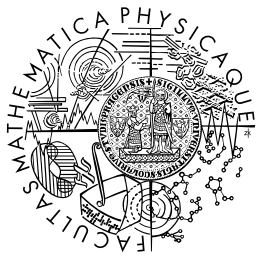

FACULTY
OF MATHEMATICS
AND PHYSICS
Charles University

DOCTORAL THESIS

Jana Straková

Neural Network Based Named Entity Recognition

Institute of Formal and Applied Linguistics

Supervisor of the doctoral thesis: prof. RNDr. Jan Hajič, Dr.

Study programme: Computer Science

Specialization: Mathematical Linguistics (4I3)

Prague 2017

I declare that I carried out this doctoral thesis independently, and only with the cited sources, literature and other professional sources.

I understand that my work relates to the rights and obligations under the Act No. 121/2000 Sb., the Copyright Act, as amended, in particular the fact that the Charles University has the right to conclude a license agreement on the use of this work as a school work pursuant to Section 60 subsection 1 of the Copyright Act.

In Prague, date ....

signature of the author

I would like to express my sincere gratitude to Jan Hajič, a most inspiring scientist. His enthusiasm and broad experience always encouraged me in my work. I am very grateful that I could grow profesionally under his supervision.

I would also like to thank my colleagues at the Institute of Formal and Applied Linguistics for a friendly and inspiring work environment. I would especially like to thank Milan Straka and Zdeněk Žabokrtský for comments which helped me to improve this thesis.

Finally, all this work would not be possible without my beloved husband Milan, our two beautiful sons and their loving grandparents.

Title: Neural Network Based Named Entity Recognition

Author: Jana Straková

Institute: Institute of Formal and Applied Linguistics

Supervisor of the doctoral thesis: prof. RNDr. Jan Hajič, Dr., Institute of Formal and Applied Linguistics

Abstract: Czech named entity recognition (the task of automatic identification and classification of proper names in text, such as names of people, locations and organizations) has become a well-established field since the publication of the Czech Named Entity Corpus (CNEC). This doctoral thesis presents the author's research of named entity recognition, mainly in the Czech language. It presents work and research carried out during CNEC publication and its evaluation. It further envelops the author's research results, which improved Czech state-of-the-art results in named entity recognition in recent years, with special focus on artificial neural network based solutions. Starting with a simple feed-forward neural network with softmax output layer, with a standard set of classification features for the task, the thesis presents methodology and results, which were later used in open-source software solution for named entity recognition, NameTag. The thesis finalizes with a recurrent neural network based recognizer with word embeddings and character-level word embeddings, based on recent advances in neural network research, which requires no classification features engineering and achieves excellent state-of-the-art results in Czech named entity recognition.

Keywords: named entity recognition, Czech Named Entity Corpus, artificial neural networks, recurrent neural networks, softmax, word embeddings, character-level word embeddings

Contents

1 Introduction **1**

1.1 Organization of This Work and the Author's Contribution 2

1.2 How To Read This Thesis 5

2 Named Entity Recognition **7**

2.1 Named Entity Recognition Task 7

2.2 Named Entity Recognition Methodology 9

2.3 NER Evaluation 14

2.4 Related Work in Czech NER 15

2.5 Related Work in English NER 18

3 Czech Named Entity Corpus **21**

3.1 Czech Named Entity Corpus 1.0 21

3.2 Using the Corpus for NE Recognizers 26

3.3 Czech Named Entity Corpus 2.0 32

3.4 Conclusions 35

4 Efficient Log-linear Modeling and Softmax NNs **37**

4.1 Introduction 37

4.2 Statistical Modeling 38

4.3 Log-linear Models 39

4.4 Neural Networks 40

4.5 Case Studies 45

4.6 Results and Discussion 50

4.7 Conclusions 54

5 Czech NE Recognizer with Softmax Neural Network **55**

5.1 System Overview 55

5.2 Softmax Neural Network Classifier 56

5.3 Decoding 57

5.4 Classification Features 57

5.5 Results and Discussion 60

5.6 Conclusions 61

6 NameTag: An Open-Source Tool for NER **63**
6.1 Introduction 63
6.2 NER Methodology 64
6.3 Software Performance 64
6.4 Release 65
6.5 Conclusion 66

7 Neural Networks for Featureless NER in Czech **67**
7.1 Introduction 67
7.2 Word Embeddings 68
7.3 Experiment Setting: Data and Evaluation 70
7.4 Related Work 70
7.5 The Artificial Neural Network Classifier 72
7.6 Results and Discussion 75
7.7 Conclusions 78

8 Conclusions **81**

9 Acknowledgments **85**

Bibliography **87**

List of Figures **101**

List of Tables **103**

List of Abbreviations **105**

Appendix **107**
A.1 Efficient Log-linear Modeling and Softmax Neural Networks . . . 107
A.2 Classification Features 108

Chapter 1

Introduction

Human ability to communicate through natural language has fascinated scientists for decades. Neurolinguists have thoroughly studied the way human brain works by means of examining the effects of losing the ability to perceive or produce natural language (i.e. due to a lesion in a particular brain area) and more recently, following recent advances in brain imaging technology, with experiments carefully designed to reveal the nature of human brain functioning in natural language production or comprehension. Classical linguists and philosophers have also wondered about natural language from a different perspective and have come forward with philosophical theories about the inner features of language, its structure and universal features. Computational linguists have been trying to quantitatively describe natural language and the human brain ability to use it, with means of mathematical, especially statistical models, and following a giant leap in information technologies, this pursue moved from purely mathematical quest and resulted in tools which help people in using language: machine translators, dictionaries, information retrieval tools and much more.

Natural language processing (NLP) is a field of science which seeks to automatically capture the ability to use natural language and model human way of language comprehension or production. As such, natural language processing spans over multiple disciplines such as artificial intelligence, computer science, mathematics (especially statistics) and of course, linguistics. The tasks of NLP include challenges such as morphological, syntactic and semantic analysis of unstructured text, sentiment analysis, information retrieval, question answering or machine translation. Obviously, most of these large scale objectives can be divided into related sub-problems: part-of-speech tagging, lemmatization, dependency parsing, or the topic of this thesis, named entity recognition, and others.

Natural language research is by its very nature an interdisciplinary field and

the author of this thesis was fortunate enough to be supported and encouraged to follow and carry out exciting research in multiple disciplines: neurolinguistics (Kim and Straková, 2012), semantics (Agirre et al., 2009), information retrieval (Straková and Pecina, 2010), artificial neural networks (Chapters 4, 5 and 7), dependency parsing with artificial neural networks (Straka et al., 2015), part-of-speech tagging (Straková et al., 2014) and named entity recognition (Kravalová and Žabokrtský, 2009; Straková et al., 2013; Straková, 2015; Straková et al., 2016; Straková et al., 2017).

The software which emerged as a result tackles classical tasks of automatic natural language processing: MorphoDiTa (Morphological Dictionary and Tagger, (Straková et al., 2014)[1]) is a free software which performs morphological analysis (with lemmatization), morphological generation, tagging and tokenization with state-of-the-art results for Czech. NameTag (Straková et al., 2014)[2] is a free software for named entity recognition which achieves state-of-the-art performance for Czech. Parsito (Straka et al., 2015)[3] is a free software for dependency parsing based on neural networks. The tagging and parsing tools[4] are united in a simple-to-use tool UDPipe (Straka et al., 2016)[5], a pipeline processing CoNLL-U-formatted files, which performs tokenization, morphological analysis, part-of-speech tagging, lemmatization and dependency parsing for nearly all treebanks of Universal Dependencies 1.2 (the pipeline is currently available for more than 30 languages).

This thesis contains the results of the author's long-standing research interest: named entity recognition, particularly in Czech language. The following sections 1.1 and 1.2 describe the organization of this thesis and are intended to facilitate the kind reader's orientation in reading of this work. Section 1.1 also contains a description of the author's contribution.

1.1 Organization of This Work and the Author's Contribution

This thesis publishes results in the field of named entity recognition (a task of automatic identification and classification of named entities, such as names of people, cities, organizations) with excellent state-of-the-art results for Czech. The

[1] `http://ufal.mff.cuni.cz/morphodita`
[2] `http://ufal.mff.cuni.cz/nametag`
[3] `http://ufal.mff.cuni.cz/parsito`
[4] except NameTag
[5] `http://ufal.mff.cuni.cz/udpipe`

organization of this work is as follows:

Chapter 1 opens this work with author's motivations, facilitates the reader's navigation in the organization of this thesis and presents an overview of the author's research contribution.

Chapter 2 describes the task of named entity recognition, especially in the Czech language. It describes the (relatively short) history of Czech named entity recognition research and related work. The text is intended as an introduction to named entity recognition and may easily be skipped by an advanced reader. It is based on an encyclopedic entry (Straková, 2015) written by the author for Karlík et al. (2015) and further extended.

Chapter 3 describes the Czech Named Entity Corpus. It is partially taken from Kravalová and Žabokrtský (2009) and from the author's submission (Straková et al., 2017) to forthcoming Ide and Pustejovsky (2017).

We decided not to include a chapter on SVM-based named entity recognizer (Kravalová and Žabokrtský, 2009) as we consider this recognizer outdated, especially in the light of more recent work (Straková et al., 2013, 2016).

Chapter 4 presents our yet unpublished in-house experiments with artificial neural network architectures, namely with drop-in replacement of log-linear models trained using maximum entropy principle by artificial neural networks with a softmax output layer trained by stochastic gradient descent.

Chapter 5 describes a Czech named entity recognizer published in paper Straková et al. (2013) and Chapter 6 follows with a NameTag software description (Straková et al., 2014).[6]

Chapter 7 presents our most recent research with featureless named entity recognizer based on artificial neural networks with word embeddings and character-level word embeddings (Straková et al., 2016).

Finally, Chapter 8 concludes the thesis with final remarks.

The contribution of this thesis is that it describes major improvements in Czech named entity recognition, which have been forming Czech named entity recognition state of the art in the recent years (Kravalová and Žabokrtský (2009); Straková et al. (2013, 2016), Chapters 5 and 7) and are also available to the community as an open-source project NameTag (Straková et al. (2014), Chapter 6). It also offers a decent insight into named entity recognition challenges, conceptions and methodology (Chapter 2).

Specifically, the contributions of the thesis author are the following:

The SVM-based recognizer (Kravalová and Žabokrtský, 2009) was completely

[6]http://ufal.mff.cuni.cz/nametag

designed, implemented and evaluated by the author of the thesis. The author set up the experimental pipeline, implemented the Perl prototype and carried out the experiments. The author then wrote the related methodology, results and discussion sections of Kravalová and Žabokrtský (2009).

Similarly, the softmax artificial neural network based NE recognizer (Straková et al. (2013), Chapter 5) was also completely designed and evaluated by the author. Straková et al. (2013) and Chapter 5 were written mainly by the thesis author.

The experiments with drop-in replacement for log-linear models with softmax artificial neural networks (Chapter 4) are a joint work with Milan Straka: The overall case studies architecture is the author's work and the author of this thesis experimented with case studies NER and SRL (please see Section 4.5.1 and Section 4.5.2), while Milan Straka provided the POS results (Section 4.5.3) and implemented the artificial neural network classifier in C++. The text in Chapter 4 was originally written jointly with Milan Straka, Jan Hajič and Martin Popel in 2014, and further thoroughly updated and extended by the thesis author in 2016.

The author prepared the release of CNEC 1.0, CNEC 1.1 and cooperated in CNEC 2.0 release, including the packaging, documentation and online documentation and further maintaining. The author did not propose the CNEC 1.0, nor the NE hierarchy described in Section 3.1. All the annotation work on CNEC 1.0 was done before the author joined the NE team. However, the author did participate in discussions about the CNEC 2.0 named entity hierarchy (Section 3.3). The author thoroughly evaluated the CNEC corpus and reported the results in Section 3.2.

NameTag, the NE recognizer described in Chapter 6, is a direct reimplementation of the author's Perl prototype proposed in Straková et al. (2013) (Chapter 5) and was jointly implemented in C++ by Milan Straka and the author. Milan Straka implemented the underlying softmax artificial neural network classifier as described above and in detail in Chapter 4 and the author implemented the NE-related methodology in C++. The text in Chapter 6 is based on Straková et al. (2014), a joint work with Milan Straka and Jan Hajič.

The overall system architecture, as well as experiment design and evaluation of Chapter 7 is the work of the thesis author. The system architecture is re-used from the author's previous work (Straková et al. (2013), Chapter 5), the artificial neural network classifier is the Milan Straka's implementation from Chapter 4. The author carried out all the experiments, wrote a new Perl prototype and shell

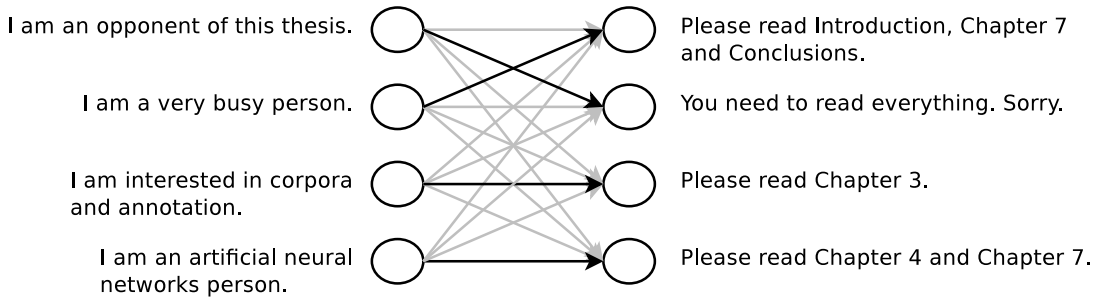

Figure 1.1: How to read this thesis. A simple artificial neural network, bold connections are activated.

pipeline for this purpose and Lua scripts for artificial neural network experiments in Torch (see Straková et al. (2016) and Chapter 7), generated the word embeddings and is the main author of Straková et al. (2016). The character-level embeddings were implemented in Lua for Torch by Milan Straka.

The author also participated as co-author in other NLP-related fields (Agirre et al., 2009; Straková and Pecina, 2010; Kim and Straková, 2012; Straka et al., 2015, 2016) and acted as an advisor in MorphoDiTa (Straková et al., 2014)[7] and Parsito (Straka et al., 2015)[8] development. This thesis however describes only the author's NER-related work.

1.2 How To Read This Thesis

We refer the kind reader to Figure 1.1, which contains vital information on how to read this thesis.

[7] http://ufal.mff.cuni.cz/morphodita

[8] http://ufal.mff.cuni.cz/parsito

Chapter 2

Named Entity Recognition

This chapter is an introductory text to named entity recognition and describes the task and its standard methodology. Advanced readers are welcome to skip this chapter. The opening Section 2.1 introduces the named entity task and is based on a translation of an encyclopedic entry (Straková, 2015) for Karlík et al. (2015). It was translated to English, updated and broadened for this thesis. The following two Sections explain the standard methodology for the NER task (Section 2.2) and its evaluation (Section 2.3). The last two Sections 2.4 and 2.5 present an overview of related work in Czech and English, respectively.

2.1 Named Entity Recognition Task

Named entity recognition (NER) is one of the tasks of automatic natural language processing (NLP). The task is to automatically identify so-called *named entities*: words or sequences of words which denote unique names, locations, organizations and so on. These entities are also sometimes called *proper names* in natural language. The task typically involves classification of these identified entities into a set of predefined classes. Therefore the task of named entity recognition is sometimes subdivided into two subtasks: *named entity identification* (the task is to correctly identify a named entity span without further classification, that is, to retrieve all named entity tokens) and *named entity classification* (the task is to correctly classify the retrieved named entity tokens into a set of predefined classes). Mostly, *named entity recognition* (NER) usually stands for both of the tasks performed jointly.

An example of named entity recognition in Czech sentence follows. Consider the following headline:[1]

[1] taken on August 17th 2016 from `http://www.ihned.cz`

Získat programátory z Ukrajiny je snazší, stačilo málo. Česku ale konkurují západní země i Moskva.

The named entity recognition system retrieves these named entities:[2]

*Získat programátory z **Ukrajiny** je snazší, stačilo málo. **Česku** ale konkurují západní země i **Moskva**.*

Named entity recognition is an often solved task of natural language processing as a preprocessing step of more complex tasks such as automatic machine translation, information retrieval, question answering and others. Since the existence of many annotated corpora and shared tasks, such as MUC7 (Chinchor, 1998) or CoNLL-2003 (Tjong Kim Sang and De Meulder, 2003), named entity recognition has become a standard task in NLP.

The input for the recognizer is usually an unprocessed and unstructured text. Sometimes, morphological analysis or part-of-speech tagging (POS tagging) is available for the NE recognizer, or they are automatically performed by the recognizer. Many recognizers utilize language independent ways of language preprocessing such as stemming (Konkol and Konopík, 2014) or word embeddings (Chapter 7 of this thesis).

A typical domain for named entity recognition is retrieval of names of people, locations and organizations in newspaper articles. However, named entity recognition obviously involves recognition of any prominent or important sequences, such as genes (genomes) in medicine and biology, or chemical entities.

Recently, named entities are also linked with their disambiguated encyclopedic entries. After the task of named entity recognition (that is, after named entity identification and classification) is solved, the task of *named entity linking* is to link all *entity mentions* (such as "Václav Havel" and later "Havel") with their disambiguated encyclopedic entry.

The complexity of the particular named entity recognition task depends on morphosyntactic characteristic of the language,[3] the amount and quality of supervised data available and obviously, on the number and hierarchy of the named entity classes. For example, in shared task CoNLL-2003 (Tjong Kim Sang and De Meulder, 2003), four classes are predicted: PER (person), ORG (organization), LOC (location) and MISC (miscellaneous), while in Czech Named Entity Corpus, tens of classes are annotated and therefore must be identified and classified by the named entity recognition system, including time expressions and

[2]For more named entity recognition examples, we also recommend an online NameTag demo at `http://ufal.mff.cuni.cz/nametag`.

[3]We use the term "morphosyntactic" very freely, in a general meaning without a specific linguistic theory in mind.

others. Generally, the annotators of a named entity corpus have to make a design choice concerning the distribution and number of the selected named entity classes, moving on a continuous scale: one extreme with a few coarse named entity classes and the other with a very fine-grained hierarchy. We touch the question of named entity classes hierarchy design in Chapter 3.

2.2 Named Entity Recognition Methodology

2.2.1 Machine Learning

The typical methodology of named entity recognition involves supervised or semi-supervised machine learning. *Machine learning* is a field of artificial intelligence, which develops and describes techniques allowing computers to automatically perform certain tasks, such as suggest a restaurant based on the previous eating habits, play a selected song based on the listener's preferences, recognize faces in images, or, in our case, recognize named entities in text. The task and its solution are usually described in the means of statistical mathematic model. In *supervised* machine learning, the machine is presented with large amount of correctly solved examples (*training data*) and a programmed way to learn how to infer solutions for new, yet unseen examples (*testing data*). Unlike humans, who are able to infer correct solutions based on a few prototypes of the problem, an automatic process with statistical approach needs a large amount of correctly solved examples (*annotated data*) to create a statistical model and then infer the new examples correctly. In named entity recognition, this purpose is achieved with a large amount of annotated data (*corpora*) where correct named entities are identified and classified by human annotators. *Semi-supervised* machine learning seeks to develop ways how to improve the system with smaller need for these annotated, labeled data and to make use of larger amount of easily available unlabeled data. For more information about statistical language modeling, we refer to standard textbook Manning and Schütze (1999). For more information about machine learning, see Mitchell (1997).

Currently, the state-of-the-art systems for named entity recognition are based on a combination of a supervised machine learning (learning from available annotated corpora) and semi-supervised machine learning (e.g. clustering and word embeddings). The state-of-the-art solutions include, but are not limited to, conditional random fields (CRFs, Lafferty et al. (2001), used for example in Konkol and Konopík (2013)), maximum entropy recognizers (used for example

in Straková et al. (2013)) and recently, excellent results have been achieved with artificial neural networks (Lample et al., 2016). More related work can be found in Sections 2.4 and 2.5.

2.2.2 Named Entity Span Encoding

Because named entity recognition is a task defined on linear sequences and because there are multi-word named entities, the classifier needs to correctly identify the named entity span. For this purpose, an encoding which denotes the named entity beginning and last token is needed. The same problem arises in data, because named entity chunks need to be marked in the selected data format. In CoNLL shared tasks, named entity beginning (B), inside (I) and outside (O) is marked, resulting in the well-known and often used BIO scheme. Another scheme is called BILOU: the classifier is learned to identify the beginning (B), inside (I), last (L), outside (O) and unit-length tokens (U). The question of named entity chunks encoding is not an academic one - the selected encoding impacts the final system performance. An interesting comparison of the BIO and BILOU encoding scheme is given in Ratinov and Roth (2009), as well as in Konkol and Konopík (2015). Following Ratinov and Roth (2009) and our own preliminary experiments, we use a modified BILOU encoding in all our solutions. For a detailed description of our encoding, see Section 5.2 in Chapter 5.

2.2.3 Classification Features

Most recognizers utilize hand-crafted rules or regular expressions and/or manually or automatically created lists of named entities (so-called *gazetteers*). These kinds of dictionary-based solutions usually yield systems with high precision but do not generalize well. Hand-crafted rules and regular expressions are therefore usually involved in supervised machine learning systems as classification features, or if high precision is required, the dictionary search can be performed as a post-processing step.

In this place, we describe a standard set of classification features for the NE task and we devote Section 2.2.6 to gazetteers.

A standard set of classification features for the NE task includes a broad variety of the following:

- current surface word form,
- word lemma (a morphologically motivated base or root form of the word,

usually an output of a morphological analysis. In a broader sense it can be considered a label of the word in a word lexicon.),

- word stem (an automatic reduction of inflected and derived word form to a base form, which necessarily does not have to be a morphologic lemma as in previous item and it even does not have to be a valid word form),
- part-of-speech tags (POS tags),
- the previous items applied on a window of a predefined number of preceding and following words,
- rule-based orthographic features describing the current or surrounding word forms, such as word capitalization, first character capitalization, information about special characters such as hyphens, dots, question marks, etc.,
- various regular expressions designed to reveal e-mail addresses, numbers, phone numbers, days, months and years, depending on the task. For example, time expression can be matched using the following regular expression `([01]?[0-9]|2[0-3])[.:][0-5][0-9]([ap]m)`,
- prefixes and suffixes of the current or surrounding surface word forms,
- capitalization patterns in current and surrounding word forms.

A simple supervised machine learning system of one's choice (such as maximum entropy classifier) armed with a reasonable amount or combination of the classification features listed above already yields a moderately successful NE recognition system. A drawback of these features is that they are human engineered, need to be designed and programmed by hand, require the knowledge of the language, and require extensive feature engineering (*feature engineering* is a term describing the process of empirical identification of the most successful set of classification features by iterative benchmarking).

2.2.4 Non-local Features

The idea behind these techniques is that the same word appearing repeatedly in the text should very probably be classified with the same label and is analogous to "one sense per discourse" (Gale et al., 1992).[4]

[4]Many thanks to Zdeněk Žabokrtský, who commented on this parallel.

- *Context aggregation* (Ratinov and Roth, 2009): The hypothesis is that contexts can be exploited from repeated occurrences of the same token in a document, thus reducing data sparsity. For each token, the classification features of all its occurrences in a fixed window are aggregated.

- *Extended prediction history* (Ratinov and Roth, 2009): An obvious observation is that when a prediction of a current token is made, the preceding occurrences of the same token have already been classified and one can use the classification of previous occurrences of the same token as classification feature for the current prediction.

- *Two-stage prediction* (Krishnan and Manning, 2006): Two-stage (or iterative) prediction is a familiar concept not only in NER. The idea is to iteratively solve the task with serially connected classifiers, and use the predictions made in the previous step as classification features in the next step. Sometimes, different classifiers are even used for the consecutive steps.

Please note that context aggregation and extended prediction history are not applicable for the CNEC (Chapter 3) because the sentences are isolated in the current form of the corpus. To obtain more context for each named entity occurrence, one would need to reconstruct the sentence context from the original corpora (Czech National Corpus[5]).

2.2.5 Unlabeled Data and Semi-Supervised Learning

As noted above, the production of human-annotated data for supervised machine learning is expensive. Therefore, many techniques were proposed, which make use of large unlabeled data, which are easy and relatively cheap to obtain. A huge amount of unstructured, unlabeled text can be used for semi-supervised learning or as a source for automatically retrieved gazetteers (see Section 2.2.6).

One of the typical usage of large unlabeled corpora is *word clustering*. Clustering is a mapping from a given vocabulary (e.g.all words in a natural language, for example English) to a radically smaller set of groups, so called *clusters*, with or without semantic motivation. For instance, a trivial example of clustering are clusters like *vehicles* (car, bus, train), *fruits* (apple, banana), etc. The clusters may also have only a mathematical, abstract meaning, without an obvious linguistic interpretation.

[5] `http://ucnk.ff.cuni.cz`

Brown clustering (Brown et al., 1992) groups words into hierarchical categories based on their contexts in unstructured texts. The idea is that words in similar contexts share also similar characteristics which is the same principle as in language modeling (see Manning and Schütze (1999)). From the NER perspective, it is important that the Brown clustering represents useful and easy to use classification features in NE classifier and is used as such (Ratinov and Roth, 2009; Straková et al., 2013).

Recently, a successful proposal for scalable clustering of tens of millions of phrases was proposed by Lin and Wu (2009).

Word embeddings (Bengio et al., 2003; Mikolov et al., 2013) represent a recent, amazingly efficient way of clustering and because they form a basis of our semi-supervised, featureless NER solution described in Chapter 7, we devote a special Section 7.2 to them.

Another interesting approach is exploitation of Wikipedia, the free encyclopedia.[6] Wikipedia is enormously interesting for NE research because it represents a large amount of data which is already partially annotated or at least easy to parse. Most of the entries describe a named entity (a city, a person, etc.) and in most of the entries, the named entity appears at the beginning of the entry (for example, *"**Prague** is the capital and largest city of the Czech Republic."*). Out of many publications dealing with Wikipedia as a source for NER, we name a recent work of Nothman et al. (2013, 2008).

The previous examples used the unlabeled data to produce gazetteers or clusters as classification features for a supervised classifier. Ando and Zhang (2005) propose one semi-supervised and one unsupervised method to learn from unlabeled data. The hypothesis is that in chunking predictive problems, there exists a shared, common predictive structure and the intuition behind the proposed approach is that by learning on large number (thousands) of auxiliary problems automatically generated from the unlabeled data, the system discovers this shared common predictor.

2.2.6 Gazetteers

Originally, a gazetteer is a list of geographical places. In named entity recognition, gazetteer is a list of any interesting group of named entities, such as names of famous people, list of cities, list of organizations, etc. Gazetteers are either collected manually, which is time-consuming, or automatically, and they are used

[6] `http://www.wikipedia.org`

in a dictionary-based named entity recognition method (that is, named entity is classified as named entity based solely on its existence in a list of gazetteers) or as a classification feature in machine learning. For an automatic, machine collection of gazetteers, Wikipedia is often used (Kazama and Torisawa, 2007).

2.3 NER Evaluation

The performance of automatic named entity recognition system is evaluated with the following mathematical measures:

Precision is the number of correctly identified (and classified, if required by the task) named entities, averaged with the number of named entities retrieved by the system. All tokens of the entity must be identified and correctly classified as marked in gold data (that is, in data annotated by human annotators).

$$precision = \frac{|gold \cap retrieved|}{|retrieved|}$$

Recall is the number of correctly identified (and classified) named entities, divided by the number of correct (gold) named entities.

$$recall = \frac{|gold \cap retrieved|}{|gold|}$$

Recall and precision are combined with harmonic mean into *F1-measure* (or *F-measure*, sometimes also *F-score*, with or without hyphen):

$$F\text{-}measure = 2 \cdot \frac{recall \cdot precision}{recall + precision}$$

Please note, that in named entity recognition evaluation, *accuracy* (percentage of words correctly recognized by the recognizer) does not describe the performance of the system. Since a vast majority of tokens in the text are not named entities and are usually correctly recognized as such, the accuracy easily achieves over $90 - 95\%$ or more. Therefore, a simple token accuracy is not considered an sufficiently informative performance measure in named entity recognition and F-measure is used instead (see paragraph above).

If nested named entities are considered, named entity recognition evaluation becomes more complex. In most datasets, flat named entities are assumed, however, this is not the case in the Czech Named Entity Corpus, which allows *nested named entities*. Named entities are nested, if one named entity is annotated within another one, such as in the following example: In the sequence *"The Bank of*

England", only the outermost named entity (*"The Bank of England"*) or the innermost named entity (*"England"*) can be recognized. If both named entities are annotated, then *"England"* is embedded inside the span of *"The Bank of England"* and the named entities are nested. To our knowledge, very little literature addresses nested entities (Finkel and Manning, 2009). In general, we can see two approaches in nested NER evaluation: either the top-level entities are evaluated or all entities are evaluated, putatively more difficult task as all named entity levels must be correctly identified and classified. The latter applies to Kravalová and Žabokrtský (2009); Straková et al. (2013) and the CNEC corpus. Nevertheless, all our NE recognizers Kravalová and Žabokrtský (2009); Straková et al. (2013, 2016) do not recognize nested named entities and deal only with top-level named entities for technological reasons.[7]

2.4 Related Work in Czech NER

Czech named entity recognition (NER) has become a well-established field since the publication of the Czech Named Entity Corpus[8] (Ševčíková et al., 2007a; Kravalová and Žabokrtský, 2009). Because Czech Named Entity Corpus (CNEC) has had a large influence on the development of the Czech NER, we devote a separate Chapter 3 to its thorough description and we also recommend both publications Ševčíková et al. (2007a); Kravalová and Žabokrtský (2009) and a technical report Ševčíková et al. (2007b) for detailed information about the corpus annotation process.

Following the publication of the Czech Named Entity Corpus in 2007, a variety of named entity recognizers for Czech has been published: Ševčíková et al. (2007b,a); Kravalová and Žabokrtský (2009); Straková et al. (2013); Král (2011); Konkol and Konopík (2011, 2013, 2014); Konkol et al. (2015); Konkol and Konopík (2015); Konkol (2015); Straková et al. (2016) and there is even a publicly available Czech named entity recognition software (NameTag[9], Straková et al. (2014)), which is also described in Chapter 6 of this thesis.

In the following paragraphs, we introduce the reader in the Czech named entity related literature.

As already mentioned above, Ševčíková et al. (2007a) and Ševčíková et al. (2007b) describe the annotation and publication of the CNEC corpus in its

[7] And are therefore penalized for each missed nested entity.

[8] `http://ufal.mff.cuni.cz/cnec`

[9] `http://ufal.mff.cuni.cz/nametag`

first version, and a simple decision tree based NE recognizer is also described in Ševčíková et al. (2007a).

Král (2011) is devoted to a search for an optimal set of features (also known as feature engineering) with focus on Czech named entity recognition. The classifier used as an underlying NER architecture is CRF (Lafferty et al., 2001).

Konkol and Konopík (2011) present a NER system for Czech based on maximum entropy classifier.

Konkol and Konopík (2013) followed up on CNEC with a new extended corpus, which contains a CoNLL-based version of CNEC with simplified annotation. Unlike the CNEC, which contains two-level hierarchical named entity classification and allows nested entities and more than one label (see Chapter 3), Konkol and Konopík (2013) published a Czech Named Entity Corpus Extension in CoNLL format[10] with one-level hierarchy containing 7 coarser classes of named entities, which are non-embedded and non-ambiguous. Furthermore, Konkol and Konopík (2013) present a CRF-based named entity recognizer with a standard set of classification features, such as bag of words, orthographic features, orthographic patterns and gazetteers. An evaluation of our work on the extended corpus of Konkol and Konopík (2013) can be found in Table 7.1.

Konkol and Konopík (2014) thoroughly examined various stemming approaches and their effect to Czech NER. The benchmarks employ classical approaches to stemming, an in-house high-precision stemmer, Majka (Šmerk, 2009), PDT 2.0 lemmatizer (Hajič et al., 2006), MorphoDiTa (Straková et al., 2014) and others. The authors evaluated these benchmarks on a simple baseline CRF NER recognizer, with a final result being that linguistically motivated stemmers yield better results than language agnostic stemmers. In other words, linguistic complexity (or know-how) built into these tools is advantageous.

Konkol et al. (2015) inspected various clustering techniques in NER. The clusters were used as classification features in NER system and proved very beneficial for the recognizer.

An interesting parallel of our work can be seen in Konkol and Konopík (2011) and Konkol et al. (2015). In Konkol and Konopík (2011), word similarity in semantic spaces are utilized, and Konkol et al. (2015) experiments with clustering techniques as classification features in NER. In Chapter 7, we use word

[10]CoNLL, or CoNLL-like format is derived from a well-known column format used in CoNLL shared tasks data. Each line represents one token, special information per token is presented in tab-separated columns and sentences are delimited with empty line.

embeddings (Section 7.2), which can be viewed in this context as semi-supervised, automatically trained real-valued vectors which describe word characteristics in a selected semantic space.

Konkol and Konopík (2015) explore various kinds of segmentation (BIO, BILOU, see Section 2.2.2). We recommend an intriguing statistical t-test comparison presented by Konkol and Konopík (2015) in Tables 2 and 3. Our only minor objection to the selected methodology is that the underlying system used too low baseline. It may happen that a more complex system would show different results. A side note to this topic, a modified BILOU scheme is used in our architectures, see Section 5.3.

Konkol (2015) describe an effort towards Czech named entity linking. Konkol (2015) present a small corpus with disambiguated personal names and explored metrics for string similarity, which were later enveloped by a larger system.

Finally, this section concludes with this thesis author's contribution. The author of this thesis joined the Czech Named Entity Corpus team two years after its publication (Ševčíková et al., 2007a) and cooperated in CNEC public release and maintenance since then.

Kravalová and Žabokrtský (2009) present both the CNEC and a SVM-based NE recognizer. Although the SVM-based NE recognizer is the work of the thesis author, we do not include Kravalová and Žabokrtský (2009) as a Chapter in this thesis, as we consider it outperformed by our own work (Straková et al., 2013, 2016).

A new state-of-the-art Czech NER recognizer was published in 2013 (Straková et al. (2013), Chapter 5). This research became a basis for an open-source Czech NER software NameTag[11] (Straková et al. (2014),Chapter 6).

In Straková et al. (2016), we proposed a new artificial neural network based named entity recognizer. All the above mentioned solutions use manually selected rule-based orthographic classification features, such as first character capitalization, presence of special characters in the word or regular expressions designed to reveal particular named entity types. Also gazetteers are extensively utilized. The major contribution of our recent work is that we proposed artificial neural networks with parametric rectified linear units, word embeddings and character-level word embeddings, which do not need manually designed classification features or gazetteers, and still surpass the current state of the art. This work can be found in Chapter 7 of this thesis or in Straková et al. (2016).

[11] `http://ufal.mff.cuni.cz/namegag`

2.5 Related Work in English NER

This section presents an overview, albeit a brief one, of related research in English NER. It does not seek to bring together a complete list of all English NER-related publications, it rather presents either publications which influenced the thesis author in her research.

As usual in most NLP tasks, English has became the most researched language in named entity recognition. In 1996, a shared task in series of Message Understanding Conferences[12] (MUC-6, Grishman and Sundheim (1996) involved named entity recognition and the term "named entity" (NE) was introduced. After the series of MUC, processing of named entities became a well established discipline within the NLP domain, usually motivated by the needs of information extraction, question answering, or machine translation.

Since then, many datasets and shared tasks were created for English: CoNLL-2002 and CoNLL-2003 (Tjong Kim Sang and De Meulder, 2003), MUC7 (Chinchor, 1998) and many more. We refer the kind reader to one of many existing comprehensive lists on the Internet.[13]

One of the most used and standard comparison dataset is the CoNLL-2003 (Tjong Kim Sang and De Meulder, 2003). In this dataset, four classes are predicted: PER (person), LOC (location), ORG (organization) and MISC (miscellaneous). It is assumed that the entities are non-embedded, non-overlapping and annotated with exactly one label.

For English, one can find literature about attempts at rule-based solutions for the NE task as well as machine-learning approaches, be they dependent on the existence of labeled data (such as CoNLL-2003 shared task data, Tjong Kim Sang and De Meulder (2003)), unsupervised (using redundancy in NE expressions and their contexts, see e.g. Collins and Singer (1999)) or a combination of both (such as Talukdar et al. (2006)).

However, a typical state-of-the-art solution involves both supervised and semi-supervised machine learning methods. One of the most influential publications of the recent years in the NER research is Ratinov and Roth (2009).[14]

Ratinov and Roth (2009) represent the English state-of-the-art with 90.80 F-measure on the CoNLL-2003 data. The authors compare and describe a broad variety of techniques BIO vs. BILOU encoding (see Ratinov and Roth (2009), Section 2.2.2), non-local features, context aggregation, two-stage prediction, ex-

[12] `http://http://cs.nyu.edu/faculty/grishman/muc6.html`

[13] such as `http://www.cs.technion.ac.il/~gabr/resources/data/nei_datasets.html`

[14] Which reached over 550 citations between 2009-2016, according to Google Scholar.

tended prediction history and semi-supervised techniques (exploitation of large unlabeled corpora, clustering, see Section 2.2.5).

Lin and Wu (2009) present a simple and scalable algorithm for clustering tens of millions of phrases and use the resulting clusters as features in discriminative classifiers (cited from (Lin and Wu, 2009)). The proposed method is evaluated also on the CoNLL-2003 shared task with the resulting F-measure 90.90.

The English NER state of the art Ratinov and Roth (2009) has recently been pushed forward by Chiu and Nichols (2015) and Lample et al. (2016). Most of the recent advances have been reached with NER systems based on artificial neural networks. We define and explain the neural network terminology, as well as describe the related artificial neural network based literature in the related Chapters 4 and 7. In this place, we mention the achievements only briefly:

Chiu and Nichols (2015) employ artificial neural networks with bidirectional LSTMs eliminating the need for feature engineering.

The most similar to system proposed by us in Straková et al. (2016) (Chapter 7) is Lample et al. (2016) with 90.94 F-measure.

We also recommend a recent intriguing work of Yang et al. (2016), based on deep hierarchical recurrent neural network for sequential tagging, which is feature engineering free and reaches 91.2 F-measure for the CoNLL-2003 shared task.

Finally, we would like to mention a recent successful proposal for joint solution of named entity recognition and named entity linking. Luo et al. (2015) presents a graph-based framework for joint learning of these two tasks, thus identifying their common properties.

Chapter 3

Czech Named Entity Corpus

We describe the Czech Named Entity Corpus (CNEC) in this chapter. The initial work on CNEC was published in the technical report Ševčíková et al. (2007b) and in Ševčíková et al. (2007a). The author joined the NE team in 2009. The Section 3.1 is taken from Kravalová and Žabokrtský (2009) and describes the annotation of CNEC 1.0.

A few years later, in 2014, we carried out an extensive evaluation of the Czech Named Entity Corpus. We focused especially on those design choices which had impact on machine learning systems (learneability) and software solutions (practical reasons). The research was carried out for the forthcoming Ide and Pustejovsky (2017). The Section 3.2 contains the author's contribution to this book (Straková et al., 2017).

3.1 Czech Named Entity Corpus 1.0

For the purpose of supervised machine learning of named entity recognition in Czech, Ševčíková et al. (2007b) annotated a Czech corpus of named entities. This section describes the Czech Named Entity Corpus 1.0 and is taken from Kravalová and Žabokrtský (2009).

3.1.1 Data Selection

For annotation, 6000 sentences were randomly selected from the Czech National Corpus[1] conforming to the query `([word=".*[a-z0-9]"] [word="[A-Z].*"])`.

This query makes the relative frequency of NEs in the selection higher than the corpus average, which makes the subsequent manual annotation much more

[1] `http://ucnk.ff.cuni.cz`

effective, even if it may slightly bias the distribution of NE types and their observed density. The query is trivially motivated by the fact that NEs in Czech (as well as in many other languages) are often marked by capitalization of the first letter. Annotation of NEs in a corpus without such selection would lower the bias, but would be less effective due to the lower density of NE instances in the annotated material.

Naturally, the decision to select only sentences with biased, unnaturally high density of named entities, was later revisited. We shall discuss this design choice later in Section 3.2 and in Conclusions (Chapter 8), but we reveal in advance, that the biased corpus did not impact the performance of our NE recognizers. (Obviously, the F-measure had to be measured in other domain than in the same, likewise biased testing data.) However, the major drawback of this decision is that the corpus consists of isolated sentences, therefore any discourse classification features cannot be used and the corpus is disqualified from entity linking annotation in its present form.[2]

3.1.2 Two-level NE Hierarchy Annotation

There is no generally accepted typology of named entities. One can see two trends: from the viewpoint of unsupervised learning, it is advantageous to have just a few coarse-grained categories (cf. the NE classification developed for MUC conferences or the classification proposed in Collins and Singer (1999), where only persons, locations, and organizations were distinguished), whereas those interested in semantically oriented applications prefer more informative (finer-grained) categories (e.g. Fleischman and Hovy (2002) with eight types of person labels, or Sekine's Extended NE Hierarchy, cf. Sekine (2003)).

In CNEC 1.0, a two-level NE classification depicted in Figure 3.1 was used. The first level corresponds to coarse categories (called *NE supertypes*) such as person names, geographical names etc. The second level provides a more detailed classification: e.g. within the supertype of geographical names, the *NE types* of names of cities/towns, names of states, names of rivers/seas/lakes etc. are distinguished.[3] If more robust processing is necessary, only the first level (NE supertypes) can be used, while the second level (NE types) comes into play

[2]The sentence context of the isolated CNEC sentences could however be reconstructed from the original corpus (Czech National Corpus, `http://ucnk.ff.cuni.cz`).

[3]Given the size of the annotated data, further subdivision into even finer classes (such as persons divided into categories such as lawyer, politician, scientist used in Fleischman and Hovy (2002)) would result in too sparse annotations.

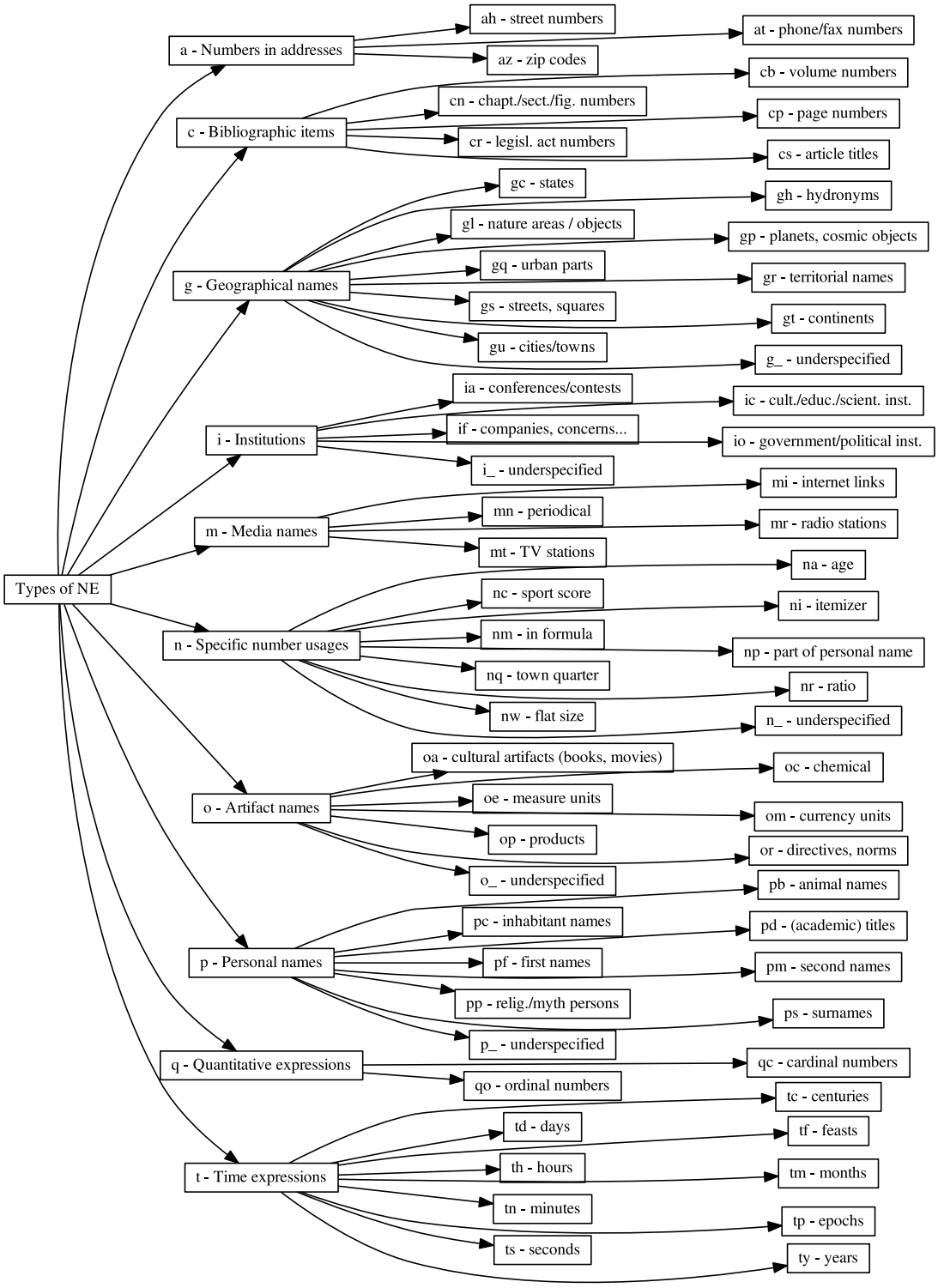

Figure 3.1: Two-level hierarchical classification of NEs used in CNEC 1.0. Note that the (detailed) NE types are divided into two columns just because of the space reasons.

when more subtle information is needed. Each NE type is encoded by a unique two-character tag (e.g., `gu` for names of cities/towns, `gc` for names of states; a special tag, such as `g_`, makes it possible to leave the NE type underspecified).

Besides the terms of NE type and supertype, we use also the term *NE instance*, which stands for a continuous subsequence of tokens expressing the entity in a given text. In the simple plain-text format, which we use for manual annotations, the NE instances are marked as follows: the word or the span of words belonging to the NE is delimited by symbols < and >, with the former one immediately followed by the NE type tag (e.g. <pf *John*> *loves* <pf *Mary*>).

The annotation scheme allows for the embedding of NE instances. There are two types of embedding. In the first case, the NE of a certain type can be embedded in another NE (e.g., the river name can be part of a name of a city as in <gu *Ústí nad* <gh *Labem*> >). In the second case, two or more NEs are parts of a (so-called) *container NE* (e.g., two NEs, a first name and a surname, form together a person name container NE such as in <P<pf *Paul*> <ps *Newman*> >). The container NEs are marked with a capital one-letter tag: `P` for (complex) person names, `T` for temporal expressions, `A` for addresses, and `C` for bibliographic items. A more detailed description of the NE classification can be found in Ševčíková et al. (2007a).

3.1.3 Annotated Data Cleaning

After collecting all the sentences annotated by the annotators, it was necessary to clean the data in order to improve the data quality. For this purpose, a set of tests was implemented. The tests revealed wrong or "suspicious" spots in the data (based e.g. on the assumption that the same lemma should manifest an entity of the same type in most its occurrences), which were manually checked and corrected if necessary. Some noisy sentences caused e.g. by wrong sentence segmentation in the original resource were deleted; the final size of the corpus is 5870 sentences.

3.1.4 Morphological Analysis of Annotated Data

The sentences have been enriched with morphological tags and lemmas using Jan Hajič's tagger shipped with Prague Dependency Treebank 2.0 (Hajič et al., 2006) integrated into the TectoMT environment (Žabokrtský et al., 2008). The motivation for this step was twofold:

- Czech is a morphologically rich language, and named entities might be subject to paradigms with rich inflection too.
- Additional features (useful for any machine learning approach) can be mined from the lemma and tag sequences.

3.1.5 Public Data Release

Manually annotated and cleaned 5870 sentences with roughly 33000 named entities were released as Czech Named Entity Corpus 1.0. The corpus consists of manually annotated sentences and morphological analysis in several formats: a simple plain text format, a simple XML format, a more complex XML format based on the Prague Markup Language (Pajas and Štěpánek, 2006) and containing also the above mentioned morphological analysis, and the html format with visually highlighted NE instances.

For the purposes of supervised machine learning, training, development and evaluation subsets are provided. The division into training, development and evaluation subsets was made by random division of sentences into three sets, in proportion 80% (training), 10% (development) and 10% (evaluation), see Table 3.1. Other basic quantitative properties are summarized in Table 3.2 and Table 3.3.

The resulting data collection, called Czech Named Entity Corpus 1.0, is now publicly available on the Internet at `http://ufal.mff.cuni.cz/cnec`.

Set	#Sentences	#Words	#NE instances
train	4696	119921	26491
dtest	587	14982	3476
etest	587	15119	3615
total	5870	150022	33582

Table 3.1: Division of the corpus into training, development and evaluation sets.

Length	#Occurrences	Proportion
one-word	23057	68.66 %
two-word	6885	20.50 %
three-word	1961	5.84 %
longer	1679	5.00 %
total	33582	100.00 %

Table 3.2: Occurrences of NE instances of different length in the corpus.

NE type	#Occurrences	Proportion
ps	4040	12.03 %
pf	3072	9.15 %
P	2722	8.11 %
gu	2685	8.00 %
qc	2040	6.07 %
oa	1695	5.05 %
ic	1410	4.20 %
ty	1325	3.95 %
th	1325	3.95 %
gc	1107	3.30 %
if	834	2.48 %
io	830	2.47 %
tm	559	1.66 %
n_	512	1.52 %

Table 3.3: Distribution of several most frequent NE types in the annotated corpus.

3.2 Using the Corpus for NE Recognizers

This section describes our experience with Czech Named Entity Corpus 1.0 in development of supervised named entity recognizer NameTag [4] based on Straková et al. (2013).

3.2.1 Corpus Size

The CNEC 1.0 corpus consists of 6,000 sentences with over 150,000 tokens, which is smaller than the CoNLL 2003 shared task corpus (Tjong Kim Sang and De Meulder, 2003) which contains more than 300,000 tokens. While the CoNLL 2003 shared task corpus uses 4 named entity types, the Czech Named Entity Corpus uses 7 supertypes in the first hierarchy level, 42 types in the second hierarchy level, and 4 types of containers. The second annotation round uses 10 supertypes in the first hierarchy level, 62 types in the second hierarchy level and 4 containers. The difference between the first and the second round annotation is that in the second round, the annotation scheme was enriched with number usages. Most Czech NE recognizers are usually trained and evaluated on the first round annotation scheme and do not deal with the numbers annotated in the second annotation round.

[4] `http://ufal.mff.cuni.cz/nametag`

The corpus represents a sufficient and consistently annotated amount of data to make a reliable source for supervised machine learned recognizers. The most interesting classes for NE recognizers using supervised machine learning — person names, surnames, cities and countries — are well represented and learnable. The two named entity recognizers trained on CNEC 1.0 which we used in real applications (Kravalová and Žabokrtský, 2009; Straková et al., 2013) achieved reliable results, generalized well on new data, and their performance received positive ratings from human evaluators.[5]

3.2.2 Classification Granularity and Distribution

The trade-off between the demand for semantically valid and exhaustive classification on one side and technical complexity and annotation costs on the other is usually one of the most interesting questions in the classification design during the annotation process. Obviously, for a particular annotation design, one classification may prove to be either too uninformative or too fine-grained.

One can see that, as the number of entity types in the annotation scheme increases within a corpus of a fixed size, the frequency of marginal classes may become too small to be learnable by supervised machine learning. On the other hand, some linguistic phenomena may be better captured and therefore better learnable in a more detailed classification because these phenomena then appear in distinctive, recognizable contexts.

We carried out a number of experiments to evaluate the impact of classification granularity as well as the distribution of types and supertypes in the corpus. Tables 3.4 and 3.5 illustrate the impact of classification granularity and entity types distribution on the supervised machine learning system developed by Straková et al. (2013). Table 3.4 presents a direct comparison of classification granularity. Three granularity levels are evaluated: 7 supertypes, 42 types, and a mapping to 4 CoNLL classes (PER, LOC, ORG, MISC). In order to ensure a fair comparison between classification granularity levels, the annotation was flattened (only the outer named entities were kept) and containers were ignored. F-measure evaluation for types and supertypes on the original CNEC annotation including embedded entities is shown in Table 3.5.

We presumed that the major challenge for supervised machine learning methods will be posed by the fine-grained, detailed second level entity type classification and expected the difficulty of recognizing automatically a much larger

[5]Licence issues unfortunately do not allow us to support this claim, as this additional evaluation was performed as an in-house project by third-party company.

Classes Trained	Classes Evaluated		
	42	7	4
42	77.52%	82.18%	79.98%
7		54.16%	52.66%
4			52.13%

(a) flattened corpus without containers

Classes Trained	Classes Evaluated		
	42	7	4
42	74.82%	79.38%	75.89%
7		71.90%	71.40%
4			74.46%

(b) flattened corpus without containers except that the container `P` is considered an entity

Table 3.4: Evaluation of Straková et al. (2013) in CNEC 1.0 trained and evaluated with a varying degree of classification granularity. To ensure fair evaluation conditions, the corpus was flattened and F-measure evaluated in a CoNLL-like fashion (outer entities only, no containers).

Data Trained	Data Evaluated			
	original		unbiased	
	types	supertypes	types	supertypes
original	79.01%	82.72%	78.93%	82.63%
unbiased	78.72%	82.33%	78.80%	82.42%

Table 3.5: Evaluation of Straková et al. (2013) on two classification levels (supertypes and types) and on original biased CNEC 1.0 and unbiased CNEC 2.0 release. The evaluation F-measure includes embedded entities and containers.

number of named entity types to be reflected in a noticeably lower F-measure. Our concern was, that from the statistical point of view, the CNEC annotation might be too detailed for the intended usage of the corpus as a training data for supervised machine learning. Due to the limited size of the current version of the corpus, some of the detailed classes are heavily underrepresented (see Fig. 3.2).

Surprisingly, it appears that the fine-grained classification is very well captured by the supervised machine learning method used by Straková et al. (2013). Even more, both Table 3.4 and Table 3.5 show that training and predicting fine-grained entity types and outputting only the coarser types leads to a performance gain. We think one of the main findings of the CNEC project is the fact that fine-grained annotation can be beneficial and is worth the while even though it is more expensive in terms of annotation costs.

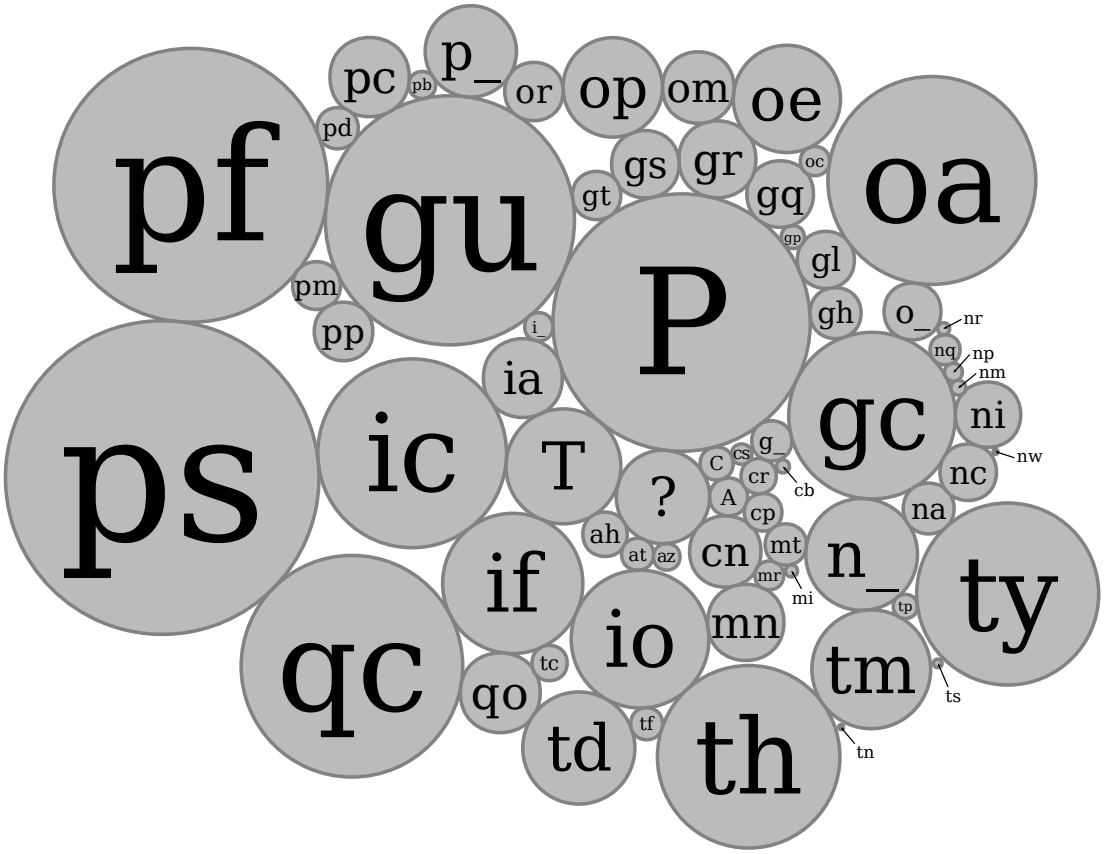

Figure 3.2: Frequency of NE types and containers in Czech Named Entity Corpus 1.0. The area displayed for each NE type is proportional to the number of its occurrences.

There is a striking performance drop in Table 3.4.a in case of first level classification (supertypes) and CoNLL-like mapping (4 classes). After a manual inspection of the prediction results, we hypothesized that this is caused by a rather confusing classification of names on the first hierarchy level. Since first names (`<pf>`) and surnames (`<ps>`) are annotated on the second hierarchy level but are merged into one entity type `<p>` on the first level and in the CoNLL-like hierarchy, it becomes too difficult for the system to recognize personal names' boundaries (`<pf><ps>` becomes `<p><p>`). Once we included one of the containers, `<P>` and marked it as a single personal name entity, we achieved much more satisfying results, which are shown in Table 3.4.b. A conclusion to this experiment is that `<p>` – personal name entity on the first level hierarchy – is causing the first level hierarchy to be uninformative.

3.2.3 Embedding of Named Entities

While the CoNLL-2003 shared task corpus entities are non-embedded, CNEC entities may be embedded and annotated with more than one type. However, most automatic named entity recognizer algorithms are designed to predict non-embedded entities. Therefore most of the above mentioned systems, including those of Kravalová and Žabokrtský (2009) and Straková et al. (2013), have to employ some kind of rule-based post-processing that combines different machine learning methods in order to recognize embedded entities and containers. Straková et al. (2013) report about 2-3% F-measure gain in a container identification post-processing step. To our knowledge, all Czech named entity recognizers ignore entity embedding and allow each token to be part of at most one named entity.

3.2.4 Representativeness and Bias

As described in Section 3.1.1, the Czech Named Entity corpus was created by selecting sentences from the Czech National Corpus matching a certain regular expression. As a result, the corpus is biased towards an unnaturally high frequency of named entities. Interestingly, the biased occurrence of named entities was more of a problem from a theoretical point of view. We were concerned that the named entity recognizer trained on biased data might produce suboptimal results, favorizing recall over precision. This was proven not to be the case: To prevent the bias in the data, a second release of CNEC has been prepared, enriched with a section containing and appropriate number of (almost) named-entity-free sentences, so that the new version of the corpus as a whole reflects the typical NE occurrence frequency in random texts. The NE recognizer of Straková et al. (2013) has been trained and evaluated on both CNEC versions for comparison. Detailed results of this experiment are displayed in Table 3.5. The recognizer performance is almost identical regardless of whether if was trained on CNEC 1.0 or CNEC 2.0, reaching slightly higher results when trained on CNEC 1.0.

3.2.5 Non-local Features

One of the main criticisms against the random selection of isolated sentences in CNEC 1.0 is the fact that such selection makes impossible the utilization of any classification features spanning more than the current sentence. Non-local features, such as context aggregation, are reported to increase named entity recognizers performance substantially (e.g. Ratinov and Roth (2009) report an

Named Entity		Errors	Relative	Named Entity		Errors	Relative
Recognized	Gold		Error	Recognized	Gold		Error
gu	ic	30	15.6	pf	p_	4	1.3
ps	p_	15	3.3	io	ic	4	7.4
ps	oa	15	3.3	gu	oa	4	2.1
pf	oa	12	3.8	gu	io	4	2.1
gc	gu	9	9.2	gu	if	4	2.1
ic	gu	8	6.8	ty	oa	3	2.4
ic	if	6	5.1	ps	op	3	0.7
gu	gq	6	3.1	ps	ia	3	0.7
ps	pm	5	1.1	ps	gu	3	0.7
oa	ic	5	4.9	pf	ia	3	1.0
if	ic	5	5.8	if	op	3	3.5
ps	ic	4	0.9	if	io	3	3.5
ps	gs	4	0.9	ic	oa	3	2.5

Table 3.6: Most frequent errors made by the recognizer of Straková et al. (2013). For every entity type misclassified by the recognizer, we present the number of errors and a relative error, which expresses the ratio of the number of misclassifications to the number of occurrences of the recognized entity.

F-measure gain of 2.97% on CoNLL 2003 test data). As we already pointed out above, one way to retrieve larger context for the corpus sentences can be a reconstruction of the sentence context from the original corpus.[6]

3.2.6 NER Error Analysis

The most frequent errors made by the recognizer of Straková et al. (2013) are presented in Table 3.6. The most common recognizer error is a confusion of cities (gu) with cultural/educational/scientific institutions (ic), whose names often contain a name of a city. A similar problem arises with personal names (pf and ps) on one side and books and movies (oa) on the other. Many of the other frequent misclassifications differ only in the second level of the NE hierarchy – e.g. the system correctly recognizes an institution (first level) but not its exact type (second level).

[6]Czech National Corpus, http://ucnk.ff.cuni.cz

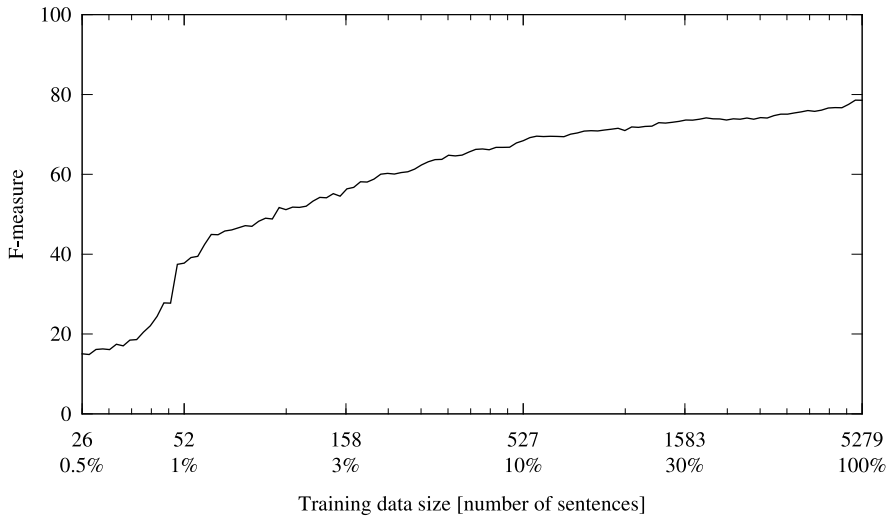

Figure 3.3: Dependence of the recognizer F-measure on the corpus size.

3.2.7 Learning Curves

The effect of the corpus size on the recognizer error rate is displayed in Figure 3.3. As expected, the F-measure increases logarithmically with respect to the increasing corpus size.

3.3 Czech Named Entity Corpus 2.0

After employing the Czech Named Entity Corpus 1.0 in a broad variety of both academic and real world applications, we slightly modified the CNEC 1.0 NE hierarchy: some of the initially proposed types were disregarded, while some type were added in CNEC 2.0. A diagram of named entity type hierarchy proposed in CNEC 2.0 is depicted in Figure 3.4.

The original set of types describing numerical and quantitative entities (`c`, `n` and `q`) proved to be too detailed and in some cases, even difficult for human annotator to distinguish. Therefore, entities of supertype `c` (bibliographic items such as page numbers, volume numbers, figure numbers, etc.) and `q` (quantitative expressions such as cardinal numbers, etc.) were all merged into hierarchy supertype `n` (number expressions).

Heavily underrepresented types were merged into other suitable types, for example `tc` (centuries) and `tp` (epochs) were merged into more general type `no` (ordinal numbers); `tn` (minutes) and `ts` (seconds) were merged into `nc` (cardinal numbers). Similarly, names of planets (`gp`) or animals (`pb`) were merged into more general types.

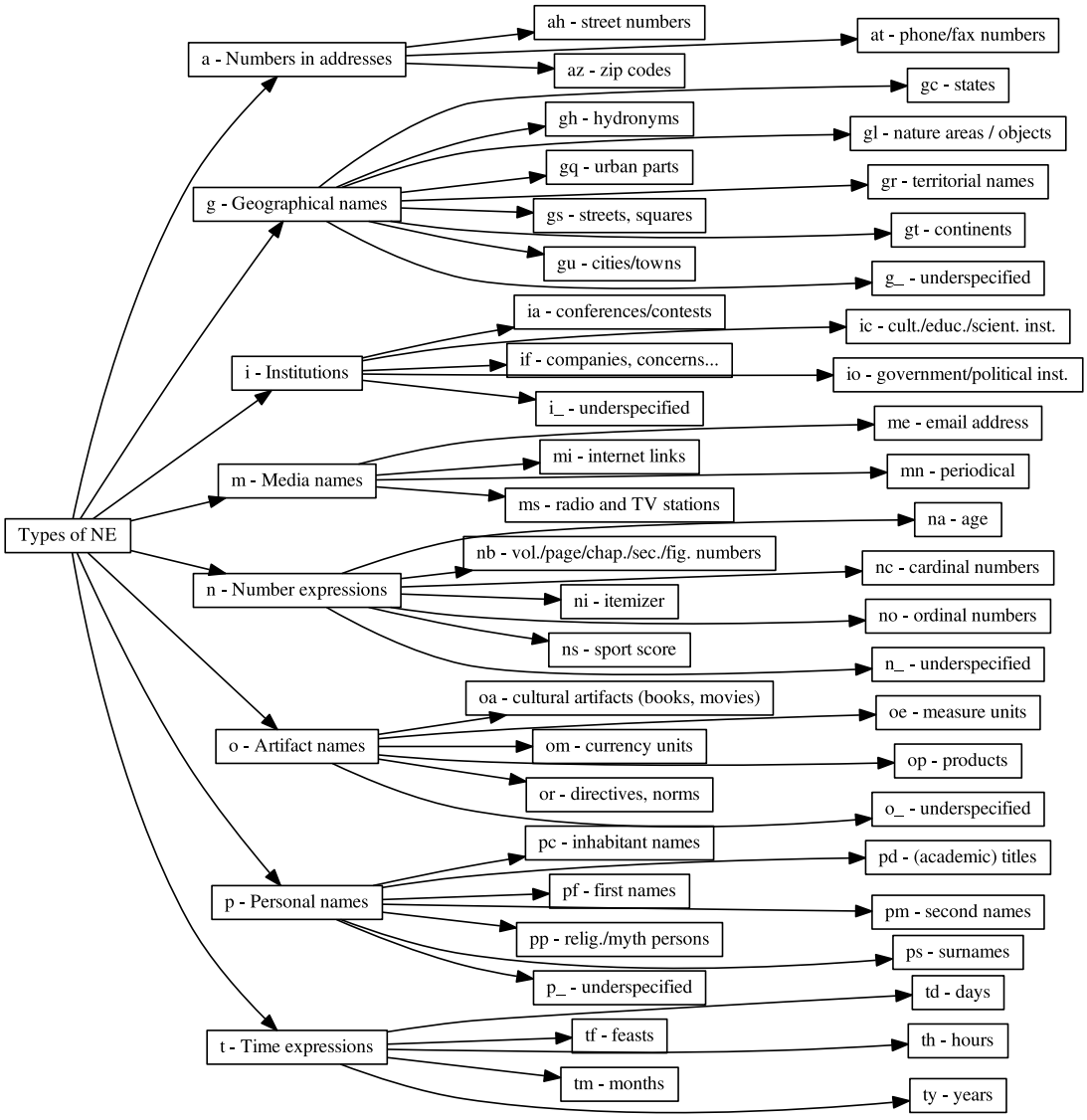

Figure 3.4: Two-level hierarchical classification of NEs proposed in CNEC 2.0.

Finally, a completely new type `me` representing e-mail was introduced.
A complete list of NE type modifications follows below:

- overhaul the number entities
  - entities of supertype c were merged into n; in order to accommodate bibliographic entities a new type nb "vol./page/chap./sec./fig. numbers" was added
    - $cs \rightarrow oa$
    - $cn \rightarrow nb$
    - $cb \rightarrow nb$
    - $cp \rightarrow nb$
    - $cr \rightarrow n\_, or$
  - entities of supetype q were moved into n
    - $qc \rightarrow nc$
    - $qo \rightarrow no$
  - low frequent entities of supertype n were removed and some renamed and merged
    - removed nm, nr, nw
    - nc was renamed to ns
    - $np \rightarrow no$
    - $nq \rightarrow n\_$
  - a few time entities were removed
    - $tc \rightarrow no$
    - $tp \rightarrow no$
    - $tn \rightarrow nc$
    - $ts \rightarrow nc$
- new entity me representing email was added
- gp entity was merged into $g\_$
- mr and mt were merged into ms
- oc entity was merged into $o\_$
- pb entity was merged into $p\_$

Furthermore, to improve the bias introduced by manual data selection during creation of the Czech Named Entity Corpus 1.0, new data was annotated and added: In order to provide annotations for the new *me*, as well as for the *A* container, which is underrepresented in CNEC 1.0, 125 sentences with many addresses and emails were added, 3000 sentences containing only a few named entities were added so that the resulting corpus better represents the density of named entities.

3.4 Conclusions

The Czech Named Entity Corpus 1.0 was the first publicly released named entity corpus for the Czech language, and since 2007, it has stimulated the research on named entities in Czech and has become the reference corpus to measure the progress in Czech named entity recognition progress (see Ševčíková et al. (2007a); Kravalová and Žabokrtský (2009); Konkol and Konopík (2011); Král (2011); Konkol and Konopík (2013); Straková et al. (2013); Konkol and Konopík (2014); Konkol et al. (2015); Konkol and Konopík (2015); Konkol (2015); Straková et al. (2016) or Section 2.4).

In the annotation process and the subsequent evaluation with Czech named entity recognizers, we concluded that fine-grained classification is beneficial and worth the increased annotation costs. We did not experience major negative consequences of the biased sentence selection in the first CNEC release. However, there is a presumable Czech NER performance loss due to the fact that the sentences in CNEC had been randomly selected from a larger sample and are therefore isolated, which makes the utilization of non-local features impossible. Also, owing to the lack of paragraphs or documents, the corpus is disqualified for named entity linking annotation. Nonetheless, we already noted above that a reconstruction of the sentence context from the original corpus[7] is possible.

Following our experience with CNEC 1.0, we released CNEC 2.0 in 2014, with a slightly modified NE hierarchy and an extended number of sentences annotated in the fourth round to achieve a more representative sample.

[7]Czech National Corpus, `http://ucnk.ff.cuni.cz/`

Chapter 4

Efficient Log-linear Modeling and Softmax Neural Networks

This chapter describes our initial experiments with artificial neural networks when we were choosing methods to be later used in our in-house software solutions for NE recognition (NameTag, Straková et al. (2014)) and dependency parsing (Parsito, Straka et al. (2015)). Although this work was never published, we believe it is worthwhile reading and include it in this thesis, because it explains why we choose artificial neural networks with a hidden layer and softmax output function in our systems.

This chapter is somewhat more technical than the previous chapters and assumes the basic knowledge of language modeling (Manning and Schütze, 1999) and artificial neural networks (Rojas, 1996), although the basic terminology is also explained in Sections 4.2,4.3 and 4.4.

The result of this research and our contribution is our own implementation of a robust, scalable and extremely efficient neural network classifier which is being successfully used in our in-house software NLP solutions: NameTag (Straková et al., 2014) and Parsito (Straka et al. (2015), with some modifications to fit the dependency parsing task).

4.1 Introduction

This chapter presents a feed-forward artificial neural network with softmax output function employed as a sequential tagger for large-scale NLP tasks. In a number of experiments in three commonly known NLP tasks, we compare standard log-linear models (for an explanation of the terminology used in this chapter, please read Section 4.2) to feed-forward neural network, and we also compare

several log-linear model optimization methods. We report that on-line stochastic gradient descent optimization proved to be the most robust and scalable optimization method for large NLP tasks compared to batch optimization methods and furthermore, that softmax neural network with hidden layer outperformed the log-linear models in all tasks and languages, both in accuracy and runtime performance.

Our contribution follows two goals:

Firstly, we verify that on-line stochastic gradient descent optimization commonly used in neural networks is better suited for large-scale NLP problems than batch algorithms, achieving higher accuracy and much faster running times than batch algorithms. Among these, especially L-BFGS (Nocedal, 1980; Liu and Nocedal, 1989) has become the method of choice, due to an extensive comparison published by Malouf (2002), mainly because it converges in substantially lower number of steps than iterative scaling (GIS and IIS, Darroch and Ratcliff (1972); Della Pietra et al. (1997); Goodman (2002); Jin et al. (2003)), and it is both fast and efficient. However, it has been already observed that for large-scale and sparse-data problems, which natural language problems often are, stochastic gradient descent tackles the estimation problem even in units of iterations (Zhang, 2004), and might therefore be considered as scalable estimation method.

Secondly, our main contribution is a careful implementation of a hidden layer neural network to capture non-linear hypotheses about the data. The implementation outperforms the log-linear models as expected, both in accuracy and runtime performance.

We performed the evaluation on three problems: Named Entity Recognition CoNLL-2003 shared task, multilingual Semantic Role Labeling CoNLL-2009 shared task and Part of Speech Tagging on Wall Street Journal, utilizing the associated complex measures (F_1, Semantic F_1 Labeled Score and accuracy, respectively).

4.2 Statistical Modeling

We began our explanation of mathematical modeling of the empirical truth in Section 2.2, in which we introduced the concept of *machine learning* by the means of *statistical mathematic modeling*. We described how the automatic system grasps the linguistic knowledge by constructing a statistical mathematical model from the observable truth, that is, from *training data*. Statistical modeling is an integral part of most automatic language processing systems, such as speech recog-

nition, machine translation, question answering, handwriting recognition, and so on. In our case, we aim to construct a statistical model to recognize and predict named entities in text.

Our goal is to define a probabilistic distribution, which would predict the probability $P(Y|X;\theta)$ of an event Y given an observation X, where θ are the model parameters to be estimated.

For example, we wish to predict the type of a named entity $y \in Y$, such as "person", "location", "organization" and "not an entity", for a given text "Václav Havel" in a context "Václav Havel served as the first president of the Czech Republic."

In the following Sections 4.3 and 4.4, we describe two approaches to estimate the model parameters θ in $P(Y|X;\theta)$: log-linear modeling (Section 4.3) and artificial neural networks with softmax output layer (Section 4.4).

4.3 Log-linear Models

The variable X in the model $P(Y|X;\theta)$ is sometimes called the *context*. In classification tasks, such as named entity recognition, it consists of *classification features* $f_i(y,x)$ (see Section 2.2.3), for example information about the word capitalization, its presence in gazetteers, its part-of-speech, etc. We aim to mathematically define the model $P(Y|X;\theta)$ in terms of available classification features $f_i(y,x)$ such that the *likelihood* of the training data is maximized in the model. One possible way of defining $P(Y|X;\theta)$ is to use the *maximum entropy principle* – as proven in Berger et al. (1996), when considering models which keep mean values of f_i, the one with maximum entropy (i.e., the most general one) can be expressed as:

$$P(Y=y|X=x;\theta) = \frac{1}{Z}\exp\Big(\sum_i \theta_i f_i(y,x)\Big) \tag{4.1}$$

where Z is the normalization term:

$$Z = \sum_y \exp\Big(\sum_i \theta_i f_i(y,x)\Big) \tag{4.2}$$

Equation 4.1 defines a *log-linear model*, because if we take logarithms on both sides, then $\log P$ is a linear combination of the weighted classification features. In this sense, the term *log-linear model* will be used throughout this chapter.

4.3.1 Parameter Estimation in Log-Linear Models

For the log-linear model parameter estimation, three wide-spread maximum entropy optimization algorithms are used:

- GIS – *generalized iterative scaling* Darroch and Ratcliff (1972) (and its variant, *improved iterative scaling*, IIS, Della Pietra et al. (1997)), is an early algorithm to estimate log-linear model parameters and it is based on expectation-maximization principle. It is known to be relatively slow, therefore other estimation methods were proposed.

- L-BFGS – *limited memory BFGS* (Nocedal, 1980; Liu and Nocedal, 1989), the abbreviation stands for the authors' names *Broyden-Fletcher-Goldfarb-Shanno*. It is a numerical optimization procedure, a quasi-Newton method, which iteratively converges to the global minimum of the optimized multivariate function in a high-dimensional space which corresponds to model parameters. The mathematical means to navigate in the space of possible updates is based on second derivatives, the *Hessian matrix*. Of this matrix, only a dense approximation needs to be saved, thus yielding a limited-memory variant of the BFGS algorithm.

- *Dual coordinate descent method* is another method of finding a minimum of a function, but it is based on cyclically repeated updates along each dimension, one at a time, thus avoiding the need for derivatives, which are computationally expensive.

An extensive comparison of multiple methods for estimating maximum entropy models is given for example in Malouf (2002). The author concluded that (at that time) widely used iterative scaling algorithms performed poorly in terms of time, and that the limited-memory second-order methods outperformed the other estimation methods. Another comparison is given in Minka (2003). Unlike these comparisons, we do not compare the estimation methods by achieved log-likelihood, but by an overall performance in large-scale practical natural language problems.

4.4 Neural Networks

Artificial neural networks have been used in the NLP field for decades, for tasks such as POS tagging (Schmid, 1994) or parsing (Henderson, 2003; Titov et al., 2009; Socher et al., 2011), as they have many properties with the potential to

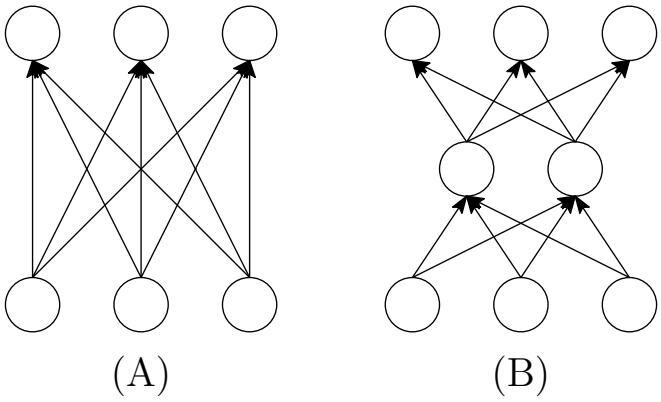

Figure 4.1: Neural network diagram: (A) neural network with direct connections from input neurons to output neurons, (B) neural network with a hidden layer.

make them more powerful compared to standard log-linear models (Section 4.3). Main motivations to use neural networks include their robustness, scalability and especially the ability to infer higher-order features from their inputs if hidden layers are added (e.g., Collobert et al. (2011b)).[1]

An artificial neural network is an interconnected group of nodes, so called *neurons*, which together form a computational model. In Figure 4.1.A, we can see a simple neural network with an input layer and an output layer. Each neuron accepts a real-valued number on input and computes an output using its *activation function*. If two neurons are connected, the output from one neuron is used as an input to the other neuron.

Artificial neural networks are being used for a broad variety of tasks. In the case of named entity recognition, we can use them as predictors for probability of a word sequence being a named entity. The input of the artificial neural network in this case are the real-valued *classification features* (see Section 2.2.3). The output of an artificial neural network for a classification task such as named entity recognition can be one neuron with categorical values ("person", "location", "organization", or "not an entity") or it can be a layer of output neurons, in which each output neuron represents each categorical value ("person", "location", "organization", or "not an entity") and its real-valued activation represents the likelihood of the respective categorical value of the neuron. From this notion, it takes only a step to an artificial neural network with *softmax output layer*.

[1] I wrote this text in 2014. Originally, the text said here: *However, neural networks are still not used widely in the NLP community.* How amusing.

4.4.1 Softmax Neural Network

In this comparison, we propose replacing a standard log-linear model with a feed-forward neural network with *softmax output function*. Because our goal is to estimate probability distributions (Equation 4.1), the output layer of the network uses the softmax activation function (Bridle, 1989, 1990), a commonly known activation function producing probability distributions. If we denote q_j as the sum of inputs of an output neuron j, the output of this neuron is:

$$output_j = \frac{\exp(q_j)}{\sum_{i=1}^{\#outcomes} \exp(q_i)}.$$

It can be verified that a simple feed-forward neural network with softmax output function computes the Equation 4.1. The difference between log-linear model and a neural network is in the numerical algorithm used to approximate the Equation 4.1. The estimation algorithms for the log-linear models can be found in in Section 4.3.1 and for artificial neural network in Section 4.4.3.

4.4.2 Neural Network Architecture

In both proposed neural networks (Figure 4.1), the input neurons directly represent F classification features $f_i(x)$ of the input instance (training event) $x \in X$. The output neurons correspond to values of categorical target variable $y \in Y$ (e.g., part of speech), and together they define the categorical distribution of the target variable y in the current context. Their output values are computed using the softmax activation function (Section 4.4.1).

The first network, displayed in Figure 4.1.A, consists of direct connections between all input neurons and all output neurons, and together with the softmax output function, it yields a direct parallel to a log-linear model (Bridle, 1989, 1990).

The second network, shown in Figure 4.1.B, contains an additional layer of neurons, in addition to the input layer and the output layer. This additional layer, usually called a *hidden layer*, is a standard part of many implementations of the neural network architecture and allows the inference of non-linear features from the input layer.

The activation function in the hidden layer neurons is the logistic function, one of the most frequently used neural network activation functions. The logistic function is a sigmoid function defined by the formula $\sigma(x) = 1/(1 + e^{-x})$.

One of the arguments against the common usage of neural networks is that the addition of a hidden layer slows down training convergence, and therefore,

the additional training cost and increased model complexity is not worth the potential performance gain.

We argue that with careful implementation of neural network optimization methods and in combination with last-decade advances in information technologies, the computational power is sufficient for a comfortable experimental procedure even on a personal computer. Our second argument in favor of higher-order models is a substantial performance gain compared to log-linear models.

4.4.3 Parameter Estimation in Neural Networks

Our goal is to obtain a network that maximizes the likelihood of the training data. Namely, we use negative log likelihood as the loss function which we try to minimize:

$$L(\theta) = -\mathbb{E}_{x,y\sim training\ data} \log P(y|x;\theta) \tag{4.3}$$

$$\theta = \underset{\theta^*}{\operatorname{argmin}}\, L(\theta^*) \tag{4.4}$$

We train the weights of the network using the stochastic gradient descent (SGD). In SGD, the loss function (the negative log likelihood in our case), which is usually an expectation over the whole training data, is approximated by using just one sample (or several samples, called a *minibatch*) from the training data. Then, parameters of the model are updated to lower the loss function, by adjusting their value in a direction opposite to the partial derivative of the loss function:

$$\theta_i \leftarrow \theta_i - \lambda \frac{\partial L(\theta)}{\partial \theta_i} \tag{4.5}$$

The *learning rate* λ controls the size of the update. In our case, we use exponentially decreasing learning rate.

As proven by Bottou (1998), SGD converges to a global optimum if the loss function is convex and a suitable sequence of learning rate factors is used. If the loss function is not convex, SGD converges at least to local optimum (again, if a suitable sequence of learning rate factors is used).

To reduce overfitting, we use Gaussian prior (Chen and Rosenfeld, 1999), which corresponds to L2-regularization term. The SGD update rule then becomes:

$$\theta_i \leftarrow \theta_i - \lambda \frac{\partial L(\theta)}{\partial \theta_i} - \sigma^2 \theta_i \tag{4.6}$$

We initialize the weights in the network 4.1.A to zero and the weights in the network 4.1.B randomly according to normal distribution. In each training iteration, we process the training data in random order.

Because of random initialization, the training algorithm produces different weights on different runs. In our experiments, all results are obtained with a fixed seed number to ensure reproducibility between runs, unless stated otherwise.

Note that it is important to represent the input data sparsely, listing non-zero classification features only. This optimization is crucial for performance, because although the input layer has usually hundreds of thousands of classification features, often only several dozens of them are non-zero in NLP tasks. Although this representation is common in log-linear model optimization methods, only a few artificial neural network implementation allow it.

Time and Memory Complexity

We discuss time and memory complexity of the individual nets (4.1.A and 4.1.B) separately. Let F denote the number of unique classification features, C number of unique outcomes, E number of training events, H the size of the hidden layer (number of hidden neurons) and N size of training data, i.e., the sum of classification features present in the training data.

The network with direct connections takes $\mathcal{O}(FC)$ space, and one iteration of training runs in $\mathcal{O}(NC)$ time. The network with hidden layer occupies $\mathcal{O}((F + C)H)$ space and one iteration of training requires $\mathcal{O}((N + EC)H)$ time.

Dealing with Unseen Weights

The size of the network 4.1.A can be much larger than the size of the training data, which happens when the number of both the unique classification features and outcomes is large (that is the case for example in predicate sense classification, see Table 4.1).

In this case, we can employ the following approximation, which is used by many implementations of maximum entropy estimators. We denote a weight corresponding to a feature-outcome pair appearing at least once in the training data as a *seen weight*. The other weights will be called *unseen weights*.

Instead of representing both the seen and unseen weights, we can represent only the seen weights. This reduces the size of the network 4.1.A to $\mathcal{O}(N)$ and its training time to $\mathcal{O}(N + EC)$, which is essentially optimal. However, when all unseen weights are ignored (which is a common practice[2]), the accuracy of the estimator decreases. This behavior is caused by the fact that corresponding feature never contributes to the outcome. (Ignoring the unseen weights is equivalent

[2]Both the Maximum Entropy Modeling Toolkit and OpenNLP ignore unseen weights (see Section 4.5 for references).

Dataset	Training					Devel. Instances	Test Instances
	Instances	Outcomes	Feature Templates	Features	Unique Features		
NER	204,567	11	184	13,711,203	717,239	51,578	46,666
SRL predicate	86,234	2,777	37	3,016,704	991,412	3,084	5,108
SRL roles	2,477,524	54	114	255,028,986	12,363,307	91,395	144,368
POS	912,344	45	18	16,422,192	256,946	131,768	129,654

Table 4.1: English dataset properties of all case studies.

to setting them to zero.) Indeed, setting the value of unseen weights to a small negative constant improves the accuracy as we discuss in Section 4.6.3.

4.5 Case Studies

We have assessed the models in an extrinsic way by incorporating each of the particular implementations into one of the following three tasks: Named Entity Recognition (NER) CoNLL-2003 shared task (Tjong Kim Sang and De Meulder, 2003), Semantic Role Labeling (SRL) CoNLL-2009 shared task (Hajič et al., 2009) and Part-of-Speech (POS) Tagging on the Wall Street Journal part of the Penn Treebank (Marcus et al., 1993). All these three tasks are commonly known and useful natural language processing applications, are non-trivial, and for each of them, a publicly available dataset exists.

We carried out our case studies in multiple languages: German and English for NER; Catalan, Chinese, Czech, English, German and Spanish for SRL;[3] and English for POS Tagging. We therefore evaluated the maximum entropy estimators on 9 datasets. As an illustration, Table 4.1 summarizes quantitative properties of the English datasets in terms of training, development and test instances and also of (unique) features and classification outcomes.

All solutions follow this scheme: The log-linear model is used to estimate the full distribution $P(Y|X;\theta)$, that is, it provides a probability of each classification outcome $y \in Y$ given a context $x \in X$. The estimated probabilities are then globally optimized either via the Viterbi algorithm (Viterbi, 1967) (dynamic programming) for NER and POS Tagging or via Linear Programming for SRL (Che et al., 2009; Punyakanok et al., 2004).

In this place, we take a step aside to point out that we use the term *dynamic programming* and *Viterbi algorithm* rather interchangeably. In a computer science context, a Viterbi algorithm is basically a version of dynamic programming.

[3]The CoNLL 2009 shared task included also Japanese, which is now unavailable due to license limitations.

On the other hand, many computational linguists prefer to denote this kind of decoding as "the Viterbi algorithm".

For the log-linear model estimation, we have used three wide-spread maximum entropy modeling toolkits which implement different optimization algorithms:

- Maximum Entropy Modeling Toolkit for Python and C++ by Zhang Le[4] using L-BFGS,
- OpenNLP Maximum Entropy Package,[5]
- LIBLINEAR Package[6] (Fan et al., 2008) using dual coordinate descent method

and compared them with our own implementation of the suggested neural network log-linear model using stochastic gradient descent (denoted *NN-simple* in Section 4.6).

We have considered several stopping criteria for these methods. Clearly, one common stopping criteria may be suboptimal for a particular task and trainer combination. In search for a stopping criterion that would ensure fair conditions for each implementation, we carried out an adaptive grid search for optimal number of iterations and smoothing parameters for each trainer and for each task and language pair. We performed the optimization experiments on a computational cluster with the overall computing capacity of 600 processor cores. The optimal parameters for each method (that is, number of iterations and Gaussian σ^2) are noted in Appendix A.1.

There was one exception in the case of the OpenNLP library, because it implements the GIS method which would need computational power exceeding our capacity. We thus only selected optimal values for the English language. We also had to set the upper limit of iterations to 500 although more iterations might still produce better evaluation results.

Our second set of experiments assessed the performance of hidden layer network (denoted *NN-hidden layer* in Section 4.6) and compared it with simple neural network (analogous to log-linear model, *NN-simple*). We do not compare our neural network implementation to other neural network toolkits, because we could not find any suitable. We inspected several popular toolkits (Neural Network Toolbox, FANN, NICO Toolkit, Encoq, Neuroph, neuralnet), but only

[4] `http://homepages.inf.ed.ac.uk/lzhang10/maxent_toolkit.html`

[5] `http://opennlp.apache.org/`

[6] `http://www.csie.ntu.edu.tw/~cjlin/liblinear/`

few provided the softmax activation function and none could represent the input sparsely, which would decrease the performance by a factor of several thousands.[7]

In all three tasks, we implemented feature templates used in the respective literature. In Sections 4.5.1,4.5.2 and 4.5.3, we refer to the original authors describing those feature templates. We opted for rather rich feature sets and did not perform any manual feature engineering. The full lists of classification features can be found in Appendix A.2.

In the following Sections 4.5.1,4.5.2 and 4.5.3, we describe the solutions adopted for each of the case studies in detail. The results reflect the situation of state of the art in 2014.

4.5.1 Named Entity Recognition

Our first case study involves named entity recognition. We tested the selected maximum entropy estimators (Sections 4.3.1 and 4.5) on the CoNLL-2003 shared task English and German datasets (Tjong Kim Sang and De Meulder, 2003). In this task, four classes are predicted: PER (person), LOC (location), ORG (organization) and MISC (miscellaneous).[8] It is assumed that the entities are non-embedded, non-overlapping and annotated with exactly one label. We evaluate our system with the publicly available evaluation script `conlleval`.[9]

The winning system of the CoNLL-2003 shared task was Florian et al. (2003) with F1 88.76 for English and 72.41 for German. Current state-of-the-art for English (90.80) was reported by Ratinov and Roth (2009) and for German (78.20) by Faruqui and Padó (2010).

We reproduced most of the methodology from Ratinov and Roth (2009) except for the estimator. Because our goal is to compare standard log-linear model estimation methods such as GIS and L-BFGS, to artificial neural networks with softmax layer, estimated via SGD, we employ these predictors instead of averaged perceptron (Collins, 2002), which was used in Ratinov and Roth (2009). The log-linear or neural network estimator outputs for each sequence of words its probability of being or not being a named entity using a selected set of classification features. The classification feature templates utilize results of morphological analysis, two-stage prediction, word clustering and gazetteers. The list of clas-

[7]This chapter was written in 2014 when we carried out our initial experiments and the situation has changed since then: for example, torch and TensorFlow were released.

[8]In order to handle multi-word named entities, we used a modified "BILOU-style" encoding, in which we use just 11 classes instead of 17 defined by the usual BILOU encoding, see Sections 2.2.2 and 5.2.

[9]`http://www.cnts.ua.ac.be/conll2000/chunking/output.html`

sification features can be found in Appendix A.2. The predicted probability distribution can already be used to recognize named entities (the output of the system would be the categorical value $y \in Y$ which achieves the largest probability), however, this approach considers each sequence of words as an isolated one. A common practice in NLP is to use the probability distribution in a global context, i.e. a sentence. For each sentence, a Viterbi (Manning and Schütze, 1999) trellis is built and the predicted probabilities are used to estimate the trellis transitions. The recognizer uses dynamic programming (Viterbi) to decode an optimal sequence labeling using probabilities estimated either by the log-linear model or an artificial neural network. The selection criterium for the optimal sequence is such that maximizes the joint probability over the path in the trellis.

In this place, we would like to add a somewhat broader note: It is usually very beneficial to design the global optimizer architecture in such a way that the selected optimalization criterium is exactly the same or corresponds closely to the evaluated measure and that the improving update steps directly correspond to improvements of the overall selected score. This is not always straightforward, as the selected measure may be complex. Let's for example mention the structural F-measure for hierarchical named entities (Section 2.3) or the semantic F1 labeled score (Section 4.5.2). A common workaround for this problem is to decode the predicted probabilities globally to maximize the joint probability of the sequence via dynamic programming. An intriguing example of a globalizing approach which optimizes the whole sequence with respect to the NER task is Lample et al. (2016), which uses an artificial neural network combined with CRF layer to decode whole sentences, thus gaining a determining advantage over our framework described in Chapter 7 (Straková et al., 2016).

4.5.2 Semantic Role Labeling

As the second case study, we have chosen the Semantic Role Labeling CoNLL-2009 shared task (Hajič et al., 2009), the closed challenge (without external data). The task is to predict semantic dependencies based on the provided (automatically predicted) morphological and syntactic analysis. The first part of the task is to determine predicate senses, i.e., to perform word sense disambiguation (WSD) on the predicates, followed by the core of the task – finding and labeling semantic dependencies of the predicates in the sentence, known as semantic role labeling (SRL). Both subtasks, predicate classification and semantic role labeling, are

then evaluated with the CoNLL-2009 shared task evaluation script[10] for measuring Semantic Labeled F_1 score. The performance of the system is evaluated with respect to a test set (gold data) by computing the recall, precision and F1 measure (2.3) of matched arcs in the semantic trees. The measure is *labeled*, that means the type of the arc is considered and must be also predicted correctly.

The best participants' semantic F1 labeled scores were 80.32 for Catalan Zhao et al. (2009), 78.60 for Chinese Björkelund et al. (2009), 86.51 for English Che et al. (2009), 79.71 for German Björkelund et al. (2009) and 80.46 for Spanish Zhao et al. (2009).

We reproduced the methodology from Che et al. (2009) and Zhao et al. (2009). For the first part of the task, the predicate sense classification, one large model for all predicate senses is built. For the semantic role labeling, probabilities of each word candidate (to be a predicate modifier) are estimated either with a log-linear model or with an artificial neural network with softmax output layer. The estimated probabilities are then used in a global decoder employing linear programming (LPsolve).[11]

4.5.3 Part of Speech Tagging

The third case study evaluates the classifiers on the part-of-speech tagging task, using the venerable Wall Street Journal corpus which is a part of the Penn Treebank Marcus et al. (1993).[12] The English language was trained on the standard training portion (Sections 0-18) of the Wall Street Journal part of the Penn Treebank, the system was tuned on the development set (Sections 19-21 in PTB/WSJ in English) and tested on the testing section (Sections 22-24 in PTB/WSJ in English). The state-of-the-art accuracy achieved without external data is 97.33 by Shen et al. (2007).[13]

We partially reproduced Spoustová et al. (2009), which is based on averaged perceptron by Collins (2002), except that instead of averaged perceptron, we utilized we used either a maximum entropy model to estimate the posterior probabilities or an artificial neural network with softmax output layer, and decoded the POS tags with the Viterbi algorithm.

[10] `http://ufal.mff.cuni.cz/conll2009-st/scorer.html`

[11] `http://lpsolve.sourceforge.net/`. Our preliminary experiments on English development data show that linear programming decoding adds about 1% to the semantic labeled F_1 measure compared to probability prediction without any (global) decoding.

[12] LDC Catalog No. LDC99T42.

[13] State of the art, as of 2014.

	English	German
Maximum Entropy Toolkit by Zhang Le (L-BFSG)	87.64	75.93
The OpenNLP Maxent (GIS)	87.46	75.49
The LIBLINEAR package (dual coordinate descent)	88.02	76.54
NN-simple (SGD)	**88.46**	**76.57**
NN-hidden layer (SGD)	**88.64**	**77.28**

Table 4.2: F_1 score in Named Entity Recognition CoNLL-2003 shared task.

	Catalan	Chinese	Czech	English	German	Spanish
Maximum Entropy Toolkit by Zhang Le (L-BFGS)	76.92	75.69	85.98	81.90	77.00	78.04
The OpenNLP Maxent (GIS)	64.08†	74.21†	72.24†	79.27†	73.03†	63.98†
The LIBLINEAR package (dual coordinate descent)	77.40	76.70	N/A†	N/A†	77.92	78.94
NN-simple (SGD)	**77.29**	**76.53**	**86.25**	**83.01**	**77.78**	**78.79**
NN-hidden layer (SGD)	**77.47**	**77.42**	**86.75**	**83.82**	**78.79**	**79.41**

Table 4.3: Semantic F_1 labeled score in Semantic Role Labeling CoNLL-2009 shared task. The results marked with † are explained in Section 4.6.1.

4.6 Results and Discussion

The results for Named Entity Recognition task are shown in Table 4.2, for Semantic Recognition task in Table 4.3 and for Part of Speech Tagging task in Table 4.4.

In all cases, the results fall well within the range of the top shared task participant results and are competitive to the state-of-the-art results. We adopted standard classification features from the respective literature without any manual feature engineering. There is, however, some a priori linguistic knowledge included in the fact that these classification features are task-specific and commonly used. The focus of our contribution lies in the comparison of estimation methods and models as they perform on the same, fixed classification feature set.

There appears a similar tendency in our case studies: statistical model trained with stochastic gradient descent (*NN-simple*) achieves consistently similar or better results compared to the log-linear estimators. In a comparison of a linear and non-linear model (*NN-simple* vs. *NN-hidden layer*), higher-order hypothesis achieved by the hidden layer in a neural network (*NN-hidden layer*) substantially adds to the improvements in all our case studies.

	English
Maximum Entropy Toolkit by Zhang Le (L-BFGS)	96.92
The OpenNLP Maxent (GIS)	96.85
The LIBLINEAR package (dual coordinate descent)	96.97
NN-simple (SGD)	**97.00**
NN-hidden layer (SGD)	**97.12**

Table 4.4: Accuracy on Part of Speech Tagging of Wall Street Journal without external data.

4.6.1 Semantic Role Labeling Dataset Issues

The results of the Semantic Role Labeling task, shown in Table 4.3, deserve further explanation. The OpenNLP estimator results noticeably differ from the others and some results of LIBLINEAR estimator are missing. Both these issues are caused by the fact that the SRL predicate sense dataset contains very large number of outcomes (2,777 for English as listed in Table 4.1). Although we could have altered the SRL solution to decrease the number of outcomes, because the potential senses are limited by the given verb lemma, we decided to keep it as a test of the computational robustness of the estimator. Furthermore, training the word senses classification in a joint model overcomes the inevitable data sparsity, should the models be split into one per each verb lemma.

As discussed at the end of Section 4.4.3, the ability to represent only the *seen weights* greatly improves the robustness.

4.6.2 Convergence Rate

Apart from the estimator accuracy, convergence rate can be crucial to the whole experimental procedure. To demonstrate the convergence rate, we measured the development data accuracy in relation to training time.[14] The elapsed time of all implementations was measured on an AMD Opteron CPU with 32 GB memory. The convergence comparison for the Named Entity Recognition task in the English language is presented in Figure 4.2. For every estimator, the plot shows accuracy after several training iterations, always starting with the accuracy after the first one. The exception is the LIBLINEAR package, where various values

[14]We measured development data accuracy (rather than overall F1 measure) because it can be calculated and printed out after each training step from inside the training procedure without the need to save and reload the model and to carry out the decoding.

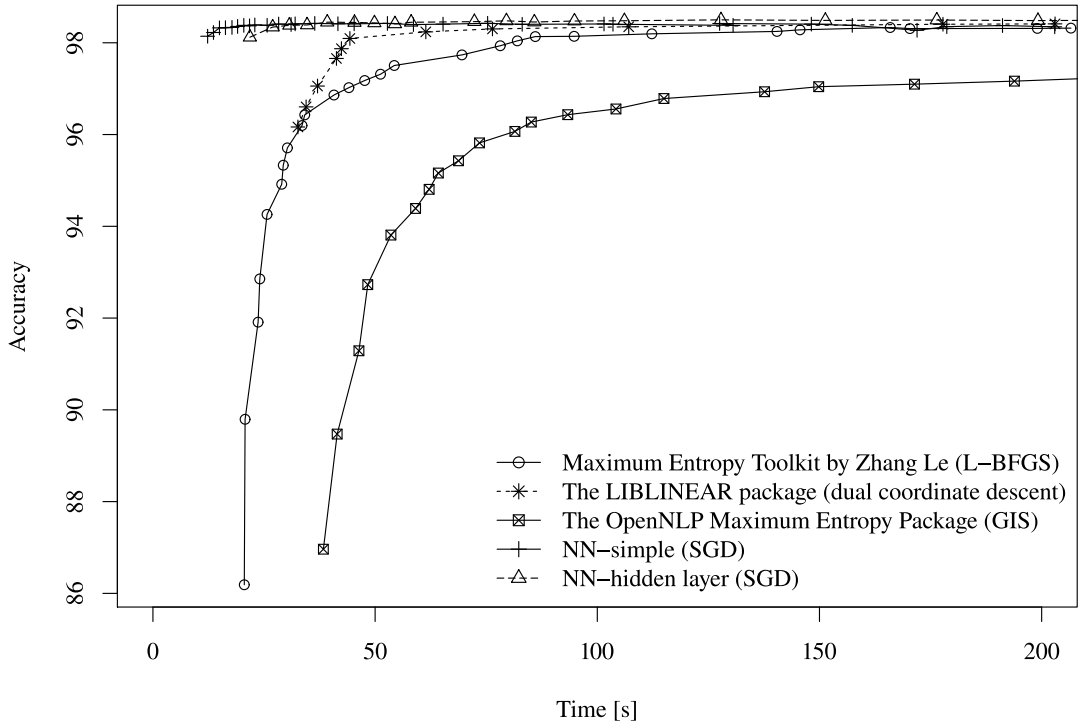

Figure 4.2: Convergence rate including the startup time on English Named Entity Recognition CoNLL-2003 shared task. The graph shows development data accuracy after elapsed user time.

of penalty parameter C were used to obtain results with different accuracy. The convergence results for other data sets were very similar, and we do not present them.

Both variants of our neural-network-based system converge to a competitive accuracy in a very fast rate and in this sense exceed the compared maximum entropy implementations.

Furthermore, the training time of a neural network with hidden layer is almost as fast as that of a simple neural network, with a negligible convergence slowdown.

4.6.3 Effect of Unseen Weight Value

As described in Section 4.4.3, the value of *unseen weights* is important in case only the seen weights are present in the model. Figure 4.3 shows the effect of the value of unseen weights. Interestingly, an estimator with an appropriate setting of the unseen weight value can outperform an estimator representing both the seen and unseen weights as well as an estimator which ignores them.

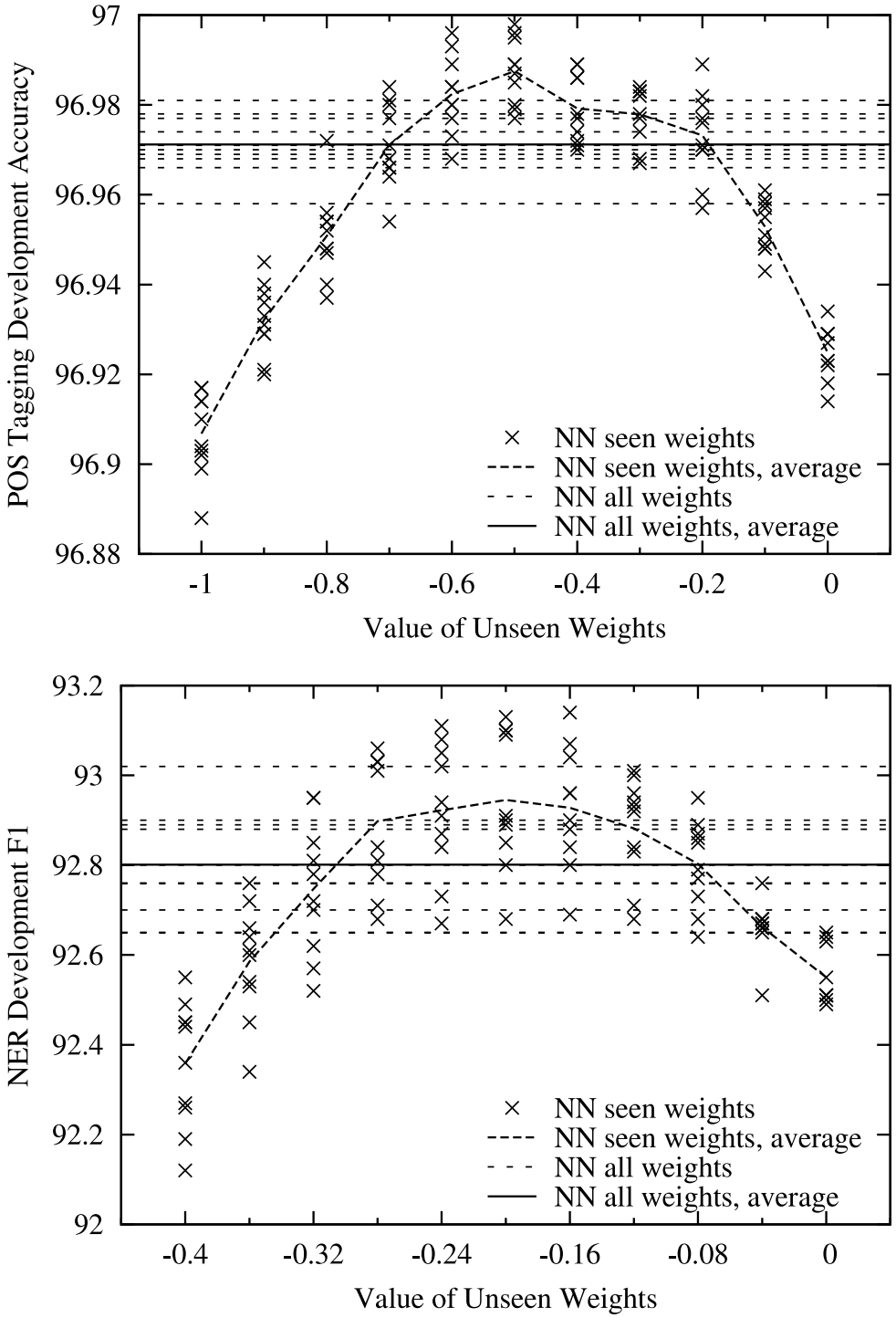

Figure 4.3: The effect of unseen weights on the score of the POS Tagging task (top) and the NER task (bottom). The dotted horizontal lines denote the score achieved with both seen and unseen weights of network 4.1.A (with random seeds), and the solid horizontal line is the average. The points correspond to neural network estimators, which represent only seen weights and the value of unseen weights is set to a constant.

4.7 Conclusions

We have described and thoroughly evaluated our own implementation of an artificial neural network classifier with softmax output layer. Its main advantages are robustness on large datasets, excellent training time and competitive accuracy. The neural network based classifier achieves better performance and faster implementations and has already been successfully used in our in-house NLP software: NameTag (Straková et al., 2014), a named entity recognizer, and Parsito (Straka et al., 2015), a dependency syntax parser.

Chapter 5

Czech Named Entity Recognizer with Softmax Neural Network

In this chapter, we describe a named entity recognizer for the Czech language with a softmax neural network and sentence-level dynamic programming decoding, which was published in Straková et al. (2013) and represented Czech state-of-the-art until Straková et al. (2016), in which it was surpassed by a deeper NN recognizer described in Chapter 7. NameTag[1], a free software for named entity recognition (Straková et al., 2014) is based on these results and is described in Chapter 6. The recognizer reaches 82.82 F-measure on the Czech Named Entity Corpus 1.0 and substantially outperforms previously published Czech named entity recognizers (Ševčíková et al., 2007a; Kravalová and Žabokrtský, 2009; Konkol and Konopík, 2011). On the English CoNLL-2003 shared task, we achieved 89.16 F-measure, reaching comparable results to the English state of the art (Ratinov and Roth, 2009). The recognizer predicts named entity label probability distribution with a softmax neural network and then decodes the optimal sequence labeling with the Viterbi algorithm (dynamic programming). The classification features utilize automatic morphological analysis, two-stage prediction, word clustering and gazetteers (see Chapter 2).

5.1 System Overview

A simple overview of our named entity recognizer is described in Figure 5.1. The system is based on softmax neural network classifier and then a Viterbi decoder globally optimizes the labels for the sequence using the estimated probability distribution on a sentence level.

[1] `http://ufal.mff.cuni.cz/nametag`

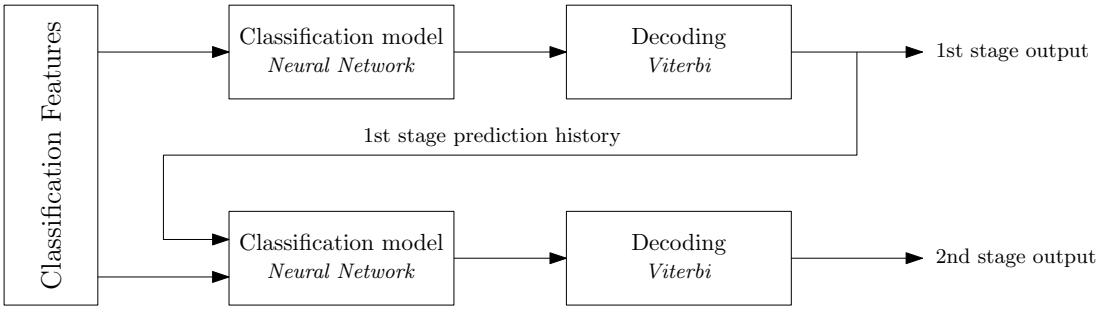

Figure 5.1: System overview

First, the softmax neural network predicts for each word in a sentence the full probability distribution of its classes and positions with respect to an entity (BILOU encoding, see Sections 2.2.2 and 5.2). Consequently, a global optimization via dynamic programming determines the optimal combination of classes and named entities chunks (lengths). This procedure deals with the innermost embedded entities and the system outputs one label per entity. Finally, the Czech system output is post-edited with four rules to add containers (see Section 3.1.2).[2]

The whole pipeline runs two times, utilizing the output from the first stage as additional classification features in the second stage.

5.2 Softmax Neural Network Classifier

In the first step, the softmax neural network task is to predict for each word the named entity type and position within the entity. The positions are described with a BILOU scheme (Ratinov and Roth (2009), see Section 2.2.2): B for multiword entity Beginning, I for Inside multiword entity, L for Last word of multiword entity, U for Unit word entity and O for Outside any entity. This scheme results in a large combination of predicted classes ($4 \times |C| + 1$, where $|C|$ is the number of classes, 4 is for B-X, I-X, L-X, U-X and +1 for O).

Our classifier is our own implementation of an artificial neural network with a softmax output layer without a hidden layer, see Figure 4.1.A in the previous Chapter 4. For the classifier training, we implemented our own stochastic gradient descent parameter estimation (Chapter 4).

Interestingly, in our first-step experiments we used a third-party maximum entropy classifier, Maximum Entropy Modeling Toolkit for Python and C++[3] to

[2]We automatically selected a subset of embedding patterns appearing in the training data by sequential adding the rule that increased F-measure the most. There are no such rules for English because the dataset does not contain embedded entities.

[3]`http://homepages.inf.ed.ac.uk/lzhang10/maxent_toolkit.html`

predict the labels probability distribution for each token. After experiments with numerous third-party toolkits and libraries, we implemented our own robust and efficient neural network classifier as described in Chapter 4.

In this place, we find it important to note that a simple artificial neural network with a softmax output layer and without a hidden layer, such as the one depicted in Figure 4.1.A, is computationally equivalent to maximum entropy classifier. Both models differ only in the training algorithm which optimizes the model parameters (see Chapter 4).

5.3 Decoding

We search for the optimal combination of named entities in the whole sentence with dynamic programming. As the initial values, the probabilities estimated by the artificial neural network with a softmax neural layer are used. As mentioned in Chapter 4, we consider the Viterbi algorithm a kind of dynamic programming and use these algorithm names rather interchangeably. In our implementation of the Viterbi algorithm, we prune the impossible trellis transitions (e.g., once B-X starts, it can be followed either by I-X or L-X). Using this observation we can decode a whole sentence using dynamic programming with $\mathcal{O}(NC)$ complexity, where N is the number of words in the sentence.

Also, we were concerned with large growth of classes predicted by the classifier in the first step. With the full BILOU scheme (see Section 2.2.2), there are 17 classes for English and 169 classes for Czech. With the previous observation, we simplified the BILOU scheme from full B-X, I-X, L-X, U-X and O, to B-X, I, L, U-X and O. With this simplified scheme, the number of predicted classes is nearly halved.

5.4 Classification Features

We use a typical set of classification features for the NE task:

- current word form
- current word lemma,
- current part-of-speech tag
- current chunk (only English)
- the previous items applied to surrounding words in window ± 2

- various orthographic features (capitalization, punctuation, lowercase and uppercase form of the word)
- suffixes and prefixes of length 4
- regular expressions identifying possible year, date and time (in Czech).

Feature selection was performed by sequentially (manually) adding new classification features to the feature set; we retained those features that have improved the classification based on development data. In English, we used forms in most of the classification features, while in Czech, we had to use lemmas because of data sparsity due to the fact that Czech is a morphologically rich language. The full list of classification features set used in our classifier can be found in Appendix A.2.

5.4.1 Two-stage Prediction

We use two-stage prediction, that is, we run our system two times in a row and in the second run, we use the predictions made in the first run. We use the information about the prediction of the previous and following five words and about the previous predictions of the candidate word in the preceding window of 500 words.

5.4.2 Morphology

The Czech Named Entity Corpus is enriched with morphological tags and lemmas (the root version of the word) with Jan Hajič's tagger shipped with Prague Dependency Treebank 2.0 (Hajič et al., 2006).

We however obtained new morphological tags and lemmas with the Featurama tagger,[4], which is more recent and achieves state-of-the-art performance in Czech. In even later versions of our NER recognizer (Straková et al. (2016),Chapter 7), we use our own implementation of morphologic dictionary and tagger, MorphoDiTa (Straková et al., 2014).[5]

Anyway, all these taggers use the Czech morphological system designed by Jan Hajič (Hajič, 2004). The part of speech tags use a positional system, in which each position within a 15-character string carries a certain morphologic information, such as part of speech, detailed part of speech, number, case (linguistic case), etc.

[4] `http://sourceforge.net/projects/featurama/`

[5] `http://ufal.mff.cuni.cz/morphodita`

Lemma type	Explanation, examples
Y	given name (formerly used as default): Petr, John
S	surname, family name: Dvořák, Zelený, Agassi, Bush
E	member of a particular nation, inhabitant of a particular territory
G	geographical name: Praha, Tatry (the mountains)
K	company, organization, institution: Tatra (the company)
R	product: Tatra (the car)
m	other proper name: names of mines, stadiums, guerilla bases, etc.
H	chemistry
U	medicine
L	natural sciences
j	justice
g	technology in general
c	computers and electronics
y	hobby, leisure, travelling
b	economy, finances
u	culture, education, arts, other sciences
w	sports
p	politics, government, military
z	ecology, environment
o	color indication

Table 5.1: Lemma term types as used in Hajič (2004) and Hajič et al. (2006).

From the NER perspective, it is important that in this morphological system, lemmas are manually annotated with labels marking proper names, such as `Y` for given names, `S` for surnames and `G` for geographical names (Hajič (2004), p. 121). These labels act as gazetteers built inside the morphology (see Section 5.4.3). A list of these lemma type descriptions is shown in Table 5.1. One can see that these types can be used as useful classification features, although they do not always directly correspond to the NE hierarchy in the Czech Named Entity Corpus.

In English, we also retagged the data with Featurama tagger. Because the Featurama tool does not support English lemmatization, the English data was lemmatized by algorithm by Popel (2009). We also chunked the English data with TagChunk[6] (Daumé III and Marcu, 2005).

[6] `http://www.umiacs.umd.edu/~hal/TagChunk/`

5.4.3 Gazetteers

Named entity recognizers rely substantially on external knowledge. For English, we used 24 manually collected gazetteers of $1.8M$ items and for the Czech language, we used 17 manually collected gazetteers of $148K$ items. We collected both manually maintained gazetteers and automatically retrieved gazetteers from the English and Czech Wikipedia (Kazama and Torisawa, 2007). We did not parse the whole Wikipedia article content, we only listed the title in gazetteer when it was filed under an appropriate category (e.g. "people", "births", "cities", etc.).

As mentioned above, the Czech lemmas as designed by Hajič (2004) contain manually inserted information about selected proper names.

5.4.4 Brown Clusters

Furthermore, we utilized Brown mutual information bigram clusters (Brown et al., 1992; Liang, 2005), which we trained on Czech Wikipedia and downloaded for English.[7] We added these clusters respective to forms (English) and lemmas (Czech) as new classification features, using cluster prefixes of length 4, 6, 10 and 20 (see Ratinov and Roth (2009)).

5.5 Results and Discussion

We call "baseline" the simplest model in which we used the common set of classification features in our model (see Appendix A.2), then selected the optimal sequention of named entities using the probability distribution given by the classifier with dynamic programming and in Czech, post-edited the result with four automatically discovered rules to retrieve container entities (see Section 5.1).

Table 5.2 shows the effect of more sophisticated classification features or processing: (A) new tagging, lemmatization and chunking, (B) two stage prediction, (C) gazetteers, (D) Brown clusters. The experiments (A), (B), (C) and (D) show the system improvement after adding the respective feature to the baseline. The last line of the table shows results after combining all features. All new features and preprocessing steps improved the system performance over the baseline and the gains were similar in both languages. In the Czech language, most of the impact of adding gazetteers (C) is formed by the manually annotated proper name labels in the morphology (Hajič, 2004; Hajič et al., 2006) while the manually collected and Wikipedia extracted gazetteers did not yield substantial improvement.

[7] `http://people.csail.mit.edu/maestro/papers/bllip-clusters.gz`

	English	Czech
baseline	83.80	74.87
(A) new tags, lemmas and chunks	84.20	75.47
(B) two stage prediction	84.93	76.14
(C) gazetteers	86.20	76.15
(D) Brown clusters	85.88	76.67
all	89.16	79.23

Table 5.2: System development. The experiments (A), (B), (C) and (D) show F-measure gains over the baseline on the test portion of the English and Czech data.

	All NEs			One-word NEs			Two-word NEs		
	P	R	F	P	R	F	P	R	F
Type:	84.46	74.61	79.23	87.70	79.97	83.66	81.85	77.10	79.40
Suptype:	88.27	78.00	82.82	92.07	84.00	87.85	84.12	79.24	81.60
Span:	91.56	82.56	86.83	94.00	87.90	90.85	90.28	86.09	88.13

Table 5.3: Detailed results for the Czech language. The table shows results for one-word, two-word and all named entities. The three measures evaluated are precision (P), recall (R) and F-measure (F).

Table 5.3 shows detailed results with precision, recall and F-measure for Czech one-word, two-word and all named entities for comparison with similar tables published in Ševčíková et al. (2007a) and Kravalová and Žabokrtský (2009). Table 5.4 and Table 5.5 compare the related work for Czech and English on the respective datasets.

5.6 Conclusions

We have presented a new named entity recognizer and evaluated it for Czech and English. We reached 82.82 F-measure for the Czech language and significantly outperformed the existing Czech state of the art at that time. For English, we achieved 89.16 F-measure. The named entity recognizer is available as an open-source software NameTag (Straková et al. (2014),Chapter 6).

Czech	Types	Supertypes
this work	**79.23**	**82.82**
Konkol and Konopík (2011)	NA	72.94
Kravalová and Žabokrtský (2009)	68.00	71.00
Ševčíková et al. (2007a)	62.00	68.00

Table 5.4: System comparison for Czech language (F-measure on test data).

English	F-measure
Ratinov and Roth (2009)	90.80
Suzuki and Isozaki (2008)	89.92
Ando and Zhang (2005)	89.31
this work	**89.16**
Florian et al. (2003)	88.76
Chieu and Ng (2003)	88.31
Finkel et al. (2005), Stanford parser	86.86

Table 5.5: System comparison for English language (F-measure on test data).

Chapter 6

NameTag: An Open-Source Tool for Named Entity Recognition

In this chapter we describe an open-source tagger: NameTag is a free software for named entity recognition which achieves state-of-the-art performance on Czech to our best knowledge. The tagger can be trained with custom data. The tool is released as free software under Mozilla Public License 2.0 and is distributed along with trained linguistic models which are free for non-commercial use under the CC BY-NC-SA license. The release includes standalone tools, C++ libraries with Java, Python, Perl and C# bindings and a web service.

This work was presented at a demo session of ACL 2014 (Straková et al., 2014) along with MorphoDiTa.[1] The original paper describes both NameTag and MorphoDita, we however include only the named entity recognizer release description in this chapter.

The binary, C++ library, Python, Perl, Java and C# bindings, the web service, demo and documentation can be found online at `http://ufal.mff.cuni.cz/nametag`.

6.1 Introduction

Our aim was to implement a NE recognizer which would

- be well suited and trainable for languages with very rich morphology and thus a large tagset of possibly several thousand plausible combinations of morphologically related attribute values,
- provide excellent, preferably state-of-the-art results for Czech,

[1] Morphological Dictionary and Tagger, `http://ufal.mff.cuni.cz/morphodita`

- be distributed along with trained linguistic models for Czech,
- allow the user to train custom models for any language,
- be extremely efficient in terms of RAM and disc usage,
- offer a full end-to-end solution for users with little computational linguistics background,
- be distributed as a library without additional dependencies,
- offer API in different programming languages,
- be open-source, free software.

Following these requirements, we developed a named entity recognizer. The software performance and resource usage are described in Section 6.3 and the release and licensing condition information is given in Section 6.4. We conclude the chapter in Section 6.5.

6.2 NER Methodology

The NE recognizer is an implementation of a research project described in Chapter 5 and in Straková et al. (2013). The recognizer is based on a softmax neural network classifier, described in Chapter 4, which predicts, for each word in a sentence, the probability distribution of its classes and positions with respect to an entity. Consequently, a global optimization via dynamic programming determines the optimal combination of classes and named entities chunks (lengths). The classification features utilize morphological analysis, two-stage prediction, word clustering and gazetteers and are described in Chapter 5.

The recognizer is available either as a pre-trained implementation with linguistic models for Czech, or as a package which allows custom models to be trained using any NE-annotated data.

For training the recognizer, Czech Named Entity Corpus (Ševčíková et al. (2007a), Chapter 3) was used. In this corpus, Czech entities are classified into a two-level hierarchy classification: a fine-grained set of 42 classes or a more coarse classification of 7 super-classes. Like other authors, we report the evaluation on both hierarchy levels.

6.3 Software Performance

For comparison with previous work, we report results for the first version of the Czech Named Entity Corpus (CNEC 1.1). The linguistic models released with

System	F-measure (42 classes)	F-measure (7 classes)
Ševčíková et al. (2007a)	62.00	68.00
Kravalová and Žabokrtský (2009)	68.00	71.00
Konkol and Konopík (2013)	NA	79.00
Straková et al. (2013)	79.23	82.82
NameTag CNEC 1.1	77.88	81.01
NameTag CNEC 2.0	77.22	80.30

Table 6.1: Evaluation of the Czech NE recognizers.

Corpus	Words / sec	RAM	Model size
CNEC 1.1	40K	54MB	3MB
CNEC 2.0	45K	65MB	4MB

Table 6.2: Evaluation of the NE recognizer throughput, RAM and model size.

NameTag are trained on the most current version of the Czech Named Entity Corpus (CNEC 2.0). We report our results for both CNEC 1.1 and CNEC 2.0 in Table 6.1.

We designed NameTag as light-weight, efficient software with low resource usage. Table 6.2 shows the system word throughput, allocated RAM and the serialized model size (on disc) for NameTag.

6.4 Release

NameTag is a free software under Mozilla Public License 2.0 and its linguistic models are free for non-commercial use and distributed under CC BY-NC-SA license, although for some models the original data used to create the model may impose additional licensing conditions. NameTag can be used as:

- a standalone tool,
- C++ library with Java, Python, Perl and C# bindings,
- a web service,
- an on-line web demo.

NameTag is platform independent and does not require any additional libraries.

The pre-compiled binaries and source code are available on GitHub, the language models are available from the LINDAT/CLARIN infrastructure and the documentation can be found at the project website.[2] The web services and demo for the Czech and English languages are also provided by LINDAT/CLARIN infrastructure.

6.5 Conclusion

We released an efficient, light-weight NE tagger, which is available to a wide audience as an open-source, free software with rich API and also as an end-to-end application. The tagger reaches state-of-the-art results for Czech and is distributed with the models. We hope the release for Czech will prove useful for broad audience, for example for shared tasks which include Czech language data.

[2] `http://ufal.mff.cuni.cz/nametag`

Chapter 7

Neural Networks for Featureless Named Entity Recognition in Czech

This chapter describes the thesis author's recent advances in the field of named entity recognition, with focus in Czech (Straková et al., 2016). The complete source code is available at GitHub.[1]

We present a completely featureless, language agnostic named entity recognition system. Following recent advances in artificial neural network research, the recognizer employs parametric rectified linear units (PReLU), word embeddings and character-level word embeddings based on gated recurrent units (GRU). Without any feature engineering, only with surface forms, lemmas and tags as input, the network achieves excellent results in Czech NER and surpasses the current state of the art of previously published Czech NER systems, which use manually designed rule-based orthographic classification features. Furthermore, the neural network achieves robust results even when only surface forms are available as input. In addition, the proposed artificial neural network can use the manually designed rule-based orthographic classification features and in this combination, it exceeds the current state of the art by a wide margin.

7.1 Introduction

Recent years have seen a dramatic progress in the field of artificial neural networks. Undoubtedly, the ascend of artificial neural networks was started by the publication of reliable and computationally feasible ways of using tokens as classi-

[1] `https://github.com/strakova/ner_tsd2016`

fication features in artificial neural networks in so called word embeddings (Bengio et al. (2003); Mikolov et al. (2013), also see Section 7.2).

Another paradigm-changing publication introduces the long short-term memory units (LSTMs, Hochreiter and Schmidhuber (1997)). In simple words, LSTMs are specially shaped small units of artificial neural networks designed to process a sequence as a whole. LSTMs have been shown to capture non-linear and non-local dynamics in sequences (Hochreiter and Schmidhuber, 1997) and have been used to obtain many state-of-the-art results in sequence labelling (Ling et al., 2015; Lample et al., 2016; Yang et al., 2016).

Recently, a gated linear unit (GRU) was proposed by Cho et al. (2014) as an alternative to LSTM, and was shown to have similar performance, while being less computationally demanding.

In this chapter, we use artificial neural networks employing parametric rectified linear units (PReLU), word embeddings and character-level word embeddings based on gated linear units (GRU). First, we introduce word embeddings and character-level word embeddings in Section 7.2 and we describe the data and the evaluation procedure in Section 7.3. The comparison to related work is provided in Section 7.4. We further describe our artificial neural network architecture in Section 7.5. We report our results and discussion in Section 7.6 and we conclude in Section 7.7.

7.2 Word Embeddings

Simply said, word embeddings (Bengio et al., 2003; Mikolov et al., 2013) are a mapping from a large, discrete space of language vocabulary into a dramatically smaller continuous space of real-valued vectors. These vectors are then used as input for some higher-level NLP problem, in a similar way as words from a lexicon. For example, they can be used as input layer of an artificial neural network.

The main motivation for word embeddings is a continuous representation of words in a language, which normally occupy a discrete space. To explain the reasons why a continuous representation of language in a R^n space is so much more advantageous, we consider that a vast majority of NLP tasks can be reduced to an estimation of probability $P(Y|X;\theta)$ of a (categorical) event Y conditioned by an even X (see Section 4.2). A major problem for such estimation in NLP is data sparsity.

Because an estimation of a probability distribution always depends on a limited amount of data, the coverage of the observable events (i.e., words of the

language) in discrete space is low and we have to deal with unseen or rare events. This problem becomes exceedingly difficult as we try to estimate probabilities of word n-grams and is referred to as the *curse of dimensionality*.

Having moved from a discrete space to a continuous one, where dimensions in R^n space correspond to certain properties in words (such as word similarity), even rare and unseen words can produce reliable results, exploiting the similarity to the seen training words.

Although the data sparsity problem is most obvious in out-of-vocabulary words, the representation of words in R^n space in which the dimensions are shared between words, is beneficial for all words, because it makes the available data dense.

Another advantage of word embeddings as they are currently understood is that they seem to "embed" linguistic knowledge in them. Certain linguistic meaning such as sentiment or gender can be assigned to certain dimensions in the vector (e.g., Bolukbasi et al. (2016)). This kind of interpretability of word embeddings is very convenient from linguists' point of view.

Last but not least, the function usually maps the tokens from a large-dimensional space of the language vocabulary in to a drastically lower space of real-valued vectors (a typical size of a word embedding vector is usually between tens and hundreds), which makes the implementation and further training of artificial neural networks with such word embeddings on the input computationally efficient.

For morphologically rich languages, word embeddings appear rather too coarse for many tasks, especially for those where the structure of the word such as prefixes and suffixes, is crucial. Therefore, the ideas go further in publication of character-level word embeddings (Santos and Zadrozny, 2014), which recently improved the state of the art in POS-tagging (Ling et al., 2015).

Finally, another objective which word embeddings follows, is that they can be estimated (trained) in a completely unsupervised way on a large, unlabeled corpus. In such case, word embeddings represent a versatile representation of words which can be further used as an input for a higher-level NLP task. Another way to construct word and character-level word embeddings is to train them specifically for the task jointly with the artificial neural network, in an end-to-end fashion. Currently, both approaches seem to be beneficial and a combination of both kinds of embeddings can be used jointly, such as in Dyer et al. (2015), in which these two approaches are called *pretrained word embeddings* and *learned word embeddings*.

7.3 Experiment Setting: Data and Evaluation

We consider Czech named entity recognition (NER): Given a text, either raw (with surface forms only) or morphologically analyzed and POS-tagged (with forms, lemmas and POS-tags), the goal of the NER system is to identify and classify special entities such as proper names, cities, etc., into pre-defined classes.

We conduct our experiments on all available Czech NER corpora, so that we are able to compare with all available related work in Czech NER: Czech Named Entity Corpus (CNEC)[2] 1.0 (Ševčíková et al., 2007a; Kravalová and Žabokrtský, 2009), CNEC 2.0, CoNLL-based Extended CNEC 1.1 (Konkol and Konopík, 2013), CoNLL-based Extended CNEC 2.0.[3].

CoNLL-based Extended CNEC 1.1 and 2.0 are based on the respective original CNEC corpora, but they use only the coarser 7 classes and assume that entities are non-nested and labeled with one label.

For a comparison with the English state of the art, we evaluated our NER system on CoNLL-2003 shared task dataset (Tjong Kim Sang and De Meulder, 2003). In this dataset, four classes are predicted: PER (person), LOC (location), ORG (organization) and MISC (miscellaneous). The named entities are non-nested and annotated with exactly one label.

For the original CNEC 1.0 and CNEC 2.0, we present results for both fine-grained and coarse-grained classes hierarchy and we evaluate our results with the script provided with the corpora, which computes F-measure of the entities annotated in the first annotation round, see Section 3.1.2 and Kravalová and Žabokrtský (2009).

For the CoNLL-based Extended CNEC 1.1 and 2.0, we present results for the 7 classes present in these corpora and evaluate our results using the standard CoNLL evaluation script `conlleval`.

Similarly, the English CoNLL-2003 dataset is evaluated with CoNLL evaluation script `conlleval`.

7.4 Related Work

Czech named entity recognition (NER) has become a well-established field. Following the publication of the Czech Named Entity Corpus (Ševčíková et al., 2007a; Kravalová and Žabokrtský, 2009), several named entity recognizers for Czech have been published: (Kravalová and Žabokrtský, 2009; Konkol and Konopík, 2011;

[2] `http://ufal.mff.cuni.cz/nametag`

[3] `http://home.zcu.cz/~konkol/cnec2.0.php`

Straková et al., 2013; Konkol and Konopík, 2013, 2014; Konkol et al., 2015) and even a publicly available Czech named entity recognizer exists, see Chapter 6.

All these works use manually selected rule-based orthographic classification features, such as first character capitalization, existence of special characters in the word, regular expressions designed to reveal particular named entity types. Also gazetteers are extensively utilized. A wide selection of machine learning techniques (decision trees (Ševčíková et al., 2007a), SVMs (Kravalová and Žabokrtský, 2009), maximum entropy classifier (Straková et al., 2013), CRFs (Konkol and Konopík, 2013)), clustering techniques (Konkol et al., 2015) and stemming approaches (Konkol and Konopík, 2014) – see Section 2.4 for details.) is applied to the task.

The contribution of this chapter is that we use artificial neural networks with parametric rectified linear units, word embeddings and character-level word embeddings, which do not need manually designed classification features or gazetteers, and still surpass the current state of the art in Czech.

In Demir and Özgür (2014), the authors present a semi-supervised learning approach based on neural networks for Czech and Turkish NER utilizing word embeddings (Mikolov et al., 2013), but there are some differences in the neural network design and in classification features used. Instead regularized averaged perceptron, we use parametric rectified linear units, character-level word embeddings and dropout. The NER system in Demir and Özgür (2014) does not use morphological analysis, it is therefore similar to our experiments with only surface forms as input. However, the system does use "type information of the window c_i, i.e. is-capitalized, all-capitalized, all-digits, ..." etc. Our system surpasses these results even without using such features.

English named entity recognition has a successful tradition in computational linguistics and the English NER state of the art Ratinov and Roth (2009) has recently been pushed forward by Lin and Wu (2009); Chiu and Nichols (2015); Luo et al. (2015); Lample et al. (2016); Yang et al. (2016). We present a comparison with these works in Section 7.6. The most similar to our proposed neural network design is Lample et al. (2016). The authors propose a very similar network with LSTMs, word embeddings and character-level word embeddings. However, while we classify each word separately and use Viterbi to perform the final decoding, Lample et al. (2016) employs LSTMs combined with CRF layer to decode whole sentences, which brings a determining advantage over our framework as we show in Section 7.6.

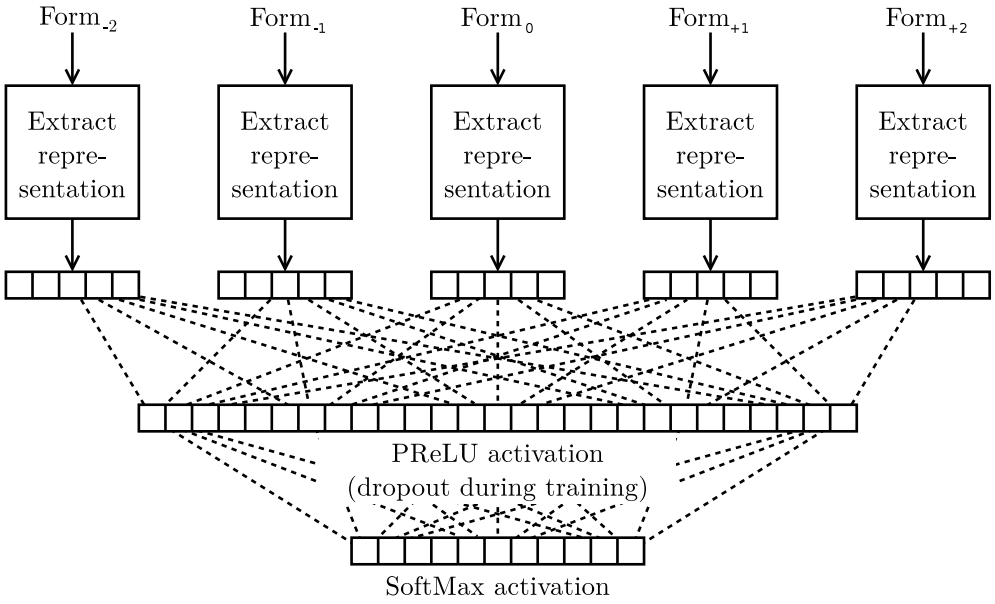

Figure 7.1: Artificial neural network, general diagram.

7.5 The Artificial Neural Network Classifier

For each word (and its context), we compute the probability distribution of labeling this word with BILOU-encoded (Ratinov and Roth (2009), Section 2.2.2) named entities. We then determine the best consistent assignment of BILOU-encoded entities to the words in the sentence using the Viterbi algorithm.

We compute the probability distribution for each word using an artificial neural network, shown in Figure 7.1. The input layer consists of representations of surface forms (and optionally lemmas and POS tags) of the word and W previous and W following words. The input layer is connected to a hidden layer of parametric rectified linear units (He et al., 2015) and the hidden layer is connected to the output layer which is a softmax layer producing probability distribution for all possible named entity classes in BILOU encoding.

We represent each word using a combination of the following (also please see Figure 7.2):

- *word embedding*: Word embeddings (see Section 7.2 are vector representations of low dimension. We generated the word embeddings using `word2vec` of Mikolov et al. (2013) and we chose the Skip-gram model with negative sampling.[4]

- *character-level word embedding*: To overcome drawbacks of word embed-

[4]We used the following options: `-cbow 0 -window 5 -negative 5 -iter 1`

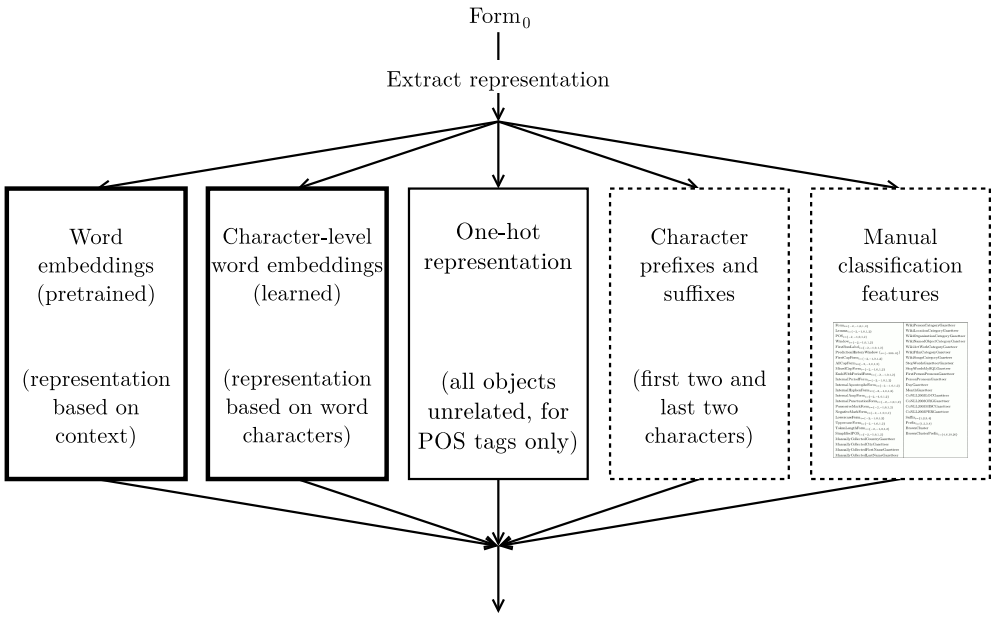

Figure 7.2: Word form representation as an input to artificial neural network.

dings (embeddings for different words are independent; unknown words cannot be handled), several orthography aware models have been proposed (Santos and Zadrozny, 2014; Ling et al., 2015), which compute word representation from the characters of the word.

We hypothesized that character-level word embeddings such as published in Ling et al. (2015) have the potential to increase the performance of Czech NER system. Our assumption was that Czech as a morphologically rich language would benefit from character-level word embeddings rather than word embeddings especially in cases where no morphological analysis is available.

We use bidirectional GRUs (Cho et al., 2014; Graves and Schmidhuber, 2005) in line with Ling et al. (2015): we represent every Unicode character with a vector of C real numbers, and we use GRUs to compute two outputs, one for word characters and the other one for reversed word characters, and we then concatenate the two outputs, as shown in Figure 7.3.

- *prefix and suffix*: For comparison with character-level word embeddings, we also include "poor man's substitution for character-level word embeddings" – we encode first two and last two characters encoded as one-hot vectors. We hypothesize that character-level word embeddings as a more sophisticated approach should perform better.

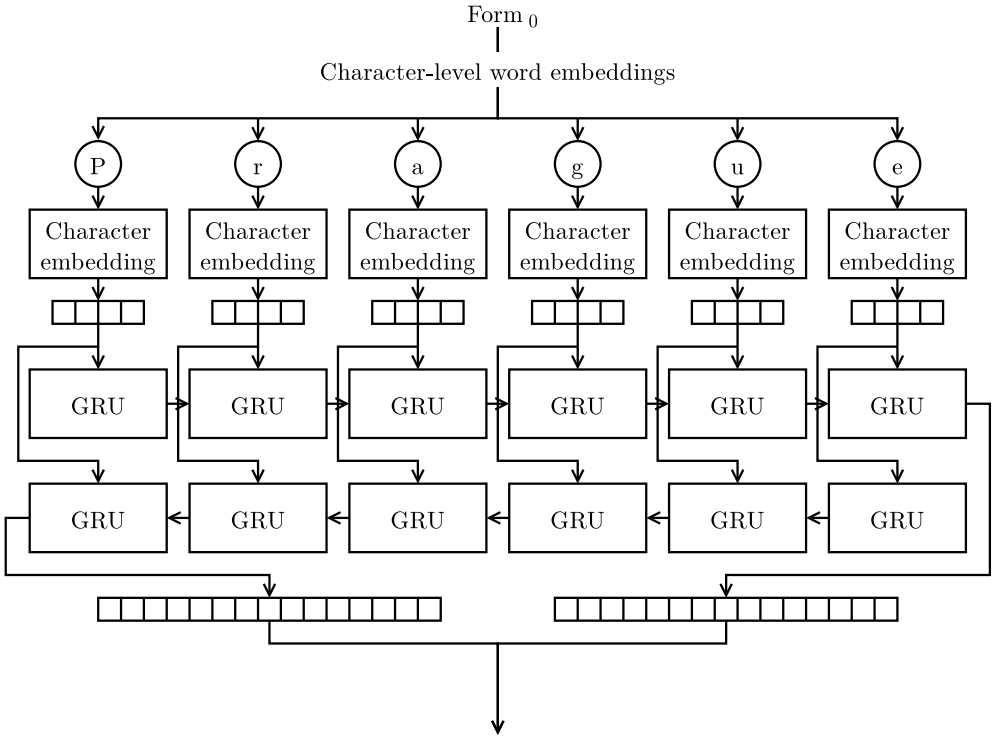

Figure 7.3: Character-level word embeddings diagram.

- *tag*: We encode part-of-speech tags as one-hot vectors.
- *manually designed classification features*: We also publish a combination of our neural network framework with traditional manually designed rule-based orthographic classification features. We use quite a limited set of classification features inspired by Straková et al. (2013): capitalization information, punctuation information, number information and Brown clusters (Brown et al., 1992). We do not use gazetteers, context aggregation, prediction history nor two-stage decoding.

The network is trained with AdaGrad (Duchi et al., 2011) and we use dropout (Srivastava et al., 2014) on the hidden layer. We implemented our neural network in Torch7 (Collobert et al., 2011a), a scientific computing framework with a wide support for machine learning algorithms.

We tuned most of the hyperparameters on development portion of CNEC 1.0 and used them for all other corpora. Notably, we utilize window size $W = 2$, hidden layer of 200 nodes, dropout 0.5, minibatches of size 100 and learning rate 0.02 with decay. We adapt the dimension C of the character-level word embeddings for every corpus separately, choosing either 32 or 64. All reported experiments use an ensemble of 5 networks, each using different random seed, with the resulting

distributions being an average of individual networks distributions. The training of a single network took half a day on a single CPU to stabilize performance on development data. During evaluation of testing data, we add the development set to the training data, a technique proposed in context of NER by Ratinov and Roth (2009).

We trained the word embeddings of dimension 200 on English Gigaword Fifth Edition corpus and on Czech SYN (Hnátková et al., 2014). We also lemmatized the corpora with MorphoDiTa (Straková et al., 2014) in order to pretrain the lemma embeddings (the word embeddings of lemmas).

7.6 Results and Discussion

We present two groups of experiments with low and high complexity depending on the available network input: experiments in which only surface forms were used, a putatively more difficult task as no linguistic knowledge is available to the NER system; and experiments with morphologically analyzed and POS-tagged text. We automatically generate lemmas and POS-tags from surface forms with MorphoDiTa (Straková et al., 2014), an open source tagger and lemmatizer.

Table 7.1 presents results of all experiments. Our baseline is an artificial neural network with only surface forms encoded as word embeddings. We then add more computational complexity to the network: WE stands for word embeddings of forms and lemmas, CLE stands for character-level word embeddings of forms and lemmas, 2CH stands for first two and last two characters of forms, lemmas and POS tags, and CF stands for experiments with traditional classification features.

We shall now present and discuss all results in Table 7.1 and we start with experiments without morphological analysis or POS-tagging, where surface forms are available only.

7.6.1 Experiments with Surface Forms in Czech

This group of experiments dealt with situations in which only surface forms are available as input. Since most of the previous literature heavily depends on manually selected language-dependent features, as well as on gazetteers and more or less linguistically motivated variants of lemmatization and stemming, the only work to be directly compared with is Demir and Özgür (2014). Demir and Özgür (2014) use a similar, semi-supervised neural network based approach. Their final system which uses word embeddings, capitalization and punctuation information,

Experiment/Related Work	Corpus						
	Original CNEC 1.0		Original CNEC 2.0		Extended CNEC 1.1	Extended CNEC 2.0	English CoNLL-2003
	Types	Supt.	Types	Supt.	Classes	Classes	Classes
f+WE (`baseline`)	63.24	69.61	63.33	68.87	63.48	63.91	67.99
f+CLE	71.43	76.13	70.50	75.80	69.59	70.06	82.65
f+WE+2CH	69.73	74.49	69.44	74.31	75.15	74.36	79.40
f+WE+CLE	73.30	78.11	73.10	77.89	73.33	73.80	84.08
f+WE+CLE+2CH	73.71	78.32	72.81	77.87	76.72	77.18	84.29
f+WE+CLE+2CH+CF	73.73	78.50	72.91	77.65	78.21	78.20	86.06
f,l,t+WE	80.07	83.21	77.45	80.92	78.42	78.18	87.92
f,l,t+CLE	75.63	80.88	74.38	79.85	75.32	76.02	83.70
f,l,t+WE+2CH	80.46	83.85	78.32	82.09	79.68	79.48	89.37
f,l,t+WE+CLE	80.64	84.06	78.62	82.48	80.11	80.41	89.74
f,l,t+WE+CLE+2CH	80.92	84.18	78.63	82.41	**80.88**	**80.79**	89.71
f,l,t+WE+CLE+2CH+CF	**81.20**	**84.68**	**79.23**	**82.78**	80.73	80.73	**89.92**
Kravalová and Žabokrtský (2009)	*68.00*	*71.00*	–	–	–	–	–
Konkol and Konopík (2011)	–	*72.94*	–	–	–	–	–
Konkol and Konopík (2013)	–	*79.00*	–	–	*74.08*	–	*83.24*
Straková et al. (2013)	*79.23*	*82.82*	–	–	–	–	–
Konkol and Konopík (2014)	–	–	–	–	*74.23*	*74.37*	–
Demir and Özgür (2014)	–	–	–	–	*75.61*	–	–
Konkol et al. (2015)	–	–	–	–	*74.08*	–	*89.44*
Ratinov and Roth (2009)	–	–	–	–	–	–	*90.80*
Lin and Wu (2009)	–	–	–	–	–	–	*90.90*
Passos et al. (2014)	–	–	–	–	–	–	*90.90*
Chiu and Nichols (2015)	–	–	–	–	–	–	*90.77*
Luo et al. (2015)	–	–	–	–	–	–	*91.20*
Lample et al. (2016)	–	–	–	–	–	–	*90.94*
Yang et al. (2016)	–	–	–	–	–	–	*91.20*

Table 7.1: Experiment results and comparison with related work. Columns denote corpora, rows our experiments or related work. The first group of rows describes our experiments with surface forms only (*f*), the second group our experiments with forms, lemmas and POS-tags (*f,l,t*). *WE* stands for word embeddings, *CLE* for character-level word embeddings, *2CH* for first two and last two characters, *CF* for traditional classification features. The third group of rows describes related work in Czech NER, and the fourth group related work in English NER.

prefixes, suffixes, context aggregation and prediction history, achieves CoNLL F-measure 75.61 for CoNLL-based Extended CNEC 1.1. We surpass these results with CoNLL F-measure 76.72, using only word embeddings, character-level word embeddings and first two and last two characters. If the traditional features are added, we even achieve CoNLL F-measure 78.21.

7.6.2 Experiments with Lemmas and POS Tags in Czech

Table 7.1 presents a comparison with related work on all available Czech NER corpora. The row denoted *f,l,t+WE+CLE+2CH+CF* is our best setting, including manually selected classification features. Our proposed network clearly exceeds the current state of the art on all Czech corpora in measures selected by the authors of the respective corpora.

We shall now focus our discussion on featureless neural networks. The proposed architecture exceeds the current Czech state of the art solely with pre-trained word embeddings (see row *f,l,t+WE* in Table 7.1), without requiring manually designed rule-based orthographic features, gazetteers, context aggregation, prediction history or two-stage decoding. The effect is even stronger with character-level embeddings and optionally first two and last two characters.

In this place, we would like to discuss the contribution of character-level word embeddings. In the introduction of this work, we hypothesized that character-level word embeddings would improve the Czech NER strongly. They did quite predictably improve some of the results, indeed in cases when morphological analysis was not available (row *f+CLE*) and they were also more successful in comparison with word embeddings (compare *f+WE* and *f+CLE*) in such settings. Character-level word embeddings also contribute more information than our "poor man's" replacement of first two and last two characters. Noticeably, character-level word embeddings did not contribute so noticeably when morphological analysis and POS-tagging was available. This effect can be easily interpreted by the training data size: while character-level word embeddings are trained end-to-end for the NER task solely on the NER corpus, the morphological analysis is trained on a much larger corpus (by a factor of 10 in the Czech case) with morphological annotation.

7.6.3 English Experiments

Our best result (row *f,l,t+WE+CLE+2CH+CF*) achieves F-measure 89.92, which is near the English state of the art. To our best knowledge, the current best result

on English was published in Luo et al. (2015), which performs joint named entity classification and linking, and Lample et al. (2016); Yang et al. (2016), which are most similar to our work.

Lample et al. (2016), also proposed neural network architecture with word embeddings and character-level word embeddings. Nevertheless, in Lample et al. (2016) sentence-level decoding using bidirectional LSTMs/GRUs with additional CRF layer is used, while our framework decodes the entities using Viterbi algorithm on probability distributions of named entity classes.

7.6.4 Drawbacks of the Proposed Architecture

The proposed neural network architecture is apparently a very strong framework. But does it have any drawbacks? We would now like to share our thoughts on problems we encountered in development of this architecture.

While the artificial neural network achieves excellent results with automatically retrieved features such as word embeddings and character-level word embeddings, one could argue that the whole framework is elegant, but conceptually complicated. Understanding the underlying mathematics and procedures to implement character-level word embeddings seems a challenging task. (We do not speak about word embeddings, as the `word2vec` tool exists, which can be used as a blackbox.)

Also, the training procedure for an artificial neural network is known to be time demanding. In our case, it took half a day to train the system in the full setting on a single CPU to stabilize accuracy on development data. On the other hand, once the neural network is fully trained, the prediction step is efficient, because it requires only a single forward pass.

Finally, a certain experience with neural networks is required to set up the architecture and tune the hyperparameters.

7.7 Conclusions

We presented an artificial neural network based NER system which achieves excellent results in Czech NER and near state-of-the-art results in English NER without manually designed rule-based orthographic classification features, gazetteers, context aggregation, prediction history or two-stage decoding. Our proposed architecture exceeds all Czech published results, using only forms, lemmas and POS tags encoded as word embeddings and achieves even better results in combination with character-level word embeddings, prefixes and suffixes. Finally, it surpasses

the current state of the art of Czech NER in combination with traditional classification features by a wide margin. The proposed neural network also yields very robust results without morphological analysis or POS-tagging, when only surface forms are available. As our future work, we plan to improve our decoding in line with Lample et al. (2016) and implement this work as part of NameTag (Straková et al., 2014). Currently, the related materials and source code are available at GitHub.[5]

[5] https://github.com/strakova/ner_tsd2016

Chapter 8

Conclusions

The goal of this thesis was to develop and describe a neural network based named entity recognizer with focus on the Czech language and we believe that we accomplished this task amply. The contribution of this thesis is a major advance in state-of-the-art of the Czech NER and it was published in peer reviewed proceedings (Kravalová and Žabokrtský, 2009; Straková et al., 2013, 2014, 2016), as encyclopedic entry (Straková, 2015) and a book chapter (Straková et al., 2017). The thesis author is the leading researcher and the main author of these publications and also published as co-author in other NLP-related fields (Agirre et al., 2009; Straková and Pecina, 2010; Kim and Straková, 2012; Straka et al., 2015, 2016). The thesis author is also one of the authors of the open-source named entity recognizer NameTag (Straková et al., 2014).[1]

The theoretical work in named entity recognition is collected in this thesis. We introduced the reader into named entity recognition field in Chapter 2 and we described the Czech Named Entity Corpus together with its thorough evaluation in Chapter 3. We published our in-house experiments with robust and efficient artificial neural networks with softmax output layer in Chapter 4 and our simple neural network based recognizer in Chapter 5. The open-source software for named entity recognition, NameTag, was described in Chapter 6. We finalized our work with a featureless, state-of-the-art named entity recognizer based on artificial neural network with softmax output layer, word embeddings and character-level word embeddings, in Chapter 7.

In the following paragraphs, we would like to share a discussion and retrospective of our previous work and design choices.

Firstly, although the bias of the Czech Named Entity Corpus towards unnaturally dense occurrence of named entities caused by the biased selection of

[1] `http://ufal.mff.cuni.cz/nametag`

sentences for annotation was not a problem for our automatic named entity systems (please see Chapter 3), we spent a nonnegligible amount of mental energy and time (=money) revising this decision, and measuring its effects. Also, the dense occurrence selection was performed on the sentence-level, resulting in a corpus consisting of independent sentences. The absence of whole documents makes entity linking in our corpus intractable, i.e., it disallows to extend the annotation of named entities by linking the entity mentions in the text to entities in a knowledge base.[2]

Furthermore, we found out that some of the heated discussions topics appeared less important when faced with real world applications. For example, a typical requirement for a named entity recognition system from a commercial subject usually comes up with a custom named entity classification system: Either an extremely simple one (only personal names are to be retrieved) or an unusual one (a certain set of classes which does not clearly map to CNEC or CoNLL annotation or is completely our of domain) is required. In this sense, the thesis author sees her future work in either an unsupervised or low resource NE system, or at least a very versatile one, maybe with data pretrained on a large unlabelled corpus in a semi-supervised way in the lines of Chapter 7. To sum up, it was beneficial for our team to step out of academic environment and deliver real-world solutions for third parties with NameTag (Chapter 6).

In the future of the field, the thesis author sees the introduction of the recent advances in artificial neural networks into named entity recognition as well as into other NLP tasks as the most probable way of this area development. Especially, word embeddings and their variants (various character-level word embeddings and end-to-end trained embeddings) are showing very promising results and will surely form the nearest future in named entity recognition and NLP in general.

In the annotation area, the thesis author is expecting the creation of entity linking corpora as well as named entities bound with ontologies and semantic networks.

To conclude, the main contribution of this thesis is a consistent research in Czech named entity recognition, with state-of-the-art results in Czech (Kravalová and Žabokrtský, 2009; Straková et al., 2013, 2016). The output of our research is available for the broad community as an open-source project NameTag[3]

[2]However, the reconstruction of the sentence context from the original corpora (Czech National Corpus, `http://ucnk.ff.cuni.cz`) is viable.

[3]`http://ufal.mff.cuni.cz/nametag`

(Straková et al., 2014). Furthermore, the most recent research (Straková et al., 2016) is also available at GitHub.[4]

[4] https://github.com/strakova/ner_tsd2016

Chapter 9

Acknowledgments

This work has been partially supported and has been using language resources and tools developed, stored and distributed by the LINDAT/CLARIN project of the Ministry of Education, Youth and Sports of the Czech Republic (project LM2015071).

This work has been using language resources developed and/or stored and/or distributed by the LINDAT-Clarin project of the Ministry of Education of the Czech Republic (project LM2010013).

This research was also partially supported by SVV project number 260 333 and by SVV project number 267 314.

Furthermore, we also received support from MSM 0021620838, GAAV ČR 1ET101120503, and MŠMT ČR LC536.

We are also grateful to all reviewers who helped us to improve our papers.

Bibliography

Eneko Agirre, Enrique Alfonseca, Keith Hall, Jana Kravalová, Marius Pasca, and Aitor Soroa. A Study on Similarity and Relatedness Using Distributional and WordNet-based Approaches. In *Proceedings of NAACL-HLT 09*, pages 19–27, Boulder, CO, USA, 2009. Association for Computational Linguistics. ISBN 978-1-932432-41-1.

Rie Kubota Ando and Tong Zhang. A high-performance semi-supervised learning method for text chunking. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, ACL '05, pages 1–9. Association for Computational Linguistics, 2005.

Eduard Bejček, Eva Hajičová, Jan Hajič, Pavlína Jínová, Václava Kettnerová, Veronika Kolářová, Marie Mikulová, Jiří Mírovský, Anna Nedoluzhko, Jarmila Panevová, Lucie Poláková, Magda Ševčíková, Jan Štěpánek, and Šárka Zikánová. Prague dependency treebank 3.0, 2013.

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Janvin. A Neural Probabilistic Language Model. *J. Mach. Learn. Res.*, 3:1137–1155, March 2003. ISSN 1532-4435.

Adam L. Berger, Vincent J. Della Pietra, and Stephen A. Della Pietra. A maximum entropy approach to natural language processing. *Comput. Linguist.*, 22 (1):39–71, March 1996. ISSN 0891-2017.

Anders Björkelund, Love Hafdell, and Pierre Nugues. Multilingual Semantic Role Labeling. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning: Shared Task*, CoNLL '09, pages 43–48, Stroudsburg, PA, USA, 2009. Association for Computational Linguistics. ISBN 978-1-932432-29-9.

Tolga Bolukbasi, Kai-Wei Chang, James Y. Zou, Venkatesh Saligrama, and Adam Tauman Kalai. Quantifying and Reducing Stereotypes in Word Embeddings. *CoRR*, abs/1606.06121, 2016.

Léon Bottou. Online Algorithms and Stochastic Approximations. In David Saad,

editor, *Online Learning and Neural Networks*. Cambridge University Press, Cambridge, UK, 1998.

John S. Bridle. Training stochastic model recognition algorithms as networks can lead to maximum mutual information estimation of parameters. In David S. Touretzky, editor, *NIPS*, pages 211–217. Morgan Kaufmann, 1989. ISBN 1-55860-100-7.

John S. Bridle. Probabilistic Interpretation of Feedforward Classification Network Outputs, with Relationships to Statistical Pattern Recognition. In Françoise Fogelman Soulié and Jeanny Hérault, editors, *Neurocomputing*, volume 68 of *NATO ASI Series*, pages 227–236. Springer Berlin Heidelberg, 1990. ISBN 978-3-642-76155-3.

Peter F. Brown, Peter V. deSouza, Robert L. Mercer, Vincent J. Della Pietra, and Jenifer C. Lai. Class-based n-gram models of natural language. *Computational Linguistics*, 18(4):467–479, December 1992. ISSN 0891-2017.

Wanxiang Che, Zhenghua Li, Yongqiang Li, Yuhang Guo, Bing Qin, and Ting Liu. Multilingual Dependency-based Syntactic and Semantic Parsing. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning: Shared Task*, CoNLL '09, pages 49–54, Stroudsburg, PA, USA, 2009. Association for Computational Linguistics. ISBN 978-1-932432-29-9.

Stanley F. Chen and Ronald Rosenfeld. A Gaussian Prior for Smoothing Maximum Entropy Models. Technical Report CMU-CS-99-108, Carnegie Mellon University, February 1999.

Hai Leong Chieu and Hwee Tou Ng. Named entity recognition with a maximum entropy approach. In *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003 - Volume 4*, CONLL '03, pages 160–163. Association for Computational Linguistics, 2003.

Nancy A. Chinchor. Proceedings of the Seventh Message Understanding Conference (MUC-7). In *Proceedings of the Seventh Message Understanding Conference (MUC-7)*, page 21 pages, April 1998.

Jason P. C. Chiu and Eric Nichols. Named Entity Recognition with Bidirectional LSTM-CNNs. *CoRR*, abs/1511.08308, 2015.

KyungHyun Cho, Bart van Merrienboer, Dzmitry Bahdanau, and Yoshua Bengio. On the Properties of Neural Machine Translation: Encoder-Decoder Approaches. *CoRR*, 2014.

Michael Collins. Discriminative Training Methods for Hidden Markov Models: Theory and Experiments with Perceptron Algorithms. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing*, pages 1–8. Association for Computational Linguistics, July 2002.

Michael Collins and Yoram Singer. Unsupervised Models for Named Entity Classification. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing and Very Large Corpora (EMNLP/VLC)*, pages 189–196, 1999.

Ronan Collobert, Koray Kavukcuoglu, and Clément Farabet. Torch7: A Matlab-like Environment for Machine Learning. In *BigLearn, NIPS Workshop*, 2011a.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. Natural Language Processing (Almost) from Scratch. *J. Mach. Learn. Res.*, 12:2493–2537, November 2011b. ISSN 1532-4435.

John N. Darroch and Douglas Ratcliff. Generalized iterative scaling for log-linear models. In *The Annals of Mathematical Statistics*, volume 43, pages 1470–1480. Institute of Mathematical Statistics, 1972.

Hal Daumé III and Daniel Marcu. Learning as search optimization: approximate large margin methods for structured prediction. In *Proceedings of the 22nd international conference on Machine learning*, ICML '05, pages 169–176. ACM, 2005. ISBN 1-59593-180-5.

Stephen Della Pietra, Vincent Della Pietra, and John Lafferty. Inducing Features of Random Fields. *IEEE Trans. Pattern Anal. Mach. Intell.*, 19(4):380–393, April 1997. ISSN 0162-8828.

H. Demir and A. Özgür. Improving Named Entity Recognition for Morphologically Rich Languages Using Word Embeddings. In *Machine Learning and Applications (ICMLA), 2014 13th International Conference on*, pages 117–122, December 2014.

John Duchi, Elad Hazan, and Yoram Singer. Adaptive Subgradient Methods for Online Learning and Stochastic Optimization. *J. Mach. Learn. Res.*, 12: 2121–2159, July 2011.

Chris Dyer, Miguel Ballesteros, Wang Ling, Austin Matthews, and Noah A. Smith. Transition-Based Dependency Parsing with Stack Long Short-Term Memory. *CoRR*, abs/1505.08075, 2015. URL `http://arxiv.org/abs/1505.08075`.

Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. LIBLINEAR: A Library for Large Linear Classification. *J. Mach. Learn. Res.*, 9:1871–1874, June 2008. ISSN 1532-4435.

Manaal Faruqui and Sebastian Padó. Training and Evaluating a German Named Entity Recognizer with Semantic Generalization. In *Proceedings of KONVENS 2010*, Saarbrücken, Germany, 2010.

Jenny Rose Finkel and Christopher D. Manning. Nested Named Entity Recognition. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1 - Volume 1*, EMNLP '09, pages 141–150, Stroudsburg, PA, USA, 2009. Association for Computational Linguistics.

Jenny Rose Finkel, Trond Grenager, and Christopher Manning. Incorporating non-local information into information extraction systems by Gibbs sampling. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, ACL '05, pages 363–370. Association for Computational Linguistics, 2005.

Michael Fleischman and Eduard Hovy. Fine Grained Classification of Named Entities. In *Proceedings of the 19th International Conference on Computational Linguistics (COLING)*, volume I, pages 267–273, 2002.

Radu Florian, Abe Ittycheriah, Hongyan Jing, and Tong Zhang. Named Entity Recognition through Classifier Combination. In Walter Daelemans and Miles Osborne, editors, *Proceedings of CoNLL-2003*, pages 168–171. Edmonton, Canada, 2003.

William A. Gale, Kenneth W. Church, and David Yarowsky. One Sense Per Discourse. In *Proceedings of the Workshop on Speech and Natural Language*, HLT '91, pages 233–237, Stroudsburg, PA, USA, 1992. Association for Computational Linguistics.

Joshua Goodman. Sequential Conditional Generalized Iterative Scaling. In *Proceedings of 40th Annual Meeting of the Association for Computational Linguistics*, pages 9–16, Philadelphia, Pennsylvania, USA, July 2002. Association for Computational Linguistics.

Alex Graves and Jürgen Schmidhuber. Framewise phoneme classification with bidirectional lstm and other neural network architectures. *Neural Networks*, pages 5–6, 2005.

Ralph Grishman and Beth Sundheim. Message Understanding Conference - 6: A Brief History. In *Proceedings of the 16th International Conference on Computational Linguistics (COLING)*, volume I, pages 466–471, 1996.

Jan Hajič. *Disambiguation of Rich Inflection: Computational Morphology of Czech*. Karolinum Press, 2004. ISBN 9788024602820.

Jan Hajič, Eva Hajičová, Jarmila Panevová, Petr Sgall, Petr Pajas, Jan Štěpánek, Jiří Havelka, and Marie Mikulová. Prague Dependency Treebank 2.0. Linguistic Data Consortium, LDC Catalog No.: LDC2006T01, Philadelphia, 2006.

Jan Hajič, Massimiliano Ciaramita, Richard Johansson, Daisuke Kawahara, Maria Antònia Martí, Lluís Màrquez, Adam Meyers, Joakim Nivre, Sebastian Padó, Jan Štěpánek, Pavel Straňák, Mihai Surdeanu, Nianwen Xue, and Yi Zhang. 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*, CoNLL '09, pages 1–18, Stroudsburg, PA, USA, 2009. Association for Computational Linguistics. ISBN 978-1-932432-29-9.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification. *CoRR*, 2015.

James Henderson. Inducing History Representations for Broad Coverage Statistical Parsing. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1*, NAACL '03, pages 24–31, Stroudsburg, PA, USA, 2003. Association for Computational Linguistics.

Milena Hnátková, Michal Křen, Pavel Procházka, and Hana Skoumalová. The SYN-series Corpora of Written Czech. In *Proceedings of the Ninth Internation-*

al Conference on Language Resources and Evaluation (LREC'14), Reykjavik, Iceland, May 2014. ELRA. ISBN 978-2-9517408-8-4.

Sepp Hochreiter and Jürgen Schmidhuber. Long Short-Term Memory. *Neural Comput.*, 9(8):1735–1780, November 1997. ISSN 0899-7667.

Nancy Ide and James Pustejovsky. *Handbook of Linguistic Annotation*. Springer Netherlands, 2017.

Rong Jin, Rong Yan, Jian Zhang, and Alex G. Hauptmann. A faster iterative scaling algorithm for conditional exponential model. In *Proceedings of the 20th International Conference on Machine Learning*, pages 282–289, 2003.

Petr Karlík et al. *Nový encyklopedický slovník češtiny*. Masarykova univerzita v Brně, Brno, Czech Republic, 2015.

Jun'ichi Kazama and Kentaro Torisawa. Exploiting Wikipedia as External Knowledge for Named Entity Recognition. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 698–707. Association for Computational Linguistics, June 2007.

Albert Kim and Jana Straková. Concurrent effects of lexical status and letter-rotation during early stages of visual word recognition: evidence from ERPs. *Brain Research*, 1468:52–62, 2012. ISSN 0006-8993.

Michal Konkol. First Steps in Czech Entity Linking. In *Text, Speech, and Dialogue*, volume 9302 of *Lecture Notes in Computer Science*, pages 489–496. Springer International Publishing, 2015. ISBN 978-3-319-24032-9.

Michal Konkol and Miloslav Konopík. Maximum Entropy Named Entity Recognition for Czech Language. In *Text, Speech and Dialogue*, volume 6836 of *Lecture Notes in Computer Science*, pages 203–210. Springer Berlin Heidelberg, 2011. ISBN 978-3-642-23537-5.

Michal Konkol and Miloslav Konopík. CRF-based Czech Named Entity Recognizer and Consolidation of Czech NER Research. In Ivan Habernal and Václav Matoušek, editors, *Text, Speech and Dialogue*, volume 8082 of *Lecture Notes in Computer Science*, pages 153–160. Springer Berlin Heidelberg, 2013.

Michal Konkol and Miloslav Konopík. Named entity recognition for highly inflectional languages: effects of various lemmatization and stemming approaches. In

Text, Speech and Dialogue, pages 267–274. Springer International Publishing, 2014.

Michal Konkol and Miloslav Konopík. Segment Representations in Named Entity Recognition. In *Text, Speech, and Dialogue*, volume 9302 of *Lecture Notes in Computer Science*, pages 61–70. Springer International Publishing, 2015. ISBN 978-3-319-24032-9.

Michal Konkol, Tomáš Brychcín, and Miloslav Konopík. Latent semantics in named entity recognition. *Expert Systems with Applications*, 42(7):3470–3479, 2015.

Pavel Král. Features for Named Entity Recognition in Czech Language. In Joaquim Filipe and Jan L. G. Dietz, editors, *KEOD*, pages 437–441. SciTePress, 2011. ISBN 978-989-8425-80-5.

Jana Kravalová and Zdeněk Žabokrtský. Czech named entity corpus and SVM-based recognizer. In *Proceedings of the 2009 Named Entities Workshop: Shared Task on Transliteration*, NEWS '09, pages 194–201, Stroudsburg, PA, USA, 2009. Association for Computational Linguistics. ISBN 978-1-932432-57-2.

Vijay Krishnan and Christopher D. Manning. An Effective Two-stage Model for Exploiting Non-local Dependencies in Named Entity Recognition, 2006.

John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. In *Proceedings of the Eighteenth International Conference on Machine Learning*, ICML '01, pages 282–289. Morgan Kaufmann Publishers Inc., 2001. ISBN 1-55860-778-1.

Guillaume Lample, Miguel Ballesteros, Kazuya Kawakami, Sandeep Subramanian, and Chris Dyer. Neural Architectures for Named Entity Recognition. *CoRR*, 2016. To appear at NAACL 2016.

Percy Liang. Semi-Supervised Learning for Natural Language. Master's thesis, Massachusetts Institute of Technology, 2005.

Dekang Lin and Xiaoyun Wu. Phrase clustering for discriminative learning. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2-Volume 2*, pages 1030–1038. Association for Computational Linguistics, 2009.

Wang Ling, Tiago Luís, Luís Marujo, Ramón Fernandez Astudillo, Silvio Amir, Chris Dyer, Alan W. Black, and Isabel Trancoso. Finding Function in Form: Compositional Character Models for Open Vocabulary Word Representation. *CoRR*, 2015.

Dong C. Liu and Jorge Nocedal. On the limited memory BFGS method for large scale optimization. *Math. Program.*, 45(3):503–528, December 1989. ISSN 0025-5610.

Gang Luo, Xiaojiang Huang, Chin-Yew Lin, and Zaiqing Nie. Joint Named Entity Recognition and Disambiguation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 879–888. Association for Computational Linguistics, 2015.

Robert Malouf. A comparison of algorithms for maximum entropy parameter estimation. In *Proceedings of the 6th conference on Natural language learning - Volume 20*, COLING-02, pages 1–7, Stroudsburg, PA, USA, 2002. Association for Computational Linguistics.

Christopher D. Manning and Hinrich Schütze. *Foundations of Statistical Natural Language Processing*. MIT Press, Cambridge, MA, USA, 1999. ISBN 0-262-13360-1.

Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. Building a Large Annotated Corpus of English: The Penn Treebank. *COMPUTATIONAL LINGUISTICS*, 19(2):313–330, 1993.

Tomáš Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeff Dean. Distributed Representations of Words and Phrases and their Compositionality. In *Advances in Neural Information Processing Systems 26*, pages 3111–3119. Curran Associates, Inc., 2013.

Thomas P. Minka. A comparison of numerical optimizers for logistic regression. Technical report, Carnegie Mellon University, Department of Statistics, 2003.

Thomas M. Mitchell. *Machine Learning*. McGraw-Hill, Inc., New York, NY, USA, 1 edition, 1997.

Jorge Nocedal. Updating Quasi-Newton Matrices with Limited Storage. *Mathematics of Computation*, 35(151):773–782, 1980.

Joel Nothman, James R. Curran, and Tara Murphy. Transforming Wikipedia into Named Entity Training Data. In *Proceedings of the Australasian Language Technology Workshop*, Hobart, Australia, 2008.

Joel Nothman, Nicky Ringland, Will Radford, Tara Murphy, and James R. Curran. Learning multilingual named entity recognition from wikipedia. *Artificial Intelligence*, 194:151 – 175, 2013.

Petr Pajas and Jan Štěpánek. XML-Based Representation of Multi-Layered Annotation in the PDT 2.0. In Richard Erhard Hinrichs, Nancy Ide, Martha Palmer, and James Pustejovsky, editors, *Proceedings of the LREC Workshop on Merging and Layering Linguistic Information (LREC 2006)*, pages 40–47, Paris, France, 2006. ISBN 2-9517408-2-4.

Alexandre Passos, Vineet Kumar, and Andrew McCallum. Lexicon Infused Phrase Embeddings for Named Entity Resolution. *CoRR*, abs/1404.5367, 2014.

Martin Popel. Ways to Improve the Quality of English-Czech Machine Translation. Master's thesis, ÚFAL, MFF UK, Prague, Czech Republic, 2009.

Vasin Punyakanok, Dan Roth, Wen-tau Yih, and Dav Zimak. Semantic Role Labeling Via Integer Linear Programming Inference. In *Proceedings of Coling 2004*, pages 1346–1352, Geneva, Switzerland, August 2004. Association for Computational Linguistics.

Lawrence R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.

Lev Ratinov and Dan Roth. Design Challenges and Misconceptions in Named Entity Recognition. In *CoNLL '09: Proceedings of the Thirteenth Conference on Computational Natural Language Learning*, pages 147–155, Morristown, NJ, USA, 2009. Association for Computational Linguistics. ISBN 978-1-932432-29-9.

Raúl Rojas. *Neural Networks: A Systematic Introduction*. Springer-Verlag New York, Inc., New York, NY, USA, 1996. ISBN 3-540-60505-3.

Cicero D. Santos and Bianca Zadrozny. Learning Character-level Representations for Part-of-Speech Tagging. In *Proceedings of the 31st International Conference on Machine Learning*, pages 1818–1826. JMLR Workshop and Conference Proceedings, 2014.

Helmut Schmid. Part-of-speech tagging with neural networks. In *Proceedings of the 15th conference on Computational linguistics - Volume 1*, COLING '94, pages 172–176, Stroudsburg, PA, USA, 1994. Association for Computational Linguistics.

Satoshi Sekine. Sekine's Extended Named Entity Hierarchy. `http://nlp.cs.nyu.edu/ene/`, 2003.

Magda Ševčíková, Zdeněk Žabokrtský, and Oldřich Krůza. Named entities in Czech: annotating data and developing NE tagger. In *Proceedings of the 10th international conference on Text, speech and dialogue*, TSD'07, pages 188–195. Springer-Verlag, 2007a. ISBN 3-540-74627-7, 978-3-540-74627-0.

Magda Ševčíková, Zdeněk Žabokrtský, and Oldřich Krůza. Zpracování pojmenovaných entit v českých textech. Technical Report TR-2007-36, Charles University in Prague, Faculty of Mathematics and Physics, Institute of Formal and Applied Lingusitics, 2007b.

Libin Shen, Giorgio Satta, and Aravind Joshi. Guided Learning for Bidirectional Sequence Classification. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 760–767, Prague, Czech Republic, June 2007. Association for Computational Linguistics.

Pavel Šmerk. Fast Morphological Analysis of Czech. In *Proceedings of the Raslan Workshop 2009*, Brno, 2009. Masarykova univerzita. ISBN 978-80-210-5048-8.

Richard Socher, Cliff Chiung-Yu Lin, Andrew Y. Ng, and Christopher D. Manning. Parsing Natural Scenes and Natural Language with Recursive Neural Networks. In Lise Getoor and Tobias Scheffer, editors, *ICML*, pages 129–136. Omnipress, 2011.

Drahomíra "johanka" Spoustová, Jan Hajič, Jan Raab, and Miroslav Spousta. Semi-Supervised Training for the Averaged Perceptron POS Tagger. In *Proceedings of the 12th Conference of the European Chapter of the ACL (EACL 2009)*, pages 763–771, Athens, Greece, March 2009. Association for Computational Linguistics.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research*, 15:1929–1958, 2014.

Milan Straka, Jan Hajič, Jana Straková, and jr. Jan Hajič. Parsing Universal Dependency Treebanks using Neural Networks and Search-Based Oracle. In *14th International Workshop on Treebanks and Linguistic Theories (TLT 2015)*, pages 208–220, Warszawa, Poland, 2015. IPIPAN, IPIPAN. ISBN 978-83-63159-18-4.

Milan Straka, Jan Hajič, and Jana Straková. UDPipe: Trainable Pipeline for Processing CoNLL-U Files Performing Tokenization, Morphological Analysis, POS Tagging and Parsing. In Nicoletta Calzolari (Conference Chair), Khalid Choukri, Thierry Declerck, Sara Goggi, Marko Grobelnik, Bente Maegaard, Joseph Mariani, Helene Mazo, Asuncion Moreno, Jan Odijk, and Stelios Piperidis, editors, *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016)*, pages 4290–4297, Paris, France, may 2016. European Language Resources Association (ELRA). ISBN 978-2-9517408-9-1.

Jana Straková. Rozpoznávání pojmenovaných entit. In Petr Karlík, Marek Nekula, and Jana Pleskalová, editors, *Nový encyklopedický slovník češtiny*. Masarykova univerzita v Brně, 2015.

Jana Straková and Pavel Pecina. Czech Information Retrieval with Syntax-based Language Models. In *Proceedings of the 7th International Conference on Language Resources and Evaluation (LREC 2010)*, pages 1359–1362, Valletta, Malta, 2010. European Language Resources Association. ISBN 2-9517408-6-7.

Jana Straková, Milan Straka, and Jan Hajič. A New State-of-The-Art Czech Named Entity Recognizer. In Ivan Habernal and Václav Matoušek, editors, *Text, Speech, and Dialogue*, volume 8082 of *Lecture Notes in Computer Science*, pages 68–75. Springer Berlin Heidelberg, 2013. ISBN 978-3-642-40584-6.

Jana Straková, Milan Straka, and Jan Hajič. Open-Source Tools for Morphology, Lemmatization, POS Tagging and Named Entity Recognition. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 13–18, Stroudsburg, PA, USA, 2014. Johns Hopkins University, Baltimore, MD, USA, Association for Computational Linguistics. ISBN 978-1-941643-00-6.

Jana Straková, Milan Straka, and Jan Hajič. Neural Networks for Featureless Named Entity Recognition in Czech. In Petr Sojka, Aleš Horák, Ivan Kopeček, and Karel Pala, editors, *Text, Speech, and Dialogue: 19th International Conference, TSD 2016, Brno , Czech Republic, September 12-16, 2016, Proceedings*,

pages 173–181, Cham, 2016. Springer International Publishing. ISBN 978-3-319-45510-5.

Jana Straková, Milan Straka, Magda Ševčíková, and Zdeněk Žabokrtský. Czech Named Entity Corpus. In Nancy Ide and James Pustejovsky, editors, *The Handbook of Linguistic Annotation*. Springer Netherlands, 2017.

Jun Suzuki and Hideki Isozaki. Semi-Supervised Sequential Labeling and Segmentation Using Giga-Word Scale Unlabeled Data. In *Proceedings of ACL-08: HLT*, pages 665–673, Columbus, Ohio, June 2008. Association for Computational Linguistics.

Partha Pratim Talukdar, Thorsten Brants, Mark Liberman, and Fernando Pereira. A Context Pattern Induction Method for Named Entity Extraction. In *Proceedings of the 10th Conference on Computational Natural Language Learning (CoNLL-X)*, pages 141–148, 2006.

Ivan Titov, James Henderson, Paola Merlo, and Gabriele Musillo. Online Projectivisation for Synchronous Parsing of Semantic and Syntactic Dependencies. In *In Proceedings of the Internation Joint Conference on Artificial Intelligence (IJCAI)*, pages 1562–1567, Pasadena, California, USA, 2009.

Erik F. Tjong Kim Sang and Fien De Meulder. Introduction to the CoNLL-2003 Shared Task: Language-Independent Named Entity Recognition. In *Proceedings of CoNLL-2003*, pages 142–147. Edmonton, Canada, 2003.

A.J. Viterbi. Error bounds for convolutional codes and an asymptotically optimal decoding algorithm. *IEEE Transactions on Information Theory*, 13:260–269, 1967.

Zhilin Yang, Ruslan Salakhutdinov, and William W. Cohen. Multi-Task Cross-Lingual Sequence Tagging from Scratch. *CoRR*, abs/1603.06270, 2016.

Zdeněk Žabokrtský, Jan Ptáček, and Petr Pajas. TectoMT: Highly Modular MT System with Tectogrammatics Used as Transfer Layer. In *Proceedings of the 3rd Workshop on Statistical Machine Translation, ACL*, 2008.

Tong Zhang. Solving large scale linear prediction problems using stochastic gradient descent algorithms. In *Proceedings of the twenty-first international conference on Machine learning*, ICML '04, pages 116–124, New York, NY, USA, 2004. ACM. ISBN 1-58113-838-5.

Hai Zhao, Wenliang Chen, Chunyu Kit, and Guodong Zhou. Multilingual Dependency Learning: A Huge Feature Engineering Method to Semantic Dependency Parsing. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning: Shared Task*, CoNLL '09, pages 55–60, Stroudsburg, PA, USA, 2009. Association for Computational Linguistics. ISBN 978-1-932432-29-9.

List of Figures

1.1 How to read this thesis. 5

3.1 Two-level hierarchical classification of NEs in CNEC 1.0. 23
3.2 Frequency of NE types and containers in CNEC 1.0. 29
3.3 Dependence of the recognizer F-measure on the corpus size. 32
3.4 Two-level hierarchical classification of NEs in CNEC 2.0. 33

4.1 Neural network diagram. 41
4.2 Convergence rate on English NER CoNLL-2003 shared task. . . . 52
4.3 The effect of unseen weights on the score of the POS tagging task. 53

5.1 System overview 56

7.1 Artificial neural network, general diagram. 72
7.2 Word form representation as an input to artificial neural network. 73
7.3 Character-level word embeddings diagram. 74

List of Tables

3.1 Division of the corpus into training, development and evaluation sets. . . . 25
3.2 Occurrences of NE instances of different length in the corpus. . . 25
3.3 Distribution of several most frequent NE types in the annotated corpus. . . . 26
3.4 Evaluation of Straková et al. (2013) in CNEC 1.0 with a varying degree of classification granularity. . . . 28
3.5 Evaluation of Straková et al. (2013) on two classification levels (supertypes and types). . . . 28
3.6 Most frequent errors made by Straková et al. (2013) on CNEC 1.0. 31

4.1 English dataset properties of all case studies. . . . 45
4.2 F_1 score in Named Entity Recognition CoNLL-2003 shared task. . 50
4.3 Semantic F_1 labeled score in SRL CoNLL-2009. . . . 50
4.4 Accuracy on Part of Speech Tagging of Wall Street Journal without external data. . . . 51

5.1 Lemma term types as used in Hajič (2004) and Hajič et al. (2006). 59
5.2 System development. The experiments (A), (B), (C) and (D) show F-measure gains over the baseline on the test portion of the English and Czech data. . . . 61
5.3 Detailed results for Czech. . . . 61
5.4 System comparison for Czech language (F-measure on test data). 62
5.5 System comparison for English language (F-measure on test data). 62

6.1 Evaluation of the Czech NE recognizers. . . . 65
6.2 Evaluation of the NE recognizer throughput, RAM and model size. 65

7.1 Experiment results and comparison with related work. . . . 76

A.1 Optimal number of iterations and Gaussian σ^2 for NER CoNLL-2003 shared task. . . . 107
A.2 Optimal number of iteratiosn and Gaussian σ^2 for SRL CoNLL-2009 shared task. . . . 107

A.3 Optimal number of iterations and Gaussian σ^2 (normalized by the number of training examples) for SRL CoNLL-2009 shared task, semantic roles assignment. 108
A.4 Optimal number of iterations and Gaussian σ^2 for POS Tagging. . 108
A.5 Classification feature templates for NER CoNLL-2003 shared task. 109
A.6 Classification feature templates for SRL CoNLL-2009 shared task, predicate sense classification. 109
A.7 Classification feature templates for SRL CoNLL-2009 shared task, semantic roles assignment. 110
A.8 Classification feature templates for POS tagging. 110

List of Abbreviations

CNEC – Czech Named Entity Corpus, Chapter 3.

CoNLL – yearly organized Conference on Natural Language Learning.

CRF – conditional random fields, probabilistic models for segmenting and labeling sequence data (Lafferty et al., 2001).

BILOU – a way od encoding the beginning (B), inside (I), last (L), outside (O) and unit-length (U) named entity in a sequence, please see Section 2.2.2 in Chapter 2.

BIO – a way of encoding the beginning (B), inside (I) and outside (O) of a named entity in a sequence, please see Section 2.2.2 in Chapter 2.

GRU – gated linear units (Cho et al., 2014), alternative to LSTMs, Chapter 7.

HMM – hidden Markov model (Rabiner, 1989).

L-BFGS – an algorithm for parameter estimation in machine learning. L stands for limited memory, the four letters are acronyms of authors Nocedal (1980); Liu and Nocedal (1989), Chapter 4.

LSTMs – long short-term memory units (Hochreiter and Schmidhuber, 1997), Chapter 7.

MUC – series of Message Understanding Conferences, Section 2.5 in Chapter 2.

NAACL – North American Chapter of the Association for Computational Linguistics

NER – named entity recognition, identification and classification of proper names in text, for example names of people, cities, organizations and others. Please see Section 2.1 in Chapter 2.

NLP – natural language processing, a field of computer science, artificial intelligence, and computational linguistics concerned with the interactions between computers and human (natural) languages (cited from `http://www.wikipedia.org`). Also please see Manning and Schütze (1999).

NN – (artificial) neural networks, please see a textbook Rojas (1996).

PDT – Prague Dependency Treebank (Bejček et al. (2013)[1]).

POS tagging – part-of-speech tagging, a task assigning each word its part of speech, Chapter 4.

PReLU – parametric rectified linear units, please see Chapter 7.

[1] `http://ufal.mff.cuni.cz/prague-dependency-treebank`

SGD – stochastic gradient descent, a stochastic approximation for minimizing (or maximizing) an objective function by iterative steps, Chapter 4.
SRL – semantic role labeling, the task of finding and labeling semantic dependencies of predicates in the sentence, please see Section 4.5.2 in Chapter 4.
SVM – support vector machines, a machine learning method, please see Mitchell (1997).
TSD – yearly organized conference on Text, Speech and Dialogue.
UD – Universal Dependencies, cross-linguistically consistent annotation for many languages[2].
WSD – word sense disambiguation, the task of determining one of the possible word senses, please see e.g. Manning and Schütze (1999).
WSJ - Wall Street Journal corpus, part of Penn Treebank (LDC Catalog No. LDC99T42).

[2] `http://www.universaldependencies.org`

Appendix

A.1 Efficient Log-linear Modeling and Softmax Neural Networks

This Appendix includes detailed information about the methodology used in Chapter 4.

Tables A.1, A.2, A.3 and A.4 describe the trainer settings for the respective tasks.

System	English	German
Maximum Entropy Toolkit by Zhang Le (L-BFGS)	80, 0.5	150, 0.5
OpenNLP Maxent (GIS)	500, 0	500, 0
LIBLINEAR	$C = 1$	$C = 1$
NN-simple (SGD)	40, 0.5	30, 0.5
NN-hidden layer (SGD)	50, 0.5	20, 0.5

Table A.1: Optimal number of iterations and Gaussian σ^2 for Named Entity Recognition CoNLL-2003 shared task. LIBLINEAR optimized parameter is the penalty parameter C.

System	Catalan	Chinese	Czech	English	German	Spanish
Maximum Entropy Toolkit by Zhang Le (L-BFGS)	40, 0	50, 0.5	100, 1	50, 1	40, 1	80, 0
OpenNLP Maxent (GIS)	500, 0					
LIBLINEAR	$C = 1$	$C = 1$	$C = 1$	$C = 1$	$C = 1$	$C = 1$
NN-simple (SGD)	40, 1	10, 0.5	10, 0.5	20, 0.5	20, 0.5	50, 0
NN-hidden layer (SGD)	30, 0	50, 0	40, 0	40, 0	30, 0	50, 0

Table A.2: Optimal number of iterations and Gaussian σ^2 for Semantic Role Labeling CoNLL-2009 shared task, predicate senses subtask. LIBLINEAR optimized parameter is the penalty parameter C.

System	Catalan	Chinese	Czech	English	German	Spanish
Maximum Entropy Toolkit by Zhang Le (L-BFGS)	130, 0.5	130, 0.5	150, 0.5	150, 0.5	120, 0	120, 0.5
OpenNLP Maxent (GIS)	500, 0					
LIBLINEAR	$C = 1$	$C = 1$	$C = 1$	$C = 1$	$C = 1$	$C = 1$
NN-simple (SGD)	30, 0.5	50, 0	40, 0	20, 0	20, 1	50, 0
NN-hidden layer (SGD)	50, 0.5	20, 1	30, 0.5	50, 1	20, 0.5	40, 1

Table A.3: Optimal number of iterations and Gaussian σ^2 for Semantic Role Labeling CoNLL-2009 shared task, semantic roles subtask. LIBLINEAR optimized parameter is the penalty parameter C.

System	English
Maximum Entropy Toolkit by Zhang Le (L-BFGS)	200, 1
OpenNLP Maxent (GIS)	50, 0.5
LIBLINEAR	$C = 1$
NN-simple (SGD)	25, 0.5
NN-hidden layer (SGD)	30, 1

Table A.4: Optimal number of iterations and Gaussian σ^2 for Part of Speech tagging. LIBLINEAR optimized parameter is the penalty parameter C.

A.2 Classification Features

This Appendix lists classification features used in our classifiers described in Chapter 4, Chapter 5 and Chapter 7.

Tables A.5, A.6, A.7 and A.8 list classification feature templates used in Named Entity Recognition, Semantic Role Labeling predicate sense classification and semantic role classification and Part of Speech Tagging, respectively.

$\text{Form}_{i=\{-2,-1,0,1,2\}}$	WikiPersonCategoryGazetteer
$\text{Lemma}_{i=\{-2,-1,0,1,2\}}$	WikiLocationCategoryGazetteer
$\text{POS}_{i=\{-2,-1,0,1,2\}}$	WikiOrganizationCategoryGazetteer
$\text{Window}_{i=\{-2,-1,0,1,2\}}$	WikiNamedObjectCategoryGazeteer
$\text{FirstRunLabel}_{i=\{-2,-1,0,1,2\}}$	WikiArtWorkCategoryGazeteer
PredictionHistoryWindow $(_{i=\{-500..0\}})$	WikiFilmCategoryGazeteer
$\text{FirstCapForm}_{i=\{-2,-1,0,1,2\}}$	WikiSongsCategoryGazeteer
$\text{AllCapForm}_{i=\{-2,-1,0,1,2\}}$	StopWordsGazetteerGazeteer
$\text{MixedCapForm}_{i=\{-2,-1,0,1,2\}}$	StopWordsMySQLGazeteer
$\text{EndsWithPeriodForm}_{i=\{-2,-1,0,1,2\}}$	FirstPersonPronounGazetteer
$\text{InternalPeriodForm}_{i=\{-2,-1,0,1,2\}}$	PersonPronounGazetteer
$\text{InternalApostropheForm}_{i=\{-2,-1,0,1,2\}}$	DayGazetteer
$\text{InternalHyphenForm}_{i=\{-2,-1,0,1,2\}}$	MonthGazetteer
$\text{InternalAmpForm}_{i=\{-2,-1,0,1,2\}}$	CoNLL2003LOCGazetteer
$\text{InternalPunctuationForm}_{i=\{-2,-1,0,1,2\}}$	CoNLL2003ORGGazetteer
$\text{PossessiveMarkForm}_{i=\{-2,-1,0,1,2\}}$	CoNLL2003MISCGazetteer
$\text{NegativeMarkForm}_{i=\{-2,-1,0,1,2\}}$	CoNLL2003PERGazetteer
$\text{LowercaseForm}_{i=\{-2,-1,0,1,2\}}$	$\text{Suffix}_{i=\{1,2,3,4\}}$
$\text{UppercaseForm}_{i=\{-2,-1,0,1,2\}}$	$\text{Prefix}_{i=\{1,2,3,4\}}$
$\text{TokenLengthForm}_{i=\{-2,-1,0,1,2\}}$	BrownCluster
$\text{SimplifiedPOS}_{i=\{-2,-1,0,1,2\}}$	$\text{BrownClusterPrefix}_{i=\{4,6,10,20\}}$
ManuallyCollectedCountryGazetteer	
ManuallyCollectedCityGazetteer	
ManuallyCollectedFirstNameGazetteer	
ManuallyCollectedLastNameGazetteer	

Table A.5: Classification feature templates for Named Entity Recognition CoNLL-2003 shared task inspired by Ratinov and Roth (2009).

ConstituentPOSPattern	Predicate
ConstituentPOSPattern+DepRelation	PredicateChildrenPOS
ConstituentPOSPattern+DepwordLemma	PredicateChildrenPOSNoDup
ConstituentPOSPattern+HeadwordLemma	PredicateChildrenREL
DepRelation	PredicateChildrenRELNoDup
DepRelation+Headword	PredicateLemma
DepRelation+HeadwordLemma	PredicatePOS
DepRelation+HeadwordPOS	$\text{PredicateLemma}_{-1}$
FirstLemma	$\text{PredicateLemma}_{-1}$+PredicateLemma
FirstPOS	PredicateLemma+$\text{PredicateLemma}_{+1}$
FirstWord	$\text{PredicateLemma}_{+1}$
Headword	Predicate_{-1}+Predicate
HeadwordLemma	Predicate_{-1}
HeadwordPOS	Predicate+Predicate_{+1}
LastLemma	Predicate_{+1}
LastPOS	PredicatePOS_{-1}
LastWord	PredicatePOS_{+1}
PFEAT	
PFEATSplit	
PFEATSplitRemoveNULL	

Table A.6: Classification feature templates for Semantic Role Labeling CoNLL-2009 shared task, predicate sense classification subtask. Features adopted from Che et al. (2009) and Zhao et al. (2009).

ChildrenPOS	DepwordLastWord
DepwordChildrenPOS	Path
ChildrenPOSNoDup	Path+RelationPath
DepwordChildrenPOSNoDup	PathLength
ChildrenREL	PFEAT
DepwordChildrenREL	DepwordPFEAT
ChildrenRELNoDup	PFEATSplit
DepwordChildrenRELNoDup	DepwordPFEATSplit
ConstituentPOSPattern	PFEATSplitRemoveNull
DepwordConstituentPOSPattern	DepwordPFEATSplitRemoveNull
ConstituentPOSPattern+DepRelation	PositionWithPredicate
DepwordConstituentPOSPattern+DepwordDepRelation	Predicate
ConstituentPOSPattern+DepwordLemma	Depword
DepwordConstituentPOSPattern+DepwordLemma	PredicateLemma
ConstituentPOSPattern+HeadwordLemma	PredicateSense
DepwordConstituentPOSPattern+DepwordHeadwordLemma	PredicateSense+DepRelation
DepRelation	PredicateSense+DepwordLemma
DepwordDepRelation	PredicateSense+DepwordPOS
DepRelation+DepwordLemma	SiblingsPOS
DepwordDepRelation+DepwordLemma	DepwordSiblingsPOS
DepRelation+Headword	SiblingsPOSNoDup
DepwordDepRelation+DepwordHeadword	DepwordSiblingsPOSNoDup
DepRelation+HeadwordLemma	SiblingsREL
DepwordDepRelation+DepwordHeadwordLemma	DepwordSiblingsREL
DepRelation+HeadwordLemma+DepwordLemma	SiblingsRELNoDup
DepwordDepRelation+DepwordHeadwordLemma+DepwordLemma	DepwordSiblingsRELNoDup
DepRelation+HeadwordPOS	RelationPath
DepwordDepRelation+DepwordHeadwordPOS	UpPath
DepwordLemma	UpPathLength
DepwordLemma+HeadwordLemma	UpRelationPath
DepwordLemma+DepwordHeadwordLemma	UpRelationPath+HeadwordLemma
DepwordLemma+RelationPath	Distance
PredicatePOS	Predicate_{-1}
DepwordPOS	Depword_{-1}
DepwordPOS+HeadwordPOS	$\text{PredicateLemma}_{-1}$
DepwordPOS+DepwordHeadwordPOS	DepwordLemma_{-1}
FirstLemma	PredicatePOS_{-1}
DepwordFirstLemma	DepwordPOS_{-1}
FirstPOS	Predicate_{+1}
DepwordFirstPOS	Depword_{+1}
FirstPOS+DepwordPOS	$\text{PredicateLemma}_{+1}$
DepwordFirstPOS+DepwordPOS	DepwordLemma_{+1}
FirstWord	PredicatePOS_{+1}
DepwordFirstWord	DepwordPOS_{+1}
Headword	PredicateSense+PredicateLemma
DepwordHeadword	PredicateSense+PredicatePOS
HeadwordLemma	PredicateSense+DepwordPOS
DepwordHeadwordLemma	FirstDepRelation
HeadwordLemma+RelationPath	Predicate+ChildrenRELNoDup
DepwordHeadwordLemma+RelationPath	PredicatePOS+ChildrenRELNoDup
HeadwordPOS	DepwordFirstDepRelation+Depword
DepwordHeadwordPOS	DepwordLastDepRelation+Depword
LastLemma	$\text{Depword+Depword}_{+1}$
DepwordLastLemma	Depword+DepwordChildrenPOS
LastPOS	DepwordPOS+DepwordChildrenPOSNoDup
DepwordLastPOS	DepwordLemma+PredicateLemma
LastWord	$\text{PredicateLemma+PredicateLemma}_{+1}$

Table A.7: Classification feature templates for Semantic Role Labeling CoNLL-2009 shared task, semantic roles subtask. Features adopted from Che et al. (2009) and Zhao et al. (2009).

Form	Num	Suffix1	Prefix1
Form_{-1}	Cap	Suffix2	Prefix2
Form_{-2}	Dash	Suffix3	Prefix3
Form_{+1}	POS_{-1}	Suffix4	Prefix4
Form_{+2}	POS_{-1} POS_{-2}		

Table A.8: Classification feature templates for POS tagging.