi et al. (2013) with 56 million tweets (800 million token) and by Ritter et al. (2011) with roughly comparable 52 million tweets. Rehbein (2013)

⁴<https://bitbucket.org/gchrupala/lda-wordclass/>, last accessed 8 May 2017

⁵<https://github.com/percyliang/brown-cluster>, last accessed 8 May 2017

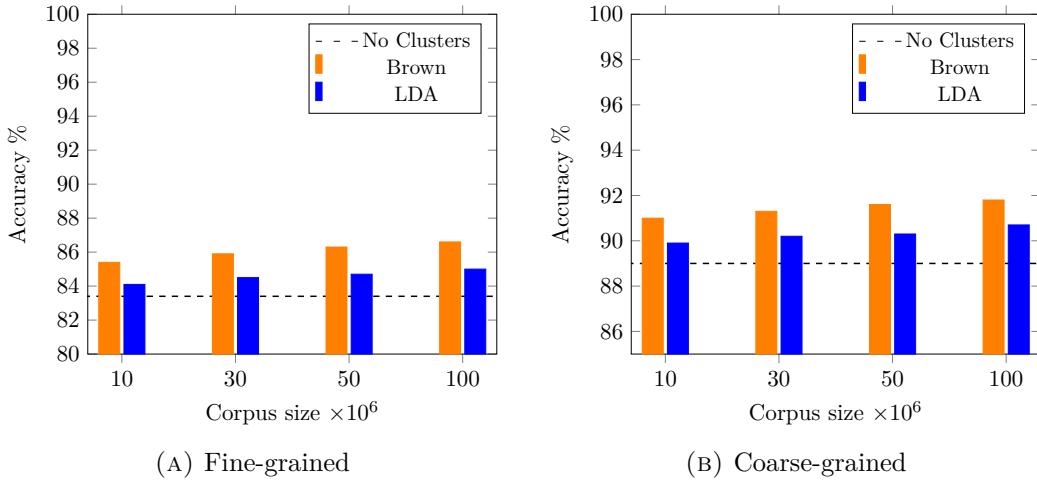


FIGURE 4.9: Improvements by using Brown and LDA clusters (10fold CV)

used LDA clustering over 200 million tokens of German Twitter messages. To learn how many tokens of plain text we need, we experiment with clusters created from 10, 30, 50 and 100 million tokens of Twitter plain text that have been sampled between the years 2011 and 2017 to avoid a time-bias. Occurrences of Twitter-specific phenomena like at-mention, URL, and hashtags have been replaced with constant values before applying the clustering algorithm and all tokens are lowercased. We use the parametrization for the cluster algorithms that we find in the literature (Rehbein, 2013; Owoputi et al., 2013), i.e. we create 1,000 Brown clusters with a minimal word frequency of 10, and 50 LDA clusters with a minimal word frequency of 10.

Figure 4.9 shows the results on fine-grained and coarse-trained tags. Both approaches account for improvements of several percent points with Brown clustering reaching a higher accuracy in all cases. The best result on the fine-grained tagset is achieved by the largest source corpus with 100 million tokens. The slope indicates further (minor) improvements if more data is added. The most relevant information is obtained by using already a few million tokens which seem to sufficiently cover the most common spelling variations of this domain. We hypothesize that the overall better performance by Brown clusters is explainable by the way the clusters are identified. Brown clustering is a hierarchical clustering algorithm and cluster ids are bit-code ids. Clusters, thus, contain information about higher or lower similarity to other clusters by comparing the overlap of those ids. This additional information about (partial) similarity between the clusters provides additional information to the classifier. The details how the bit-string information is provided plays an important role. We experimented with various methods and found that providing the bit string information as prefixes of increasing length to work best⁶ (Miller et al., 2004; Koo et al., 2008; Owoputi et al., 2013).

⁶i.e. for each word with an entry in the created word clusters, we provide for each word several cluster features each holding the bit-string in an increasing length, i.e. 2, 4, 6,..., N.

4.3 Combining Approaches

While the strategies for *more knowledge* are easy to combine, it is not reasonable to combine all *more data* strategies, i.e. voting and oversampling did not show advantages that would justify the effort or prolonged training time of these methods. Thus, we combine re-training a tagger on the TWITTER dataset using the PoS dictionary and word clusters as resources of the *more knowledge* strategy, and provide additional 100k *more data* of newswire text.

In Figure 4.10, we show a comparison of the combined approach to all other approaches. Furthermore, we make a distinction between all tokens and only OOV⁷ tokens. The combined approach reaches the best accuracy on the fine-grained and coarse-grained tagset, and also on OOV words, which shows that the individual approaches add up well.

Improvements by Tag So far, we focused on the accuracy as evaluation metric. Some approaches accounted for quite large improvements and increased the accuracy by several percent points. To learn more about which tags improve using which approach, we will compare the best working configuration of each approach by computing the F-Score for each tag. For reasons of readability, we focus on the fine-grained tags belonging to the four major word classes adjective, adverbs, nouns, and verbs. All reported results are again averaged results computed by 10fold cross-validation over the TWITTER corpus. We report for each tag its frequency in the TWITTER corpus as reference for the relative importance of a tag.

The results are shown in Table 4.1, bold faced results indicate the best number achieved for a tag. The best individual approaches are highlighted in gray. We see that all approaches have their largest effect on noun and verb related tags. We suspect that many improvements are due to added language knowledge as the TWITTER corpus is standalone just too small. The improvements by *100k Mixed RT* provide more lexical knowledge (from a standard English text domain), which should allow improvements on standard English words that also occur in *Twitter* but might not have occurred in the training data. The *Voted Data 70k* approach confirms this, too, as this approach is highly similar, i.e. the created annotated data is similar to newswire text. *Oversampling x10* achieves similar results by overweighting the same information multiple times. Thus, either the strategies improve by providing more standard language knowledge or by overweighting the small social media training data. In almost all cases either the PoS dictionary or the word clusters account for the largest improvements, which is not surprising as these two resources contain standard and non-standard language knowledge. Furthermore, the combination of the approaches improves tagging on verb related tags. In particular, cases such as *VBN* (past participle verb) that require local context in order to be disambiguated from a normal past tense verb improve when all data and knowledge sources are combined.

⁷We define *known vocabulary* as (lower-cased) words which occur in the training part of the TWITTER corpus.

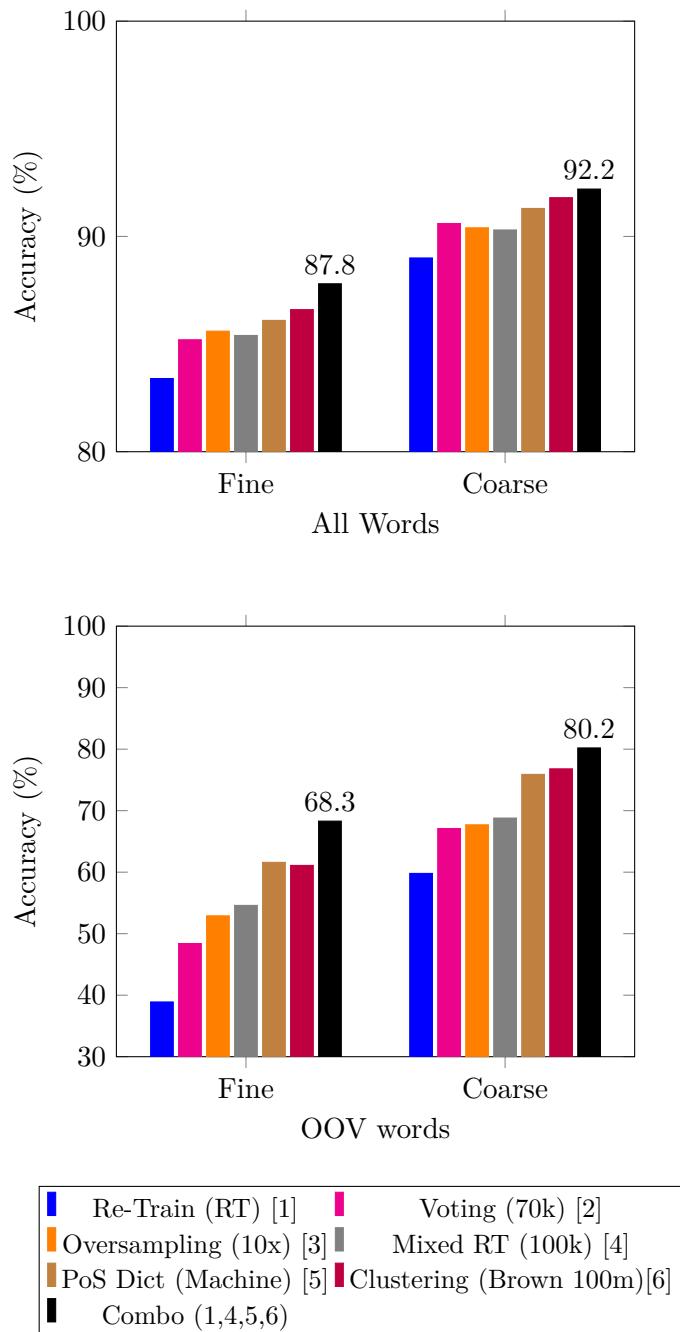


FIGURE 4.10: Results for each approach and in combination on the English dataset for all and OOV words (10fold CV), the numbers in round brackets for *Combo* refer to the approaches that have been combined

PoS Tag		Re-Train Freq.	100k Mixed RT	Over-sample x10	Voted Data 70k	Dict Machine	100M Brown	Combo
ADJ	JJ	670	.64	.72 (+.08)	.71 (+.07)	.71 (+.07)	.72 (+.08)	.77 (+.13)
	JJR	31	.42	.65 (+.23)	.68 (+.26)	.60 (+.19)	.58 (+.17)	.62 (+.20)
	JJS	26	.69	.88 (+.19)	.92 (+.23)	.79 (+.10)	.88 (+.19)	.84 (+.15)
ADV	RB	680	.80	.84 (+.04)	.83 (+.03)	.86 (+.05)	.85 (+.05)	.86 (+.06)
	RBR	20	.42	.36 (-.06)	.43 (+.01)	.55 (+.13)	.59 (+.17)	.52 (+.10)
	RBS	3	.00	.75 (+.75)	.86 (+.86)	.67 (+.67)	.40 (+.40)	.21 (+.21)
NOUN	NN	1,931	.75	.80 (+.05)	.80 (+.05)	.78 (+.03)	.80 (+.05)	.79 (+.04)
	NNP	1,159	.55	.60 (+.05)	.60 (+.05)	.56 (+.02)	.60 (+.05)	.62 (+.08)
	NNPS	8	.00	.09 (+.09)	.00 (+.00)	.00 (+.00)	.00 (+.00)	.00 (+.00)
	NNS	393	.70	.81 (+.10)	.81 (+.10)	.79 (+.08)	.79 (+.09)	.76 (+.05)
VERB	MD	181	.96	.96 (+.00)	.96 (+.00)	.96 (+.00)	.95 (-.01)	.97 (+.01)
	VB	660	.80	.80 (+.00)	.81 (+.01)	.80 (+.00)	.83 (+.03)	.83 (+.03)
	VBD	306	.80	.84 (+.05)	.83 (+.04)	.83 (+.03)	.81 (+.02)	.80 (+.00)
	VBG	303	.87	.88 (+.02)	.89 (+.02)	.89 (+.02)	.90 (+.04)	.90 (+.04)
	VBN	140	.57	.69 (+.12)	.69 (+.13)	.67 (+.10)	.71 (+.14)	.65 (+.08)
	VBP	527	.82	.82 (+.00)	.82 (+.00)	.82 (+.00)	.84 (+.01)	.84 (+.02)
	VBZ	342	.81	.88 (+.06)	.87 (+.05)	.87 (+.06)	.87 (+.06)	.87 (+.06)

TABLE 4.1: F-Score improvements for the tags in the four major word classes on the Twitter dataset. Shading highlights best improvements for a tag by an individual approach and bold face shows best overall result per tag

We also find that names and named entities improve considerably. When putting the frequency of the tags in perspective, it becomes clear why word clusters improve the tagging accuracy to such an extent, they do not just cover spelling variations of words but also a large number of (proper) nouns. As word clusters are inexpensive to create from plain text, they are by far the most promising approach to adapt a tagger to the social media domain.

4.4 Transferability to Other Languages

In the previous section, we investigated the effect of various domain adaptation approaches to increase model robustness on an English social media corpus. In the following sections, we apply these approaches to a German and Italian social media dataset to investigate the general applicability of these approaches to other languages.

4.4.1 Case Study: German Social Media and Web Text

This case study is centered around the dataset of the “Empirikom Shared Task” by Beßwenger et al. (2016). The main objective of this shared task lay on constructing a PoS tagger which is able to tag various kinds of social media text with high accuracy.

Dataset The shared task provided a corpus of 23k tokens of PoS annotated text, split up in an official train and test set. The text consists of computer mediated communication (i.e. WhatsApp chats, Twitter, blog comments, Wiki talk pages, social and professional chats) and web text about lifestyle or traveling. This dataset is larger than the English Twitter data that we have been using before but the official train-test data split cuts the dataset nearly in half. Furthermore, the dataset is annotated with

Extended STTS tags	Frequency	
	Train	Test
EMOASC	115	72
PTKMA	103	85
PTKIFG	99	133
AKW	49	60
HST	46	42
ADR	35	48
PTKMWL	28	24
EMOIMG	22	63
URL	18	21
VVPPER	7	6
VAPPER	4	4
DM	3	6
PPERPPER	1	1
ONO	1	2
KOUSPPER	1	2
VMPPER	1	0
ADVART	1	3
EML	0	1

TABLE 4.2: Frequency of the newly introduced tags in the German dataset

an extended version of the Stuttgart-Tübingen-Tagset (STTS) (Schiller et al., 1999), which adds 18 additional PoS tags to the 54 tags in the canonical STTS to account for social media phenomena. The additional tags assign user-mentions, hashtags and alike and own tag but also introduces new tags for word contractions that occur in informal but conceptually “oral” written discourse.

We analyzed the frequency of the newly added tags in the training and testing set in Table 4.2. We find that nine out of eighteen tags occur less than 10 times. Interestingly, we even have one tag in each set that occurs zero times showing that the dataset contains many rare phenomena. Thus, the German *Empirikom* dataset appears to be more challenging as it is a mixture of informal text domains with a highly fine-grained PoS tagset of which many tags occur only rarely.

Experimental Setup The shared task made a distinction in its test data between data from the CMC genre and web text but also reported overall results. To ensure comparability of our results to the shared task setup, we will adapt to this three-way evaluation and also report separate results on the CMC and web subset but also overall results over the entire test set.

We apply the domain adaption strategies as follows: To provide *more data*, we will apply mixed re-training by providing 100k tokens of annotated newswire text from the Tiger (Brants et al., 2004) corpus. The extended STTS is compatible to the canonical STTS used in the Tiger, which allows using data annotated with the canonical STTS. We provide *more knowledge* from Brown clusters and a PoS dictionary:

- *Brown clusters*: We create Brown clusters from 170 million tokens of German Twitter messages which we crawled between 2011 and 2017. All tokens have

been lowercased. Hashtags, user mentions and URLs are replaced by constant values in a pre-processing step using regular expressions.

- *PoS dictionary:* We create a PoS dictionary which stores the three most frequent PoS tags of a word. We build the dictionary using the Hamburg Dependency Treebank (Foth et al., 2014b), which contains STTS annotated text from a technical German newswire website. We choose this corpus for its large size of several million tokens and its technical nature, which seems to be more suited for the social media domain than a business newswire corpus.

We use the same feature set as for the English experiment. We will re-evaluate each approach individually and determine the combined results.

Results In Figure 4.11, we show the results for each approach on all and OOV tokens. We can confirm the findings we already made for the English dataset, the combined approach reaches the best result. Mixed RT, word clusters and the PoS dictionary reach highly comparable results, with the dictionary reaching higher improvements than the word clusters. We assume that the higher annotation quality and size of the corpus from which the dictionary was created is accountable for the better performance (unlike for the English dataset where the annotation was created by automatically tagging). Tagging OOV words on the fine-grained tagset shows that the combined approach does not reach the highest accuracy. We assume that the added newswire data account for the slight decrease of the combined approach by introducing an increased amount of intra-class errors (e.g. confusion of verb past tense with present tense). An assumption we find confirmed when looking on the OOV performance on the coarse-grained mapped tags where the combined approach reaches the best overall performance.

Improvements by Sub-Genre In Table 4.3, we show the results for the CMC and Web subset of the test data in contrast to the averaged result over both subsets. The number in brackets show the improvement relative to the Re-Train baseline that is trained only on the shared-task training data. Bold-faced accuracies show the best overall result per subset. Gray highlighting shows the best individual approach per subset. We see in general a similar picture as for the English dataset, with an exception for mixed re-training that improves performance the most. This clearly shows that the shared task dataset alone is too small to allow learning the rich German morphology. On *Web*, the PoS dictionary is considerably more effective than on the CMC subset. Since the dictionary was created from a large newswire (standard text) corpus, we see the main reasons in an improved named entity coverage. Furthermore, web text appears to be less challenging than CMC. The gap between both sub-genres is with over five percent points quite large. The coarse-grained mapping confirms that CMC with 90.7% is a lot more challenging than Web that reaches 95.6%. The combined tagger reaches on the fine-grained tagset on CMC, Web and All statistical

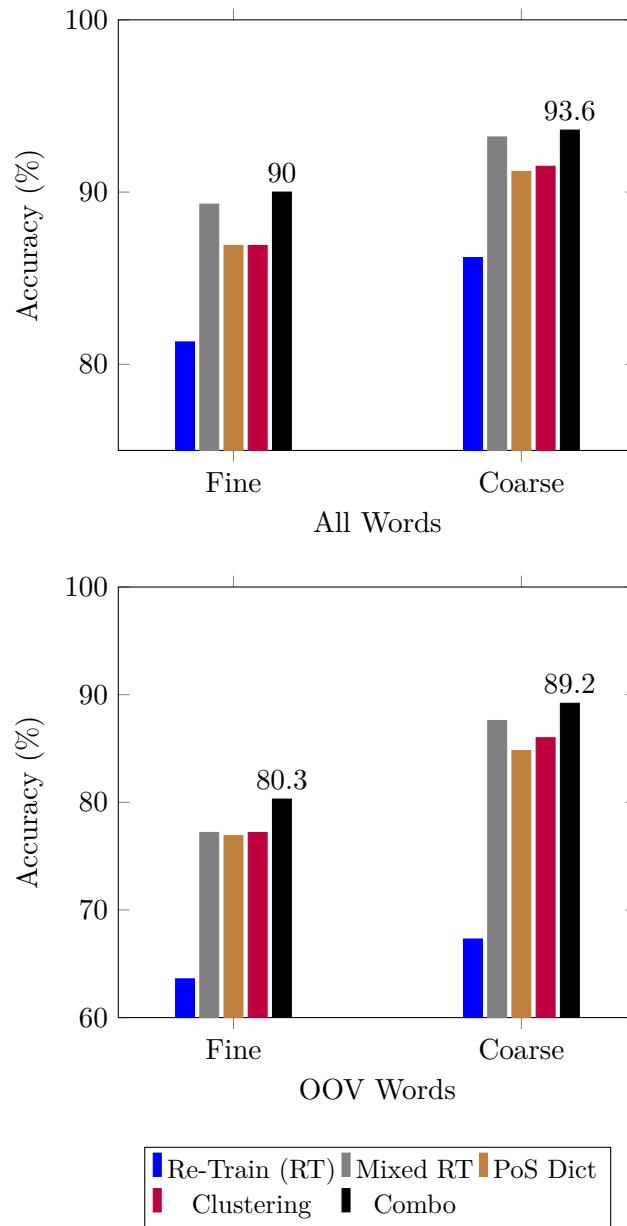


FIGURE 4.11: Results for each approach and in combination on the German test dataset for all and OOV words (CMC+Web)

	Accuracy (%)					
	Fine-grained Tagset			Coarse-grained Tagset		
	CMC	Web	Both \emptyset	CMC	Web	Both \emptyset
TreeTagger	74.1	91.7	84.4	79.3	93.2	87.5
Re-Train (RT) (<i>Baseline</i>)	80.4	81.9	81.3	85.7	86.5	86.2
+Mixed RT	86.2 (+5.8)	91.5 (+9.6)	89.3 (+8.0)	90.7 (+5.0)	95.0 (+8.5)	93.2 (+7.0)
+Clusters	85.5 (+5.1)	87.8 (+5.9)	86.9 (+5.6)	90.4 (+4.7)	92.2 (+5.7)	91.5 (+5.3)
+PoSDict	84.7 (+4.3)	88.4 (+6.5)	86.9 (+5.6)	89.8 (+4.1)	92.1 (+5.6)	91.2 (+5.0)
+All	86.8 (+6.4)	92.3† (+10.4)	90.0† (+8.7)	90.7 (+5.0)	95.6 (+9.1)	93.6 (+7.4)

TABLE 4.3: Results on German test data subsets per approach, for the fine-grained tagset we show in bold face the best result per subset and in gray the best individual strategy. The combination of all fine-grained approaches are statistical significant improvements compared to the RT baseline, results marked with † are significant compared to *Mixed RT*, i.e. just adding foreign domain data, (McNemar’s test, $p < 0.05$)

Rank	Configurations	Accuracy (%)		
		CMC	Web	All
1	UdS-distributional	87.3	93.6	90.4
2	UdS-retrain	86.4	92.8	89.6
3	UdS-surface	86.5	92.4	89.4
4	<i>Our tagger</i>	86.1	92.1	89.1

TABLE 4.4: Official results of the German Empirikom shared task of our tagger and the winner tagger(s) by Prange et al. (2016)

significant improvements in a McNemar’s test, $p < 0.05$, compared to the RT baseline. Results marked with a † are significant improvements to MixedRT, which is the simplest adaption approach of just adding foreign, i.e. newswire, domain data.

Improvements by Tag In Table 4.5, we show the F-Score for the fine-grained PoS tag belonging to the four major word classes. Reported results are obtained by testing on both, CMC and Web, datasets. We show the frequency of occurrence in the train and test dataset as reference for the relative importance of a tag in the corpus. We show the improvements per tag, per approach, and in combination. Highest improvements for each tag by a single approach are highlighted in gray, overall best result for a tag is shown in bold-face.

We find that the accuracy improvements from Table 4.3 originate in noun-related improvements. Nouns, NN and NE, are by far the most frequent tags and we see for all methods considerable improvements for these two tags. Adjective and adverb tags also improve considerably. On verbs, we also see improvements on reasonably frequent tags but especially the F-Score on the newly added tags remain poor. We even have a the zero-block for tags such as VMINF, VMPP or VMPPER. For two tags, VMPP and VMPPER, we even have no testing instances. Hence, we see that the size of the corpus is too small to cover all typical social media phenomena. We will come back in Chapter 7 to the problem of tagging under-represented phenomena, which occur too infrequently to be learned during model training.

PoS Tag	Frequency		Re-Train (RT)			Word Clusters	Combo
	Train	Test		Mixed RT	PoS Dict		
ADJ	ADJA	390	647	.76	.91 (+.16)	.88 (+.12)	.86 (+.11) .93 (+.18)
	ADJD	294	410	.59	.83 (+.24)	.80 (+.21)	.74 (+.16) .84 (+.25)
ADV	ADV	473	577	.69	.76 (+.08)	.78 (+.09)	.76 (+.08) .77 (+.09)
	PAV	53	78	.74	.92 (+.18)	.87 (+.13)	.85 (+.11) .91 (+.17)
VERB	PTKVZ	46	69	.37	.74 (+.37)	.53 (+.16)	.50 (+.12) .75 (+.38)
	VAFIN	299	416	.94	.97 (+.03)	.96 (+.02)	.96 (+.02) .97 (+.03)
	VAINF	26	52	.73	.80 (+.08)	.77 (+.04)	.77 (+.05) .81 (+.09)
	VAPP	3	1	.67	.50 (-.17)	.33 (-.33)	.33 (-.33) .50 (-.17)
	VAPPER	4	4	.00	.40 (+.40)	.40 (+.40)	.33 (+.33) .40 (+.40)
	VMFIN	113	160	.88	.98 (+.10)	.97 (+.08)	.96 (+.08) .98 (+.10)
	VMINF	3	4	.00	.00 (±.00)	.00 (±.00)	.00 (±.00) .00 (±.00)
	VMPP	1	0	.00	.00 (±.00)	.00 (±.00)	.00 (±.00) .00 (±.00)
	VMPPER	1	0	.00	.00 (±.00)	.00 (±.00)	.00 (±.00) .00 (±.00)
	VVFIN	376	433	.67	.85 (+.17)	.79 (+.12)	.80 (+.13) .86 (+.19)
	VVIMP	18	32	.10	.17 (+.07)	.05 (-.05)	.11 (+.01) .16 (+.06)
	VVINF	241	212	.63	.86 (+.22)	.78 (+.15)	.77 (+.13) .85 (+.22)
	VVIZU	7	15	.26	.90 (+.64)	.53 (+.27)	.48 (+.22) .93 (+.67)
	VVPP	141	182	.64	.86 (+.23)	.76 (+.12)	.77 (+.13) .87 (+.24)
	VVP PER	7	6	.20	.44 (+.24)	.29 (+.09)	.37 (+.17) .60 (+.40)
NOUN	NN	1,644	2,357	.80	.91 (+.11)	.88 (+.08)	.88 (+.09) .92 (+.12)
	NE	407	482	.51	.67 (+.16)	.64 (+.13)	.63 (+.12) .72 (+.21)

TABLE 4.5: F-Score of each tag of the major word classes per approach and in combination on the German test set (CMC and Web). Shading highlights best improvements by approach and bold face highlights best overall improvement

Shared Task Results We participated in this shared task (see Horsmann and Zesch (2016b) for the exact details of our submitted tagger) and are the second-best team on the PoS tagging task (Beißenwenger et al., 2016). Our submitted tagger used essentially the previously described domain adaptation approaches. The slight variation in our achieved results to the one in the shared task origin in a few subtle changes to the learning parameters during model training (after the gold annotated test set has been released). The official results of the shared task are shown in Table 4.4 (each team was allowed to submit several runs). The key difference of the winner’s taggers by Prange et al. (2016) are additional social media training data. They re-annotated a corpus with the extended shared task tagset and used this extra data for training. This corpus has not been available to other participants. Significance values have not been reported but we consider our tagger to still be competitive to the taggers by Prange et al. (2016). When reproducing our own shared task setup, we reach on *All* an accuracy of 90.0% (see Table 4.3) without the extra social media data used by the winning tagger.

4.4.2 Case Study: Italian Tweets

This case study is based on the dataset of the Italian shared task by Bosco et al. (2016). The main objective of this task lay on tagging Italian Twitter messages.

Dataset The organizers of this task provided a dataset of 6,700 Twitter messages with 120k tokens, which are split into an official train dataset (115k tokens) and test dataset (5k tokens). The dataset is annotated with a coarse-grained tagset which consists of 25 tags in total. The tagset is based on the Universal Dependency tagset which has 17 tags but was refined with additional tags to deal with Twitter phenomena such as user mention and alike, and to account for a customized tokenization that was applied in this shared task. This tagset is, thus, by far much more coarse-grained while the corpus size is magnitudes larger than in the datasets we used for English and German. Thus, we have to learn only few tag distinctions while having magnitudes more annotated social media data.

Experimental Setup We will use again training and evaluating on the official training dataset as reference point for investigating the effect of the following domain adaptation approaches:

- *Mixed Re-Train*: We use 100k tokens of formal text from the Italian corpus in the Universal Dependencies (UD) project (Nivre et al., 2016). The corpus is annotated with the standard UD tagset, this shared task uses a slightly modified version of this UD tagset.
- *Brown clusters*: We train Brown clusters on 565 million tokens of Italian Twitter messages. All tokens have been lowercased and hashtags, user mentions and URLs are replaced by constant values.
- *PoS dictionary*: We create a PoS dictionary which stores the three most frequent PoS tags of a word. We build the dictionary from a machine-tagged corpus of Italian Wikipedia entries for lack of availability of other resources⁸.

We use the same feature set as for the English experiment. To the best of our knowledge there is no existing tagger that uses the same tagset as in the shared-task. Hence, we will report baseline results by using the Italian model of the TreeTagger and map the predicted tags to coarse-grained tags. This should still serve as a fair approximation as both tagsets are highly similar.

Results In Figure 4.12, we show the results on all and OOV tokens for both tagset granularities. The difference between re-training and combining approaches is considerably smaller than the one we saw for English or German. Furthermore, mixed Re-training shows in this setting no improvements. In fact, the additional foreign domain data harm overall performance even a bit compared to just re-training on the provided training data. This is explainable by (i), the large size of the social media training corpus and (ii), the rather coarse-grained tagset, which has a few modifications over the tagset used in the foreign domain data. We assume that the cases in which the annotation between both corpora conflicts account for the overall

⁸<http://wacky.sslmit.unibo.it/doku.php?id=corpora>, last accessed 17 May 2017

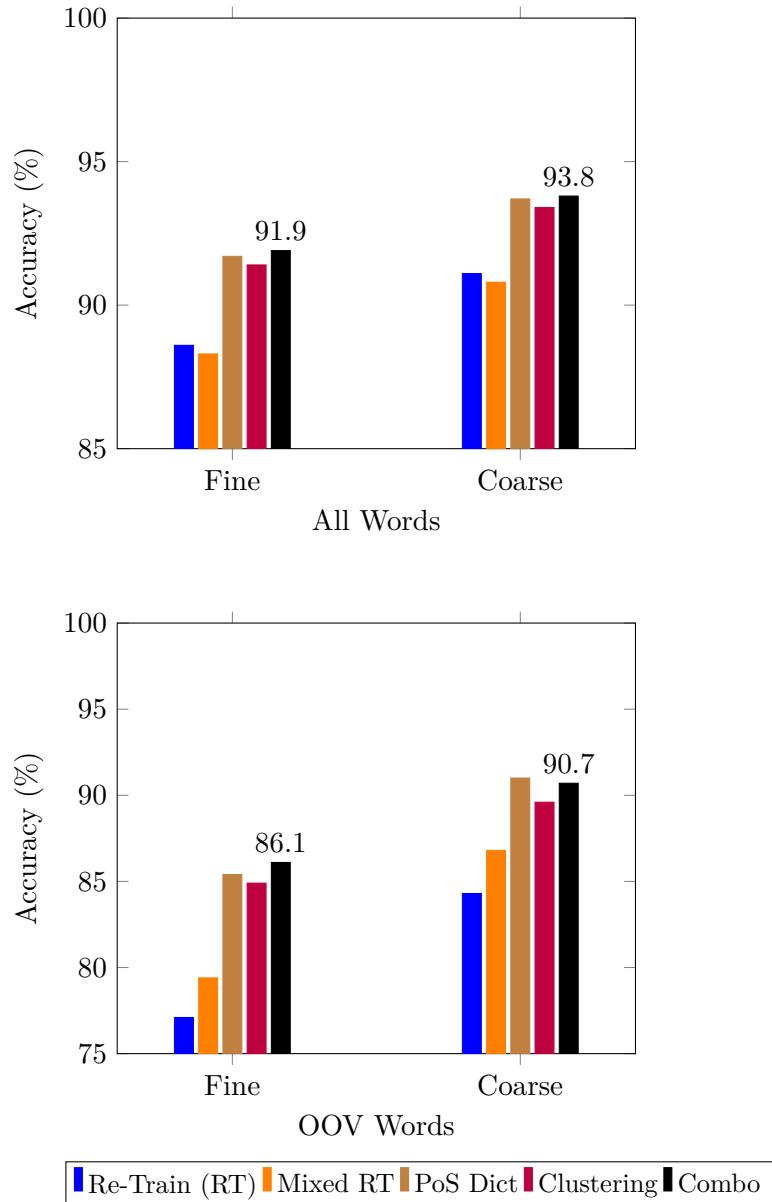


FIGURE 4.12: Results for each approach and in combination on the Italian test dataset for all and OOV words, Combo consists of RT with PoS Dict and Clustering excluding the Mixed RT approach as it does not offer improvements over simple Re-training

PoS Tag	Test Freq.	(RT) Re-Train	Mixed RT	PoS Dict	Word Clusters	Combo
ADJ	210	.75	.78 (+.03)	.84 (+.09)	.83 (.08)	.85 (+.10)
ADV	322	.90	.90 ($\pm .00$)	.91 (+.01)	.92 (+.02)	.92 (+.02)
VERB	595	.87	.87 ($\pm .00$)	.91 (.04)	.92 (+.05)	.93 (+.06)
VERB_CLIT	27	.73	.78 (+.05)	.86 (.13)	.93 (+.20)	.91 (+.18)
NOUN	607	.84	.85 ($\pm .00$)	.89 (+.05)	.88 (-.01)	.89 (+.05)

TABLE 4.6: F-Score of each tag of the major word classes per approach and in combination on the Italian test set. Shading highlights best improvements by approach and bold face highlights best overall improvements. Combo uses RT, PoS Dict and Word Clusters (Mixed RT is excluded)

reduced performance, although we see a slight improvement on OOV words when adding foreign domain data. The added external resources reach an improvement of a bit more than two percent points. They have their greatest effect on the OOV accuracy where the re-training approach performs only poorly but the resources account for large improvements. The combo approach combines the two resources but excludes the foreign domain data as they harmed overall performance. The improvements of combo over Re-Train on the fine-grained tagset are statistical significant in a McNemar’s test with $p < 0.05$.

Improvements by Tag Table 4.6 shows the F-Score of the four major word classes (that are represented by five tags in the shared-task tagset). We see the same improvements in F-Score that we have seen for the English and German dataset. A remarkable exception is the adverb tag that improves considerably after adding the word clusters. Also, the newly introduced tag for verb clitics shows remarkable improvements by using external resources.

Shared Task Results We participated in this shared task (see Horsmann and Zesch (2016a) for the exact details of our submitted tagger) and reached the second place (Bosco et al., 2016). The results are shown in Table 4.7, the best system by Cimino and Dell’Orletta (2016) uses a bidirectional Long-Short-Term-Memory (Hochreiter and Schmidhuber, 1997; Graves et al., 2005) neural network. They combine a common word-level LSTM and a bag-of-character LSTM. The bag-of-character LSTM uses hand-crafted information such as capitalization or lowercasing of letters in a word. Significance values have not been reported by the organizers but the six best systems have an absolute difference of less than one percent point between the winning tagger and the sixth placed tagger. This shows that all taggers perform highly similar on this dataset, one reason might be the large amount of available training data that lets technical differences between implementations become less important i.e. 115k tokens of annotated training data evaluated on 5k tokens in the test set.

Rank	Configurations	Acc (%)
1	(Cimino and Dell'Orletta, 2016)	93.2
2	<i>Our Tagger</i>	92.9
3	(Tamburini, 2016)	92.8
4	(Paci, 2016)	92.7
5	(Tamburini, 2016)	92.5
6	(Plank and Nissim, 2016)	92.3

TABLE 4.7: Official results of the Italian shared task showing the result of our tagger and the winning tagger (teams were allowed to submit two runs)

4.5 Chapter Conclusion

In this chapter, we investigated the impact of approaches proposed in the literature to improve tagging robustness for social media text. We evaluated the effectiveness of each approach in isolation and in combination when using a fine-grained and coarse-grained tagset. The domain adaptation strategies showed that in particular methods improving word form coverage account for substantial accuracy improvements. Utilizing word distributional knowledge obtained from word clusters and retrieving most common tags from PoS dictionaries showed to be very effective. Due to the fact that social media text is easily available, clusters are an inexpensive choice to reach substantial improvements. We found Brown clusters to be particularly effective as the additionally encoded (partial) similarity between created clusters turned out to be an extremely valuable information for training more robust models. Even mixing annotated foreign domain text with social media corpora showed rather high improvements. This effectiveness seems to originate in the low amount of lexical knowledge in the small-sized social media corpora.

We confirmed the general applicability of these approaches by applying them also to German and Italian social media text and found similar improvements as for English. Although the approaches are easy to combine, they only partially solve the robustness problem. A high tagging accuracy on informal text domains is still out of reach. In case of Italian, we had magnitudes of more social media training data available than for English or German. Despite of having that much more data, we still did not reach comparable results as when tagging formally written text. This stresses once more that annotation of more (social media) data is no satisfying solution, either, even if combined with domain adaption approaches. Hence, we will introduce an entirely different approach for robust tagging in the next chapter.

Chapter 5

Domain Robustness - Two-Step Tagging

In this chapter, we discuss a new Part-of-Speech (PoS) tagging approach to improve domain robustness when tagging across different text domains. Chapter 3 showed that the accuracy of tagger models drops considerably when applied to foreign text domains, e.g. social media. In Chapter 4, we reviewed the present state of research for improving domain robustness of tagger models. The discussed approaches account for substantial improvements but do not solve the general robustness problem. A rather fundamental issue when tagging across text domains lies in the high amount of uncertainty that the tagger model faces when dealing with unknown (foreign domain) phenomena. This is further complicated by choosing a tag from an often very fine-grained tagset. The approach we discuss in this chapter tackles the robustness challenge by breaking the tagging down into two steps. Each step has to choose from a small number of tags compared to directly making a poorly informed tagging decision by choosing from all tags in a tagset. We consider this approach as a step towards a single tagger that is able to cope with a variety of text domains.

Motivation of Two-Step Tagging Usually, when assigning a tag, the tagger determines a tag from a rather large pool of possible tags. The tagset size varies from a dozen tags up to several hundred depending on the language and domain. Models trained on a large amount of training data perform this task well when the plain text and the model training data are from the same domain. If little or no domain data is available, directly choosing the fine-grained tag might not be the best strategy.

Tagging in two steps is based on an intermediate step in which a coarse-grained tag assignment precedes the fine-grained tag assignment. This breaks the tagging task down into two sub-problems which are, so our assumption, easier to solve than directly making a potentially poorly informed fine-grained tagging decision. Figure 5.1 provides an example of this two-step tagging approach assuming that the Penn Treebank (Marcus et al., 1993) tagset is used as fine-grained tagset (45 tags). In the first step, we assign only a coarse-grained PoS tag on a granularity level of the Universal PoS tagset (Petrov et al., 2012) or the Universal Dependency tagset (Nivre et al., 2016). In a second step, only few fine-grained tags remain as possible tags,

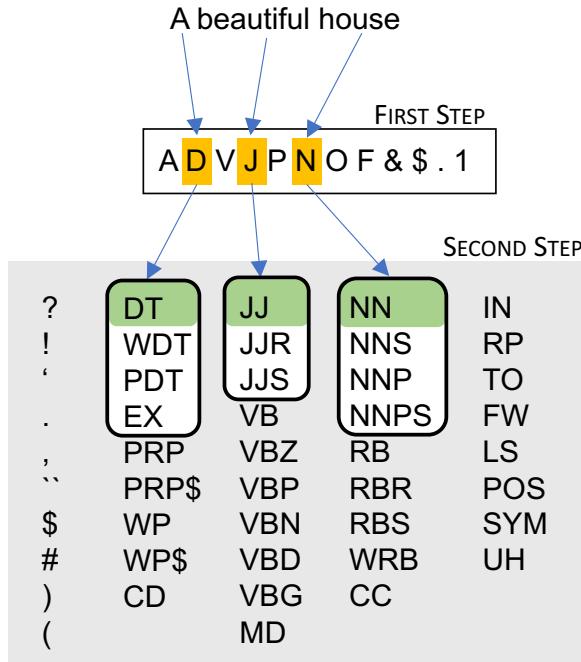


FIGURE 5.1: PoS tagging in two steps. The first step assigns a coarse-grained tag (yellow shading), the second step considers only fine-grained candidate tags belonging to this coarse-grained tag (round-boxed area) to determine the final fine-grained tag (green shading)

as the first tagging step already excluded a large portion of all possible fine-grained tags. Separating the tagging into two steps simplifies the tagging, as it solves two considerably smaller problems, for instance, the prediction of *determiner* (from 12 tags when one assumes Petrov et al. (2012)'s tagset) for the first word, excluded 41 candidate tags in the second step. The tagging in the second step then has to pick the final tag out of four remaining fine-grained determiner candidates.

The fundamental assumption of this approach is that a model trained on coarse-grained tags is more robust and avoids tagging errors a fine-grained model would make. A coarse-grained model will naturally achieve a higher accuracy simply because confusions between fine-grained tags are removed, e.g. confusions between verb tenses. The improvements we have in mind are avoiding confusions between word class categories, for instance, a model trained on a fine-grained tagset might confuse in a particular case adverbs with adjectives, while a model trained on coarse-grained tags would avoid making this error. Thus, we will first analyze how well coarse tagging can actually be done and then turn to tagging in two steps.

5.1 Potential of Coarse-grained Tagging

Two-step tagging requires a robust and highly accurate coarse-grained tagging. We will start with experiments to learn more about the possible improvements between fine-grained and coarse-grained tagging. Since we are interested in robustness, we

expect coarse-grained tagging to offer advantages for tagging foreign text domains, especially if training data is limited.

Corpora We conduct our comparison on four corpora of four text domains: *News*, *Web*, *Chat*, and *Twitter*. This covers a wide range of text domains and ensures that the results are not bound to a certain domain. Newswire text has a formal nature and contains few language errors, as it is usually carefully edited. Text from the web is (on average) less formal containing informal expressions and orthographic errors. Chat conversations are highly informal and often contain many non-standard abbreviations and orthographic errors. Twitter contains highly diverse text as the platform is used for all kinds of purposes ranging from chat-like discussions to formal announcements. As *News* corpus, we use 46k tokens of the Wall Street Journal (WSJ) (Marcus et al., 1993), for *Web* we use 44k tokens from the GUM corpus (Zeldes, 2016) containing semi-formal text from various Wiki-platforms, as *Chat* corpus we use the NPS (Forsyth and Martell, 2007) corpus with 32k tokens, and a *Twitter* (Ritter et al., 2011) corpus with 15k tokens.

5.1.1 Mapping Fine-Grained Tags as a Baseline

There are two ways to obtain coarse-grained tags: First, by mapping fine-grained tags after prediction to the corresponding coarse-grained tag and second, by directly training on coarse-grained tags where no fine-grained information enters the training or prediction step. The mapping will remove fine-grained intra-class confusions, e.g. confusions between verb inflections, and will serve as a strong baseline for the coarse-grained tagging. The expectation is that by having to learn a less complex tagset, the task is simplified to an extent that the model trained directly on coarse-grained tags learns to avoid some errors that the fine-grained model would make.

Experimental Setup We will train a model on each corpus and compare the coarse-grained tagging results to results that have been predicted by a model trained on fine-grained tags, which have been coarse-mapped afterwards. This allows a direct comparison between the tagging quality and judge the differences between both approaches. We train the models using Conditional Random Fields (CRF) (Lafferty et al., 2001) as implemented in FlexTag (Zesch and Horschmann, 2016). We use a tri-gram context window and provide the 750 most frequent character uni-grams, bi-grams, three-grams and four-grams as features. We map the tags of each corpus to a slightly modified version of the Universal tagset by Petrov et al. (2012), which additionally uses a tag for interjections. This tagset has 13 tags and is used in all subsequently discussed experiments in this chapter that work with (direct) coarse-grained tagging. Please note that we intentionally do not include any additional resources (at this point) when training these models as we want to study the impact of the tagset differences. We use 10fold cross-validation on each corpus and report averaged results over all runs.

Setup	Accuracy (%)			
	News	Web	Chat	Twitter
fine-mapped	94.5	93.1	92.0	89.0
coarse	94.3	92.8	92.3	89.7

TABLE 5.1: Comparison of fine-mapped and coarse-grained tagging (10fold CV)

Results Table 5.1 shows the results. We see only small differences between fine-mapped and directly working on coarse-grained tags. Direct coarse-grained tagging shows small advantages on informal corpora while fine-mapping works a bit better on formal text. This indicates a (small) advantage of directly using coarse-grained tagging on informal text domains, but we found no statistical significance when using a McNemar’s (McNemar, 1947) test with $p < 0.05$.

5.1.2 Amount of Necessary Training Data

When the tagger has to learn fewer tags, it should also require less training data to learn these few tags.

Experimental Setup To confirm this hypothesis, we run a learning curve experiment to see the effect of the amount of training data on fine-mapped and direct coarse-grained tagging. We adapted our 10-fold cross validation experiment to train on a decreasing number of chunks, i.e. instead of training on 9 folds and test on the remaining one, we train only on 8 and test on the 10th fold, then train on 7 and test on the 10th fold, and so on. We run this experiment ten times to ensure that every fold was once in the test set and report averaged results over all runs.

Results Figure 5.2 shows the results on *News* and *Twitter*. We see a small advantage of using direct coarse-grained tagging for smaller amounts of training data. Especially on the low-resourced social media corpus, we see the expected advantage of coarse-grained tagging. The difference between fine-mapped and direct coarse-grained tagging starts to vanish once a large part of the data is used for training. The advantage of coarse-grained tagging over fine-mapped tagging is on the Twitter corpus constantly visible but not significant in a McNemar’s test with $p < 0.05$.

5.2 Coarse-grained Cross-Domain Robustness

So far, we have compared mapped fine-grained and coarse-grained tagging by comparing results when training and testing on text of the same corpus. In this setup, we find only small differences between both methods, which are not enough to justify using coarse-grained tags instead of fine-grained ones. Furthermore, training and testing on data of the *same* corpus is an evaluation under ideal circumstances (Giesbrecht and Evert, 2009) because the text during training and the one for testing are most similar

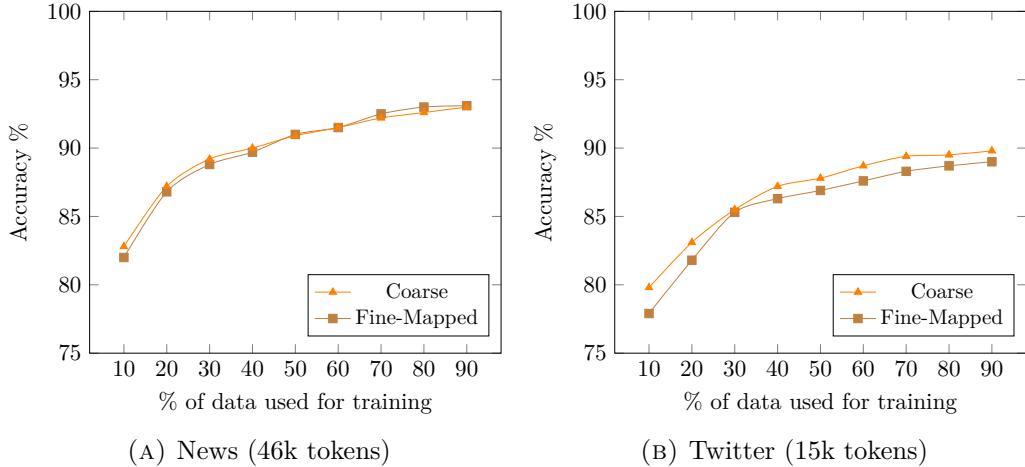


FIGURE 5.2: Learning curves for the *News* and the *Twitter* dataset comparing mapping fine-grained tags to coarse-grained tags to directly working on coarse-grained tags

to each other. The results allow no conclusion how coarse-grained tagging behaves when the dissimilarity between training and testing data increases.

The next reasonable step would be to continue the evaluation by using two corpora coming from the same domain, i.e. *in-domain* robustness. Unfortunately, there are not enough corpora available to conduct such an *in-domain* evaluation. As a consequence, we directly move on to evaluate *across domains*. Cross-domain evaluation requires only one corpus of each domain with a compatible tagset, which we have with the corpora we used already above.

5.2.1 Cross-Domain Learning Curve

We continue the learning curve experiment from the previous section but train this time on one text domain and evaluate on another one.

Experimental Setup We run each experiment ten times and increase in each iteration the number of training data while testing in each iteration against the full out-of-domain corpus. We use again *News* and *Twitter*. In the first experiment, we train on an increasing amount of *News* while testing on the full *Twitter* corpus, and in the second experiment, we train on an increasing amount of *Twitter* while testing on the full *News* corpus. We limit this experiment again to *News* and *Twitter* which are the cleanest and noisiest corpus in the evaluation setup.

Results We show the results in Figure 5.3. When tagging cross-domain, we find that the model learned on the smaller *Twitter* corpus predicts *News* considerably more accurately than the other way around. *Twitter* contains additionally to non-standard language also some portion of standard language, it thus is not surprising to see that the model learns useful information from this corpus to tag *News*. When testing on *Twitter*, the *News* training data provides no helpful information how to deal with the many non-standard phenomena. We find no statistical significance after adding

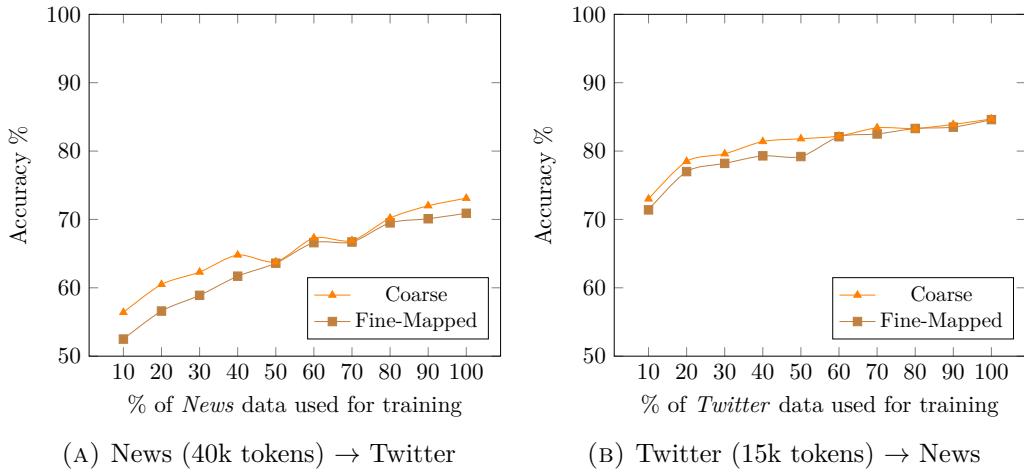


FIGURE 5.3: Cross-domain learning curves that compare mapping fine-grained tags to coarse-grained tags to directly working on coarse-grained tags when tagging foreign text domains

all training data but when working with only one data fold, direct coarse-grained tagging is significantly better in a McNemar's (McNemar, 1947) test with $p < 0.05$ than fine-mapping on *News*→*Twitter*. Thus, direct coarse-grained tagging performs better when training on small amounts of data and tagging across text domains. This confirms our assumption that coarse-grained tagging is of advantage, when working with extremely small amounts of training data.

5.2.2 Real-world Comparison

The comparisons we made until now used only the training data for predicting tags. This was necessary to compare fine-mapped tagging to direct coarse-grained tagging.

Experimental Setup We are now turning to more realistic setups where additional resources are provided for improving tagging performance to investigate the differences between fine-mapped and coarse-grained tagging under more realistic conditions. Furthermore, we now compare the performance of several taggers to also investigate if some taggers are better equipped to deal with domain transfers than others. We use the following tagger implementations: We use a CRF tagger using FlexTag (Zesch and Horschmann, 2016), the LSTM tagger by Plank et al. (2016) based on a bidirectional Long-Short-Term-Memory (LSTM) (Hochreiter and Schmidhuber, 1997; Graves et al., 2005) neural network and the Hidden-Markov-Model tagger HunPos (Halász et al., 2007). We use the same corpora for evaluation as in the above sections, except for *News*, which we redefine to be section 22-24 of the WSJ, which is the commonly used subset for tagger evaluations (Collins, 2002). The *Twitter* evaluation corpus contains Twitter-specific PoS tags for hashtags or user-mentions that do not exist in the training data, we set these tags in a post-processing step to their correct tag, which is easily possible using regular expressions (Ritter et al., 2011).

Coarse-grained Models We train the coarse-grained models on the slightly customized Universal tagset with one additional tag for interjections. A key advantage of directly training on coarse-grained tags is a considerably larger pool of training data. While several corpora may exist for a language they often use incompatible tagsets and cannot be directly combined. By mapping these corpora to a more coarse-grained representation, a combination of these corpora is possible. As training data, we use the WSJ section 0-18 and the Twitter datasets by Owoputi et al. (2013) and Jørgensen et al. (2016). By providing a mixture of formally written text and social media text, we aim at achieving an increased robustness of our coarse-grained model. Adding the Twitter datasets is possible for the coarse-grained model but not for the fine-grained model (discussed below).

We will train two models for the best working coarse-grained tagger (i), using only the WSJ data for a direct comparability to the fine-mapped model and (ii), using WSJ+Twitter to learn about the improvements of adding coarse-grained mapped training data. The CRF model adds as additional resources Brown (Brown et al., 1992) clusters trained on 100 million token of English tweets and a PoS dictionary created from the British National Corpus (Leech et al., 1994). The LSTM tagger is run with three different pre-trained word embeddings, two versions of the 100-dimensional GloVe (Pennington et al., 2014) word embeddings, one created from Wikipedia text and one from Twitter text, and the 64-dimensional Polyglot (Al-Rfou et al., 2013) word embeddings that have been created from Wikipedia. The HMM tagger uses no resources and is trained as-is on the training data using default parametrizations.

Fine-grained Baseline Models The fine-grained models are trained on WSJ section 0-18. The Twitter corpus we add for the coarse-grained model cannot be used here as the fine-grained tagset of this corpus is incompatible to the PTB tagset of the WSJ. The CRF model uses the same Brown clusters and PoS dictionary as the coarse-grained models. The HMM model is trained on the training data without any additional resources.

Results Table 5.2 shows the results of tagging the *Web*, *Chat*, and *Twitter* datasets. We also show the results on *News* to provide the usual reference value on WSJ section 22-24. The macro average result is computed over Web, Chat and Twitter as our focus lies on the cross-domain tagging performance.

The best working configuration is the CRF tagger using the WSJ and the Twitter corpus as training data. When compared to training only on the WSJ data, we see statistically significant improvements in a McNemar’s test with $p < 0.05$ on Chat and Twitter. Table 5.3 shows a comparison of the F-Score of the major word classes and interjections for both coarse-grained CRF setups (with and without extra Twitter data) for the *Chat* corpus, which showed the largest improvements by the additional Twitter training data. While we see a general improvement on the major word classes, the improvements on interjections are the highest. In a qualitative analysis, we find

	Setup	Accuracy (%)				Macro \emptyset
		News	Web	Chat	Twitter	
Fine-Mapped	HMM	97.6	95.8	79.3	78.9	84.7
	CRF	97.4	96.0	87.4	89.6	91.0
	LSTM-PolyEmb	97.9	96.1	81.3	85.7	87.7
	LSTM-WikiEmb	97.9	96.0	79.9	83.3	86.3
	LSTM-TwitEmb	97.9	96.2	81.1	85.9	87.7
Coarse	HMM	97.0	95.4	88.2	86.1	89.9
	CRF (only WSJ)	96.9	95.4	85.9	88.7	90.0
	CRF (WSJ+Twitter)	96.9	95.6	90.6*	91.6*	92.6
	LSTM-PolyEmb (WSJ+Twitter)	97.9	96.4	87.0	88.4	90.6
	LSTM-WikiEmb (WSJ+Twitter)	97.9	96.3	88.5	89.0	91.3
	LSTM-TwitEmb (only WSJ)	98.0	96.2	80.8	86.0	87.7
	LSTM-TwitEmb (WSJ+Twitter)	97.9	96.4	89.4†	89.2	91.7

TABLE 5.2: Accuracy of fine- and coarse-grained models for tagging across domains. Macro average computed over Web, Chat and Twitter. Fine-mapped taggers are only trained on the WSJ corpus because the fine-grained tagset of the Twitter corpus is not compatible to the WSJ. Improvements of the coarse-grained CRF and LSTM tagger on *Chat* and *Twitter* are significant (marked with *) in a McNemar’s test $p < 0.05$ when adding additionally Twitter training data. Using GloVe embeddings instead of Polyglot reaches also significant improvements (marked with †) for the LSTM tagger on the *Chat* corpus. Best results are bold-faced

that smiles and frequently occurring interjections such as *omg* or *lol* are tagged wrongly by the CRF tagger when trained only on the WSJ data. These typical chat phenomena do not occur in the WSJ data but do occur in the additional Twitter data. Thus, by adding only a small amount of social media training data, we reached significant improvements by covering highly frequent linguistic phenomena of the social media domain. We find similar improvements for the best coarse-grained LSTM tagger (LSTM-TwitEmb). Unsurprisingly, the LSTM that uses the word embeddings created from Twitter reach the best performance on the informal corpora and even a significant result on Chat compared to using the Polyglot embeddings.

We also analyzed the difference between the best CRF and the best LSTM-TwitEmb tagger. We find that the CRF tagger deals better with peculiarities of the training data that we use. In the WSJ, the word “my” is used as pronoun, while in the extra Twitter data, it is almost exclusively annotated as determiner. Therefore, the LSTM tags the word “my” in the *Chat* and *Twitter* corpus frequently as pronoun although it is a determiner. Furthermore, we find in *Chat* frequent one-letter words that are annotated as nouns, for instance the letter “m” is used as abbreviation for “male” that is found in sentences such as *any girls wanna chat with 24 / m ?*. The LSTM tags these cases as “other” while the CRF manages to assign the correct noun tag in these cases.

Tag	Freq.	F-Score	
		w/o Twitter	with Twitter
ADJ	1.514	.784	.812
ADV	1.920	.836	.870
VERB	6.038	.885	.925
NOUN	6.633	.817	.890
INTJ	1.556	.297	.799

TABLE 5.3: F-Score for the coarse-grained CRF taggers on selected tags with and without training additionally on Twitter training data evaluated on the *Chat* corpus

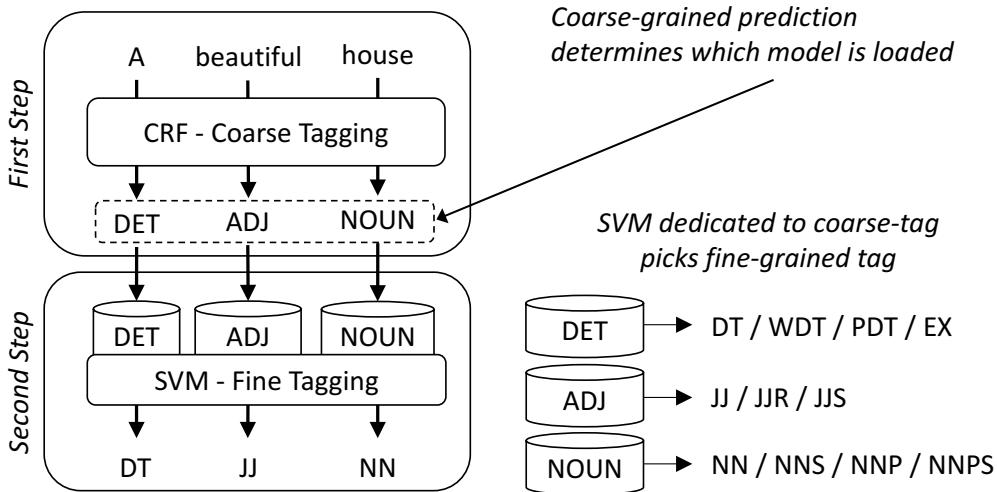


FIGURE 5.4: Two-step tagging using a CRF coarse-grained tagger in the first step, depending on the predicted coarse-grained tag a dedicated SVM model is used to assign the fine-grained PTB tag belonging to the predicted coarse-grained word class

5.3 Tagging in Two-steps

We will now turn to the actual two-step tagging approach to learn how well two-step tagging works in practice.

Implementation of Two-step Tagging We show the implementation of this approach in Figure 5.4. The coarse-grained tagging decision of the CRF in the first step is used to load a dedicated SVM tagger in the second step that assigns (one of) the fine-grained PTB tags, which belong to the coarse-grained word class which was predicted. The obvious drawback is the error propagation from the first to the second step. A wrongly predicted coarse-grained tag cannot be corrected during the second step. However, the advantage is that the models in the second step can focus on a much smaller sub-problem, which simplifies the tag prediction by having to choose from only a few tags.

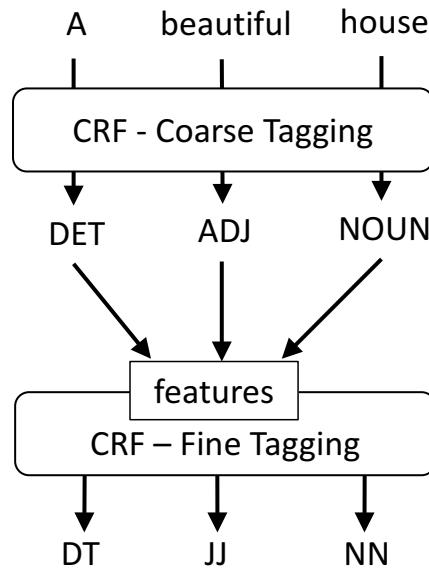


FIGURE 5.5: Stacking two CRF taggers and providing the predicted coarse-grained tags as features into a second CRF to predict fine-grained tags

Baseline: Stacked CRFs A further possibility to utilize the coarse-grained information is to stack two sequence classifiers e.g. CRFs. The first CRF would predict coarse-grained tags that are provided to the second CRF as shown in Figure 5.5. The coarse-grained information is utilized by using additional features in the second CRF, which provide for the word in focus, in a ± 2 context window¹, the predicted coarse-grained PoS tags of the first step. The fundamental difference to two-step tagging is that the coarse-grained prediction for a word is not binding i.e. the second CRF might predict a fine-grained tag that belongs to another coarse-grained word class than the one predicted in the first step. This allows recovering from a tagging error in the first step, which is not possible for two-step tagging.

Experimental Setup For coarse-grained prediction, we use the CRF model discussed in Section 5.2. The coarse-grained stacked CRFs baseline uses the same features as the coarse-grained model with additional features to provide the coarse-grained information.

The SVMs in the second step of two-step tagging are implemented with LibLinear (Fan et al., 2008). We use a context window of ± 2 words around the word in focus as features. We additionally include the 1000 most frequent character uni-grams, bi-grams and tri-grams, use Brown clusters created over 100 million token of Twitter messages and a PoS dictionary created from the British National Corpus as resources. The SVMs are trained on the WSJ section 0-18, i.e. we have no social media corpus available with the PTB tagset that we would need for a mixed training of the SVMs.

¹e.g. pos2Before=PUNCT, pos1Before=DET and so on

Setup		Accuracy (%)				Macro \emptyset
		News	Web	Chat	Twitter	
CRF fine baseline		95.7	92.2	74.6	83.4	86.5
Stacked CRFs		96.0	93.4*	78.3 *	85.0	88.2
TS	Plain	95.4	92.6	81.3 *†	85.1	88.6
	Coarse-context	95.6	92.8	81.4*†	85.4	88.8
	Stacked CRFs	99.0	97.5	85.8 *	91.7	93.5
	Plain	98.4 *	96.6 *	88.6 *	92.2 *	94.0
Oracle	Coarse-context	98.6 *	97.0 *	89.1 *†	92.3 *	93.7

TABLE 5.4: Accuracy of tagging in TS=two steps compared to coarse-induced tagging.

Plain uses no coarse-grained information in second step while *Context* does. We additionally show results assuming an oracle condition in the first step – results marked with an * are statistically significant improvements against the baseline, † (no-oracle) and ‡ (oracle) are significant against *Stacked CRF* (McNemar’s test, $p < 0.05$). The coarse-grained accuracy of the first step in two-step tagging (no-oracle) are on *News*: 96.9%, *Web*: 95.6%, *Chat*: 90.6% and on *Twitter*: 91.6% which set the upper bound for the fine-grained tagging in the second step (see Table 5.2)

The two candidate corpora that principally are available are in our evaluation dataset, namely *Chat* and *Twitter*.

We conduct an additional experiment to investigate whether two-step tagging improves if the SVMs are provided with coarse-grained PoS context from the first step. We compare two setups: *Coarse-context*, which uses the coarse-grained tags predicted in the first step as additional PoS context features and *Plain*, without coarse-grained features. Furthermore, as one can expect errors propagating from the first step into the second step, we run both experiments again, this time assuming an oracle condition in the first step, i.e. we assume a perfect coarse-grained tagging.

Results We show the results in Table 5.4 and additionally show a macro-averaged value over all four evaluation corpora to better compare the configurations. All configurations exceed the performance of the baseline system showing an improved average performance across the four domains.

Stacked CRF and *two-step tagging* are competitive. The stacked CRFs approach works better on the more formal corpora, which is reasonable to expect, as the training data are still mostly newswire data. Two-step tagging shows advantages on the less formal corpora. On the *Chat* corpus, two-step tagging is statistically significantly better than the *Stacked CRF* approach (McNemar’s test, $p < 0.05$). Providing coarse-grained PoS context (*coarse-context*) from the first step as additional features in the second step leads to insignificant improvements on all corpora.

When assuming an oracle condition for the prediction in the first step, we see the huge unused potential of this approach. Stacked CRF improves in a comparable scale to two-step tagging and both approaches stay competitive. We see again insignificant

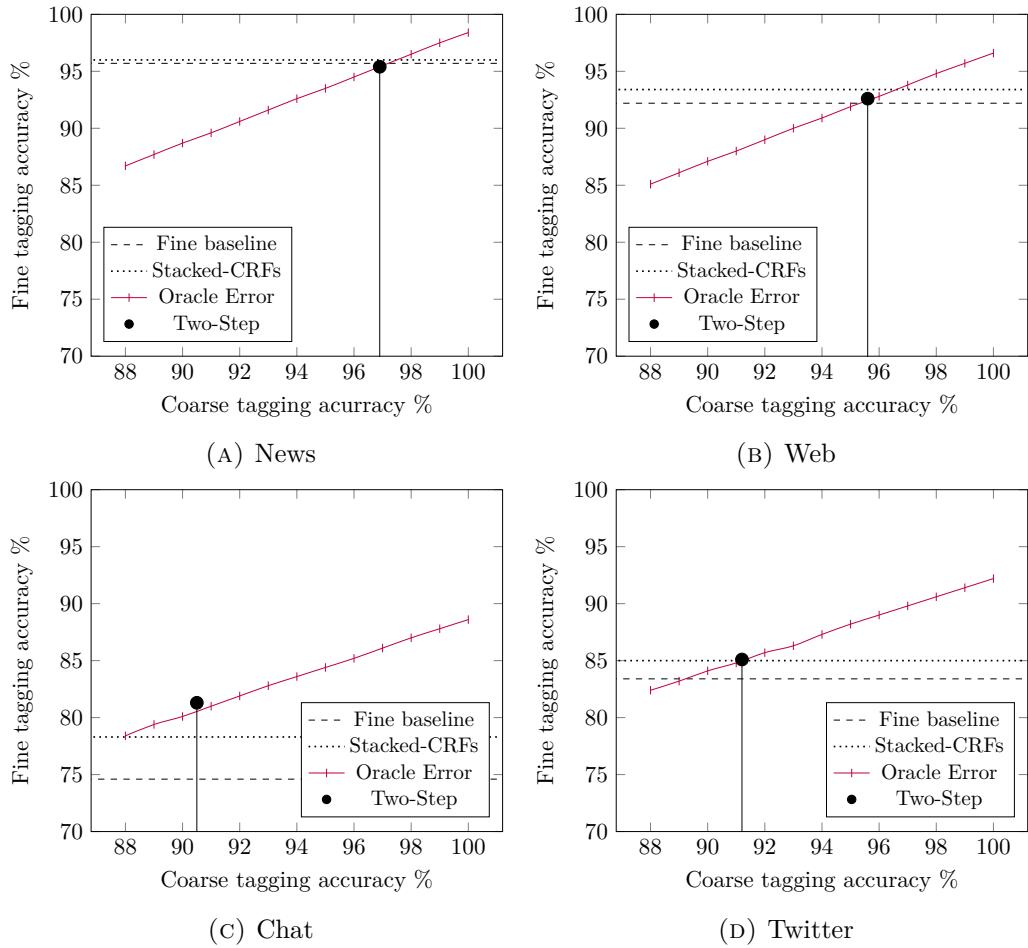


FIGURE 5.6: Relationship between coarse-grained accuracy and fine-grained accuracy of two-step tagging. *Oracle error* shows an oracle condition for the first step with an artificially added error to degenerate the oracle's performance

improvements for two-step tagging when using additionally coarse-grained information in the SVMs (coarse-context) of the second step against using no coarse-grained information in the SVMs (plain).

Two-step Tagging Error Propagation Analysis The results that assume an oracle condition for the first step reach an outstanding accuracy. The difference to the results without oracle condition show the impact of the error propagation between the two steps for the fine-grained tagging. In order to better understand the influence of the coarse-grained tagging error in the first step, we simulate a certain level of error propagation by using an oracle that only returns a wrong tag with a certain probability (uniformly sampled over all tags). This allows a gradual degeneration of the oracle condition and shows the influence of the coarse-grained tagging error. Figure 5.6 shows the resulting relationship between a certain level of coarse-grained accuracy and the resulting fine-grained accuracy. We see that both are in a similar linear relationship for all datasets. Measuring the slope, we can approximate that a one percent point improvement in coarse accuracy translates into an improvement

	Tag	News	Web	Chat	Twitter
ADJ	JJ	.999	.998	.999	.996
	JJR	.991	.970	.976	.928
	JJS	.995	.997	.970	.980
ADV	RB	.995	.997	.998	.992
	RBR	.938	.943	.921	.870
	RBS	.961	1.00	1.00	1.00
NOUN	NN	.988	.962	.875	.885
	NNP	.969	.925	.766	.785
	NNPS	.348	.486	.005	.064
	NNS	.979	.955	.848	.840
VERB	MD	.999	.996	.776	.923
	VB	.975	.935	.798	.859
	VBD	.954	.911	.857	.878
	VBG	.998	.999	.907	.942
	VBN	.928	.915	.658	.732
	VBP	.949	.887	.733	.805
	VBZ	.998	.997	.887	.965

TABLE 5.5: F-Score of fine-grained tags for two-step tagging under oracle condition for the best two-step tagging setup with coarse-context (see Table 5.4)

between .8 and .9 percent points in fine-grained accuracy. The accuracy of the best two-step tagging is marked by a black dot. Its location is always on or very close to the hypothetical performance of the oracle. We can thus conclude that our predictions are quite accurate and a better coarse performance will really lead to better fine-grained tagging performance.

Tagging Errors of Oracle Two-step Tagging Even if we assume a flawless coarse-grained tagging in the first step, two-step tagging still makes a substantial amount of tagging errors on the fine-grained level. We show in Table 5.5 the F-Scores for the best working two-step tagger (coarse-context) on the fine-grained tags under oracle condition. We focus on fine-grained tags belonging to the major word classes adjectives, adverbs, nouns and verbs to gain insights on the tagging errors that remains under oracle condition.

We focus our discussion on the noun and verb related tags as the F-Score differences on these tags is more severe than on the adjective and adverb related tags. The more informal the text domain is, the more the F-Scores on *NN* (common noun, singular) and *NNP* (proper noun, singular) decrease i.e. the tagging error on these tags increase. Proper nouns are a kind of noun and the distinction between common and proper noun is not a strictly syntactic distinction anymore. As these two tags occur in the fine-grained PTB tagset we use here, it is not surprising to find an increased error rate on these tags.

For verb-related tags, tagging *VBN* (past participle verb) drops considerably on *Chat* and *Twitter*, where it is often confused with *VBD* (past tense verb). We analyzed erroneous cases and find that phrases that are typical for spoken discourse are also these in which *VBN* is not tagged correctly, for instance in *i've got/VBN a brother in Naples*, *VBN* is confused as *VBD*. A reason for the increased number of *VBN* confusions can also be found in the training data of the SVMs that we use in the second step. We train the SVMs on WSJ data as the social media corpora with a compatible tagset are in our evaluation dataset and, thus, cannot be used for training the SVMs. The words that are annotated in the *Chat* and *Twitter* corpus as *VBN* are almost in all instances annotated as *VBD* in the WSJ. This is, thus, a bias problem learned from the WSJ training data. The *Chat* corpus contains many modal verbs, tag *MD*, which are not tagged correctly. An examination of these cases showed that many non-standard spelling variation of frequent words have been erroneously tagged as modal verbs, for instance *wanna*, or *dont* and *wont* (missing apostrophe). Thus, few systematic errors account for a substantial number of tagging errors on the informal corpora.

We saw in Table 5.2 the huge impact on the coarse-grained tagging accuracy by already a small amount of social media training data. One can assume to reach similar improvements on the fine-grained tagset, too, if the SVMs are trained on at least a few training samples from the social media domain.

5.4 Chapter Conclusion

In this chapter, we investigated how to improve cross-domain PoS tagging robustness. We suggested to split the tagging task into two steps in which a first tagging step assigns only a coarse-grained tag. This coarse-grained tag is then refined in a second step to the fine-grained PoS tag. The underlying assumption of this approach is that when tagging out-of-domain text, ones solves an easier problem first, i.e. coarse-grained tagging, and then refines this tag to the final fine-grained tag.

At first, we investigated coarse-grained tagging in detail as key pre-requisite for tagging in two steps. We compared the differences between training models on fine-grained tags and mapping them afterwards to coarse-grained tags to directly training and evaluating coarse-grained. We find that the differences are only small. A big advantage of coarse-grained tagging lies in the opportunity to ease the access to additional training corpora. The fine-grained tagsets of many corpora are not compatible and cannot be combined for training models. Mapping these fine-grained tags to a more coarse-grained tagset allows a harmonization of tagsets and enables training models on an enlarged corpora pool. This allows informing the coarse-grained model about data of several text domains and increase its robustness, which is an important pre-requisite of the two-step tagging approach. We find that adding already a small Twitter corpus to the coarse-grained model reached substantial improvements on informal text.

We implemented two-step tagging as a combination of a coarse-grained CRF tagger in the first step and dedicated SVMs in the second step to assign the final fine-grained tag. We compared two-step tagging to stacking two CRF taggers. The first CRF predicts coarse-grained tags that are then injected as features values into the second CRF to make the final fine-grained prediction. On formal text, we find that the stacked CRF tagger reaches the best results while two-step tagging reaches on informal text the best results. This finding is not surprising, as models trained on large newswire corpora do not face many out-of-vocabulary (OOV) words when being applied to newswire-like domains. Two-step tagging aims at lowering the severity of (potentially wrong) decisions that are made under poorly informed situations when facing many OOV words, for instance in social media. The biggest remaining challenge for two-step tagging lies in dealing with the error propagation occurring in the first step, which lead to classification errors in the second step. A simulation assuming a perfect coarse-grained tagging in the first step showed a huge untapped potential when the tagging error can be further reduced. The overall result showed that two-step tagging is the most suited multi-domain tagger of all taggers that we compared in our setup.

Chapter 6

Language Robustness

In the previous chapters, we focused on the robustness of Part-of-Speech (PoS) taggers when tagging different domains of the same language. In this chapter, we take a step back and look at a more general kind of robustness, namely language robustness. Many tagger implementations are available today, we have evaluated some of the available English and German taggers in Chapter 3. One finds many reports in the literature that a particular implementation works well (or better than another) for a certain language (Singh et al., 2006; Spouštová et al., 2007), while other taggers can cope with more than just a single language (Halász et al., 2007). We will investigate in this chapter the need for tailoring PoS taggers to a particular language. From a technical point-of-view, PoS taggers follow usually a common schema of a machine learning process that uses contextual and morphological features to learn a model. Hence, one can assume that a tagger is generally able to deal with more than just a single language, as shown in Figure 6.1. Thus, we investigate language robustness by using several tagger implementations and train models of a variety of languages to find out which taggers work well on more than just one language. Furthermore, we will compare these language-independent taggers to implementations that have been tailored to a particular language.

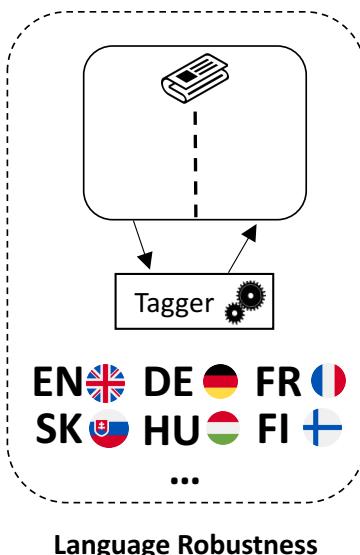


FIGURE 6.1: Evaluation for language robustness

Fallback Tagger The choice for a tagger usually depends on the expectation how accurately the tagger is able to learn a model of a certain language. Ideally, one would choose an off-the-shelf tagger tailored to a certain language. This would ensure that the tagger can deal with the individual properties of a language to achieve good results. Such tagger implementations are often available for well-resourced languages. If no such reference implementation exists, two options remain: i) implementation of an own tagger, or ii) using the next best off-the-shelf tagger that hopefully works reasonably accurately to learn models for other languages. Both options set a high threshold, i.e. i) requires experience and expertise to even be able to attempt tailoring an implementation to a language and ii) requires a high familiarity with all available tagger implementations to narrow down candidate taggers that are worth a try. Thus, both options are not easily implemented. Having a well-evaluated language robust tagger would certainly ease this problem. This puts the question in focus how to achieve language robustness, i.e. which technical properties are necessary to construct a language robust tagger.

Previous Work on Language Robustness The work by Plank et al. (2016) investigates a similar question regarding multilingual tagger robustness by comparing a tagger based on Long Short Term Memory (LSTM) (Hochreiter and Schmidhuber, 1997; Graves et al., 2005) neural networks to a Conditional Random Field (CRF) (Lafferty et al., 2001) and Hidden Markov Model (HMM) tagger. They train and compare model accuracy on the languages contained in the Universal Dependencies (UD) (Nivre et al., 2016) project. They find that the LSTM tagger reaches the best results i.e. is highly language robust. However, these corpora are all annotated with the language-independent UD tagset that distinguishes 17 tags. Learning a coarse-grained tagset is an easier task than using a language-specific fine-grained tagsets that easily reach hundreds or even more PoS tags. A language robust tagger must be able to also cope with fine-grained tagsets granularities. Thus, we conduct a replication of Plank et al. (2016)'s setup and investigate language robustness when working with

	Plank et al. (2016)	Our Replication
Tagger	Bidirectional LSTM	<i>same</i>
Baseline	CRF <i>Unpublished</i>	self-implemented (see Section 6.2.3)
	HMM TNT (Brants, 2000)	HunPoS (Halász et al., 2007)
Data	22 corpora 22 languages	27 corpora 21 languages
Tagset	coarse-grained (17 tags)	varying granularity (12 up to 1,500 tags)

TABLE 6.1: Comparison of the replication setup to the setup by Plank et al. (2016)

	Corpus Id	Source	Tokens (10 ³)	Tagset Size	Annotation	Reference
Germanic	Danish	Copenhagen DTB	255	36	manual	(Buch-Kromann and Korzen, 2010)
	Dutch	Alpino	200	20	manual	(Bouma et al., 2001)
	English	Brown	1,100	180	manual	(Nelson Francis and Kučera, 1964)
	German-1	Hamburg DTB	4,800	54	manual	(Foth et al., 2014b)
	German-2	Tiger	880	54	manual	(Brants et al., 2004)
	German-3	Tüba-D/Z	1,500	54	manual	(Telljohann et al., 2004)
	Icelandic	Mim	1,000	703	auto	(Helgadóttir et al., 2012)
	Norwegian	Norwegian DTB	1,300	19	manual	(Solberg et al., 2014)
	Swedish-1	Talbanken	96	25	manual	(Einarsson, 1976)
	Swedish-2	Stockholm-Umea	1,100	153	manual	(Ejerhed and Källgren, 1997)
Language Groups	Romanic	Braz.Portug.	MAC-Morpho	1,000	82	manual
		French-1	Multitag	370	992	manual
		French-2	Sequoia	200	29	manual
		Italian	Turin Parallel	80	15	auto
		Spanish	IULA DTB	550	241	manual
Slavic	Croatian-1	Croatian DTB	200	692	manual	(Željko Agić and Ljubešić, 2014)
	Croatian-2	Hr500k	500	769	manual	(Ljubešić et al., 2016)
	Czech	Prague DTB	2,000	1,574	manual	(Bejček et al., 2013)
	Polish	National Corpus	1,000	27	manual	(Przepiórkowski et al., 2008)
	Russian	Open Corpus	1,700	22	manual	(Bocharov et al., 2013)
	Slovak	MULTEXT-East	84	956	manual	(Erjavec, 2010)
	Slovene-1	IJS-ELAN	540	1,181	auto	(Erjavec, 2002)
Others	Slovene-2	SSJ	590	1,304	manual	(Krek et al., 2013)
	Afrikaans	AfriBooms	50	12	manual	(Augustinus et al., 2016)
	Finnish	FinnTreebank	170	1573	manual	(Voutilainen, 2011)
	Hebrew	HaAretz Corpus	11,000	22	auto	(Itai and Wintner, 2008)
	Hungarian	Szeged Treebank	1,200	1,085	manual	(Cséndes et al., 2005)

TABLE 6.2: Multi-lingual corpora collection for comparing PoS taggers robustness for their suitability to train models on a variety of languages

fine-grained tagsets. In Table 6.1, we show a direct comparison of the key elements in our replication to the work by Plank et al. (2016).

In the remainder of this chapter, we present a new set of multilingual evaluation corpora annotated with the typical fine-grained tagsets for the respective language. Based on this new dataset, we replicate the experiments by Plank et al. (2016) to investigate the impact of tagset granularity on language robustness. Furthermore, we analyze the technical properties that make tagger language robustness.

6.1 Corpora

A key requirement for our experiment is a large set of evaluation corpora that are annotated with the respective tagset that is commonly used for a language. In Table 6.2, we show the in total 27 corpora from 21 languages and 4 language groups that we collected. We collected these corpora by screening through the literature for corpus resources that are freely available for research purposes, which is a key prerequisite for reproducible research. Our dataset provides more than just one corpus for each language if several ones are available.

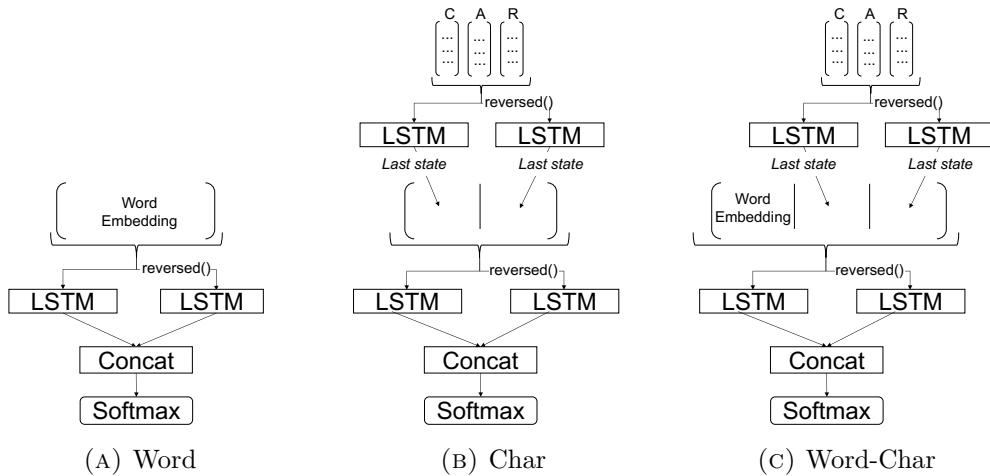


FIGURE 6.2: LSTM architectures in our replication setup

6.2 Tagger Implementations

In this section, we discuss and analyze the tagger implementations that we use in the replication experiments.

6.2.1 Hidden Markov Model (HMM) Tagger

We use HunPos (Halácsy et al., 2007) as HMM tagger, which is a freely available implementation of the TNT tagger (Brants, 2000). HunPos has been used before for training models of various languages and tagsets (Seraji, 2011; Attardi et al., 2010; Hládek et al., 2012), which makes this tagger a suited black-box baseline. We will use HunPoS using the default parameters.

6.2.2 Long Short Term Memory (LSTM) Neural Network Tagger

The replication is based on the LSTM tagger by Plank et al. (2016) of which we use the provided¹ reference implementation. The tagger by Plank et al. (2016) uses a stacked LSTM architecture which combines results of a word- and character level LSTM. In order to better understand the reported results, we will analyze the tagger architecture in detail by re-implementing the tagger. We compare the results of our own implementation to results obtained with the reference implementation by Plank et al. (2016). Word embeddings are a crucial variable when conducting experiments using neural networks. To account for this importance, we also ensure comparability of the pre-trained word embeddings. We retrieve $15 \cdot 10^6$ tokens plain text for each language in our setup from the Leipzig Corpus Collection (Quasthoff et al., 2006) and use *fasttext* (Bojanowski et al., 2017) for training the embeddings. We show the LSTM architectures that we re-implement in Figure 6.2.

¹<https://github.com/bplank/bilstm-aux>, last accessed 18 January 2018

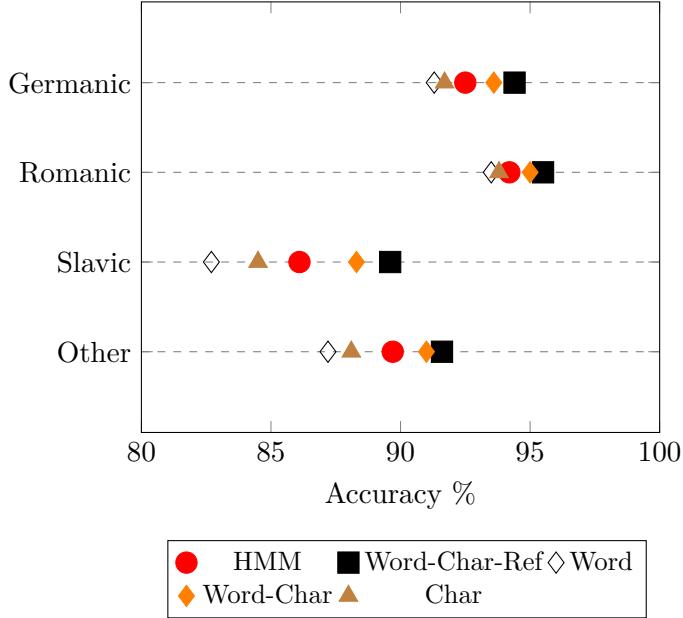


FIGURE 6.3: Results of the LSTM architectures per language group (10fold CV)

Word In this setup, we use word-level information for training the LSTM tagger. This setup will serve as baseline and is shown in Figure 6.2a.

Char The character embeddings of a word are provided to a bidirectional LSTM. The last state of the forward and the backward character LSTM are combined (Ling et al., 2015) and provided to another bidirectional LSTM layer. The character embeddings are trained on the fly during model training. This setup is shown in 6.2b.

Word-Char This architecture is a combination of the previous two architectures. The last state of the character LSTMs is added to the word embedding information before it is provided to the next LSTM layer. This setup is shown in Figure 6.2c.

Word-Char-Ref The reference implementation by Plank et al. (2016). This tagger reported state-of-the-art results on the coarse-grained tagset of the UD corpora.

Experimental Setup We implement the architectures *Word*, *Char* and *Word-Char* in DyNet (Neubig et al., 2017), which is the same framework used as by Plank et al. (2016). We use the hyper-parameter settings by Plank et al. (2016), i.e. we train 20 epochs using Statistical-Gradient-Descent with a learning rate of 0.1 and adding Gaussian noise of 0.2 to the embedding layer. To control also the amount of training data, we use (for now) of each corpus 50k tokens in a 10fold cross-validation setup for comparability.

Results In Figure 6.3, we show the averaged results per language group for the LSTM architectures. The *Word-Char-Ref* tagger performs best followed by our reimplementation *Word-Char*. The difference in accuracy between the various architectures is small for the Germanic and Romanic languages. However, for Slavic and Other, which use much more fine-grained tagsets than Germanic and Romanic, the differences become more clearly visible. It is surprising to see that *Char* performs slightly better than *Word* for all four language groups. We assume that the *Word* architecture is challenged by a high number of unknown word forms. To some extend this is also explainable by the training corpus size that we are using in this experiment for comparability, i.e. the *Word* architecture faces many unknown word forms. The *Char* model, which uses only sub-word information, seems to have considerable advantages here by focusing on a word’s morphology. Furthermore, focusing on morphological properties require considerably less training data than a word-based model. Consequently, it is not surprising that the combination of *Word* and *Char* reaches a better performance. The character level model works essentially as fallback in the combined *Word-Char* architecture for unknown word forms. The HMM baseline tagger is superior to the *Word* and *Char* LSTMs but stays behind the results achieved by combining these two LSTM architectures.

We show in Figure A.2 and Table A.2 in the Appendix the detailed results per language that we discussed here.

6.2.3 Conditional Random Field (CRF) Tagger

We implement an own CRF tagger using FlexTag (Zesch and Horsmann, 2016). As feature set we use a trigram word window of ± 1 words to the right and left, the 750 most frequent character ngrams of length [1..4] and add semantic knowledge from Brown (Brown et al., 1992) clusters.

We arrived at this feature set by running a series of evaluation experiments. We reviewed the recent literature for commonly used features and found word ngrams, fixed character sequences focusing on either pre-, in-, or suffixes of words and word distributional knowledge for PoS taggers of various languages (Brants, 2000; Halácsy et al., 2007; Ljubetić et al., 2016). To the best of our knowledge there is no agreement for parametrization of word- and character ngrams that work best for a single or multiple languages. To arrive at a multilingual feature set (parametrization), we evaluate all parameter permutations using 10fold cross-validation to find a parametrization that works reasonably well across all languages in our setup.

Word Features We experiment with adding the $\{1, 2, 3\}$ words to the right and left of the current word as lower-cased string features.

Character Features Which character ngram is discriminative for a language depends on the language and its morphological properties. To avoid a language bias, we use a frequency-based approach in which we select the N most frequently occurring

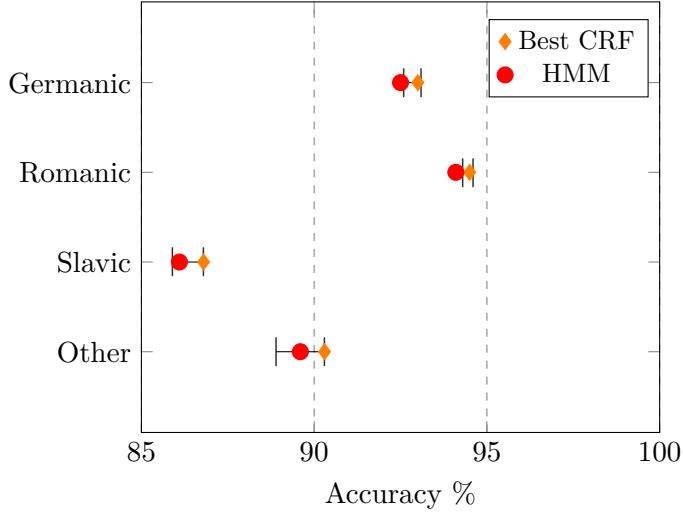


FIGURE 6.4: Averaged results of the CRF feature parametrization experiment across language groups (10fold CV)

character ngrams of length $\{1, 2, 3, 4\}$ from the training dataset. We experiment with a frequency cut-off that uses the $N \in \{250, 500, 750, 1000\}$ most frequent character ngrams. The intuition is that only frequent character ngrams are informative. This grants the character ngrams a flexibility to automatically consider frequent ngrams with respect to a language. These N features are boolean and are set to 1 if the respective character ngram occurs in the current word.

Semantic Features We use Brown clustering to create word clusters for each language. We use the same $15 \cdot 10^6$ tokens of plain text from the Leipzig Corpus Collection that we used for training the word embeddings for the LSTM experiment. We provide the cluster ids in substrings of varying length² to the classifier (Miller et al., 2004; Koo et al., 2008; Owoputi et al., 2013).

Experimental Setup We use of each corpus 50k tokens and report averaged results of running each feature variation as 10fold cross-validation.

Results In Figure 6.4, we show the results of our parameter search experiment averaged by language group. The diamond symbol shows the configuration which works best over all corpora. We show additionally the averaged performance of the HMM baseline tagger as point of reference. The horizontal line marks the variance between worst and best performance across all parameter configurations and languages. The *Best CRF* configuration that worked best across all languages uses a word-context window of 1 word to the left and right and the 750 most frequent character [1..4] grams with additionally adding word clusters.

²For each word with an entry, we create several cluster-related features of which each encodes the cluster id, which is a bit string, in increasing length, i.e. 2, 4, 6, ..., N

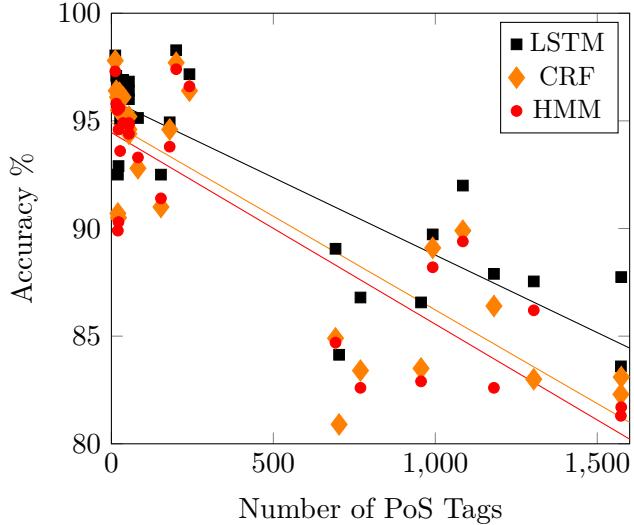


FIGURE 6.5: Effect of tagset size on accuracy for a HMM, CRF and LSTM tagger

When looking at the detailed results per language in Figure A.1 or in Table A.1 in the appendix, we find that *Best CRF* does not necessarily reach the best results for each individual language but is always among the best working ones. In particular, for the *Slavic* and *Other* group, which contain languages richer in morphology than *Germanic* or *Romanic*, the variance in performance is quite large. We account this wider variance to the higher morphological complexity of these languages but also to the usually considerably more fine-grained tagsets used for these languages. On Slavic languages the character ngrams perform much better than using only word ngrams or clusters, but we also see that a low number of character ngrams perform extremely poorly, i.e. Slavic languages require a higher number of character ngrams to account for the higher morphological complexity. Hence, a rather naïve strategy to achieving a decent performance on almost any language is to just use all kinds of character ngrams.

Table A.1 shows that also the cluster feature performs better than using only word ngrams. Considering that we had to limit the amount of data for cluster creation for reasons of comparability, we can conclude that this feature has more potential when using larger data sizes (Derczynski et al., 2015).

The combination of all features shows that the features address quite different information and add up well. A rather important observation are the differences to the HMM baseline tagger. The HMM tagger is often competitive, which shows that off-the-shelf taggers do not necessarily have a disadvantage over constructing an own tagger. We will subsequently use the *Best CRF* configuration when discussing CRF tagger results.

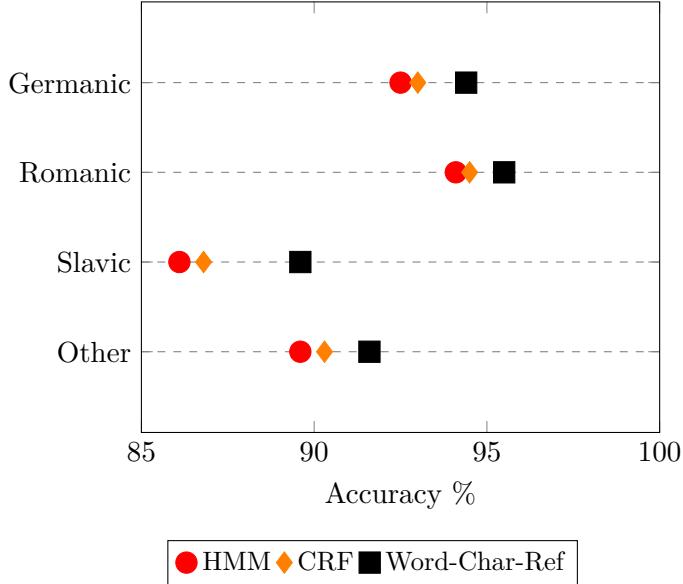


FIGURE 6.6: Results of a HMM, CRF and LSTM tagger on multilingual corpora (10fold CV). Results are shown as aggregate results over all corpora in a language group

6.3 Direct Comparison of Taggers

Figure 6.6 shows the results for each of the three taggers for a direct comparison. CRF is the *Best CRF* discussed in Section 6.2.3 and LSTM is the tagger implementation by Plank et al. (2016). The evaluation is again conducted on 50k tokens of each corpus and reported results are averaged values over 10fold cross-validation.

The CRF and HMM taggers are for Germanic and Romanic languages rather close to the LSTM results. In particular on Slavic languages, the difference between taggers grows larger which might be explicable by the more fine-grained tagsets used by morphologically rich languages. A detailed comparison for each individual language is found again in the appendix in Figure A.3.

We show in Figure 6.5 the relationship between accuracy and tagset size. For each PoS tagger, a regression trendline is plotted, which indicates the average loss in accuracy with an increasing tagset size. For one-hundred additional PoS tags, the LSTM tagger loses .72 points in accuracy, while the CRF HMM tagger have a much steeper decay of .87 to .89 points. Hence, with growing tagset size the tagger choice becomes increasingly more important.

6.4 Comparison to State-of-the-art Taggers

Our experiments until now were limited to the fixed dataset size that we set at the beginning for comparability. In particular, for the morphologically fine-grained tagsets this might have been problematic, i.e. it is doubtful if all PoS tags of a morphological tagset do even occur on 50k tokens. Furthermore, one can expect that there is some difference in accuracy between a language-independent tagger and one that is fitted to a certain language. To quantify this difference and learn more

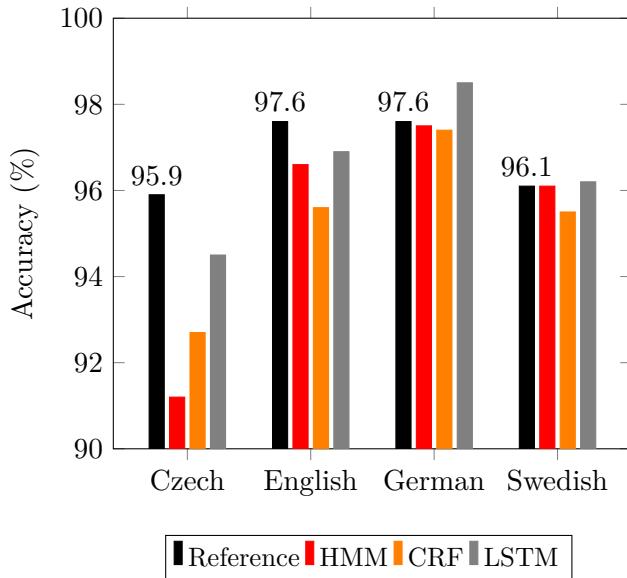


FIGURE 6.7: Results of reproducing setups in the literature with a HMM, CRF and LSTM tagger utilizing the full corpus size for selected languages

about the trade-off in accuracy when picking a language-independent tagger over a language-fitted one, we compare the language-independent taggers to reference results found in the literature.

Experimental Setup This experiment limits the number of comparisons to corpora that we have in our evaluation dataset and for which we also find reference values in the literature.

This constraint leaves us with the following setups that we can reproduce here. For *Czech*, the setup by Spouštová et al. (2009), with training on 10^6 and evaluation on $2 \cdot 10^5$ tokens, for *German-2*, the setup by Giesbrecht and Evert (2009) and for *Swedish-2*, the setup by Östling (2013), which both use 10fold cross-validation over the full corpus size. Taggers for Slavic languages often make use of additional resources such as morphological dictionaries, such additional knowledge sources might be a reason why a language-fitted tagger reaches improvements over a tagger without such a resource. We include in the taggers in our setup intentionally no human-crafted resources to give no language an (unfair) advantage. This leads to a limitation to train models only on the training data and provide distributional knowledge that is usually available for any language. Thus, we do not expect to reach state-of-the-art performance but we want to quantify the difference in accuracy that is accompanied by using a language independent tagger, compared to using language-fitted reference implementation.

We also add one additional corpus to our setup, sections 22-24 of the Wall Street Journal (WSJ) (Marcus et al., 1993) that we compare to the best reported results by Choi (2016), i.e. section 22-24 are the commonly used evaluation dataset of this corpus (Collins, 2002).

Results In Figure 6.7, we show the results. On *German* and *Swedish*, the LSTM tagger is able to reach better results than the reported reference values. This confirms our previous observations once more; the LSTM tagger fits better than the CRF and HMM to the data of a certain corpus. For *Czech* and *English*, none of the taggers reaches the reference value but the LSTM still performs better than the HMM and CRF tagger. For the English corpus, the CRF tagger is considerably more challenged than the HMM and LSTM model. This might be accountable to the feature set that we use for this tagger, i.e. a trigram window with one word to the right and left. It is for good reason common to use a context window that considers the previous two words when building taggers specifically for English. This is a clear disadvantage in this case. The best result reported for the English WSJ corpus can be suspected to be highly fitted to this particular corpus for its frequent use as de-facto standard reference value for evaluating English PoS taggers. Therefore, the close approximation of the HMM and LSTM tagger are still good. For Czech, we see the largest difference to the reference value in this comparison. The LSTM tagger reaches with a difference of about 1.5 percent points the best approximation. The accuracy of CRF and HMM are clearly dissatisfactory, which is easily explainable by the highly fine-grained tagset size of the Czech corpus (> 1500 tags). We suspect that the feature set used by the HMM and CRF tagger is not suited to learn this many fine-grained distinctions.

Thus, we find that language-fitted PoS tagger might still reach better results than a language-independent LSTM tagger, but we also find that differences between tagger are small when the tagset of the corpus is small.

6.5 Chapter Conclusion

In this chapter, we investigated multilingual robustness of PoS taggers by replicating the setup by Plank et al. (2016). We compared a HMM, CRF and LSTM PoS taggers with each other on a multilingual corpus evaluation set that we newly collected composing of 27 corpora of 21 languages. We find that language robustness is in general a less severe challenge than domain robustness, which we investigated in the previous chapters. The differences between taggers remain small as long training corpora use small tagsets. The choice of the tagger becomes increasingly more important for large tagsets. In particular, the LSTM performs considerably better on morphologically fine-grained tagsets compared to HMM and CRF. This also means that a language is not difficult per-se but rather that the PoS granularity of the annotation makes a particular corpus challenging to train an accurate model on.

Despite our efforts to keep things fair and comparable, the LSTM tagger has a small advantage as it learns the most suited feature representation by itself for each language. This is certainly a merit of using a LSTM but the results in this chapter should not be understood as proof of a general superiority of LSTMs over a HMM or CRF tagger. When evaluating different feature sets for the CRF tagger (see Section 6.2.3), we saw already that more suited feature sets exist. If more knowledge

about a particular language is available, configurations might be found that perform better than the LSTM tagger.

A comparison of tagging results to reference values found in the literature showed that language robust taggers do not necessarily perform worse than language-fitted taggers. If language-fitted taggers rely on human-created resources, the LSTM was the only tagger that achieved an acceptable approximation. This also showed that language-specific knowledge from external resources is the key factor that makes a certain tagger implementation superior on a particular language.

Chapter 7

Long Tail Robustness

In the previous chapters, we discussed domain and language robustness. In this chapter, we focus on a tagger's robustness with respect to dealing with a lack of training data of linguistic low-frequency phenomena. As long as a sufficient amount of training data (of a phenomenon) is available, a model can be trained that is able to learn to recognize a certain phenomenon accurately. When working in low-resourced domains such as social media, the assumption of sufficient training data is usually not fulfilled. Many domain-specific phenomena are often under-represented in the small training corpora. If one assumes a Zipf-like distribution as shown in Figure 7.1, many vocabulary items occur in the long tail of this distribution, i.e. occur extremely infrequently. Consequently, they occur equally rarely in an annotated corpus. This means that many domain-specific phenomena are not learnable during model training due to their infrequency.

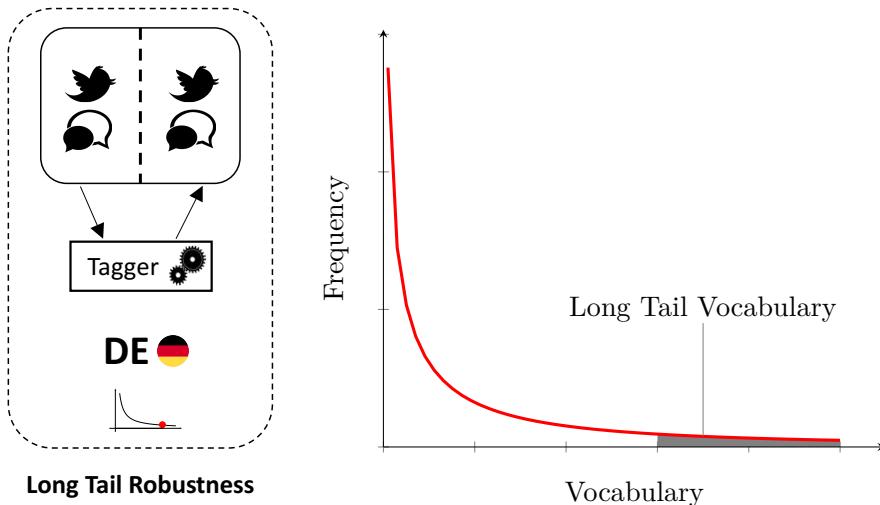


FIGURE 7.1: Long tail robustness

Thus, even when using the approaches discussed in Chapter 4 or Chapter 5 to improve domain robustness, the unique phenomena of a domain are still difficult to tag accurately. This is a particularly severe problem as these domain-specific phenomena often are the main reason for working in low-resourced domains. A linguist who wants to study domain peculiarities such as shifts in word usage or domain-specific word formations require datasets with a sufficient number of samples of such rare

phenomena. A rather obvious approach to deal with this lack of data is to collect and annotate more samples that contain a certain phenomenon. While this would certainly work, it is also an extremely expensive and time-consuming process that scales poorly. Furthermore, it is unclear how many instances one would have to annotate to reach an acceptable performance.

Use-case The work in this chapter is motivated by a practical example for German in which a linguist wants to use a PoS tagger as corpus query tool. Usually, linguists screen many plain text samples to find relevant instances of a phenomenon. Working on social media means that this manual screening is highly time-intensive as many irrelevant postings have to be screened, too. The idea of using a tagger as query tool is based on a specific PoS tagset that assigns a phenomenon of interest an own PoS tag. If a tagger would accurately tag the phenomenon with its tag, a large number of postings without this tag could be easily ignored. This would speed-up the screening of plain text samples considerably and allow acquiring new examples of this phenomena with only a fraction of the manual effort. Furthermore, for studying such a phenomenon, the tagger should not just tag instances correctly that occurred in the training data but also be able to correctly tag out-of-vocabulary instances of this phenomenon. As this phenomenon is comparatively rare and occurs only a few times in a small annotated dataset that is available to us, training a model that accurately tags this phenomenon is not possible.

Thus, in this chapter, we investigate how to deal with the lack of training data and improve tagging of such under-represented phenomena, while keeping the manual annotation effort at a minimum.

Dataset We base this work on the German social media corpus by Beißwenger et al. (2016). The corpus contains 23k tokens from a mixture of social media sub-domains, such as Twitter, WhatsApp chats, Blog comments, etc. This dataset is annotated with the extended version (Beißwenger et al., 2015) of the Stuttgart-Tübingen tagset (STTS) (Schiller et al., 1999), which adds 18 additional tags to the canonical STTS with 54 tags. These additional tags target many typical phenomena of informal and colloquial utterances, which are not found in standard German. We showed in Section 4.4.1 (see Table 4.5) that the newly added tags are usually not learned by a tagger due to their infrequency, even if domain adaptation approaches are applied.

7.1 Fitting Towards a Phenomenon

In this section, we will focus on word contractions of a full verb with a personal pronoun. This word contraction is annotated with an own tag, *VVPPE*, in the shared-task dataset. Table 7.1 shows examples of this type of contractions taken from the Dortmund Chat Corpus (Beißwenger, 2013). This word contractions are not found in standard German which prevents using formal German corpora to obtain

wiederhols = wiederholen (to repeat) + es (it)
 ich **wiederhols** nochmal, ihr redet hier öffentlich!
I repeat it [repeat-it] *again, you're talking in public!*
 I REPEAT IT ONCE MORE, YOU'RE TALKING IN PUBLIC!

kommste = kommen (to come) + du (you)
 wieso? wo **kommste** denn her?
why? where do [come-you] *from?*
 WHY? WHERE DO YOU COME FROM?

findeste = finden (to find) + du (you)
 nò,dat beste **findeste** eigentlich wenn du gar nich suchst ...
nope, you find [find-you] *the best when you're not searching for it*
 NO, YOU WILL FIND THE BEST WHEN YOU'RE NOT SEARCHING FOR IT

TABLE 7.1: Examples of full-verb with personal pronoun contractions (VVPPer).
 Bold face shows the contracted words

more data of this phenomenon. We, hence, start with an investigation how to deal with infrequent phenomena.

7.1.1 Experiment: Dealing with Infrequency

The German social media corpus contains in total 13 VVPPer instances, which are not sufficient to train a model that can reliably recognize this tag. Thus, in this experiment, we test different strategies to improve the tagging of VVPPer instances. In order to train the tagger but also to arrive at meaningful results during evaluation, we will need more than the 13 instances we have at the moment. Hence, we are not able to avoid annotation entirely. However, since we are interested in only a particular tag anyway, we don't have to annotate the entire posting. We can focus on correcting the tag of this one word, which avoids a large part of the manual annotation effort.

Inexpensive Annotation of More Data We apply the following strategy to produce more annotated data as shown in Figure 7.2. This strategy requires only a minimalistic amount of manual annotation. We asked the expert (i.e. the linguist) to provide us with additional plain text examples in which the word contraction phenomenon occurs. We obtained 230 additional postings with this phenomenon from the Dortmund Chat Corpus. We machine tagged these additional data by using the Stanford (Toutanova and Manning, 2000) tagger with one of the provided models for German that assigns PoS tags of the canonical STTS. We use a model trained on newswire data as no German social model is available.

The results from Chapter 3.3, the evaluation of available PoS tagger models, showed that newswire-trained model perform poorly on social media text. We yet decided to use a newswire model for lack of alternatives and to keep the annotation

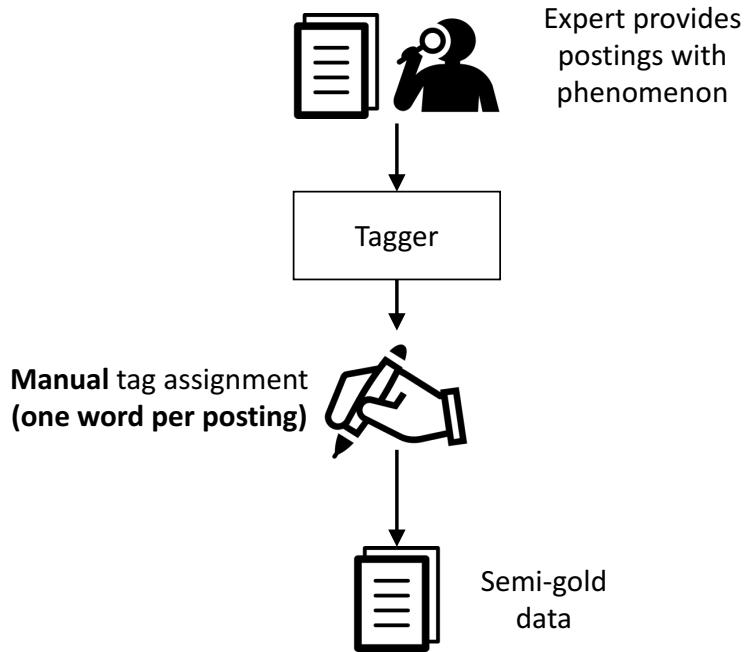


FIGURE 7.2: Inexpensive production of training data for under-resourced phenomena

process as simple and straight-forward as possible, which is sufficient to reach improvements, as we will see soon. The choice of the Stanford tagger is motivated by its good reputation, the ease of installing and using this tagger and for the availability of a German model that assigns STTS tags. Any other tagger which provides a German STTS compatible model should be equally suited. If a tagger or model is available that is adapted to German social media text, this one should be of course preferred. However, this is just another fine-tuning step. We focus here on the principle approach and use a newswire-trained model for automatically tagging.

The tag of the word contraction is the only tag that is manually corrected and set to VVP PER of the extended STTS. This is the most minimalistic amount of manual annotation one can possibly perform. We will soon see that this is enough to reach considerable improvements, despite of the poor performance of newswire-trained models on social media text.

Experimental Setup An option to circumvent annotation of a larger amount of data is boosting the signal for a certain PoS tag in the already existing data. This can either be done by oversampling (Daumé III, 2007) or downsampling as alternative to provide more (fresh) training data. Oversampling adds the few available instances N times to the training set. Downsampling removes sequences *without* the PoS tag of interest, i.e. VVP PER. Both approaches lead to an increased frequency weight of the phenomenon relative to the other PoS tags in the corpus. Our evaluation dataset is created from the few existing instances and the 230 freshly created ones. Of the 230 additionally obtained instances, we add one half to the testing set and one-sixth to the training set. The remaining two-sixths are our development set and are held back

for the moment. Thus, the train set for this experiment contains 45 (38+7) postings with the phenomenon and the test set 121 (115+6) postings. We use an intentionally large test set to see how well the tagger is able to generalize to this phenomenon.

Oversampling/new Instances: We choose oversampling rates that add a number of instances which we can also provide from the held back annotated data. This allows a direct comparison between oversampling instances and adding fresh ones. We oversample two and three times, and compare this to adding the same number of instances from the set of new sequences in the held back development set.

Downsampling: We remove 25, 50 and 75 percent of the training data instances that do not contain any full verb contractions.

Furthermore, we conduct these experiments with several taggers to learn about the empirical differences between tagger implementations. The taggers essentially face two challenges (i), infrequency of training instances and (ii), the noisy nature of our additionally created data. Our assumption is that some taggers might be better suited for dealing with such a setup than others. We use two taggers that are frequently used in the literature where we can assume that they work accurately. Additionally, we add a rather recently published tagger to our setup based on neural networks, which reported good results on a variety of languages, and we include the two-step tagging approach that we discussed in Chapter 5.

Stanford (Toutanova et al., 2003). A PoS tagger that is frequently used in the community due to its good reputation and high accuracy.

HunPos (Halácsy et al., 2007). A further tagger with a good reputation based on Hidden-Markov models and a re-implementation of the TNT tagger (Brants, 2000).

LSTM (Plank et al., 2016). A deep learning PoS tagger that is based on Long-Short-Term-Memory (Hochreiter and Schmidhuber, 1997) neural networks. We use the same parametrization as Plank et al. (2016) and a self-trained German word embedding trained on $195 \cdot 10^6$ tokens German Twitter messages.

Two-step Tagging Briefly summarized, a coarse-grained tagging is performed in a first step, which is refined to the fine-grained tag in the second step. The idea is to reduce the complexity of the task by dealing with smaller sub-problems that are easier to solve and, hence, reach an improved accuracy. The second tagger is tailored towards recognizing the tag of interest while the first tagging step constraints the application of the second tagger. We implement this approach by using a CRF tagger (Lafferty et al., 2001) in the first step and a Support Vector Machine (SVM) in the second step. For training the coarse-grained sequence model, we map the extended-STTS tags of the training data to the coarse-grained tagset used by the Universal Dependency project and map VVPPER to *verb*. We include a PoS dictionary and Brown (Brown et al., 1992) clusters created over German Twitter messages to compensate for the

Setup	F_1	
	All	OOV
HunPos	Baseline	.78 .52
	Downsampling 75%	.78 .54
	Downsampling 50%	.79 .53
	Downsampling 25%	.79 .53
	Oversampling x2	.79 .53
	Oversampling x3	.79 .53
	Annotated x2	.83 .65
Two-Step	Annotated x3	.88 .75
	Baseline	.77 .48
	Downsampling 75%	.78 .51
	Downsampling 50%	.80 .55
	Downsampling 25%	.79 .53
	Oversampling x2	.77 .48
	Oversampling x3	.77 .48
Annotated x2	Annotated x2	.81 .59
	Annotated x3	.85 .69

TABLE 7.2: F-Score on full-verb with personal pronoun contractions for all and OOV words. Bold face shows best overall F-Score for each tagger and gray shading shows best results only on OOV words

lack of training data. The coarse-grained CRF tagger uses a trigram word context window, the 750 most frequent character bi-grams, tri-grams and four-grams, and boolean features for capitalization and numeric values. We use as SVM features character bi-grams and tri-grams over all verb forms and the local bi-gram and tri-gram word context. Please note, we ignore the other word classes since we are only interested in tagging VVPVER. This coarse-grained tagger reaches a F_1 of 0.93 on the tag *Verb* in the test data, which means that some VVPVER instances will be missed because the coarse model did not predict *verb*.

Results In Figure 7.3, we show the results of the three strategies on the VVPVER tag. We focus on *out-of-vocabulary* instances which perform considerably poorer than *in-vocabulary* instances (F_1 between 0.96 to 0.99), and thus, offer more opportunities for improvements. We see that neither downsampling nor oversampling helps to reach a substantial improvement on the tag. Furthermore, downsampling shows that the anyway low amount of training data becomes a large problem for the LSTM if further reduced. The Stanford tagger stays behind the other taggers with both sampling methods. The only effective method is, without much surprise, providing new data. The LSTM needs considerably more data to improve while the other taggers improve linearly with each new dataset.

Discussion Table 7.2 shows the F-Score of the two best taggers HunPoS and Two-Step in each experiment. The overall F-Score shows that both taggers reach a rather similar overall performance. The only effective approach was providing fresh data. Oversampling showed no effect which suggest that the improvements origin in lexical knowledge from the added data. This also means that the taggers seem to focus too

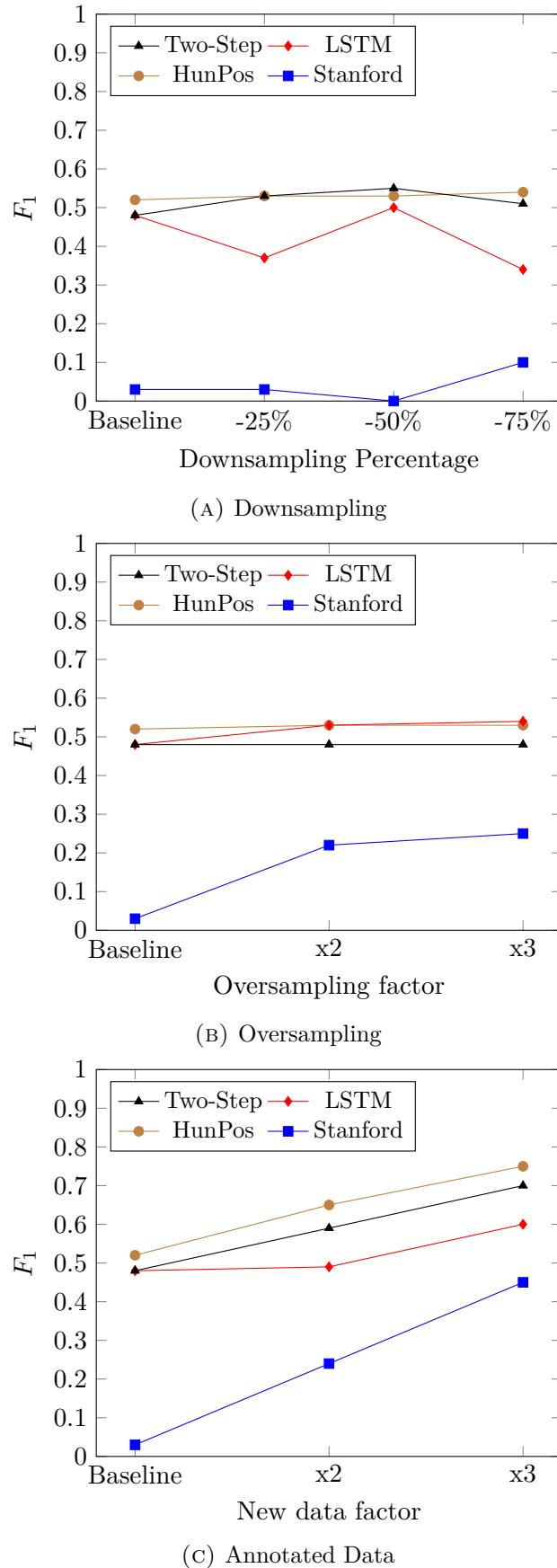


FIGURE 7.3: Results of tagging out-of-vocabulary instances of *full-verb with personal pronoun contractions (VVPPER)* for each tagger and each approach

Configuration	F_1	
	All	OOV
Baseline	.81 (+.04)	.57 (+.09)
Annotated x3	.86 (+.01)	.73 (+.04)

TABLE 7.3: Improvements of the contextualised two-step tagging on full-verb with personal pronoun contractions

much on the word form and less on the word context. This has important implications for the tagger’s ability to find unknown instances. Without weighting the word context sufficiently, this tagger is not well suited for retrieving new instances of a phenomenon. Thus, we investigate next if we can improve the performance of Two-Step by forcing it to focus more strongly on the local word context.

7.1.2 Experiment: Forced Generalization

In this experiment, we try to improve generalization of the Two-step tagger by forcing the tagger to rely more on the local word context to improve the recall. This investigation focuses only on Two-Step as this tagger is fully self-implemented. We alter the feature space of the SVM and exclude all features that inform the classifier about the lexical form of the VVP PER instances. Thus, the SVM is not aware of any lexical forms, it must now rely on the word context to recognize VVP PER instances.

Results In Table 7.3, we show the changes in performance of the contextualized Two-Step tagger. In parentheses, we show the differences to the not contextualized tagger in Table 7.2. For both setups, we see an improved F-Score but especially the recall increases for out-of-vocabulary instances. The F-Score by HunPos (.88) in Table 7.2 is still superior. However, Two-Step’s trade-off between precision and recall (not shown) supports better the use case in which the tagger functions as a precise filtering tool with decent recall, i.e. a higher precision than HunPoS.

7.1.3 Experiment: Field Trial in Social Media

So far, we have only simulated our use case of a linguist who uses a tagger as a filtering tool, while now, we turn to a real setting and apply the tagger to plain text Twitter messages for finding *full verb-pronoun contractions*.

Working on plain text means that the ground truth of how many instances occur is unknown, which prevents calculating an F-Score. We focus instead on precision as main evaluation metric. We choose the Twitter domain for its ease of obtaining data but also for its linguistic diversity that ranges from informal, interactional language to messages that are similar to formally written text. This domain provides us with a challenging testbed that should allow to determine a conservative, lower-bound performance for our approach. We will use the contextualized Two-Step tagger for its higher precision while providing a reasonable high recall.

Twitter Data We use a subsample of 50k tweets (about 1.7 million tokens) crawled between 2011 and 2017 from the public Twitter API that we language-filtered for German. All occurrences of user-mentions, hashtags and URLs are replaced by a text constant and the Tweets are tokenized by Gimpel et al. (2011)'s ArkTools tokenizer.

Tagger Setup We train the coarse-grained model and the SVMs on the full shared task dataset including the additionally annotated data. To provide more lexical knowledge and increase the robustness when facing standard language text, we also add 100k tokens of German newswire text of the Tiger (Brants et al., 2004) corpus to both steps.

Evaluation Setup We evaluate the tagged instances with two annotators. The annotators make four distinctions: *strict*, *relaxed*, *all* and *none*. *Strict* are full verb contractions with personal pronoun, the exact phenomenon we intended to tag. *Relaxed* counts all verb contractions with personal pronoun as correct, this includes also modal and auxiliary verbs. *All* counts all contractions phenomena as correct, this additionally includes, for instance, contractions of conjunctions with personal pronouns. The remaining cases are no contractions and are, thus, false positives.

We will evaluate two setups. The first one selects the first 250 of all found instances, which will be the overall evaluation. The second evaluation focuses on out-of-vocabulary instances in which we remove all tagged instances that are known from the training set until we gather 250 instance and, thus, evaluate how reliably new instances are found.

Results In total, we found 1,091 instances in 50k tweets tagged as *VVPPE*. The two annotators reached a perfect agreement on the subset of the first 250 instances that we evaluated manually. Figure 7.4a shows the precision of the overall evaluation. The *strict* result shows that the majority of found instances are the targeted full verb contractions. Including modal and auxiliary verbs in the *relaxed* mode, even three-quarter are verb contractions. When including also miscellaneous contractions in *all*, almost all instances are contractions.

In Figure 7.4b, we take a closer look on the performance of detecting new contractions, e.g. out-of-vocabulary instances. We focus our discussion on the *strict* results. The precision is drastically decreased to almost half of what we reach when including all instances. We also computed the type/token ratio which is at 0.69 almost twice as high as in the overall evaluation in Figure 7.4a. This confirms that the tagger is able to recognize many new instances of the phenomenon. Furthermore, when ignoring the known instances almost every correct instance is a new lexical form.

Analysis Table 7.4 shows examples of found instances for each of the three contraction classes (bold face). Many of the provided VVPPE training instances end on *s* or *'s*, which is a common morphological property of contractions in German. On the

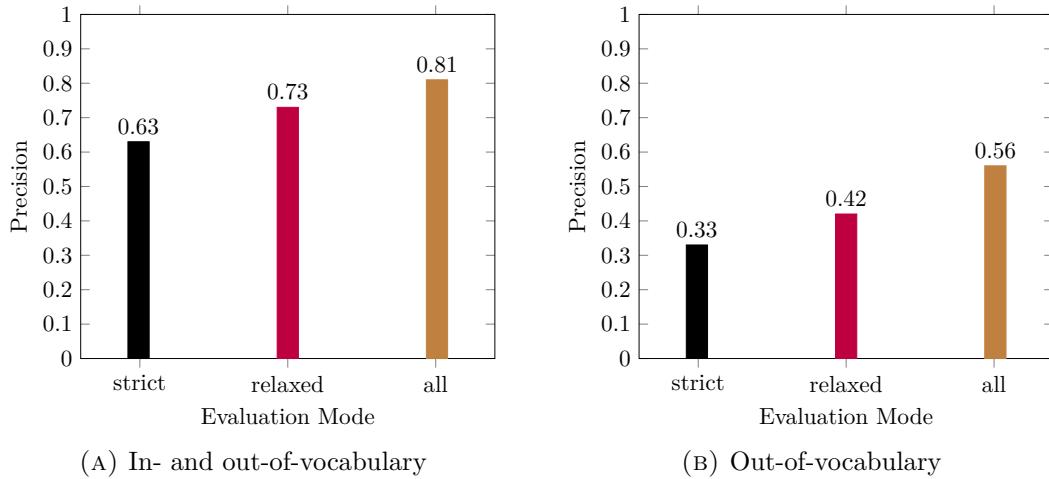


FIGURE 7.4: Evaluation for precision of tagging full-verb with personal pronoun contractions in Twitter plain text. *Strict* are correctly tagged VVPPER postings, *Relaxed* are all verb forms with personal pronoun contractions including auxiliary and modal verbs and *All* are any kind of word contraction of all word classes

one hand, this bias introduces a substantial number of false positives, for instance the verb *weiß* (*to know*) occurs frequently as *weiss* in social media. On the other hand, this enables the SVM to also tag similar contraction cases of other word classes in *relaxed* or *all*.

7.2 Generalization of the Approach

The methods we introduced in Section 7.1 focus on a particular verb contraction form that we chose as leading example to develop the fitting approach. This naturally raises the question if this fitting works likewise for other phenomena that are similarly under-represented. We reproduce in this section our own experiments for two other phenomena to confirm the general applicability of this approach.

7.2.1 Fitting to Contractions of Adverbs with Articles

In this experiment, we adapt our tagger to a contraction of adverbs with articles, examples of this phenomenon are shown in Figure 7.5. This phenomenon is annotated in the shared task dataset with the tag ADVART. We have one occurrence of this tag in the shared task training data and three in the testing data. Naturally, (3+1) instances are not sufficient to learn to recognize this tag.

Experiment: Tagging ADVART At first, we need again additional samples of the phenomenon for constructing a sufficiently large evaluation dataset. We use again the Dortmund Chat Corpus from which we obtain 18 postings in total. Furthermore, most of the newly retrieved instances are of the same type. Hence, the additionally retrieved instances are magnitudes lower than for the verb contraction phenomena that we discussed in Section 7.1. This case will provide us with a lower bound

Found Contractions			
Strict	lernste = lernst (learn) + du (you)	Da lernste pragmatisch zu sein .	
	There <i>[learn-you]</i> to be pragmatic		
	YOU WILL LEARN TO BE PRAGMATIC THERE		
Relaxed	sachs = sagen (tell) + es (it)	Ich sachs dir noch .	
	<i>I</i> <i>[told-it]</i> <i>you so</i>		
	I TOLD YOU SO		
Relaxed	häts = hätte (had) + es (it)	Wer häts gedacht .	
	<i>Who would</i> <i>[had-it]</i> <i>thought of</i>		
	WHO WOULD HAVE THOUGHT OF IT		
All	wills = will (want) + es (it)	Ich wills nicht ich will aber auch nicht [...]	
	<i>I don't</i> <i>[want-it]</i> <i>but I don't want ... either</i>		
	I DON'T WANT THAT BUT I DON'T WANT [...] EITHER		
All	du's = du (you) + es (it)	Würdest du's mir heute noch sagen?	
	<i>Would</i> <i>[you-it]</i> <i>still tell me today?</i>		
	WOULD YOU STILL TELL ME TODAY?		
All	für's = für (for) + das (the)	Danke für's retweeten	
	<i>Thanks</i> <i>[for-the]</i> <i>retweeting</i>		
	THANKS FOR RETWEETING		

TABLE 7.4: Twitter postings that have been tagged as full-verb with personal pronoun contractions (VVPPer). *Strict* are correctly tagged VVPPer postings, *Relaxed* are all verb forms with personal pronoun contractions including auxiliary and modal verbs and *All* are any kind of word contraction of all word classes

estimation of the performance one can expect if more data is hard to obtain. We automatically tag these sentences with the Stanford tagger and hand correct the tag for the ADVART instances. The enhanced training set contains now 10+1 instances and the testing set 8+3 instances. This means we have now some additional data samples with barely any additional types of the phenomenon.

We use again Two-Step as tagger, the coarse-grained sequence tagging model is trained on the enhanced training set that is mapped to coarse-grained tags. The ADVART tag is mapped to the coarse tag for *adverb*. We add Brown clusters created from German Twitter messages as resources. For the second step, we train a contextualized (see Section 7.1.2) SVM that uses the same Brown clusters as the coarse-grained model. We use HunPos as baseline tagger.

son = so (such) + einen (an)
noch nie gesehen, son archiv
<i>never seen</i> <i>[such-an]</i> <i>archive</i>
(I HAVE) NEVER SEEN SUCH AN ARCHIVE
auchn = auch (also) + eine (an)
oz hat auchn octavia
<i>oz has</i> <i>[also-an]</i> <i>octavia</i>
OZ HAS AN OCTAVIA
nurn = nur (just) + ein (a)
is halt nurn bischen teuer ...
<i>it is</i> <i>[just-a]</i> <i>bit expensive</i>
IT IS JUST A BIT EXPENSIVE

TABLE 7.5: Examples of adverb with article contractions (ADVART). Bold faced words are the contracted words

Results We reach for both taggers comparable results for tagging ADVART, as shown in Table 7.6. Two step tagging reaches a considerably better performance on OOV words. We also analyzed the F-Score for only known instances (not shown) and find that known instances are accurately tagged. Thus, two-step tagging supports the use case of a corpus query tool better than HunPoS for tagging ADVART contractions, although these results have to be taken with caution as the evaluation dataset contains only eleven instances of six types.

Experiment: Finding ADVART Instances in Plain Text In this experiment, we use the Two-Step tagger for finding ADVART instances in plain text. We use the setup as for the verb contraction experiment described in Section 7.1.3. Briefly summarized, we train the sequence model and the SVM on the shared task data, 100k tokens of the Tiger corpus and the manually retrieved instances from the Dortmund Chat Corpus and tag 50k plain text Twitter postings.

Results The tagger tags in 50k Twitter messages in total 83 instances as ADVART. Figure 7.5 shows the precision of the detection. We found an additional kind of contraction in our data in which the adverb and article occurred as own words (thus, not a contraction in the actual sense) but where the article was shortened to a single letter. For instance, “so n (such an)” is not a single word form but a strongly related

Tagger	F_1	
	All	OOV
Two-Step	0.84	0.67
HunPoS	0.84	0.40

TABLE 7.6: F-Score results on ADVART instances

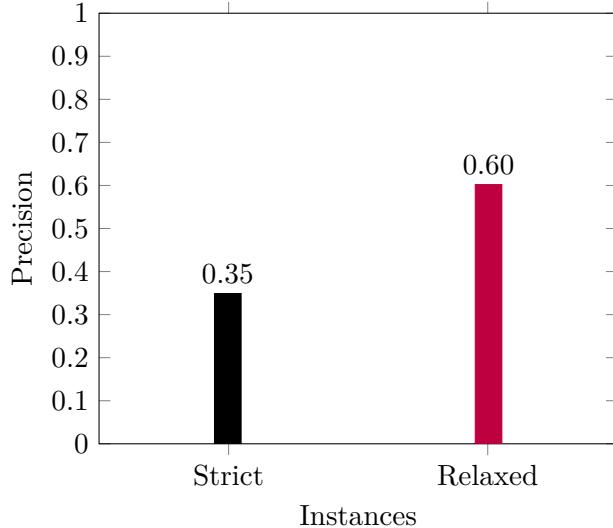


FIGURE 7.5: Evaluation for precision of tagging adverb with article contractions (ADVART) in Twitter plain text. *Strict* are correctly tagged ADVART instances, *Relaxed* are instances that still occur as two words with the second word being contracted to a single letter

phenomenon that we consider additionally in our evaluation. The *strict* result show found ADVART instances while *relaxed* also includes the partially contracted cases that still appear as two own words.

We find in total only two new OOV instances, which is easily explained by a low number of training instances for this phenomenon, i.e. the 22 instances in our training data compose of nine types of which one occurs ten times. Thus, the tagger had only very few information to learn how to recognize new instances, which led to an unsurprisingly low number of OOV instances. Thus, with few training instances, we find two-step tagging to be highly conservative yielding only a low number of actual hits with barely any new ones.

As frequent error case, we find instances in which an indirect article such as “ein/e (a)” is contracted to a single letter “n”, which indicates that the taggers put a strong emphasis on the suffix of an instance to recognize ADVART instances. We show examples of the found instances in Table 7.7.

7.2.2 Fitting to Contractions of Preposition With Articles

In this experiment, we fit a tagger to contractions of *preposition with articles* that are labeled with the tag APPRART. This tag does already exist in the canonical STTS as this kind of contractions also occurs in standard German. For instance, the word “im” is such a contraction of “in dem (in the)”. However, in social media text we find a large number of other contractions that are not part of standard German. These instances are also labeled as ADVART in the shared task dataset. We show some of those non-standard cases in Table 7.8. The contracted words classes, preposition and article, are both closed word classes which means that there is a limit in the number

Found Contractions		
	nurn = nur (only) + ein (a)	
Strict	... hmm aber nurn glas	
	... <i>hmm but</i> [<i>only-a</i>] <i>glass</i>	
	(I WILL DRINK) ONLY ONE GLASS	
	mitem = mit (with) + ein (a)	
	ja is mitem Firmenlogo	
	<i>yes it is</i> [<i>with-a</i>] <i>company logo</i>	
	YES, IT HAS A COMPANY LOGO	
	noch n = noch (still) + ein (a)	
Relaxed	bin mir unsicher ... hab noch n komisches Gefühl	
	<i>I am unsure ... have</i> [<i>still-a</i>] <i>strange feeling</i>	
	I AM NOT SURE, I STILL HAVE A STRANGE FEELING	
	nur n = nur (only) + ein (a)	
	Da wurde halt nur n bisschen geläster	
	<i>There was</i> [<i>only-a</i>] <i>bit gossiping</i>	
	THERE WAS SOME GOSSIPING	

TABLE 7.7: Twitter postings that have been tagged as adverb with article contractions (ADVART). *Strict* are correctly tagged ADVART instances, *Relaxed* are instances that still occur as two words with the second word being contracted to a single letter

of words that can be contracted but not necessarily in the spelling variations in which this contraction might occur.

Two-step tagging reaches an F-Score of .96 on the tag APPRART which suggest that this class is easy to tag. However, the majority of the 187 instances in the shared task dataset are *standard language forms* and only *five are non-standard*. We re-computed the F-Score considering only non-standard instances and arrive at an F-Score of .57 (over five instances in total). Thus, the total number of non-standard APPRART instances is too low to arrive at meaningful conclusions.

Experiment: Tagging APPRART The first step for reaching meaningful results is to create a dataset that allows verification of our adaptation to this phenomenon. The dataset is enhanced by manually retrieving 81 sentences with non-standard text instances of APPRART from the Dortmund Chat Corpus. We use again the Stanford tagger for machine tagging and manually correct the tag of the APPRART instances. The enhanced dataset contains now 40+5 non-standard APPRART instances in the training set and 41+1 instances in the testing set.

We use again two-step tagging for our experiment. The coarse-grained sequence tagging model in the first step is trained on the shared task training data, include the Brown cluster create over German Twitter messages. The APPRART instances are mapped to the coarse tag for *Adpositions*. The coarse tagging reaches on the enhanced test dataset an F-Score of .95 on the ADP tag to which the APPRART tag is mapped. When focusing on non-standard APPRART forms, we find that 72.5% of

annen = an (on) + den (the)			
aua.... doch net	annen	kopf	
ouch .. but not	[at-the]	head	
OUCH DON'T HIT MY HEAD			
durch's = durch (through) + das (the)			
Wir haben uns	durch's	internet	kennen gelernt
We have met	[through-the]	internet	
WE MET ON THE INTERNET			
fürn = für (for) + einen (a)			
Was quatscht der	fürn	müll	
What	[for-a]	garbage	is he talking
HE IS TALKING NONSENSE			

TABLE 7.8: Examples of preposition with article contractions (APPRART). Bold faced words are contractions

Tagger	F₁	
	All	OOV
Two-Step	.84	.61
HunPoS	.63	.17

TABLE 7.9: F-Score on preposition with article contractions for all and on OOV words

the cases are tagged correctly as adposition. Hence, about one third of those instances will be missed by a wrong prediction in the first tagging step.

For the second step, we train a contextualized (see Section 7.1.2) SVM on the enhanced training dataset and included a Brown cluster created over German Twitter messages. As baseline tagger, we use HunPos.

Results In Table 7.9, we show the results of tagging APPRART instances. The reported results focus on non-standard APPRART text forms only (we excluded standard text forms by excluding all instances found in the Tiger corpus). Two-Step and HunPoS reach a close to perfect F-Score on instances contained in the training data but on out-of-vocabulary word forms, the advantages of Two-Step are striking. Two-Step clearly outperforms HunPoS for this phenomenon.

Experiment: Finding APPRART Instances in Plain Text We use the same setup as for the verb contraction experiment, described in Section 7.1.3, for finding instances in plain text. Briefly summarized, we train the sequence model and the SVM on the shared task data, 100k token of the Tiger corpus and the manually retrieved instances from the Dortmund Chat Corpus. We modify the setup to account for the circumstance that the tag APPRART contains standard text and non-standard text phenomena. As our interest lies only on the non-standard phenomena and the standard text forms are rather frequently encountered, we want to avoid screening

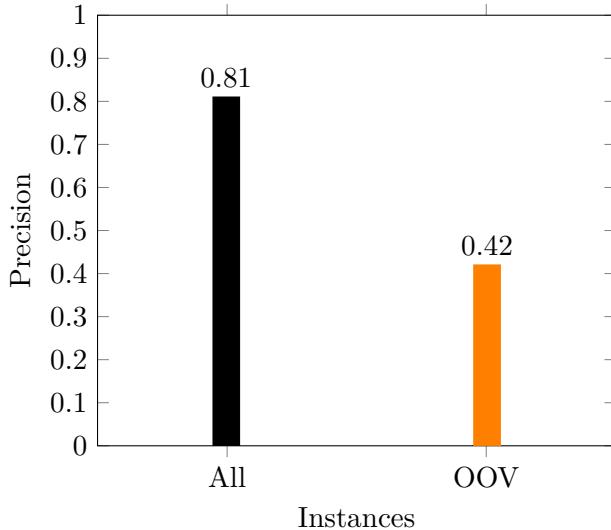


FIGURE 7.6: Evaluation for precision of tagging preposition with article contractions (APPRART) in Twitter plain text

through countless standard text instances. Hence, for training the SVM, we assign to all standard text forms an own tag which allows it to distinguish standard from non-standard APPRART instances. We treat all APPRART instances that occur in the Tiger corpus as standard text instances. This allows a better targeting of the interesting word forms.

Results In the 50k Twitter messages, we find 249 non-standard APPRART instances in total¹. Figure 7.6 shows the precision of detection after a manual verification of each found instance. Unlike with the open word class experiments in Section 7.1.3 and Section 7.2.1, we do not obtain any related or similar contraction forms. This might be accountable to the circumstance that the APPRART contraction composes of two closed word classes. We, thus, distinguish all and OOV instances. *All* shows the precision for all 249 instances which indicates that two-step tagging is quite conservative but in return highly precise in finding APPRART contractions. *OOV* reports the precision on the out-of-vocabulary instances of which we have 79 in total. Less than half of the OOV instances are actual APPRART contractions but among the 33 correct ones we have two-third new types. We, thus, find many new instances of this phenomenon, we show examples in Table 7.10.

We also analyzed the errors and found a particular interjection to be frequently confused as adverb forms. The interjection *ne* (*nope*) is a typical suffix of APPRART instance in which “ne” means “a/the”, as in “sone (so eine = such a). We find “ne” to occur most the time as article but only once as interjection while in the Twitter data “ne” is frequently used as interjection. We, thus, suspect the lack of training instances as cause, i.e. the tagger is not given the chance to distinguish these forms.

¹We find 9,463 instances without distinguishing between standard and non-standard forms

übern = über (over) + den (the)			
sie hat sich immer des öl	übern	körper geschüttet	
<i>she always poured the oil</i>	<i>[over-the]</i>	<i>body.</i>	
SHE ALWAYS POURED THE OIL OVER HER BODY.			
aufem = auf (on) + dem (the)			
Ich wohne aufem Dorf			
<i>I live</i>	<i>[in-the]</i>	<i>village</i>	
I LIVE IN THE VILLAGE			
nach'm = nach (after) + dem (the)			
Bitte aber erst nach'm Mittag			
<i>But please</i>	<i>[after-the]</i>	<i>noon</i>	
BUT PLEASE AFTER NOON			

TABLE 7.10: Twitter postings that have been tagged as preposition with articles

7.3 Chapter Conclusion

In this chapter, we discussed and analyzed strategies to overcome the notorious lack of training data often found in under-resourced text domains. Motivated from the use case of constructing a tagger as corpus query tool to find such under-represented phenomena, we experimented with two strategies to dealing with the lack of training data, namely adjusting the frequency weight and annotation of new data. We altered the frequency weight of the under-represented phenomenon in the corpus by over- and undersampling data. We found that neither strategies improve tagging of the phenomenon under observation and that annotation of more data is unavoidable.

However, we also showed that additional data can be produced in an inexpensive fashion. We produced additional training data of a certain phenomenon by mostly relying on an annotation of a single word per posting and automatically tagging of the remaining words with a newswire-trained model. Furthermore, we compared various PoS taggers for their ability to deal with infrequent phenomena. We find that some taggers are much better equipped to deal with such phenomena than others. The best trade-off between precision and recall was achieved by two-step tagging after forcing the tagger to focus on the local word context, i.e. contextualized tagging. As proof-of-concept, we fitted two-step tagging to three infrequent phenomena and evaluated their performance by tagging Twitter messages. We find that our approach performs rather conservatively but precisely and returns many instances of the desired phenomenon. Among the found instances, we additionally find a high number of new instances which is important for studying such phenomena.

The basic version of data production offers of course many opportunities for achieving additional improvements. We focused on manual annotation of a single word to keep things as simple as possible. A certainly reasonable next step would be to also experiment with a larger manually annotated PoS context. In particular, with

respect to the requirement of being able to find out-of-vocabulary instances, providing a verified local PoS context promises additional improvements. As starting point, one could use a social media adapted tagger model instead of a newswire-trained one, which might be a less expensive methods to achieve more improvements.

Thus, the results of the presented annotation method should be understood as a lower bound, which promises realistic improvements if more effort is invested.

Chapter 8

Technical Prerequisites

In this chapter, we discuss the software tools we used and developed during the course of this thesis. It has been a key requirement that the tools we use are beyond the state of a mere research prototype. This is of particular importance for reproducibility of experiments where running a machine learning algorithm is only a small part of all necessary steps to come to a result.

We subsequently discuss the two projects FlexTag and DeepTC, which are both based on DKPro Text Classification (DKPro TC) (Daxenberger et al., 2014). FlexTag is a highly flexible Part-of-Speech (PoS) tagger that we used for many experiments in this thesis. As the name already suggest, the unique characteristic of FlexTag is its high flexibility compared to other PoS taggers. DeepTC is a deep learning extension to DKPro TC that provides a software environment for end-to-end shareable text classification experiments based on neural networks. DeepTC improves the reproducibility of deep learning code, which is often released in a prototypical and incomplete state. Consequently, such code is *not* easily executable for third-party researchers. FlexTag is based on DKPro TC and development progress in DKPro TC leads consequently also to improvements and extensions for FlexTag. The DeepTC extension adds support for deep learning classifiers to DKPro TC, which have not been supported so far. Subsequently, we will discuss FlexTag and Deep TC in detail.

8.1 FlexTag – A Flexible PoS Tagger

The experiments we conducted entailed experimenting with various information that was provided to the PoS tagger during model training. We varied, for instance, the size of the word context window, number of character ngrams or providing knowledge from external resources. While off-the-shelf taggers allow training of new models, changing the feature set is often not possible. This enforces that each model learning process must always use the same feature set. The conducted experiments in this thesis frequently required changes to the feature set and experimenting with various feature parametrizations. These modifications are not easily possible for off-the-shelf taggers, which treat the used feature set as a fixed, internal component of the tagger that is not exposed to the user. This led to the development of FlexTag, which grants the user a new level of flexibility by enabling a modification of the feature

set. A further objective is to make this flexibility also accessible to non-experts. Feature space modifications are a highly technical task that require a rather high amount of technical expertise. Altering the feature space in FlexTag is easy and less technical experienced user can perform this modification in a straight-forward manner. Furthermore, trained models are easily serializable and can be shared with other researchers, which eases the reproducibility of tagging experiments.

8.1.1 PoS Tagger Architectures

In order to better understand in which way FlexTag differs from existing PoS taggers, we distinguish four commonly used architectures, i.e. see Figure 8.1 for an overview.

Fixed Model Some taggers cannot be changed at all (without rewriting the tagger code itself) because model and features are hard-coded in the implementation. This is often the case for proof-of-concept implementations that might directly implement an optimized machine learning classifier or a domain-specific feature set. Figure 8.1a shows how taggers with a fixed model look from the user’s perspective. A fixed-model tagger is a black box that accepts raw text as input and outputs tagged text.

Replaceable Model The next step on the flexibility continuum are taggers with replaceable models as shown in Figure 8.1b. Here, the user can change the behavior of the tagger by choosing from a set of provided models, but the tagger itself provides no means for training a model. An example is the rule-based Hepple tagger (Hepple, 2000), where a rule set for English is provided. Rulesets for other languages can be specified, but there is no method offered by the tagger itself for training new models.

Trainable Model A major step towards really custom-made taggers is to let users train their own models as shown in Figure 8.1c. While the tagger is still a black box, it provides an additional interface to turn PoS annotated training data into a custom-made model. Once the model is trained, the tagger works exactly as in the *replaceable model* case.

Flexibly Trainable Model The hard-coded feature set of trainable taggers complicates the process of adapting a tagger to new domains. In our FlexTag architecture, which is shown in Figure 8.1d, we provide the user full control over the features that are extracted and which machine learning algorithm is used. A similar approach was taken by SVMTool (Giménez and Màrquez, 2004), which allowed the user to change the parametrization of features but not adding new ones or removing old ones. When a tagger uses a fixed feature set, the model is a persisted version of the weights learned during the training process. In the case of FlexTag, the model additionally includes the feature extractors that were used during model training. FlexTag loads during prediction the used feature extractors from the model to extract the same features as used during the training phase. This enables an easy experimenting with feature set

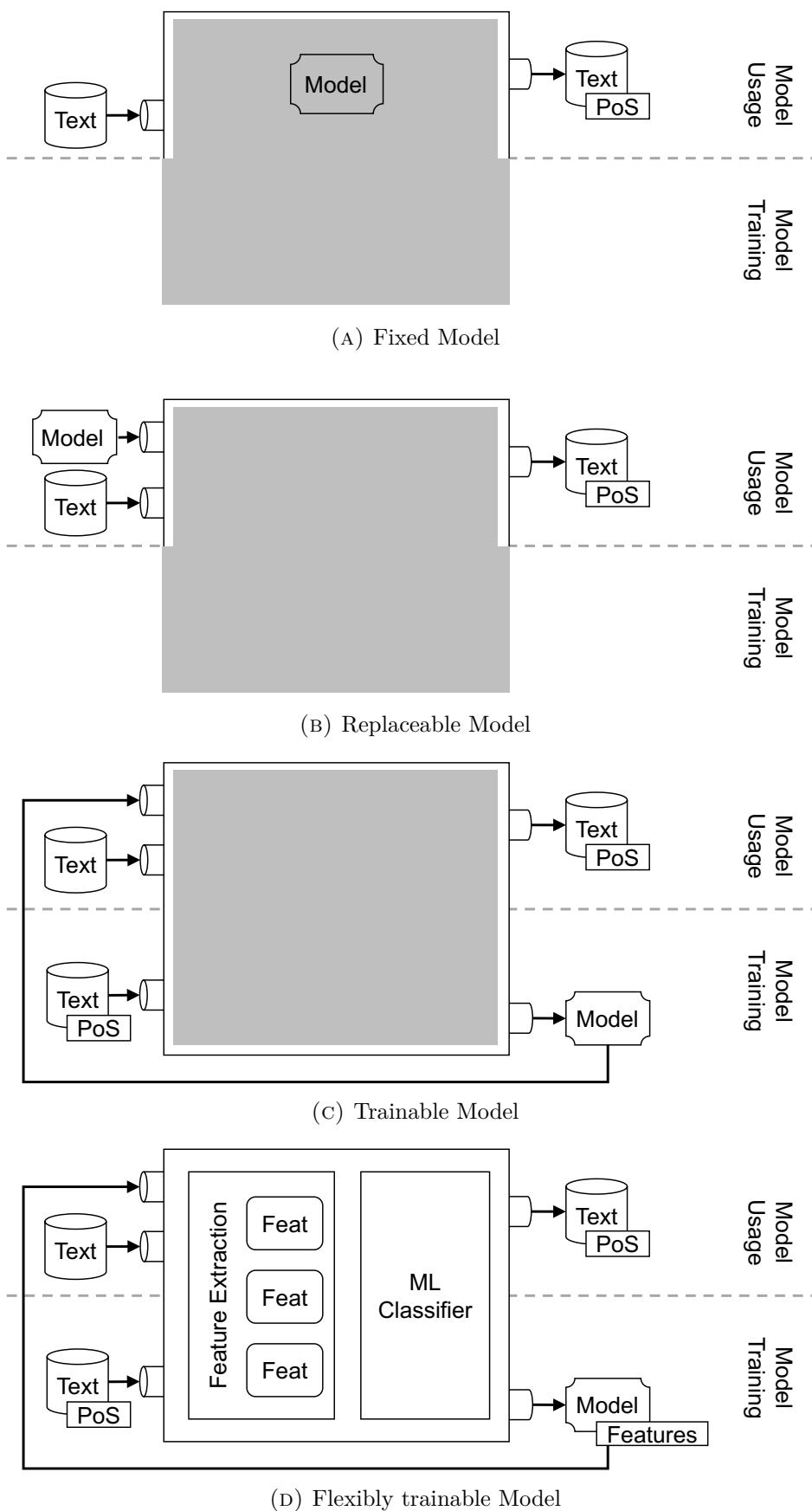


FIGURE 8.1: Flexibility levels of PoS taggers

configurations and sharing trained models with other researchers, as the used feature set is now a part of the model and not of the tagger implementation.

The main challenge that remains when exposing more parameters of a PoS tagger is how to keep the tagger usable, also for non-experts.

8.1.2 Trade-Off between Usability and Flexibility

A central question when exposing more functionality to the user is how to keep things useable. We now explain how we sustained usability while allowing a considerably higher degree of flexibility.

Using FlexTag As FlexTag is a Java/Maven-based implementation, it does not need to be manually installed and runs wherever a JVM is available. Everything is downloaded and setup upon first usage without any additional user intervention. Also, mapping of fine-grained tags to coarse-grained tags is provided automatically. In order to make this simplicity possible, FlexTag relies on DKPro Core (Eckart de Castilho and Gurevych, 2014) for usage and model loading, and DKPro TC (Daxenberger et al., 2014) for feature extraction and classification. FlexTag can be used standalone or as an Apache UIMA component (Ferrucci and Lally, 2004).

Training FlexTag Most other trainable taggers only support one input format and users are supposed to transform their data in this required format. In contrast, FlexTag makes no assumptions about the input format and relies on the UIMA reader concept supporting all readers that are compatible with the DKPro type system. For most common data formats (e.g. BNC, Brown, IMS-CWB, Negra, PTB, or TEI), many DKPro Core readers are already available that convert the corpus format in the correct format used by UIMA/FlexTag.

```
public class IsUserMention
    extends FeatureExtractorResource_ImplBase
    implements FeatureExtractor
{
    public static final String FEATURE_NAME = "isUserMention";

    public Set<Feature> extract(JCas aView, TextClassificationTarget aTarget)
        throws TextClassificationException
    {
        String text = aTarget.getCoveredText();
        boolean isUserMention = text.startsWith("@");
        return new Feature(FEATURE_NAME, isUserMention ? 1 : 0).asSet();
    }
}
```

FIGURE 8.2: Feature extractor that detects user mentions in Twitter

Adding Features FlexTag already comes with a wide range of implemented feature extraction modules that can be enabled when needed. However, as it is impossible

to foresee all future uses, one can easily implement own ones. For example, when processing Twitter data it might be useful to detect user mentions (like @DummyName) in order to reliably assign a specific tag (Ritter et al., 2011).

Technically, users need to provide a self-contained Java class that implements the FlexTag interface for feature extractors. In case of PoS tagging, this is a unit or sequence classification interface (see Daxenberger et al. (2014) for the different classification modes) where features are separately extracted for each token. The extractor interface exposes an in-memory representation of the whole text, i.e. the UIMA CAS (Götz and Suhre, 2004). This in-memory representations allow accessing all pre-processing results as stand-off-annotation. Figure 8.2 shows the full source code of a feature extractor that detects user mentions in tweets. In this case, we simply request the text of the current token and check whether it starts with an @ sign. However, more complicated actions such as accessing information about neighboring tokens are easily possible, too.

Additionally, a large number of existing feature extractors are pre-defined such as word or character ngrams which keeps the necessity to implement own feature extractors to a minimum.

Switching Classifiers As FlexTag relies on DKPro TC, we can easily switch between all the provided classifiers by changing the configuration without having to change any code. We will discuss the available classifiers in the following section where we introduce DKPro TC in greater detail.

8.1.3 Proof of Concept

Even if FlexTag is all about defining your own features, there is a set of standard features that usually work well and are thus activated by default. Default features include context features, the top 1000 most frequent 1-4 character ngrams, and boolean features testing if a word uses capitalized letters, hyphenation, periods, or is numeric. Users are however free to *not* use these features, i.e. the full feature space is fully customizable and if other features definition for the task at hand seem more suited, FlexTag does not prevent the user from building an own feature space. Training and testing a PoS tagger trained with this feature set reaches on the Wall-Street-Journal (Marcus et al., 1993) default data split (Collins, 2002) an accuracy of 96.6%. This is in range of the state-of-the-art accuracy that lies between 96.5% (Brants, 2000) and 97.6% (Choi, 2016).

8.2 Extending DKPro Text Classification

Experiments based on deep learning neural networks pose huge challenges to reproducibility. An experiment consists not just of the actual neural network architecture but also of a potentially large number of processing steps to prepare the data. Furthermore, countless network parameters exist which can greatly affect a networks'

performance. Reproduction attempts lead to a considerable amount of time spent with constructing comparable processing setups. Even if the deep learning code is released, code that applies all preprocessing to a dataset is often missing. Additional effort is often necessary to install and configure required third-party tools.

A potential solution to these reproducibility challenges is DKPro TC. DKPro TC ensures that the same pre-processing is automatically applied to any (new) dataset and provides convenience services such as an automatic installation of third-party tools. DKPro TC experiments are end-to-end shareable, enabling a quick and easy execution of experiments by other researchers. However, until now¹, DKPro TC only supported *shallow learning* frameworks. To offer the merits of DKPro TC also to deep learning researcher, we implemented an extension that also supports the deep learning paradigm, which is highly different to shallow learning. As proof-of-concept of this extension, we integrated the deep learning frameworks Keras (Chollet et al., 2015), DyNet (Neubig et al., 2017) and DeepLearning4J (DeepLearning4J, 2017). We start with a brief introduction of DKPro TC before we discuss the deep learning extension in detail.

8.2.1 DKPro Text Classification

DKPro TC is a Java based open-source software framework build upon the UIMA architecture. DKPro TC provides an intermediate software layer that harmonizes the use of various machine learning frameworks. The same experimental setup is easily executed with one or more classifiers, which enables a direct comparison of different implementations. The user defines feature extractors, which collect the information the classifier uses for training a model. DKPro TC transforms the extracted feature information into the data format required by the respective classifier. Hence, the user is completely shielded from the intrinsic data format details required by a certain implementation. Furthermore, DKPro TC allows running experiments as train-test or cross-validation setups and takes care of all data-splitting operations, execution, and aggregation of results. Required pre-processing components are automatically downloaded and installed on the users' computer. In summary, this allows sharing self-contained and executable DKPro TC experiments with other researchers. As of version 0.9.0, DKPro TC supports: Weka (Hall et al., 2009), LibLinear (Fan et al., 2008), LibSvm (Chang and Lin, 2011), SvmHmm (Joachims, 2008), and CrfSuite (Okazaki, 2007) that cover the common machine learning tasks in NLP, i.e. single outcome (e.g. sentiment analysis), multi-outcome (e.g. categorization), sequence classification (e.g. PoS tagging), and regression (e.g. text reading difficulty).

DKPro TC is designed around three design goals: (i) reproducibility, (ii) convenience, and (iii) applicability.

¹Version 0.9.0

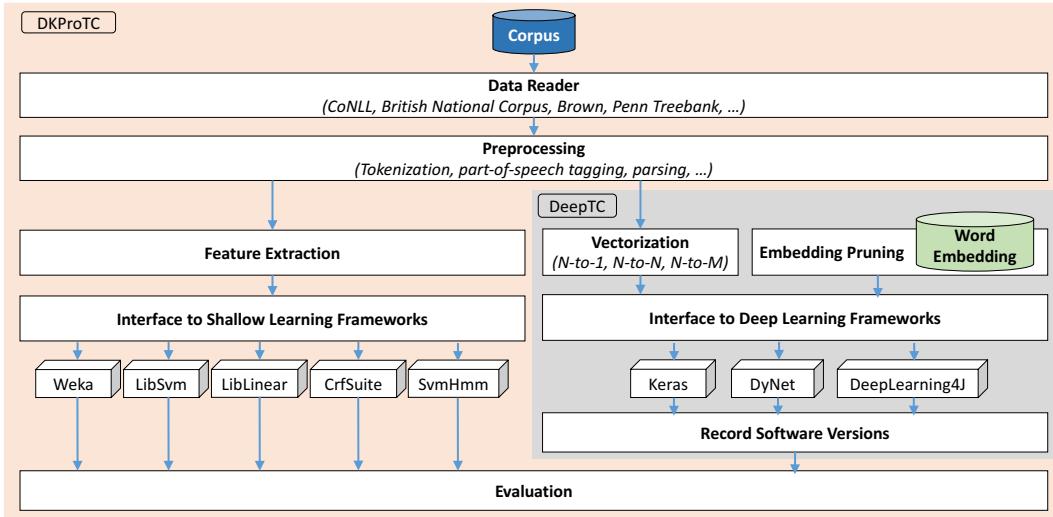


FIGURE 8.3: Processing schema for experiments in DKPro TC

Reproducibility We achieve reproducibility by using only software components that are released in public repositories such as Maven Central. This ensures that software remains available even if components are no longer maintained. Furthermore, all parametrization details of the experiment, for instance classifier parametrization, features, and configuration of pre-processing tools, are automatically stored in a DKPro TC project for sharing the project right away.

Convenience We achieved convenience by (i) easy-to-implement feature extractors with frequently needed ones being already pre-defined and (ii) automatic installation of third-party components from public repositories. Additionally, DKPro TC integrates DKPro Core (Eckart de Castilho and Gurevych, 2014) and thus provides a rich source of tools, such as tokenizers, part-of-speech taggers or lemmatizers, which can be added in a plug-and-play fashion as processing component. These tools are also automatically downloaded and installed as Maven artifacts. This provides a high degree of flexibility in terms of experimenting with various pre-processing tools and picking the best working one for a certain task. Of course, researchers can always implement their own UIMA processing components.

Applicability DKPro TC supports all common machine learning setups related to text classification tasks. This ensures a wide applicability of DKPro TC to different kinds of text classification tasks.

8.2.2 Shallow Architecture

Figure 8.3 shows a conceptual overview of DKPro TC and DeepTC. We focus our discussion for the moment on the DKPro TC part of the figure.

Reader The corpus data is read into DKPro TC by a reader component. DKPro Core support dozens of common NLP formats, for instance CoNLL, TEI or Penn Treebank (Marcus et al., 1993).

Preprocessing In this step, an optional pre-processing can be applied, which might entail tasks such as tokenization, part-of-speech tagging, syntactic parsing, etc.

Feature Extraction The feature extractors are applied to the data with access to information created during the pre-processing step. The extracted information is temporarily stored in an intermediate data format.

Interface to Shallow Learning Frameworks The feature information is transformed into the data format of the selected machine learning framework.

Evaluation If test data is provided, the trained model is applied to this dataset (after running through the same pre-processing and feature extraction as the train data). Many commonly used metrics such as accuracy, F-Score or Spearman correlation can be computed during evaluation. In case of cross-validation, aggregated results over all folds are automatically provided.

8.2.3 Deep Learning Extension

The conceptual differences between shallow and deep learning make an extension challenging, i.e. the shallow paradigm learns a model from a representation created from human defined features, while the deep paradigm learns a suited representation by itself. Furthermore, a meaningful extension must not just work on a technical level, but also sustain the advantages of taking workload from the user. Consequently, we conducted an analysis of common deep learning setups in the literature to learn about the challenges to reproducibility and convenience. This led to the *DeepTC* extension shown in Figure 8.3.

Format Many experiments assume a flat file format. The most common format is a whitespace or tabulator separation of text and labels. This format is quite popular and wide-spread as it allows a rather easy transformation of the textual data into an integer representation. Any corpus must, hence, first be transformed into this format before an experiment with a new dataset can be executed. In case of more complex data formats, such as XML, this leads to considerable additional effort. This challenge is solved by the many data format readers included in DKPro Core. The seamless integration of DKPro Core into DKPro TC allows it to access all corpus data and annotations (e.g. lemmas, part-of-speech tags, etc.) from DKPro TC enabling a quick and easy exchange of corpora and data formats. Of course, own readers for highly specific data formats can be easily written, too.

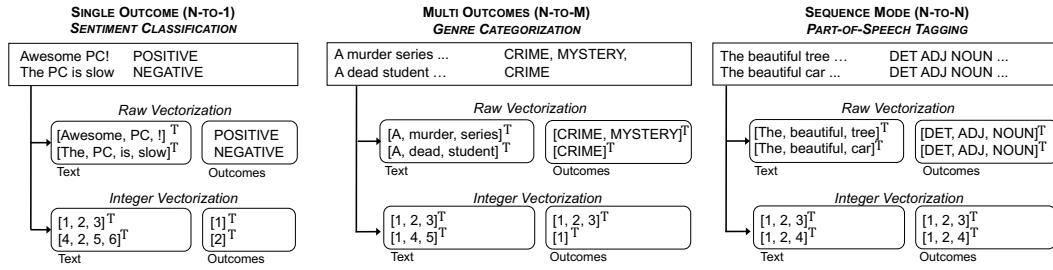


FIGURE 8.4: Vectorization N-to-1, N-to-M and N-to-N

Vectorization All textual information has to be transformed into a numerical vector representation before it can be provided to the deep learning framework. This vectorization entails mapping words and labels to integer values. When applying a prototype to unlabeled plain text, the integer values have to be mapped back to the original label to obtain human interpretable results. This is a mandatory task that can be easily automatized. While the general task of vectorization appears straightforward, its details depend on the kind of classification task of which we distinguish the three variants shown in Figure 8.4:

Single Outcome (N-to-1): For a text document with N tokens, an outcome have to be predicted. In classification the outcomes are labels, in case of regression they are numeric values. Use cases for single outcome classification are sentiment analysis or scoring the reading difficulty of a text.

Multi-Outcome (N-to-M): For a text document with N tokens, M outcomes have to be predicted. For instance, categorization of books into genres where a single book might have more than just one genre.

Sequence (N-to-N): For a text document with N tokens, an equal amount of N labels has to be predicted. The sequence in which the tokens occur is furthermore informative for predicting the labels. A prominent example is part-of-speech tagging.

The user is given control as to whether a vector is created with textual information (raw vectorization) or if the words have already been mapped to an integer representation (integer vectorization). Integer vectorization fits most setups and leads to further reduction of user-specific preprocessing code as the mapping process is done automatically by DeepTC. If the network architecture also considers sub-word information, e.g. character- or byte-level information, integer vectorization would be premature as the networks requires access to the actual word forms. For such cases, raw vectorization allows providing the actual words to the deep learning framework, as trade-off, the deep learning code has then to take care of mapping the raw data to an integer representation. This allows DeepTC to be flexible for more complex tasks, but still provide convenience features for common NLP setups.

Word Embeddings It is common to use pre-trained word embeddings, which are often quite large with negative effects on the start-up time of experiments. As a consequence, embeddings are usually pruned to contain only words that occur in the

vocabulary. Furthermore, in some tasks words without pre-trained embedding are either dropped or vectors are randomly initialized instead.

We provide a processing step in which the word embedding is pruned to contain only the occurring vocabulary. The user is given control as to whether words missing in the embeddings are removed or shall be initialized with a random vector. In case no word embedding is provided, this step performs no operation.

Interface to Deep Learning Framework The prepared data is provided to the deep learning framework. All necessary files are written to disk and the framework code is executed. The file locations are passed as parameters to the framework code. The framework code is expected to create a file at a specified location which contains the results of the execution. The break between programming environments, i.e. Java to Python to Java, are tackled by defining a protocol of data exchange. For each of the three defined classification tasks, i.e. single outcome, multi-outcome and sequence classification, a data format is expected in which the framework code provides the predicted outcomes. This allows bridging to deep learning frameworks based on non-Java technologies. As non-Java frameworks work internally with their own data structures, the framework code has to wrap the vectorized data into the respective data format. In case of Keras, for instance, which is based on Python, the vectorized data has to be transformed into the NumPy data type.

Record Software Versions An important challenge to reproducibility is keeping track of the software versions that are being used for running an experiment. Many deep learning frameworks are still under rapid development and, thus, change quickly with bugs being fixed and APIs updated. If code is released, it is often not reported which software version was used. For instance, for Keras, which depends on a backend such as TensorFlow, we record not just the Keras version but also the version of the backend and the NumPy library as primary data structure. The software versions that are recorded are highly dependent on the respective deep learning framework. This provides a basic software versioning record, which can be released with the experimental code.

8.2.4 Limitations

The rapid development of deep learning software creates practical limitations to reproducibility and convenience.

The convenience of automatically installing needed components is easily provided for Java/Maven-based software. For non-Java frameworks, this is not as easily possible and the task of installing software is delegated to the user. We would require a method to serialize the deep learning framework environment into a container that would allow deployment on a third-party computer, i.e. in the case of Keras. This would also entail the respective backend and their dependencies. To the best of our knowledge, such a method is not available with support for all major operating

systems. A further challenge is to track the names and versions of all involved components the user has to install to reproduce an environment. This would lead to an extremely long list with some software components being more important than others to the reproducibility of results. As a compromise, we record the software components of primary importance for the respective framework. A further limitation occurs if researchers work on unstable software versions. It is not uncommon that researchers compile their deep learning software from the latest version in a source-code repository to make certain features available, i.e. a bleeding edge version. One would have to record the exact hash-id of the source-code repository from which the software was built to enable reproducibility, i.e. detecting such setups is beyond an automatic detection by DeepTC.

Nonetheless, we provide with DeepTC a substantial improvement to reproducibility despite of lacking functionalities to automatically create software environments on the users' computer.

8.3 Chapter Conclusion

In this chapter, we discussed the two major software projects that we have been developing in the course of this thesis. One is FlexTag, a highly flexible part-of-speech tagger and the second one is Deep TC, for improving the reproducibility of deep learning experiments. FlexTag allows changing the feature set, which grants an advantage over the many available off-the-shelf taggers that do not permit any changes to the feature set. This flexible tool is the prerequisite for the many tagging experiments conducted in this thesis. The second project is DKPro TC, which is the underlying framework used by FlexTag. DKPro TC is an environment for text classification experiments that aims on avoiding repetitive work and eases experiment reproducibility. We discussed a deep learning extension for DKPro TC to improve the reproducibility of deep learning experiments by embedding them into an end-to-end shareable environment, i.e. as DKPro TC experiments. As underlying framework to FlexTag, this extension will also enable the use of deep learning frameworks in FlexTag, in future releases.

Chapter 9

Conclusion

In this chapter, we summarize the content of this thesis and give an overview of our main contributions. Furthermore, we discuss limitations and provide pointers for future work.

9.1 Summary

In this thesis, we focused on the robustness of Part-of-Speech tagging from the viewpoint of domain, language and long tail robustness.

Domain Robustness Our first objective was on domain robustness which we investigated in Chapter 3, Chapter 4 and Chapter 5. The central questions focused on determining which is the best and most robust tagger model and on how to improve tagging if tagging performance is inadequate on a certain text domain.

In Chapter 3, we started with an analysis of informal text by the example of social media, and continued with an empirical evaluation of PoS taggers to learn about the currently provided robustness. We conducted this evaluation for English and German taggers and their available models by evaluating them on three inherently different text domains – formal text, transcripts of spoken discourse, and social media text. We find that the taggers are well equipped to deal with formal text but often perform poorly on less formal text domains. In particular, text containing colloquial, informal utterances that are also often found in oral discourse poses a huge challenge to taggers. A central problem is the availability of training data. Most training data is formally written text such as newswire. Furthermore, sufficient training data of foreign text domains is not available, which prevents training new models that are more suited for a certain text domain.

In Chapter 4, we investigated currently available methods to improve tagging robustness, which showed that two main strategies exist. First, providing more training data for the taggers (without annotating new data), and the second is injecting external knowledge into the model training process. We find that injecting external knowledge works best on social media text as it provides knowledge about common spelling variations of canonical word forms. While these methods substantially improve tagging on informal text, the overall performance remains dissatisfactory.

In Chapter 5, we introduced a new cross-domain tagging approach with an increased robustness called two-step tagging. In two-step tagging, we first apply a highly robust coarse-grained tagger followed by the second tagging step in which the predicted coarse-grained tag is refined to the fine-grained tag. An evaluation on four text domains showed that this approach performs better than the baselines systems and has a huge latent potential. The question that remains is how to tap this potential; we will discuss starting points for improving two-step tagging further below.

Language Robustness In Chapter 6, we investigated the language robustness of taggers. The central research question here is whether or not a construction of a language-independent tagger is possible that is competitive to a language-fitted tagger. We compare taggers based on Hidden Markov Models (HMM), Conditional Random Fields (CRF) and Long Short Term Memory (LSTM) neural networks on 21 languages for their language robustness. We find that the differences between taggers and languages remain small as long as the tagset size is small. On large tagset sizes, we find that the LSTM tagger performs considerably better than the HMM and CRF tagger. A reproduction of state-of-the-art results for selected languages showed that the LSTM tagger is indeed suited as a language-independet PoS tagger. When learning models on morphologically rich languages, we did find that language-fitted taggers that use additional human-crafted resources might reach better results. Thus, language robustness is a considerably less severe challenge than domain robustness.

Long Tail Robustness In Chapter 7, we investigated long tail robustness of taggers. This work is centered around the research question on how to train taggers in order to recognize phenomena that occur infrequently in the training corpus. We proposed an inexpensive method to produce additional training data for such under-represented phenomena. This method uses an off-the-shelf tagger to automatically tag a sentence, which is followed by a manual correction of a single word, i.e. the phenomenon of interest. This procedure is easily applied and avoids a manually annotation of the entire sequence. We compared the quality of the produced data to altering the frequency weight of a phenomenon by over- and undersampling the data in the training corpus. We find that annotation of additional data is unavoidable but that the proposed data production method is sufficient to reach substantial improvements in tagging such phenomena.

Enabling Technologies In Chapter 8, we discussed our two technical contributions that were developed in the course of this thesis. These tools provide valuable functionalities that are not found in other tools. The tagger FlexTag provides a high flexibility that researchers require to experiment with various feature configurations. DeepTC is a contribution towards reproducibility of deep learning experiments by providing an end-to-end shareable environment for such experiments.

9.2 Limitations and Outlook

The strong bias of available taggers to mostly support formal text domains prevents these taggers from being directly applicable to other text domains. The domain adaptation approaches that we discussed reached substantial improvements but still stay behind the extremely high accuracies reported on formal newswire corpora.

More Knowledge We found that providing knowledge from resources such as distributional word clusters is a valuable resource of information to improve tagging. These resources are often created from large amounts of plain text Twitter messages. The reason for choosing Twitter lies for one in the heterogeneity of the text found in this domain but also in the ease of accessing plain text from this domain. We expect to see further improvements if such resources are created from text that is similar to the text that shall be tagged. Thus, for tagging chat messages such clusters should be ideally created from a chat domain corpus. Here, the challenge lies in getting access to a sufficiently large amount of plain text of the respective text domain.

Two-Step Tagging On average, two-step tagging showed an improved performance over the baseline taggers. This higher cross-domain robustness makes it reasonable to pursue this approach further. Ideally, a PoS tagger can be used as a black-box that works reasonably well on all domains. Our analysis showed that essentially every tagger fails in some domain due to larger differences between training and application text domain. In such cases, users who rely on the black-box assumption that a tagger will perform equally well on all text domains, will face many situations in which tagging performs poorly. Two-step tagging is certainly a promising step towards a single tagger that can deal in an acceptable fashion with many text domains. In the future, this will hopefully liberate the user from worrying about training and application domains of the tagger model.

An opportunity to improve two-step tagging lies in harmonizing the sub-corpora used to train the coarse-grained model in the first step. Each annotated corpus follows its own annotation schema that often also entails differences with respect to tokenization decisions. These differences lead to sub-corpora with contradicting information, which in turn lead to additional tagging errors. A harmonization of the PoS annotation and also a unification of the corpora tokenization should lead to further improvements. While a harmonization by a human annotator would certainly lead to the best result, less labor-intensive strategies such as the application of regular expressions to equalize systematical differences would be an inexpensive strategy to start with.

In this thesis, we trained a single coarse-grained model that combines text from two domains. The limitation to two text domains was mostly motivated by the circumstance that other candidate corpora have been used in our evaluation set. It would be worthwhile to explore how training of several domain-fitted coarse-grained models perform. By measuring the out-of-vocabulary rate from the input sentence

to the training data of the respective coarse-grained model, the most suited model could be determined. This might allow tapping the latent potential of this approach, which mostly depends on the coarse-grained tagging.

Another approach lies in using multi-task learning (Caruana, 1997). Multi-task learning solves a task by learning from the source data of two or more problems, which appear to be not directly related to the problem one tries to solve. The assumption is that the machine learning process can learn to avoid certain errors from additionally learning from these indirectly related data. Multi-task learning can be applied in two different ways: First, instead of training the coarse-grained model only on coarse-grained tags, the tagger could be simultaneously trained on coarse-grained and fine-grained tags. This would allow for the learning of coarse-grained tags by taking advantage of distinctions only learnable from a more fine-grained level of PoS annotation. This would be a possibility to improve the coarse-grained tagging accuracy. Second, one trains a single model that directly predicts fine-grained tags, while during model training, in addition to the correctness of the fine-grained predictions, one considers also the correctness of the coarse-mapped predictions. The coarse-grained tagging would thus provide some additional constraints leading to a more accurate fine-grained tagging. This would replace the two-step tagging architecture by a single model. We discussed that in particular adding of social media data to the coarse-grained model improved tagging accuracy considerably, which was only possible due to the harmonization of the tagsets to coarse-grained tags. Both ways require that all training corpora use the same fine-grained tagset, which means losing the advantage of combining the data of several domains.

9.3 Closing Remarks

With the work in this thesis, we have demonstrated that many of the extremely good results reported for PoS tagging cannot be reached under realistic conditions. Tagging quality rarely reaches the quality level known from newswire text or other highly formal text domains. We provided with two-step tagging an approach which has the potential of becoming a multi-domain tagger that can sustain a high accuracy even under text domain shifts.

Appendix A

Language Robustness - Results per Corpus

We provide here detailed results for each corpus in the experiments in Chapter 6 that we showed in aggregated form per language group.

Lang. Group	Corpus Id	Word Top 750						CRF		HMM	
		Ngrams ±1		Char	Ngrams	Clusters		Best All	CRF OOV	All	OOV
		All	OOV	All	OOV	All	OOV	All	OOV	All	OOV
Germanic	Danish	90.9	53.3	90.3	69.3	89.5	67.6	96.1	82.4	94.9	74.2
	Dutch	86.5	66.9	85.0	71.7	88.0	77.7	90.7	83.7	89.9	80.6
	English	87.5	45.1	90.3	70.1	89.1	64.0	94.6	80.2	93.8	77.7
	German-1	88.5	62.4	90.3	77.7	90.8	73.7	94.6	84.6	94.4	83.7
	German-2	87.2	60.3	90.9	77.7	90.8	76.1	95.2	87.1	94.9	85.4
	German-3	86.3	58.5	91.7	76.8	91.6	77.6	94.4	85.0	94.4	83.9
	Icelandic	67.5	14.2	76.5	45.1	68.3	28.9	80.9	53.6	79.8	51.9
	Norwegian	92.4	77.1	91.6	80.6	92.8	82.7	96.1	89.7	95.5	86.5
	Swedish-1	91.1	70.6	92.9	82.2	92.3	79.9	96.3	90.3	95.6	85.9
	Swedish-2	78.7	29.7	87.2	67.3	81.4	48.8	91.0	74.6	91.4	77.6
Romanic	B-Portug.	86.9	62.8	87.8	73.6	89.7	76.0	92.8	83.8	93.3	84.2
	French-1	81.9	40.1	85.9	66.5	81.6	58.2	89.2	75.7	88.2	71.8
	French-2	95.4	67.3	93.8	74.5	91.9	79.3	97.7	88.2	97.4	82.4
	Italian	93.3	68.6	91.6	74.8	91.7	75.5	96.4	86.5	95.8	80.8
	Spanish	88.5	45.5	94.5	78.2	88.1	58.8	96.4	83.5	96.6	83.6
Slavic	Croatian-1	69.0	18.6	80.6	56.3	75.2	47.2	84.9	65.4	84.7	66.7
	Croatian-2	66.3	15.9	78.5	54.4	73.5	44.8	83.4	63.9	82.6	63.9
	Czech	64.1	14.4	79.2	56.0	75.2	39.2	83.1	62.9	81.7	60.9
	Polish	82.9	58.1	92.5	86.9	86.5	72.5	95.5	91.5	93.6	85.4
	Russian	83.7	53.7	93.0	83.5	88.2	70.9	95.5	87.5	94.6	83.6
	Slovak	67.7	14.9	80.5	57.8	65.6	31.9	83.5	63.8	82.9	61.6
	Slovene-1	72.6	17.4	83.5	55.6	72.4	39.4	86.4	62.5	82.6	59.6
	Slovene-2	65.4	12.1	78.2	50.5	73.0	39.0	83.0	59.4	86.2	59.5
Other	Afrikaans	95.7	75.0	95.3	80.3	95.8	81.9	97.8	89.6	97.3	85.5
	Finnish	62.6	10.0	77.1	48.5	67.8	33.8	82.3	56.7	81.3	55.8
	Hebrew	82.3	41.7	81.3	60.9	76.3	53.3	90.5	68.5	90.3	60.1
	Hungarian	72.7	13.9	86.7	63.3	72.0	31.7	89.9	69.6	89.4	69.5

TABLE A.1: Accuracy of CRF tagger configurations on multi-lingual corpora (10fold CV). Best results per language are highlighted in bold-face. We show additionally OOV performance and the results with a HMM tagger as baseline

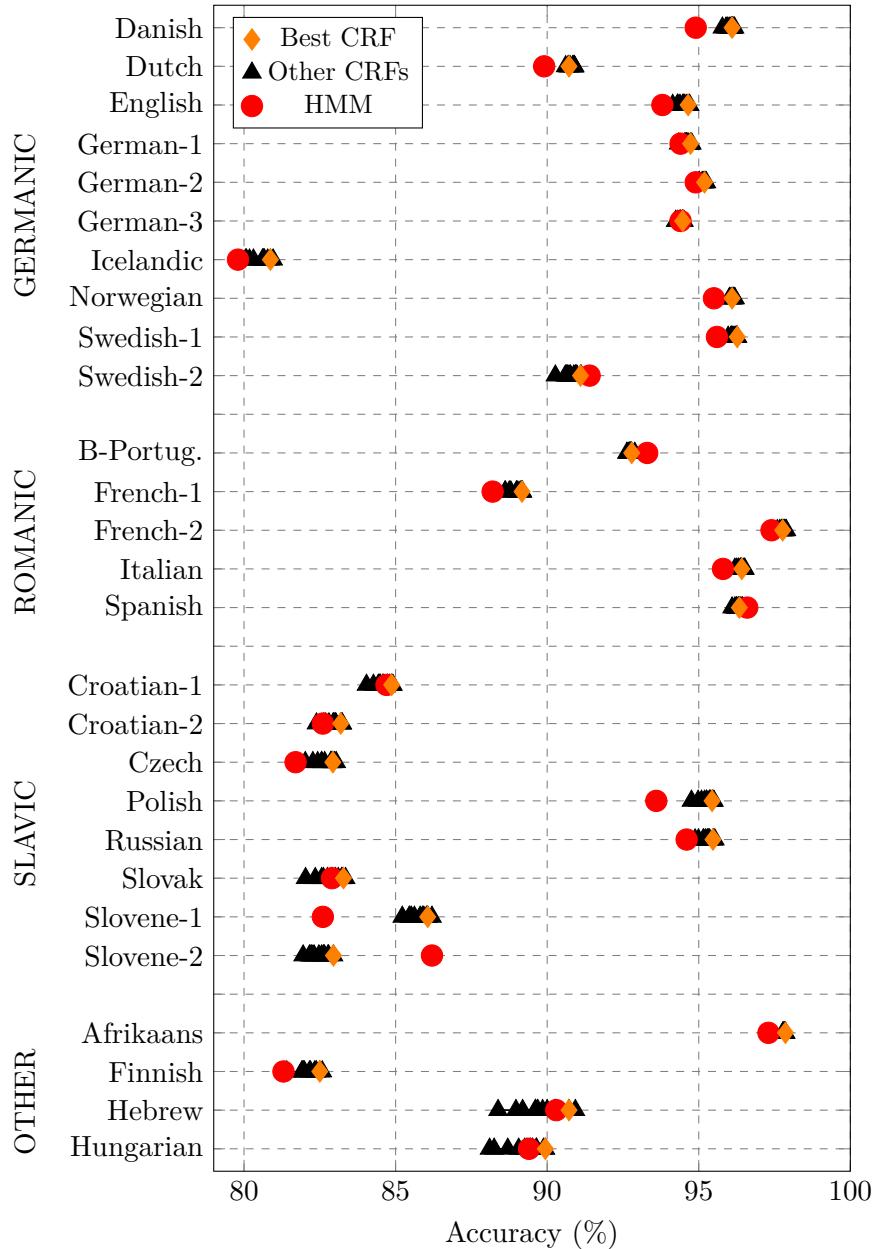


FIGURE A.1: Results of the various parameterizations of the CRF tagger feature set on multi-lingual corpora (10fold CV). Triangles mark the result of a particular configuration, the diamond symbol is the overall best working configuration. HMM results are additionally shown as baseline

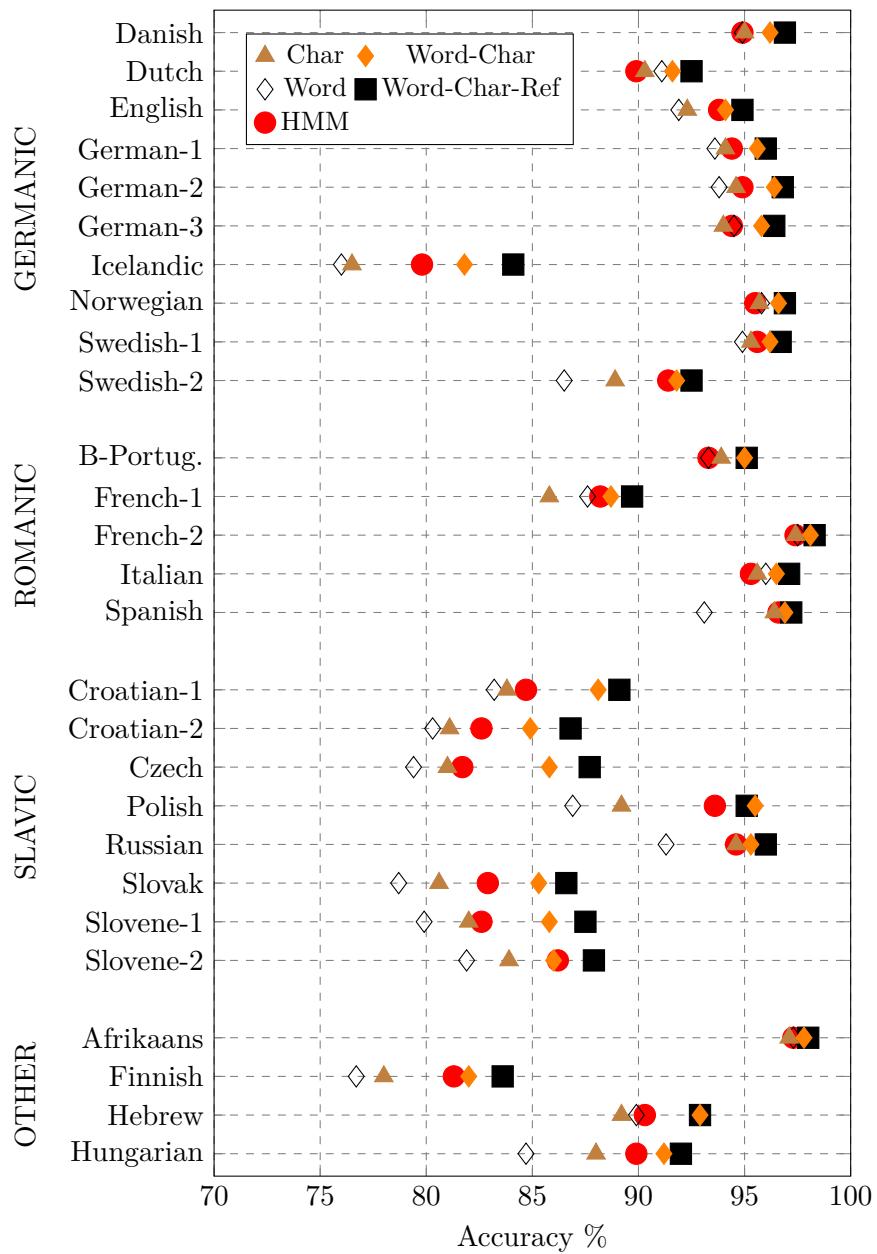


FIGURE A.2: Results of the LSTM architectures on multilingual corpora (10fold CV)

Lang. Group	Corpus Id	Word		Char		Word-Char		Reference		HMM	
		All	OOV	All	OOV	All	OOV	All	OOV	All	OOV
Germanic	Danish	94.9	72.7	95.0	79.1	96.4	82.5	96.9	83.4	94.9	74.2
	Dutch	91.1	82.3	90.3	83.6	91.6	85.7	92.5	87.1	89.9	80.6
	English	91.9	65.9	92.3	77.4	94.1	79.6	94.9	80.9	93.8	77.7
	German-1	93.6	78.3	94.1	84.5	95.6	87.6	96.0	88.3	94.4	83.7
	German-2	94.5	82.4	94.6	87.1	96.4	90.1	96.8	91.5	94.4	85.4
	German-3	93.8	80.3	94.0	84.9	95.8	88.6	96.4	89.8	94.4	83.9
	Icelandic	76.0	34.8	76.5	49.3	81.8	56.2	84.1	60.6	79.8	51.9
	Norwegian	95.8	86.2	95.7	88.2	96.6	90.3	96.9	90.3	95.5	86.5
	Swedish-1	94.9	81.4	95.3	86.7	96.2	89.0	96.7	89.8	95.6	85.9
	Swedish-2	86.5	54.3	88.9	74.3	91.8	78.5	92.5	80.4	91.4	77.6
Romanic	B-Portug.	93.3	82.4	93.9	87.4	95.0	90.3	95.1	90.8	93.3	84.2
	French-1	87.6	67.0	85.8	72.0	88.7	77.4	89.7	78.7	88.2	71.8
	French-2	97.5	80.4	97.4	83.4	98.1	87.7	98.3	88.7	97.4	82.4
	Italian	96.0	81.3	95.6	84.2	96.5	85.9	97.1	86.9	95.8	80.8
	Spanish	93.1	63.3	96.4	85.5	96.9	86.1	97.2	87.0	96.6	83.6
Slavic	Croatian-1	83.2	55.5	83.8	67.5	88.1	72.8	89.1	75.2	84.7	66.9
	Croatian-2	80.3	52.4	81.1	63.8	84.9	69.1	86.8	72.4	82.6	63.9
	Czech	79.4	49.1	81.0	62.7	85.8	68.7	87.7	72.4	81.7	60.9
	Polish	86.9	73.6	89.2	84.7	95.5	91.2	95.1	91.0	93.6	85.4
	Russian	91.3	73.2	94.6	85.8	95.3	86.9	96.0	88.4	94.6	83.6
	Slovak	78.7	44.9	80.6	65.0	85.3	69.7	86.6	71.4	82.9	61.6
	Slovene-1	81.9	44.5	83.9	61.1	86.0	62.6	87.9	65.7	82.6	59.6
	Slovene-2	79.9	47.9	82.0	63.4	85.8	67.4	87.5	70.1	86.2	59.5
Other	Afrikaans	97.3	82.8	97.1	85.8	97.8	88.4	98.0	90.0	97.3	85.5
	Finnish	76.7	42.7	78.0	57.6	82.0	58.9	83.6	61.2	81.3	55.8
	Hebrew	89.9	60.2	89.2	66.9	92.2	69.7	92.9	72.1	90.3	60.1
	Hungarian	84.7	53.3	88.0	73.1	91.2	76.9	92.0	79.0	89.4	69.5

TABLE A.2: Accuracy of LSTM taggers on multi-lingual corpora (10fold CV). Best results for a language are highlighted in bold-face

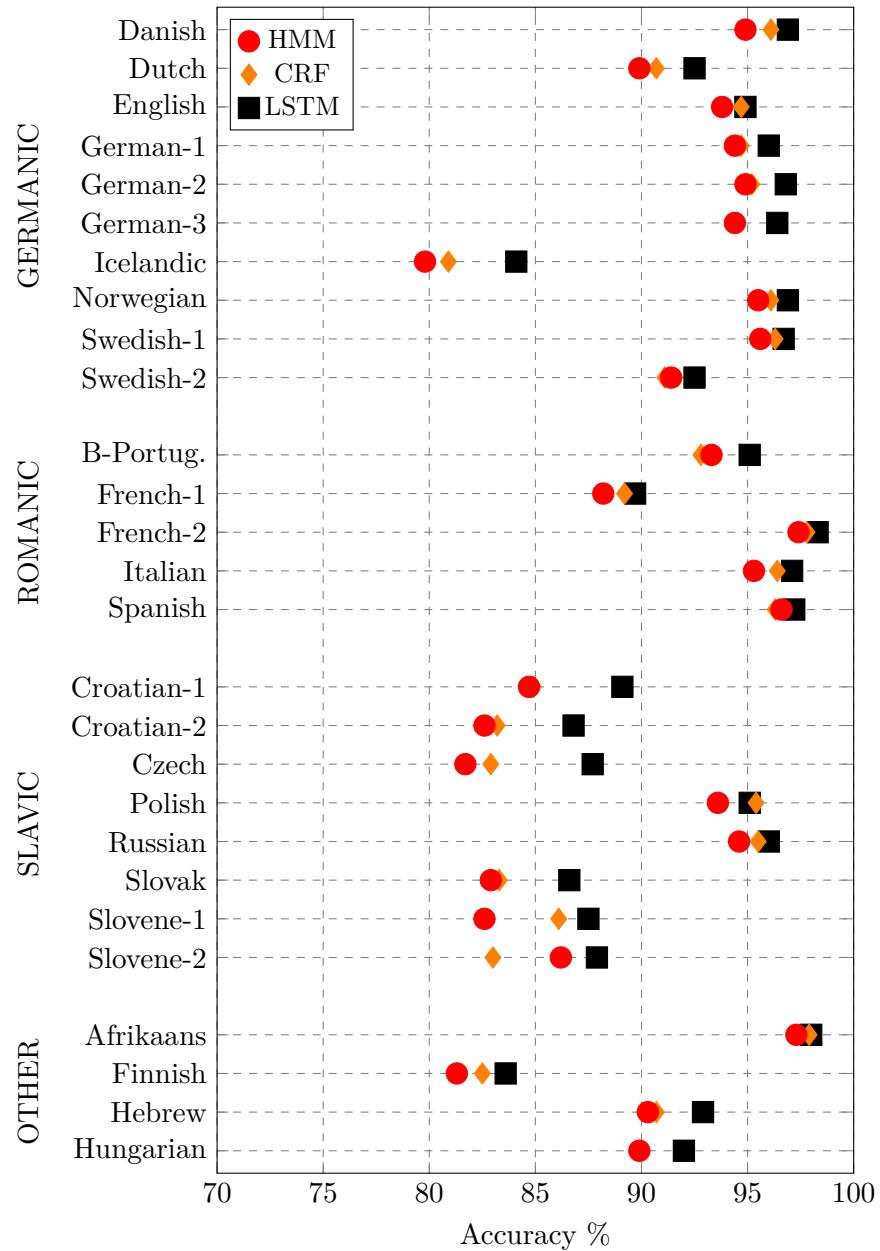


FIGURE A.3: Results of a HMM, CRF and LSTM tagger on multilingual corpora
(10fold CV)

Bibliography

- [Agarwal et al. 2011] AGARWAL, Apoorv ; XIE, Boyi ; VOVSHA, Ilia ; RAMBOW, Owen ; PASSONNEAU, Rebecca: Sentiment Analysis of Twitter Data. In: *Proceedings of the Workshop on Languages in Social Media*. Portland, Oregon : Association for Computational Linguistics, 2011, pages 30–38
- [Željko Agić and Ljubešić 2014] AGIĆ Željko ; LJUBEŠIĆ, Nikola: The SETimes.HR Linguistically Annotated Corpus of Croatian. In: *Proceedings of the International Conference on Language Resources and Evaluation (LREC)*. Reykjavik, Iceland : ELRA, 2014, pages 1724–1727
- [Al-Rfou et al. 2013] AL-RFOU, Rami ; PEROZZI, Bryan ; SKIENA, Steven: Polyglot: Distributed Word Representations for Multilingual NLP. In: *Proceedings of the Seventeenth Conference on Computational Natural Language Learning*. Sofia, Bulgaria : Association for Computational Linguistics, 2013, pages 183–192
- [Aluísio et al. 2003] In: ALUÍSIO, Sandra ; PELIZZONI, Jorge ; MARCHI, Ana R. ; OLIVEIRA, Lucélia de ; MANENTI, Regiana ; MARQUIAFÁVEL, Vanessa: *An Account of the Challenge of Tagging a Reference Corpus for Brazilian Portuguese*. Faro, Portugal, 2003, pages 110–117
- [Attardi et al. 2010] ATTARDI, Giuseppe ; ROSSI, Stefano D. ; PIETRO, Giulia D. ; LENCI, Alessandro ; MONTEMAGNI, Simonetta ; SIMI, Maria: A Resource and Tool for Super-sense Tagging of Italian Texts. In: *Proceedings of the International Conference on Language Resources and Evaluation (LREC)*. Valletta, Malta : ELRA, 2010
- [Augustinus et al. 2016] AUGUSTINUS, Liesbeth ; DIRIX, Peter ; NIEKERK, Daniel V. ; SCHUURMAN, Ineke ; VANDEGHINSTE, Vincent ; EYNDE, Frank V. ; HUYSTEEN, Gerhard V.: AfriBooms: An Online Treebank for Afrikaans. In: *Proceedings of the International Conference on Language Resources and Evaluation (LREC)*. Portorož, Slovenia : ELRA, 2016, pages 677–682
- [Aw et al. 2006] AW, AiTi ; ZHANG, Min ; XIAO, Juan ; SU, Jian: A Phrase-based Statistical Model for SMS Text Normalization. In: *Proceedings of the International Conference on Computational Linguistics (COLING)*. Sydney, Australia : Association for Computational Linguistics, 2006, pages 33–40

- [Baldwin et al. 2013] BALDWIN, Timothy ; COOK, Paul ; LUI, Marco ; MACKINLAY, Andrew ; WANG, Li: How noisy social media text, how diffrent social media sources. In: *International Joint Conference on Natural Language Processing*. Nagoya, Japan, 2013, pages 356–364
- [Baroni et al. 2009] BARONI, Marco ; BERNARDINI, Silvia ; FERRARESI, Adriano ; ZANCHETTA, Eros: The WaCky wide web: a collection of very large linguistically processed web-crawled corpora. In: *Language Resources and Evaluation* 43 (2009), Nr. 3, pages 209–226
- [Bartz et al. 2013] BARTZ, Thomas ; BEISSWENGER, Michael ; STORRER, Angelika: Optimierung des Stuttgart-Tübingen-Tagset für die linguistische Annotation von Korpora zur internetbasierten Kommunikation: Phänomene, Herausforderungen, Erweiterungsvorschläge. In: *Journal for Language Technology and Computational Linguistics (JLCL)* 28 (2013), pages 157–198
- [Behera et al. 2015] BEHERA, Atul Kr. Ojha P. ; SINGH, Srishti ; JHA, Girish N.: Training & Evaluation of POS Taggers in Indo-Aryan Languages: A Case of Hindi, Odia and Bhojpuri. In: *Language & Technology Conference: Human Language Technologies as a Challenge for Computer Science and Linguistics*. Poznan, Poland, 2015, pages 524–529
- [Beißwenger 2013] BEISSWENGER, Michael: Das Dortmunder Chat-Korpus. In: *Zeitschrift für germanistische Linguistik* 41 Bd. 1, 2013, pages 161–164
- [Beißwenger et al. 2016] BEISSWENGER, Michael ; BARTSCH, Sabine ; EVERT, Stefan ; WÜRZNER, Kay-Michael: EmpiriST 2015: A Shared Task on the Automatic Linguistic Annotation of Computer-Mediated Communication and Web Corpora. In: *Proceedings of the 10th Web as Corpus Workshop (WAC-X) and the EmpiriST Shared Task*. Berlin, Germany : Association for Computational Linguistics, 2016, pages 44–56
- [Beißwenger et al. 2015] BEISSWENGER, Michael ; BARTZ, Thomas ; STORRER, Angelika ; WESTPFAHL, Swantje: Tagset und Richtlinie für das Part-of-Speech-Tagging von Sprachdaten aus Genres internetbasierter Kommunikation / Tagset and guidelines for the PoS tagging of langauge data from genres of computer-mediated communication. In: *EmpiriST guideline document (German and English version)*, 2015
- [Bejček et al. 2013] BEJČEK, Eduard ; HAJIČOVÁ, Eva ; HAJIČ, Jan ; JÍNOVÁ, Pavlína ; KETTNEROVÁ, Václava ; KOLÁŘOVÁ, Veronika ; MIKULOVÁ, Marie ; MÍROVSKÝ, Jiří ; NEDOLUZHKO, Anna ; PANEVOVÁ, Jarmila ; POLÁKOVÁ, Lucie ; ŠEVČÍKOVÁ, Magda ; ŠTĚPÁNEK, Jan ; ZIKÁNOVÁ, Šárka: Prague Dependency Treebank 3.0. (2013)
- [Bengio et al. 2006] In: BENGIO, Yoshua ; SCHWENK, Holger ; SENÉCAL, Jean-Sébastien ; MORIN, Frédéric ; GAUVAIN, Jean-Luc: *Neural Probabilistic Language*

- Models*. Berlin, Heidelberg : Springer Berlin Heidelberg, 2006. – ISBN 978–3–540–33486–6, pages 137–186
- [Berg-Kirkpatrick et al. 2012] BERG-KIRKPATRICK, Taylor ; BURKETT, David ; KLEIN, Dan: An Empirical Investigation of Statistical Significance in NLP. In: *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Stroudsburg, PA, USA : Association for Computational Linguistics, 2012, pages 995–1005
- [Biemann 2006] BIEMANN, Chris: Unsupervised Part-of-Speech Tagging Employing Efficient Graph Clustering. In: *Proceedings of the COLING/ACL 2006 Student Research Workshop*. Sydney, Australia : Association for Computational Linguistics, 2006, pages 7–12
- [Bishop 2011] BISHOP, Christopher M.: *Pattern Recognition and Machine Learning*. Springer, 2011. – ISBN 978–0387310732
- [Björkelund et al. 2010] BJÖRKELUND, Anders ; BOHNET, Bernd ; HAFDELL, Love ; NUGUES, Pierre: A High-performance Syntactic and Semantic Dependency Parser. In: *Proceedings of the International Conference on Computational Linguistics (COLING): Demonstration Volume*. Beijing, China : Association for Computational Linguistics, 2010, pages 33–36
- [Blei et al. 2003] BLEI, David M. ; NG, Andrew Y. ; JORDAN, Michael I.: Latent Dirichlet Allocation. In: *The Journal of Machine Learning Research* 3 (2003), pages 993–1022
- [Blitzer et al. 2011] BLITZER, John ; KAKADE, Sham ; FOSTER, Dean: Domain Adaptation with Coupled Subspaces. In: *Proceedings of the International Conference on Artificial Intelligence and Statistics* Bd. 15. Fort Lauderdale, FL, USA, 2011 (Proceedings of Machine Learning Research), pages 173–181
- [Bocharov et al. 2013] BOCHAROV, V. V. ; ALEXEEVA, S.V. ; GRANOVSKY, D.V. ; PROTOPOPOVA, E.V. ; STEPANOVA, M.E. ; SURIKOV, A.V.: *Crowdsourcing morphological annotation*. <http://opencorpora.org>, last accessed 15 January 2018, 2013
- [Bojanowski et al. 2017] BOJANOWSKI, Piotr ; GRAVE, Edouard ; JOULIN, Armand ; MIKOLOV, Tomas: Enriching Word Vectors with Subword Information. In: *Transactions of the Association for Computational Linguistics* 5 (2017), 135–146. <https://transacl.org/ojs/index.php/tacl/article/view/999>
- [Bosco et al. 2012] BOSCO, Cristina ; SANGUINETTI, Manuela ; LESMO, Leonardo: The Parallel-TUT: a multilingual and multiformat treebank. In: *Proceedings of the International Conference on Language Resources and Evaluation (LREC)*. Istanbul, Turkey : ELRA, 2012

- [Bosco et al. 2016] BOSCO, Cristina ; TAMBURINI, Fabio ; BOLIOLI, Andrea ; MAZZEI, Alessandro: Overview of the EVALITA 2016 Part Of Speech on TWitter for ITALian Task. (2016), pages 78–84
- [Bouma et al. 2001] BOUMA, Gosse ; NOORD, Gertjan van ; MALOUF, Robert: Alpino: Wide Coverage Computational Analysis of Dutch. In: *Computational Linguistics in the Netherlands 2000* 37 (2001), pages 45–59
- [Brants et al. 2004] BRANTS, Sabine ; DIPPER, Stefanie ; EISENBERG, Peter ; HANSEN, Silvia ; KÖNIG, Esther ; LEZIUS, Wolfgang ; ROHRER, Christian ; SMITH, George ; USZKOREIT, Hans: TIGER: Linguistic Interpretation of a German Corpus, Journal of Language and Computation, 2004, pages 597–620
- [Brants et al. 2002] BRANTS, Sabine ; DIPPER, Stefanie ; HANSEN, Silvia ; LEZIUS, Wolfgang ; SMITH, George: The TIGER Treebank. In: *Proceedings of the Workshop on Treebanks and Linguistic Theories*, 2002
- [Brants 2000] BRANTS, Thorsten: TnT: A Statistical Part-of-speech Tagger. In: *Proceedings of the Conference on Applied Natural Language Processing*. Seattle, Washington : Association for Computational Linguistics, 2000, pages 224–231
- [Brill 1992] BRILL, Eric: A Simple Rule-based Part of Speech Tagger. In: *Proceedings of the Third Conference on Applied Natural Language Processing*. Trento, Italy : Association for Computational Linguistics, 1992, pages 152–155
- [Brody and Diakopoulos 2011] BRODY, Samuel ; DIAKOPOULOS, Nicholas: Coooooooooooooolllllllllllll!!!!!! Using Word Lengthening to Detect Sentiment in Microblogs. In: *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Association for Computational Linguistics, 2011, pages 562 – 570
- [Brown et al. 1992] BROWN, Peter F. ; DESOUZA, Peter V. ; MERCER, Robert L. ; PIETRA, Vincent J D. ; LAI, Jenifer C.: Class-Based n-gram Models of Natural Language. In: *Computational Linguistics* 18 (1992), pages 467–479
- [Buch-Kromann and Korzen 2010] BUCH-KROMANN, Matthias ; KORZEN, Iørn: The Unified Annotation of Syntax and Discourse in the Copenhagen Dependency Treebanks. In: *Proceedings of the Fourth Linguistic Annotation Workshop*. Uppsala, Sweden : Association for Computational Linguistics, 2010, pages 127–131
- [Bundschus et al. 2008] BUNDSCHEIS, Markus ; DEJORI, Mathaeus ; STETTER, Martin ; TRESP, Volker ; KRIEGEL, Hans-Peter: Extraction of Semantic Biomedical Relations from Text Using Conditional Random Fields. In: *BMC bioinformatics* 9 (2008), Nr. 1, pages 207
- [Candito et al. 2014] CANDITO, Marie ; PERRIER, Guy ; GUILLAUME, Bruno ; RIBEYRE, Corentin ; FORT, Karén ; SEDDAH, Djamel ; VILLEMONTE DE

- LA CLERGERIE, Éric: Deep Syntax Annotation of the Sequoia French Treebank. In: *Proceedings of the International Conference on Language Resources and Evaluation (LREC)*. Reykjavik, Iceland : ELRA, 2014, pages 2298–2305
- [Caruana 1997] CARUANA, Rich: Multitask Learning. In: *Machine Learning* 28 (1997), Nr. 1, pages 41–75
- [Eckart de Castilho and Gurevych 2014] CASTILHO, Richard Eckart d. ; GUREVYCH, Iryna: A broad-coverage collection of portable NLP components for building shareable analysis pipelines. In: *Proceedings of the Workshop on Open Infrastructures and Analysis Frameworks for HLT*. Dublin, Ireland : Association for Computational Linguistics and Dublin City University, 2014, pages 1–11
- [Chang and Lin 2011] CHANG, Chih-Chung ; LIN, Chih-Jen: LIBSVM: A Library for Support Vector Machines. In: *ACM Transactions on Intelligent Systems and Technology (TIST)* 2 (2011), Nr. 3, pages 1–27
- [Cho et al. 2014] CHO, Kyunghyun ; MERRIENBOER, Bart van ; GÜLÇEHRE, Caglar ; BOUGARES, Fethi ; SCHWENK, Holger ; BENGIO, Yoshua: Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. In: *CoRR* abs/1406.1078 (2014)
- [Choi 2016] CHOI, Jinho D.: Dynamic Feature Induction: The Last Gist to the State-of-the-Art. In: *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*. San Diego, California : Association for Computational Linguistics, 2016, pages 271–281
- [Choi and Palmer 2012] CHOI, Jinho D. ; PALMER, Martha: Fast and Robust Part-of-speech Tagging Using Dynamic Model Selection. In: *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*. Jeju Island, Korea : Association for Computational Linguistics, 2012, pages 363–367
- [Chollet et al. 2015] CHOLLET, François et al.: *Keras*. 2015. – <https://github.com/fchollet/keras>, last accessed 15 January 2018
- [Choudhury et al. 2007] CHOUDHURY, Monojit ; SARAF, Rahul ; JAIN, Vijit ; MUKHERJEE, Animesh ; SARKAR, Sudeshna ; BASU, Anupam: Investigation and modeling of the structure of texting language. In: *International Journal of Document Analysis and Recognition (IJDAR)* 10 (2007), Nr. 3, pages 157–174. – ISSN 1433–2825
- [Christodoulopoulos et al. 2010] CHRISTODOULOUPOULOS, Christos ; GOLDWATER, Sharon ; STEEDMAN, Mark: Two Decades of Unsupervised POS Induction: How Far Have We Come? In: *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Cambridge, Massachusetts : Association for Computational Linguistics, 2010, pages 575–584

- [Chrupala 2014] CHRUPALA, Grzegorz: Normalizing tweets with edit scripts and recurrent neural embeddings. In: *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*. Baltimore, Maryland, USA : Association for Computational Linguistics, 2014, pages 680–686
- [Chrupala 2011] CHRUPALA, Gzegorz: Efficient induction of probabilistic word classes with LDA. In: *Proceedings of the Fifth International Joint Conference on Natural Language Processing (IJCNLP)*. Chiang Mai, Thailand, 2011, pages 363–372
- [Cimino and Dell’Orletta 2016] CIMINO, Andrea ; DELL’ORLETTA, Felice: Building the state-of-the-art in POS tagging of Italian Tweets. In: *Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (EVALITA)* Bd. 1749, CEUR-WS.org, 2016 (CEUR Workshop Proceedings)
- [Collins 2002] COLLINS, Michael: Discriminative Training Methods for Hidden Markov Models: Theory and Experiments with Perceptron Algorithms. In: *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Philadelphia, USA : Association for Computational Linguistics, 2002, pages 1–8
- [Collobert and Weston 2008] COLLOBERT, Ronan ; WESTON, Jason: A Unified Architecture for Natural Language Processing: Deep Neural Networks with Multitask Learning. In: *Proceedings of the International Conference on Machine Learning*. Helsinki, Finland : ACM, 2008. – ISBN 978-1-60558-205-4, pages 160–167
- [Collobert et al. 2011] COLLOBERT, Ronan ; WESTON, Jason ; BOTTOU, Léon ; KARLEN, Michael ; KAVUKCUOGLU, Koray ; KUKSA, Pavel: Natural Language Processing (Almost) from Scratch. In: *The Journal of Machine Learning Research* 12 (2011), pages 2493–2537
- [Csendes et al. 2005] In: CSENDES, Dóra ; CSIRIK, János ; GYIMÓTHY, Tibor ; KOC-SOR, András: *The Szeged Treebank*. Karloy Vary, Czech Republic, 2005, pages 123–131
- [Cucerzan and Yarowsky 1999] CUCERZAN, Silviu ; YAROWSKY, David: Language Independent Named Entity Recognition Combining Morphological and Contextual Evidence. In: *Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*, 1999, pages 90–99
- [Dandapat et al. 2007] DANDAPAT, Sandipan ; SARKAR, Sudeshna ; BASU, Anupam: Automatic Part-of-speech Tagging for Bengali: An Approach for Morphologically Rich Languages in a Poor Resource Scenario. In: *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*. Prague, Czech Republic : Association for Computational Linguistics, 2007, pages 221–224

- [Das et al. 2015] DAS, Bishwa R. ; SAHOO, Smrutirekha ; PANDA, Chandra S. ; PATNAIK, Srikanta: Part of Speech Tagging in Odia Using Support Vector Machine. In: *Procedia Computer Science* 48 (2015), Nr. Supplement C, pages 507 – 512
- [Das and Petrov 2011] DAS, Dipanjan ; PETROV, Slav: Unsupervised Part-of-speech Tagging with Bilingual Graph-based Projections. In: *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*. Portland, Oregon : Association for Computational Linguistics, 2011, pages 600–609
- [Daumé III 2007] DAUMÉ III, Hal: Frustratingly Easy Domain Adaptation. In: *Conference of the Association for Computational Linguistics (ACL)*. Prague, Czech Republic : Association for Computational Linguistics, 2007, pages 256–263
- [Daxenberger et al. 2014] DAXENBERGER, Johannes ; FERSCHKE, Oliver ; GUREVYCH, Iryna ; ZESCH, Torsten: DKPro TC: A Java-based Framework for Supervised Learning Experiments on Textual Data. In: *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL): System Demonstrations*. Baltimore, Maryland : Association for Computational Linguistics, 2014, pages 61–66
- [DeepLearning4J 2017] DEEPLARNING4J: *DeepLearning4J: Open-source distributed deep learning for the JVM*. <http://deeplearning4j.org>, last accessed 15 January 2018, 2017
- [Deng et al. 2013] DENG, Li ; LI, Jinyu ; HUANG, Jui-Ting ; YAO, Kaisheng ; YU, Dong ; SEIDE, Frank ; SELTZER, Michael L. ; ZWEIG, Geoffrey ; HE, Xiaodong ; WILLIAMS, Jason D. ; GONG, Yifan ; ACERO, Alex: Recent Advances in Deep Learning for Speech Research at Microsoft. In: *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, IEEE, 2013, pages 8604–8608
- [Denton et al. 2015] DENTON, Emily L. ; CHINTALA, Soumith ; SZLAM, arthur ; FERGUS, Rob: Deep Generative Image Models using a Laplacian Pyramid of Adversarial Networks. In: *Advances in Neural Information Processing Systems (NIPS)*. Curran Associates, Inc., 2015, pages 1486–1494
- [Derczynski et al. 2015] DERCYNSKI, Leon ; CHESTER, Sean ; BØGH, Kenneth S.: Tune Your Brown Clustering, Please. In: *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP)*. Hissar,Bulgaria, 2015, pages 110–117
- [Derczynski et al. 2013] DERCYNSKI, Leon ; RITTER, Alan ; CLARK, Sam ; BONTCHEVA, Kalina: Twitter Part-of-Speech Tagging for All: Overcoming Sparse and Noisy Data. In: *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP)*, Association for Computational Linguistics, 2013, pages 198 – 206

- [Ebrahim Kafoori and Ahadi 2017] EBRAHIM KAFOORI, Kian ; AHADI, Seyed M.: Robust Recognition of Noisy Speech Through Partial Imputation of Missing Data. In: *Circuits, Systems, and Signal Processing* (2017). <http://dx.doi.org/10.1007/s00034-017-0616-4>. – DOI 10.1007/s00034-017-0616-4. – ISSN 1531-5878
- [Einarsson 1976] EINARSSON, Jan: *Talbankens skriftspråkskonkordans*. 1976
- [Eisenstein 2013] EISENSTEIN, Jacob: What to do about bad language on the internet. In: *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*. Atlanta, Georgia, USA : Association for Computational Linguistics, 2013, pages 359–369
- [Ejherhed and Källgren 1997] EJERHED, Eva ; KÄLLGREN, Gunnel: Stockholm Umeå Corpus (SUC) version 1.0. In: *Department of Linguistics, Umeå University* (1997)
- [Ekbal and Bandyopadhyay 2008] EKBAL, Asif ; BANDYOPADHYAY, Sivaji: Part of Speech Tagging in Bengali Using Support Vector Machine. In: *International Conference on Information Technology*, 2008, pages 106–111
- [Erhan et al. 2010] ERHAN, Dumitru ; BENGIO, Yoshua ; COURVILLE, Aaron ; MANZAGOL, Pierre-Antoine ; VINCENT, Pascal ; BENGIO, Samy: Why Does Unsupervised Pre-training Help Deep Learning? In: *The Journal of Machine Learning Research* 11 (2010), pages 625–660. – ISSN 1532-4435
- [Erjavec 2002] ERJAVEC, Tomaž: Compiling and Using the IJS-ELAN Parallel Corpus. In: *Informatica*, 2002, pages 299–307
- [Erjavec 2010] ERJAVEC, Tomaž: MULTEXT-East Version 4: Multilingual Morphosyntactic Specifications, Lexicons and Corpora. In: *Proceedings of the International Conference on Language Resources and Evaluation (LREC)*. Valletta, Malta : ELRA, 2010
- [Erjavec and Krek 2008] ERJAVEC, Tomaž ; KREK, Simon: The JOS Morphosyntactically Tagged Corpus of Slovene. In: *Proceedings of the International Conference on Language Resources and Evaluation (LREC)*. Marrakesh, Morocco : ELRA, 2008
- [Fan et al. 2008] FAN, Rong-En ; CHANG, Kai-Wei ; HSIEH, Cho-Jui ; WANG, Xiang-Rui ; LIN, Chih-Jen: Liblinear: A Library for Large Linear Classification. In: *The Journal of Machine Learning Research* 9 (2008), pages 1871–1874
- [Farzindar and Inkpen 2015] FARZINDAR, Atefeh ; INKPEN, Diana: *Natural Language Processing for Social Media*. Morgan & Claypool Publishers, 2015
- [Ferrucci and Lally 2004] FERRUCCI, David ; LALLY, Adam: UIMA: An Architectural Approach to Unstructured Information Processing in the Corporate Research Environment. In: *Natural Language Engineering* 10 (2004), Nr. 3-4, pages 327–348. – ISSN 1351-3249

- [Flach 2012] FLACH, Peter: *Machine Learning: The Art and Science of Algorithms that Make Sense of Data*. Cambridge, United Kingdom : Cambridge University Press, 2012. – ISBN 978–1–107–09639–4
- [Forsyth and Martell 2007] FORSYTH, Eric N. ; MARTELL, Craig H.: Lexical and Discourse Analysis of Online Chat Dialog. In: *Proceedings of the International Conference on Semantic Computing*. Washington, DC, USA : IEEE Computer Society, 2007 (ICSC '07), pages 19–26
- [Foster et al. 2011] FOSTER, Jennifer ; ÇETINOGLU, Özlem ; WAGNER, Joachim ; ROUX, Joseph L. ; HOGAN, Stephen ; NIVRE, Joakim ; HOGAN, Deirdre ; GENABITH, Josef van: #hardtoparse: POS Tagging and Parsing the Twitterverse. In: *Analyzing Microtext*, 2011, pages 20–25
- [Foth et al. 2014a] FOTH, Kilian A. ; KÖHN, Arne ; BEUCK, Niels ; MENZEL, Wolfgang: Because Size Does Matter: The Hamburg Dependency Treebank. In: *Proceedings of the International Conference on Language Resources and Evaluation (LREC)*. Reykjavik, Iceland : European Language Resources Association (ELRA), 2014, pages 2326 – 2333
- [Foth et al. 2014b] FOTH, Kilian A. ; KÖHN, Arne ; BEUCK, Niels ; MENZEL, Wolfgang: Because Size Does Matter: The Hamburg Dependency Treebank. In: *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC)*. Reykjavik, Iceland : ELRA, 2014
- [Garside et al. 1997] GARSIDE, R.G. ; LEECH, Geoffrey ; MCENERY, Anthony M.: *Corpus Annotation: Linguistic Information from Computer Text Corpora*. Routledge, 1997. – ISBN 978–1138148581
- [Gers and Schmidhuber 2000] GERS, Felix A. ; SCHMIDHUBER, Juergen: Recurrent Nets That Time and Count. Istituto Dalle Molle Di Studi Sull'Intelligenza Artificiale, 2000. – Forschungsbericht
- [Giesbrecht and Evert 2009] GIESBRECHT, Eugenie ; EVERET, Stefan: Is Part-of-Speech Tagging a Solved Task? An Evaluation of POS Taggers for the German Web as Corpus. In: *Proceedings of the Web as Corpus Workshop*. San Sebastian, Spain, 2009
- [Giménez and Màrquez 2004] GIMÉNEZ, Jesús ; MÀRQUEZ, Lluís: SVMTool: A general POS Tagger Generator Based on Support Vector Machines. In: *Proceedings of the International Conference on Language Resources and Evaluation (LREC)*. Lisbon, Portugal : ELRA, 2004
- [Gimpel et al. 2011] GIMPEL, Kevin ; SCHNEIDER, Nathan ; O'CONNOR, Brendan ; DAS, Dipanjan ; MILLS, Daniel ; EISENSTEIN, Jacob ; HEILMAN, Michael ; YOGATAMA, Dani ; FLANIGAN, Jeffrey ; SMITH, Noah A.: Part-of-speech Tagging for

- Twitter: Annotation, Features, and Experiments. In: *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*. Portland, Oregon : Association for Computational Linguistics, 2011, pages 42–47
- [Goldman and Zhou 2000] GOLDMAN, Sally ; ZHOU, Yan: Enhancing Supervised Learning with Unlabeled Data. In: *Proceedings of the International Conference on Machine Learning*, Morgan Kaufmann, 2000, pages 327–334
- [Goldwater and Griffiths 2007] GOLDWATER, Sharon ; GRIFFITHS, Tom: A fully Bayesian approach to unsupervised part-of-speech tagging. In: *Proceedings of the Annual Meeting of the Association of Computational Linguistics (ACL)*. Prague, Czech Republic, 2007, pages 744–751
- [Gong et al. 2016] GONG, Maoguo ; LI, Hao ; JIANG, Xiangming: A multi-objective optimization framework for ill-posed inverse problems. In: *CAAI Transactions on Intelligence Technology* 1 (2016), Nr. 3, pages 225 – 240. <http://dx.doi.org/https://doi.org/10.1016/j.trit.2016.10.007>. – DOI <https://doi.org/10.1016/j.trit.2016.10.007>. – ISSN 2468–2322
- [Goodfellow et al. 2016] GOODFELLOW, Ian ; BENGIO, Yoshua ; COUVILLE, Aaron: *Deep Learning*. MIT Press, 2016. – ISBN 9780262035613
- [Goodfellow et al. 2014] GOODFELLOW, Ian ; POUGET-ABADIE, Jean ; MIRZA, Mehdi ; XU, Bing ; WARDE-FARLEY, David ; OZAIR, Sherjil ; COURVILLE, Aaron ; BEN-GIO, Yoshua: Generative Adversarial Nets. In: *Advances in Neural Information Processing Systems (NIPS)*, 2014, pages 2672–2680
- [Goodfellow 2017] GOODFELLOW, Ian J.: NIPS 2016 Tutorial: Generative Adversarial Networks. In: *CoRR* abs/1701.00160 (2017)
- [van der Goot 2016] GOOT, Rob van d.: Normalizing Social Media Texts by Combining Word Embeddings and Edit Distances in a Random Forest regressor. In: *In Normalisation and Analysis of Social Media Texts* (2016)
- [van der Goot et al. 2017] GOOT, Rob van d. ; PLANK, Barbara ; NISSIM, Malvina: To Normalize, or Not to Normalize: The Impact of Normalization on Part-of-Speech Tagging. In: *CoRR* abs/1707.05116 (2017)
- [Götz and Suhre 2004] GÖTZ, Thilo ; SUHRE, Oliver: Design and Implementation of the UIMA Common Analysis System. In: *IBM Systems Journal* 43 (2004), Nr. 3, pages 476–489
- [Graves et al. 2005] GRAVES, Alex ; FERNÁNDEZ, Santiago ; SCHMIDHUBER, Jürgen: Bidirectional LSTM Networks for Improved Phoneme Classification and Recognition. In: *Proceedings of the 15th International Conference on Artificial Neural Networks: Formal Models and Their Applications - Volume Part II*. Warsaw, Poland : Springer-Verlag, 2005. – ISBN 3–540–28755–8, 978–3–540–28755–1, pages 799–804

- [Gui et al. 2017] GUI, Tao ; ZHANG, Qi ; HUANG, Haoran ; PENG, Minlong ; HUANG, Xuanjing: Part-of-Speech Tagging For Twitter With Adversarial Neural Networks. In: *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2017, pages 2401–2410
- [Hajič et al. 2009] HAJIČ, Jan ; CIARAMITA, Massimiliano ; JOHANSSON, Richard ; KAWAHARA, Daisuke ; MARTÍ, Maria A. ; MÀRQUEZ, Lluís ; MEYERS, Adam ; NIVRE, Joakim ; PADÓ, Sebastian ; ŠTĚPÁNEK, Jan ; STRAÑÁK, Pavel ; SURDEANU, Mihai ; XUE, Nianwen ; ZHANG, Yi: The CoNLL-2009 Shared Task: Syntactic and Semantic Dependencies in Multiple Languages. In: *Proceedings of the Thirteenth Conference on Computational Natural Language Learning: Shared Task*. Boulder, Colorado, 2009, pages 1–18
- [Halácsy et al. 2007] HALÁCSY, Péter ; KORNAI, András ; ORAVECZ, Csaba: HunPos: An Open Source Trigram Tagger. In: *Proceedings of the Annual Meeting of the Association of Computational Linguistics (ACL): System Demonstrations*. Prague, Czech Republic, 2007, pages 209–212
- [Hall et al. 2009] HALL, Mark ; FRANK, Eibe ; HOLMES, Geoffrey ; PFAHRINGER, Bernhard ; REUTEMANN, Peter ; WITTEN, Ian H.: The WEKA Data Mining Software: An Update. In: *SIGKDD Explorations* 11 (2009), Nr. 1, pages 10–18
- [Han and Baldwin 2011] HAN, Bo ; BALDWIN, Timothy: Lexical Normalisation of Short Text Messages: Makn Sens a #Twitter. In: *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*. Stroudsburg, PA, USA : Association for Computational Linguistics, 2011, pages 368–378
- [Hara et al. 2005] HARA, Tadayoshi ; MIYAO, Yusuke ; TSUJII, Jun’ichi: Adapting a Probabilistic Disambiguation Model of an HPSG Parser to a New Domain. In: *International Conference on Natural Language Processing (IJCNLP)* Bd. 3651. Jeju Island, Korea : Springer-Verlag, 2005, pages 199–210
- [He et al. 2015] HE, Kaiming ; ZHANG, Xiangyu ; REN, Shaoqing ; SUN, Jian: Deep Residual Learning for Image Recognition. In: *CoRR* abs/1512.03385 (2015). <http://arxiv.org/abs/1512.03385>
- [Helgadóttir et al. 2012] HELGADÓTTIR, Sigrún ; SVAVARSDÓTTIR Ásta ; RÖGNVALDSSON, Eiríkur ; BJANADÓTTIR, Kristín ; LOFTSSON, Hrafn: The Tagged Icelandic Corpus (MIM). In: *Proceedings of the Workshop on Language Technology for Normalisation of Less-Resourced Languages*. Istanbul, Turkey : ELRA, 2012
- [Hepple 2000] HEPPEL, Mark: Independence and Commitment: Assumptions for Rapid Training and Execution of Rule-based POS Taggers. In: *Proceedings of the Annual Meeting on Association for Computational Linguistics (ACL)*. Hong Kong : Association for Computational Linguistics, 2000, pages 278–277

- [Herring 1996] HERRING, Susan C.: *Computer-Mediated Communication: Linguistics, social and cross-cultural perspectives*. John Benjamins Publishing Company, 1996
- [Hládek et al. 2012] HLÁDEK, D. ; STAŠ, J. ; JUHÁR, J.: Dagger: The Slovak morphological classifier. In: *Proceedings ELMAR-2012*, 2012, pages 195–198
- [Hochreiter 1998] HOCHREITER, Sepp: The Vanishing Gradient Problem During Learning Recurrent Neural Nets and Problem Solutions. In: *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems* 06 (1998), Nr. 02, pages 107–116
- [Hochreiter and Schmidhuber 1997] HOCHREITER, Sepp ; SCHMIDHUBER, Jürgen: Long Short-Term Memory. In: *Neural Computation* 9 (1997), Nr. 8, pages 1735–1780
- [Horsmann and Zesch 2016a] HORSMANN, Tobias ; ZESCH, Torsten: Building a Social Media Adapted PoS Tagger Using FlexTag – A Case Study on Italian Tweets. In: *Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (EVALITA)*. Naples, Italy, 2016, pages 95–98
- [Horsmann and Zesch 2016b] HORSMANN, Tobias ; ZESCH, Torsten: LTL-UDE @ EmpiriST 2015: Tokenization and PoS Tagging of Social Media Text. In: *Proceedings of the 10th Web as Corpus Workshop (WAC-X) and the EmpiriST Shared Task*. Berlin, Germany : Association for Computational Linguistics, 2016, pages 120–126
- [Huber and Ronchetti 2009] HUBER, Peter J. ; RONCHETTI, Elvezio M.: *Robust Statistics*. Hoboken, New Jersey : John Wiley & sons Inc., 2009
- [Itai and Wintner 2008] ITAI, Alon ; WINTNER, Shuly: Language resources for Hebrew. In: *Language Resources and Evaluation* 42 (2008), Nr. 1, pages 75–98
- [Jakubíček et al. 2011] JAKUBÍČEK, Miloš ; KOVÁŘ, Vojtěch ; ŠMERK, Pavel: Czech Morphological Tagset Revisited. In: *Proceedings of Recent Advances in Slavonic Natural Language Processing (RASLAN)*, 2011, pages 29–42
- [Japkowicz and Shah 2011] JAPKOWICZ, Nathalie ; SHAH, Mohak: *Evaluating Learning Algorithms: A Classification Perspective*. Cambridge University Press, 2011. – ISBN 978-1-107-65311-5
- [Java et al. 2007] JAVA, Akshay ; SONG, Xiaodan ; FININ, Tim ; TSENG, Belle: Why We Twitter: Understanding Microblogging Usage and Communities. In: *Proceedings of the 9th WebKDD and 1st SNA-KDD 2007 Workshop on Web Mining and Social Network Analysis*. San Jose, California : ACM, 2007, 56–65

- [Jiang and Zhai 2007] JIANG, Jing ; ZHAI, Chengxiang: Instance weighting for domain adaptation in NLP. In: *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*. Prague, Czech Republic, 2007, pages 264–271
- [Jin 2015] JIN, Ning: NCSU-SAS-Ning: Candidate Generation and Feature Engineering for Supervised Lexical Normalization. In: *Proceedings of the Workshop on Noisy User-generated Text*, Association for Computational Linguistics, 2015, pages 87–92
- [Joachims 2008] JOACHIMS, Thorsten: *SvmHmm: Sequence Tagging with Structural Support Vector Machines*. 2008. – https://www.cs.cornell.edu/People/tj/svm_light/svm_hmm.html, last accessed 15 January 2018
- [Jørgensen et al. 2016] JØRGENSEN, Anna ; HOVY, Dirk ; SØGAARD, Anders: Learning a POS tagger for AAVE-like language. In: *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL)*. San Diego, California : Association for Computational Linguistics, 2016, pages 1115–1120
- [Jurafsky and Martin 2009] JURAFSKY, Dan ; MARTIN, James H.: *Speech and Language Processing*. Person Education, 2009. – ISBN 978-0-13-504196-3
- [Jurafsky and Martin 2017] JURAFSKY, Dan ; MARTIN, James H.: *Speech and Language Processing*. 2017. – Preprint draft of Chapter 10 in 3rd edition <https://web.stanford.edu/~jurafsky/slp3/10.pdf>, last accessed 15 January 2018
- [Kaplan and Haenlein 2010] KAPLAN, Andreas M. ; HAENLEIN, Michael: Users of the world, unite! The challenges and opportunities of Social Media. In: *Business Horizons* 53 (2010), Nr. 1, pages 59–68. – ISSN 0007-6813
- [Kaufmann and Kalita 2010] KAUFMANN, Max ; KALITA, Jugal: Syntactic normalization of Twitter messages. In: *International Conference on Natural Language Processing*. Kharagpur, India, 2010
- [Kitano 2004] KITANO, Hiroaki: Cancer as a Robust System: Implications for Anti-cancer Therapy. In: *Nature Reviews Cancer* 4 (2004), pages 227–235
- [Koo et al. 2008] KOO, Terry ; CARRERAS, Xavier ; COLLINS, Michael: Simple Semi-supervised Dependency Parsing. In: *Proceedings of Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NACCL)*. Columbus, Ohio : Association for Computational Linguistics, 2008, pages 595–603
- [Krek et al. 2013] KREK, Simon ; ERJAVEC, Tomaž ; DOBROVOLJC, Kaja ; MOŽE, Sara ; LEDINEK, Nina ; HOLZ, Nanika: *Training corpus ssj500k 1.3.* 2013. – Slovenian language resource repository CLARIN.SI

- [Lafferty et al. 2001] LAFFERTY, John D. ; MCCALLUM, Andrew ; PEREIRA, Fernando C N.: Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. In: *Proceedings of the International Conference on Machine Learning (ICML)*. San Francisco, CA, USA : Morgan Kaufmann Publishers Inc., 2001, pages 282–289
- [Leech et al. 1994] LEECH, Geoffrey ; GARSIDE, Roger ; BRYANT, Michael: CLAWS4: The tagging of the British National Corpus. In: *Proceedings of the International Conference on Computational Linguistics*. Kyoto, Japan, 1994, pages 622–628
- [Levandowsky and Winter 1971] LEVANDOWSKY, Michael ; WINTER, David: Distance between Sets. In: *Nature* 234 (1971), pages 34–35
- [Li and Liu 2015] LI, Chen ; LIU, Yang: Joint POS Tagging and Text Normalization for Informal Text. In: *Proceedings of International Joint Conference on Artificial Intelligence (IJCAI)*, AAAI Press, 2015, pages 1263–1269
- [Li et al. 2016] LI, Jinyu ; DENG, Li ; HAEB-UMBACH, Reinhold ; GONG, Yifan: Oxford : Academic Press, 2016. – ISBN 978-0-12-802398-3
- [Li et al. 2017] LI, Yitong ; COHN, Trevor ; BALDWIN, Timothy: Robust Training under Linguistic Adversity. In: *Proceedings of the European Chapter of the Association for Computational Linguistics (EACL)*. Valencia, Spain : Association for Computational Linguistics, April 2017, pages 21–27
- [Liang 2005] LIANG, Percy: *Semi-Supervised Learning for Natural Language*, Massachusetts Institute of Technology, Master thesis, 2005
- [Ling et al. 2015] LING, Wang ; DYER, Chris ; BLACK, W. A. ; TRANCOSO, Isabel ; FERMANDEZ, Ramon ; AMIR, Silvio ; MARUJO, Luis ; LUIS, Tiago: Finding Function in Form: Compositional Character Models for Open Vocabulary Word Representation. In: *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Lisbon, Portugal : Association for Computational Linguistics, 2015, pages 1520–1530
- [Ljubešić et al. 2016] LJUBEŠIĆ, Nikola ; KLUBIČKA, Filip ; AGIĆ Željko ; JAZBEC, Ivo-Pavao: New Inflectional Lexicons and Training Corpora for Improved Morphosyntactic Annotation of Croatian and Serbian. In: *Proceedings of the International Conference on Language Resources and Evaluation (LREC)*. Portorož, Slovenia : ELRA, 2016, pages 4264–4270
- [Manning 2011] MANNING, Christopher D.: Part-of-Speech Tagging from 97% to 100%: Is It Time for Some Linguistics? In: *Proceedings of Computational Linguistics and Intelligent Text Processing (CICLing)*. Tokyo, Japan : Springer Berlin Heidelberg, 2011, pages 171–189

- [Marcus et al. 1993] MARCUS, Mitchell P. ; MARCINKIEWICZ, Mary A. ; SANTORINI, Beatrice: Building a Large Annotated Corpus of English: The Penn Treebank. In: *Computational Linguistics* 19 (1993), Nr. 2, pages 313–330
- [Marimon et al. 2014] MARIMON, Montserrat ; BEL, Núria ; FISAS, Beatriz ; ARIAS, Blanca ; VÁZQUEZ, Silvia ; VIVALDI, Jorge ; MORELL, Carlos ; LORENTE, Mercè: The IULA Spanish LSP Treebank. In: *Proceedings of the International Conference on Language Resources and Evaluation (LREC)*. Reykjavik, Iceland : ELRA, 2014, pages 782–788
- [McCallum and Li 2003] MCCALLUM, Andrew ; LI, Wei: Early Results for Named Entity Recognition with Conditional Random Fields, Feature Induction and Web-enhanced Lexicons. In: *Proceedings of the Conference on Natural Language Learning (CONLL)*. Edmonton, Canada : Association for Computational Linguistics, 2003, pages 188–191
- [McNemar 1947] McNEMAR, Quinn: Note on the sampling error of the difference between correlated proportions or percentages. In: *Psychometrika* 12 (1947), Nr. 2, pages 153–157
- [Mikolov et al. 2013] MIKOLOV, Tomas ; SUTSKEVER, Ilya ; CHEN, Kai ; CORRADO, Greg S. ; DEAN, Jeff: Distributed Representations of Words and Phrases and their Compositionality. In: *Advances in Neural Information Processing Systems (NIPS)*. Curran Associates, Inc., 2013, pages 3111–3119
- [Miller et al. 2004] MILLER, Scott ; GUINNESS, Jethran ; ZAMANIAN, Alex: Name Tagging with Word Clusters and Discriminative Training. In: *Proceedings of Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NACCL)*. Boston, Massachusetts, USA : Association for Computational Linguistics, 2004, pages 337–342
- [Mueller et al. 2013] MUELLER, Thomas ; SCHMID, Helmut ; SCHÜTZE, Hinrich: Efficient Higher-Order CRFs for Morphological Tagging. In: *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Seattle, Washington, USA : Association for Computational Linguistics, 2013, 322–332
- [Müller and Schuetze 2015] MÜLLER, Thomas ; SCHUETZE, Hinrich: Robust Morphological Tagging with Word Representations. In: *Proceedings of Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NACCL)*. Denver, Colorado : Association for Computational Linguistics, 2015, pages 526–536
- [Nelson Francis and Kuçera 1964] NELSON FRANCIS, W. ; KUÇERA, Henry: *Manual of Information to Accompany a Standard Corpus of Present-day Edited American English, for use with Digital Computers*. Department of Linguistics, Brown University, 1964

- [Neubig et al. 2017] NEUBIG, Graham ; DYER, Chris ; GOLDBERG, Yoav ; MATTHEWS, Austin ; AMMAR, Waleed ; ANASTASOPOULOS, Antonios ; BALLESTEROS, Miguel ; CHIANG, David ; CLOTHIAUX, Daniel ; COHN, Trevor ; DUH, Kevin ; FARUQUI, Manaal ; GAN, Cynthia ; GARRETTE, Dan ; JI, Yangfeng ; KONG, Lingpeng ; KUNCORO, Adhiguna ; KUMAR, Gaurav ; MALAVIYA, Chaitanya ; MICHEL, Paul ; ODA, Yusuke ; RICHARDSON, Matthew ; SAPHRA, Naomi ; SWAYAMDIPTA, Swabha ; YIN, Pengcheng: *DyNet: The Dynamic Neural Network Toolkit - Technical Report*. <https://arxiv.org/abs/1701.03980>. Version: 2017
- [Neubig et al. 2014] NEUBIG, Graham ; SUDOHI, Katsuhito ; ODA, Yusuke ; DUH, Kevin ; TSUKADA, Hajime ; NAGATA, Masaaki: The NAIST-NTT TED Talk Treebank. In: *International Workshop on Spoken Language Translation (IWSLT)*. Lake Tahoe, USA, 2014
- [Neunerdt et al. 2013] NEUNERDT, Melanie ; REYER, Michael ; MATHAR, Rudolf: A POS Tagger for Social Media Texts trained on Web Comments. In: *Polibits* 48 (2013), pages 61 – 68
- [Neunerdt et al. 2014] NEUNERDT, Melanie ; REYER, Michael ; MATHAR, Rudolf: Efficient Training Data Enrichment and Unknown Token Handling for POS Tagging of Non-standardized Texts. In: *Conference on Natural Language Processing (KONVENS)*. Hildesheim, Germany, 2014, pages 186–192
- [Ng and Jordan 2002] NG, Andrew Y. ; JORDAN, Michael I.: On Discriminative vs. Generative Classifiers: A comparison of logistic regression and naive Bayes. In: *Advances in Neural Information Processing Systems*. MIT Press, 2002, pages 841–848
- [Nivre et al. 2016] NIVRE, Joakim ; MARNEFFE, Marie-Catherine de ; GINTER, Filip ; GOLDBERG, Yoav ; HAJIC, Jan ; MANNING, Christopher D. ; McDONALD, Ryan ; PETROV, Slav ; PYYSALO, Sampo ; SILVEIRA, Natalia ; TSARFATY, Reut ; ZEMAN, Daniel: Universal Dependencies v1: A Multilingual Treebank Collection. In: *Proceedings of the International Conference on Language Resources and Evaluation (LREC)*. Portorož, Slovenia : ELRA, 2016
- [Obar and Wildman 2015] OBAR, Jonathan A. ; WILDMAN, Steve: Social media definition and the governance challenge: An introduction to the special issue. In: *Telecommunications Policy* 39 (2015), Nr. 9, pages 745 – 750. – ISSN 0308–5961
- [Okazaki 2007] OKAZAKI, Naoaki: *CRFsuite: a fast implementation of Conditional Random Fields (CRFs)*. 2007. – <http://www.chokkan.org/software/crfsuite/>, last accessed 15 January 2018
- [Owoputi et al. 2013] OWOPUTI, Olutobi ; DYER, Chris ; GIMPEL, Kevin ; SCHNEIDER, Nathan ; SMITH, Noah A.: Improved part-of-speech tagging for online conversational text with word clusters. In: *Proceedings of the Conference of the North*

- American Chapter of the Association for Computational Linguistics (NAACL). Atlanta, USA, 2013, pages 380–390
- [Paci 2016] PACI, Giulio: Mivoq Evalita 2016 PosTwITA tagger. In: *Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (EVALITA)*. Naples, Italy, 2016, pages 99–103
- [Paroubek 2000] PAROUBEK, Patrick: Language Resources as by-Product of Evaluation: The MULTITAG Example. In: *Proceedings of the International Conference on Language Resources and Evaluation (LREC)*. Athens, Greece : ELRA, 2000
- [Patterson and Gibson] PATTERSON, Josh ; GIBSON, Adam: *Deep Learning: A Practitioner's Approach*. O'Reilly Media, Inc.. – ISBN 978-1-491-91425-0
- [Peng and Dredze 2017] PENG, Nanyun ; DREDZE, Mark: Multi-task Domain Adaptation for Sequence Tagging. In: *Proceedings of the 2nd Workshop on Representation Learning for NLP*, Association for Computational Linguistics, 2017, pages 91–100
- [Pennell and Liu 2011] PENNELL, Deana L. ; LIU, Yang: A character-level machine translation approach for normalization of sms abbreviations. In: *In Proceedings of the Joint Conference on Natural Language Processing*. Chiang Mai, Thailand, 2011, pages 974–982
- [Pennington et al. 2014] PENNINGTON, Jeffrey ; SOCHER, Richard ; MANNING, Christopher D.: GloVe: Global Vectors for Word Representation. In: *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)* Bd. 14. Doha, Qatar, 2014, pages 1532–1543
- [Petrov et al. 2012] PETROV, Slav ; DAS, Dipanjan ; McDONALD, Ryan: A Universal Part-of-Speech Tagset. In: *Proceedings of the International Conference on Language Resources and Evaluation (LREC)*. Istanbul, Turkey : ELRA, 2012, pages 2089 – 2096
- [Plank et al. 2014] PLANK, Barbara ; HOVY, Dirk ; McDONALD, Ryan ; SØGAARD, Anders: Adapting taggers to Twitter with not-so-distant supervision. In: *Proceedings of the International Conference on Computational Linguistics (COLING)*. Dublin, Ireland, 2014, pages 1783–1792
- [Plank and Nissim 2016] PLANK, Barbara ; NISSIM, Malvina: When silver glitters more than gold: Bootstrapping an Italian part-of-speech tagger for Twitter. In: *Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (EVALITA)*. Naples, Italy, 2016, pages 108–113
- [Plank et al. 2016] PLANK, Barbara ; SØGAARD, Anders ; GOLDBERG, Yoav: Multilingual Part-of-Speech Tagging with Bidirectional Long Short-Term Memory Models and Auxiliary Loss. In: *Proceedings of the Annual Meeting of the Association of Computational Linguistics (ACL)*. Berlin, Germany : Association for Computational Linguistics, 2016, pages 412–418

- [Prange et al. 2016] PRANGE, Jakob ; HORBACH, Andrea ; THATER, Stefan: Uds-(retrain|distributional)surface): Improving POS Tagging for OOV Words in German CMC and Web Data. In: *Proceedings of the 10th Web as Corpus Workshop (WAC-X) and the EmpiriST Shared Task*, Association for Computational Linguistics, 2016, pages 63–71
- [Przeipiorkowski et al. 2008] PRZEPIÓRKOWSKI, Adam ; GÓRSKI, Rafal L. ; LEWANDOWSKA-TOMASZYK, Barbara ; LAZINSKI, Marek: Towards the National Corpus of Polish. In: *Proceedings of the International Conference on Language Resources and Evaluation (LREC)*. Marrakech, Morocco : ELRA, 2008
- [Quasthoff et al. 2006] QUASTHOFF, Uwe ; RICHTER, Matthias ; BIEMANN, Christian: Corpus Portal for Search in Monolingual Corpora. In: *Proceedings of the International Conference on Language Resources and Evaluation (LREC)*. Genoa : ELRA, 2006, pages 1799–1802
- [Ratnaparkhi 1996] RATNAPARKHI, Adwait: A Maximum Entropy Model for Part-Of-Speech Tagging. In: *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Philadelphia, USA, 1996, pages 133 – 142
- [Rehbein 2013] REHBEIN, Ines: Fine-Grained POS Tagging of German Tweets. In: *Proceedings of Language Processing and Knowledge in the Web*. Berlin, Heidelberg : Springer Berlin Heidelberg, 2013, pages 162–175
- [Ritter et al. 2011] RITTER, Alan ; CLARK, Sam ; MAUSAM ; ETZIONI, Oren: Named Entity Recognition in Tweets: An Experimental Study. In: *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Edinburgh, Scotland, United Kingdom : Association for Computational Linguistics, 2011, pages 1524–1534
- [Rudrapal et al. 2015] RUDRAPAL, Dwijen ; JAMATIA, Anupam ; CHAKMA, Kunal ; DAS, Amitava ; GAMBÄCK, Björn: Sentence Boundary Detection for Social Media Text. In: *Proceedings of the International Conference on Natural Language Processing*, 2015, pages 91–97
- [Schiller et al. 1999] SCHILLER, Anne ; TEUFEL, Simone ; STÖCKERT, Christine ; THIELEN, Christine: Guidelines für das Tagging deutscher Textcorpora mit STTS (Kleines und großes Tagset), Institut für maschinelle Sprachverarbeitung, University of Stuttgart, Germany, 1999
- [Schmid 1994a] SCHMID, Helmut: Part-of-speech Tagging with Neural Networks. In: *Proceedings of the International Conference on Computational Linguistics (COLING)*. Kyoto, Japan : Association for Computational Linguistics, 1994, pages 172–176

- [Schmid 1994b] SCHMID, Helmut: Probabilistic Part-of-Speech Tagging Using Decision Trees. In: *International Conference on New Methods in Language Processing*. Manchester, United Kingdom, 1994, pages 44–49
- [Schmid 1995] SCHMID, Helmut: Improvements In Part-of-Speech Tagging With an Application To German. In: *Proceedings of the ACL SIGDAT-Workshop*, 1995, pages 47–50
- [Schmid and Laws 2008] SCHMID, Helmut ; LAWS, Florian: Estimation of Conditional Probabilities with Decision Trees and an Application to Fine-grained POS Tagging. In: *Proceedings of the International Conference on Computational Linguistics (COLING)*. Manchester, United Kingdom : Association for Computational Linguistics, 2008, pages 777–784
- [Seraji 2011] SERAJI, Mojgan: A Statistical Part-of-Speech Tagger for Persian. In: *Proceedings of the Nordic Conference of Computational Linguistics (NODALIDA)*, 2011, pages 340–343
- [Serdyuk et al. 2016] SERDYUK, Dmitriy ; AUDHKHASI, Kartik ; BRAKEL, Philemon ; RAMABHADRAN, Bhuvana ; THOMAS, Samuel ; BENGIO, Yoshua: Invariant Representations for Noisy Speech Recognition. In: *CoRR* abs/1612.01928 (2016)
- [Sha and Pereira 2003] SHA, Fei ; PEREIRA, Fernando: Shallow Parsing with Conditional Random Fields. In: *Proceedings of Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NACCL)*. Edmonton, Canada : Association for Computational Linguistics, 2003, pages 134–141
- [Singh et al. 2006] SINGH, Smriti ; GUPTA, Kuhoo ; SHRIVASTAVA, Manish ; BHATTACHARYYA, Pushpak: Morphological Richness Offsets Resource Demand- Experiences in Constructing a POS Tagger for Hindi. In: *Proceedings of the International Conference on Computational Linguistics (COLING)*. Stroudsburg, PA, USA : Association for Computational Linguistics, 2006, 779–786
- [Skut et al. 1998] SKUT, Wojciech ; USZKOREIT, Hans ; BRANTS, Thorsten ; KRENN, Brigitte: A Linguistically Interpreted Corpus of German Newspaper Text. In: *Proceedings of the European Summer School in Logic, Language and Information (ESSLLI). Workshop on Recent Advances in Corpus Annotation*. Saarbrücken, Germany, 1998
- [Socher et al. 2013] SOCHER, Richard ; GANJOO, Milind ; MANNING, Christopher D. ; NG, Andrew Y.: Zero-shot Learning Through Cross-modal Transfer. In: *Advances in Neural Information Processing Systems (NIPS)*. Lake Tahoe, Nevada : Curran Associates Inc., 2013, pages 935–943
- [Solberg et al. 2014] SOLBERG, Per E. ; SKJÆHOLT, Arne ; ØVRELID, Lilja ; HAGEN, Kristin ; JOHANNESSEN, Janne B.: The Norwegian Dependency Treebank. In:

- Proceedings of the International Conference on Language Resources and Evaluation (LREC).* Reykjavík, Iceland : ELRA, 2014
- [Spoustová et al. 2009] SPOUSTOVÁ, Drahomíra ; HAJIČ, Jan ; RAAB, Jan ; SPOUSTA, Miroslav: Semi-Supervised Training for the Averaged Perceptron POS Tagger. In: *Proceedings of the Conference of the European Chapter of the ACL (EACL).* Athens, Greece : Association for Computational Linguistics, 2009, pages 763–771
- [Spoustová et al. 2007] SPOUSTOVÁ, Drahomíra ; HAJIČ, Jan ; VOTRUBEC, Jan ; KRBEČ, Pavel ; KVĚTOŇ, Pavel: The Best of Two Worlds: Cooperation of Statistical and Rule-based Taggers for Czech. In: *Proceedings of the Workshop on Balto-Slavonic Natural Language Processing: Information Extraction and Enabling Technologies.* Stroudsburg, PA, USA : Association for Computational Linguistics, 2007, pages 67–74
- [Sproat et al. 2001] SPROAT, Richard ; BLACK, Alan W. ; CHEN, Stanley ; KUMAR, Shankar ; OSTENDORF, Mari ; RICHARDS, Christopher: Normalization of Non-standard Words. In: *Computer Speech and Language* 15 (2001), Juli, Nr. 3, pages 287–333. – ISSN 0885–2308
- [Stamatatos 2009] STAMATATOS, Efstathios: A survey of modern authorship attribution methods. In: *Journal of the American Society for Information Science and Technology* 60 (2009), Nr. 3, pages 538–556. – ISSN 1532–2890
- [Sutton and McCallum 2012] SUTTON, Charles ; MCCALLUM, Andrew: An Introduction to Conditional Random Fields. In: *Foundations and Trends in Machine Learning* 4 (2012), Nr. 4, pages 267–373. – ISSN 1935–8237
- [Tamburini 2016] TAMBURINI, Fabio: A BiLSTM-CRF PoS-tagger for Italian Tweets Using Morphological Information. In: *Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (EVALITA).* Naples, Italy, 2016, pages 120–123
- [Taylor et al. 2003] TAYLOR, Ann ; MARCUS, Mitchell ; SANTORINI, Beatrice ; ABEILLÉ, Anne (Hrsg.): *The Penn Treebank: An Overview.* Dordrecht : Springer Netherlands, 2003. – 5–22 S. – ISBN 978–94–010–0201–1
- [Telljohann et al. 2004] TELLJOHANN, Heike ; HINRICHES, Erhard ; KÜBLER, Sandra ; KÜBLER, Ra ; TÜBINGEN, Universität: The Tüba-D/Z Treebank: Annotating German with a Context-Free Backbone. In: *Proceedings of the International Conference on Language Resources and Evaluation (LREC).* Lisbon, Portugal : European Language Resources Association (ELRA), 2004, pages 1175–1178
- [Thurlow et al. 2004] THURLOW, Chrispin ; LENGEL, Laura ; TOMIC, Alice: *Computer Mediated Communication – Social Interaction And The Internet.* Sage Publications, 2004

- [Tjong Kim Sang and De Meulder 2003] TJONG KIM SANG, Erik F. ; DE MEULDER, Fien: Introduction to the CoNLL-2003 Shared Task: Language-independent Named Entity Recognition. In: *Proceedings of Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NACCL)*. Edmonton, Canada : Association for Computational Linguistics, 2003, pages 142–147
- [Toutanova et al. 2003] TOUTANOVA, Kristina ; KLEIN, Dan ; MANNING, Christopher D. ; SINGER, Yoram: Feature-rich Part-of-speech Tagging with a Cyclic Dependency Network. In: *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics (ACL)*. Edmonton, Canada : Association for Computational Linguistics, 2003, pages 173–180
- [Toutanova and Manning 2000] TOUTANOVA, Kristina ; MANNING, Christopher D.: Enriching the Knowledge Sources Used in a Maximum Entropy Part-of-speech Tagger. In: *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Hong Kong : Association for Computational Linguistics, 2000, 63–70
- [Tsuruoka et al. 2005] In: TSURUOKA, Yoshimasa ; TATEISHI, Yuka ; KIM, Jin-Dong ; OHTA, Tomoko ; MCNAUGHT, John ; ANANIADOU, Sophia ; TSUJII, Jun’ichi: *Developing a Robust Part-of-Speech Tagger for Biomedical Text*. Berlin, Heidelberg : Springer Berlin Heidelberg, 2005. – ISBN 978-3-540-32091-3, pages 382–392
- [Ueffing and Ney 2003] UEFFING, Nicola ; NEY, Hermann: Using POS Information for Statistical Machine Translation into Morphologically Rich Languages. In: *Proceedings of the Conference of the European Chapter of the ACL (EACL)*. Budapest, Hungary : Association for Computational Linguistics, 2003, pages 347–354
- [Uličný et al. 2016] ULIČNÝ, Matej ; LUNDSTRÖM, Jens ; BYTTNER, Stefan: Robustness of Deep Convolutional Neural Networks for Image Recognition. Merida, Mexico, 2016, pages 16–30
- [Vacavant 2017] In: VACAVANT, Antoine: *A Novel Definition of Robustness for Image Processing Algorithms*. Cham : Springer International Publishing, 2017. – ISBN 978-3-319-56414-2, 75–87
- [Vapnik and Lerner 1963] VAPNIK, Vladimir ; LERNER, A.: Pattern Recognition using Generalized Portrait Method. In: *Automation and Remote Control* (1963), pages 774–780
- [Viterbi 1967] VITERBI, Andrew: Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. In: *IEEE Transactions on Information Theory* 13 (1967), pages 260–269

- [Voutilainen 2011] VOUTILAINEN, Atro: FinnTreeBank: Creating a research resource and service for language researchers with Constraint Grammar. In: *Constraint Grammar Applications* (2011), pages 41–49
- [Westpfahl and Schmidt 2016] WESTPFAHL, Swantje ; SCHMIDT, Thomas: FOLK-Gold – A Gold Standard for Part-of-Speech-Tagging of Spoken German. In: *Proceedings of the International Conference on Language Resources and Evaluation (LREC)*. Portorož, Slovenia : European Language Resources Association (ELRA), 2016, pages 1493 – 1499
- [Yang et al. 2017] YANG, Qingsong ; YAN, Pingkun ; ZHANG, Yanbo ; YU, Hengyong ; SHI, Yongyi ; MOU, Xuanqin ; KALRA, Mannudeep K. ; WANG, Ge: Low Dose CT Image Denoising Using a Generative Adversarial Network with Wasserstein Distance and Perceptual Loss. In: *CoRR* abs/1708.00961 (2017)
- [Yang and Eisenstein 2015] YANG, Yi ; EISENSTEIN, Jacob: Unsupervised Multi-Domain Adaptation with Feature Embeddings. In: *Proceedings of Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NACCL)*. Denver, Colorado : Association for Computational Linguistics, 2015, pages 672–682
- [Yang and Eisenstein 2016] YANG, Yi ; EISENSTEIN, Jacob: Part-of-Speech Tagging for Historical English. In: *CoRR* abs/1603.03144 (2016)
- [Yasunaga et al. 2017] YASUNAGA, Michihiro ; KASAI, Jungo ; RADEV, Dragomir R.: Robust Multilingual Part-of-Speech Tagging via Adversarial Training. In: *CoRR* abs/1711.04903 (2017)
- [Zeldes 2016] ZELDES, Amir: The GUM corpus: creating multilayer resources in the classroom. In: *Language Resources and Evaluation* (2016), pages 1–32
- [Zesch and Horsmann 2016] ZESCH, Torsten ; HORSMANN, Tobias: FlexTag: A Highly Flexible PoS Tagging Framework. In: *Proceedings of the International Conference on Language Resources and Evaluation (LREC)*. Portorož, Slovenia : ELRA, 2016, pages 4259–4263
- [Zhang et al. 2013] ZHANG, Congle ; BALDWIN, Tyler ; HO, Howard ; KIMELFELD, Benny ; LI, Yunyao: Adaptive Parser-Centric Text Normalization. In: *Proceedings of the Seventeenth Conference on Computational Natural Language Learning*. Sofia, Bulgaria, 2013, pages 1159–1168
- [Zheng et al. 2016] ZHENG, Stephan ; SONG, Yang ; LEUNG, Thomas ; GOODFELLOW, Ian: Improving the Robustness of Deep Neural Networks via Stability Training. In: *Computer Vision and Pattern Recognition (CVPR)*, 2016
- [Zhou and Li 2005] ZHOU, Zhi-Hua ; LI, Ming: Tri-Training: Exploiting Unlabeled Data Using Three Classifiers. In: *IEEE Transactions on Knowledge and Data Engineering* 17 (2005), Nr. 11, pages 1529–1541

- [Östling 2013] ÖSTLING, Robert: Stagger: An Open-Source Part of Speech Tagger for Swedish. In: *Northern European Journal of Language Technology (NEJLT)* 3 (2013), pages 1–18