Syntax Parsing of Morphologically Rich Languages and Its Application

PhD Thesis

Zsolt Szántó

Supervisor: Richárd Farkas, PhD

Doctoral School of Informatics

Department of Computer Algorithms and Artificial Intelligence

Faculty of Science and Informatics

University of Szeged

Szeged
2024

Contents

List of Tables

List of Figures

Chapter 1

Introduction

The internet is a vast repository of information that is constantly growing, with new pages being created and updated every day. Much of this information is in the form of text, such as news articles, blog posts, and product descriptions.

Natural language processing (NLP) is a field of computer science that deals with the interaction between computers and human language. NLP has a wide range of applications, including machine translation, text summarization, and question answering. One of the key challenges in NLP is understanding the syntactic structure of human language sentences. Syntax determines the order of words, how words of a sentence relate to each other, and how phrases and clauses are built within a sentence.

Syntactic parsing is the process of analyzing the structure of sentences to uncover their underlying grammatical relationships. Because of the complexity and ambiguity of human language, syntactic parsers mostly rely on machine learning-based solutions.

In most cases, syntactic parsing is used as a tool for higher-level text-processing applications like information extraction, machine translation, or sentiment analysis. For example, in aspect-based sentiment analysis we aim to assign different sentiment values to different parts of the text. Take the following social media post as an example:

(1) I love the last jedi, but not a fan of the rise of skywalker

There is positive sentiment about The Last Jedi Star Wars movie, but for The Rise of Skywalker the writer shared a negative opinion. If we know the syntax of the sentence we can easily see the `love` is connected to the `last jedi` and the `not a fan` is related to the `rise of skywalker`.

Features extracted from syntax parses has led to state-of-the-art machine learning application in many text processing fields (Björne et al., 2009; Lapponi et al., 2012; Johansson and Moschitti, 2013) since 2006. Since 2019, a major trend in natural language processing research has been the use of end-to-end approaches based on large, pre-trained neural language models. These models are often fine-tuned for specific applications. Although incorporating syntactic parsing as an intermediate step can further enhance the effectiveness of

these methods (Zeng et al., 2019), the added value of externally given syntactic information is much lower than previously.

While deep learning solutions often achieve high accuracy, the demand for interpretable output remains crucial in real-world language processing systems. Industrial applications are frequently fully or partially rule-based solutions, as (sufficient) training data for a pure machine learning solution is not available and each and every real-world application has its own requirements. Moreover, rule-based components provide tight control over the behavior of the systems in contrast to other approaches. Experts in a particular field can design application-specific rules based on the relationship of certain words thanks to syntactic parsing. In general, although their relevancy has been decreased, we believe that syntactic parsers are useful, even in the Large Language Model era.

The most popular approaches to syntactic parsing are constituent parsing and dependency parsing, each offering a unique perspective on sentence structure. Constituency parsing breaks down sentences into nested phrases (e.g., noun phrases, verb phrases), exposing the hierarchical structure of language. This type of analysis facilitates applications that require an understanding of how sentence components function together. Dependency parsing, on the other hand, shows the direct, grammatical connections between words, highlighting their roles within a sentence. This information is vital for tasks where accuracy depends on precise interpretation of linguistic relationships such as information extraction.

Besides potential applications, this thesis focuses on Hungarian and other morphologically rich languages, that express syntactic information at the level of the morphology of the words instead of encoding it in the word order. Among its contributions, the thesis presents techniques for constituent and dependency parsing that achieve state-of-the-art accuracy on morphologically rich languages and in some cases at least competitive results. Additionally, it introduces methods for automatic, rule-based conversion between constituent and dependency corpora, as well as between different dependency representations for Hungarian that were used to create the Hungarian Universal Dependencies dataset.

1.1 Structure of the Dissertation

The document is separated into three parts, each describing a specific field of syntactic parsing: constituent parsing, dependency parsing, and the high level application of these tools.

Part I introduces novel approaches for constituent parsing, especially for morphologically rich languages. To enhance the efficiency of parsing systems, it introduces techniques like a novel preterminal merger procedure and leveraging external corpora within the lexical model. This part also shows improvement on the reranking step of constituent parsers and demonstrates that incorporating features based on morphological details leads to improved outcomes for morphologically rich languages (Szántó and Farkas, 2014; Szántó and Farkas, 2015).

Part II provides an overview of Hungarian dependency parsing. It delves into the relationship between

			Chapters				
			3	4	5	6	7
EACL	2014	Szántó and Farkas (2014)	•				
ACTA	2015	Szántó and Farkas (2015)	•				
COLING	2014	Simkó et al. (2014)		•			
EACL	2017	Vincze et al. (2017)		•			
TSD	2023	Orosz et al. (2023)		•			
ICCIA	2017	Hangya et al. (2017)			•		
EPE	2017	Szántó and Farkas (2017)				•	
AIAI	2023	Szántó et al. (2023)					•

Table 1.1: Connection between the chapters of the thesis and the corresponding publications.

dependency and constituent parsing in Hungarian (Simkó et al., 2014). Additionally, it analyzes the development of the Universal Dependency dataset for Hungarian (Vincze et al., 2017). Finally, the chapter highlights HuSpaCy, a Hungarian language processing framework that provides a cutting-edge dependency parser (Orosz et al., 2023).

Part III introduces three application of syntactic parsing, exploring its potential in three distinct areas. Firstly, it examines how utilizing syntactic structures can enhance sentiment analysis (SA) on both individual words and entire sentences (Hangya et al., 2017). Secondly, it delves into Team Szeged's participation in the First Shared Task on Extrinsic Parser Evaluation (EPE 2017), presenting three methods that leverage the challenge's shared generalized dependency graph representation (Szántó and Farkas, 2017) for various downstream applications. Finally, the chapter explores strategies for medication event extraction and classification, demonstrating how syntax-based information extraction can lead to efficiency gains even on the top of pre-trained large language models (Szántó et al., 2023).

Table 1.1 outlines the connections between the thesis chapters and the key publications they reference.

1.2 List of Publications

- Szántó, Z., Farkas, R.: Special techniques for constituent parsing of morphologically rich languages. In: Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics. pp. 135–144. Gothenburg, Sweden (April 2014)

- Szántó, Z., Farkas, R.: Constituency parse reranking for morphologically rich languages. Acta Polytechnica Hungarica 12(8) (2015)

- Simkó, K.I., Vincze, V., Szántó, Zs., Farkas, R.: An Empirical Evaluation of Automatic Conversion from Constituency to Dependency in Hungarian. In: Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers. pp. 1392–1401 (2014)

- Vincze, V., Simkó, K., Szántó, Z., Farkas, R.: Universal Dependencies and morphology for Hungarian

- and on the price of universality. In: Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers. pp. 356–365. Association for Computational Linguistics, Valencia, Spain (Apr 2017)

- Orosz, G., Szabó, G., Berkecz, P., Szántó, Z., Farkas, R.: Advancing hungarian text processing with huspacy: Efficient and accurate nlp pipelines. In: International Conference on Text, Speech, and Dialogue. pp. 58–69. Springer (2023)

- Hangya, V., Szántó, Z., Farkas, R.: Latent syntactic structure-based sentiment analysis. In: 2017 2nd IEEE International Conference on Computational Intelligence and Applications (ICCIA). pp. 248–254. IEEE (2017)

- Szántó, Z., Farkas, R.: Szeged at epe 2017: First experiments in a generalized syntactic parsing framework. In: Proceedings of the 2017 Shared Task on Extrinsic Parser Evaluation at the Fourth International Conference on Dependency Linguistics and the 15th International Conference on Parsing Technologies. Pisa, Italy. pp. 75–79 (2017)

- Szántó, Z., Bánáti, B., Zombori, T.: Enhancing medication event classification with syntax parsing and adversarial learning. In: Maglogiannis, I., Iliadis, L., MacIntyre, J., Dominguez, M. (eds.) Artificial Intelligence Applications and Innovations. pp. 114–124. Springer Nature Switzerland, Cham (2023)

Chapter 2

Background

2.1 Syntax Representations

Syntax describes the structure of the sentence and the grammatical relations between the words. In computational linguistics, syntactic parsing generally supports higher level tasks, since the knowledge of the syntactic structure of a sentence can contribute to many natural language processing applications like machine translation and information retrieval.

Two prominent approaches to syntax representation are the constituent and dependency representation, both describe a sentence as a tree but in a very different way.

2.1.1 Constituent Representation

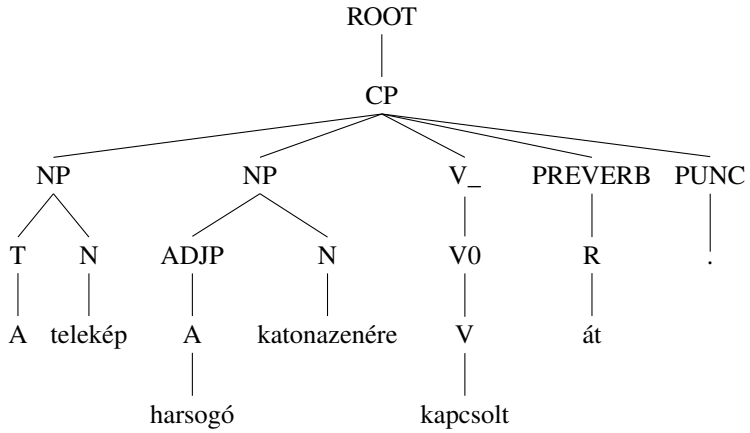
Constituent representation is based on the idea that certain groups of words, or constituents, form cohesive units that function together within a sentence. These constituents can be of various types, such as noun phrases (NP), verb phrases (VP), or prepositional phrases (PP). These constituents form higher level groups and make a tree over a sentence. Merging larger and larger constituents culminates in the complete sentence structure (Jurafsky and Martin, 2008).

The constituent representation of a sentence is a tree consisting of words at the as leaves and abstract constituent nodes over it that groups the words into cohesive units.

In Figure 2.1 we can see how noun phrases can be made up of - for example - a determiner and a noun (`A telekép`) or an adjective phrase (ADJP) and a noun (`harsogó katonazenére`) and how all the elements of the sentence combine to make a sentence (CP). The part of speech tag of each word is indicated in the node directly above it, while the higher level nodes describe the type of constituent (e.g. `V_` for the verb or `ADJP` for the adjective phrase).

Constituent representation can be formalized by context-free grammars (CFG), which are formal grammars where each rule is of the form $A \rightarrow \alpha$ where A is a nonterminal symbol, and α is a sequence of terminals

Figure 2.1: Hungarian sentence from the novel 1984, the constituent tree came from the Szeged Treebank. In English: *The telescreen had changed over to strident military music.*



and nonterminals.

For natural languages, such rules can be, for example:

- $CP \rightarrow NP\ NP\ VP$
- $NP \rightarrow N\ NP$
- $N \rightarrow\ beer$

where the first rule states that a sentence (`CP`) can consist of two noun phrases (`NP`) and a verb phrase (`VP`). According to the second rule, a noun phrase (`NP`) can be made up of a noun (`N`) and a noun phrase (`NP`). The third rule states that the noun (`N`) can be the word `beer`.

In the constituent trees that can be generated from the grammar, the parts of speech are always found directly above the words, which is why the nonterminals belonging to different parts of speech are also called preterminals.

The individual rules can be grouped according to whether they are syntactic or lexical rules. The left-hand side of a syntactic rule is a syntactic phrase and the right-hand side is made up of nonterminals, while the left-hand side of lexical rules is a preterminal (part-of-speech tag) that generates a word. Thus, the first two rules in the example above are syntactic, while the third is lexical.

2.1.2 Dependency Representation

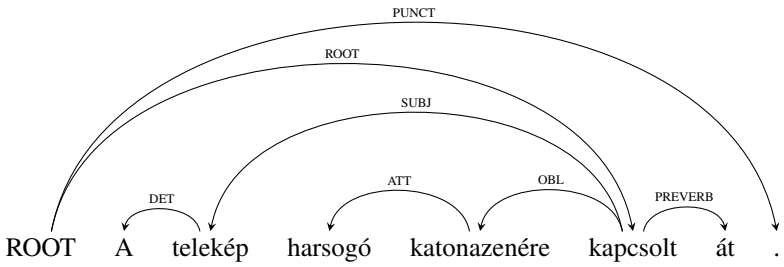
One of the basic ideas of dependency grammar is that there are components that are not next to each other in the sentence, but intuitively they still belong together. Based on this assumption, the dependency description

does not group the words but directly describes the relationships between the words of the sentence (Jurafsky and Martin, 2008).

Although the sentence is represented as a tree, similar to constituent grammar, here each node is a word and additional information is described by the labels of the edges instead of adding abstract nodes. In contrast to the traditional constituent representation, the labels directly define grammatical roles such as subject or object. The edges are directed, each word has one superordinate (or head) but may have multiple subordinates. The edges can also cross each other, these are also called non-projective trees.

In Figure 2.2 we see the dependency grammar analysis of the same sentence of 1984 as above. Here we see that the main, highest level word of the sentence is the verb, `kapcsolt`. The preverb and the verb's other arguments are linked directly to it with their grammatic role indicated on the labels of the edges. For example the subject of `kapcsolt`, `telekép` is attached with an edge labeled `subj`. The modifiers of these nouns are connected directly to them and their roles are again indicated on the labels of the edges; a determiner attached to `telekép` and an adjective modifier attached to `katonazenére`.

Figure 2.2: Hungarian sentence from the novel 1984, the dependency tree came from the Szeged Dependency Treebank. In English: *The telescreen had changed over to strident military music.*



2.1.3 Other Representations

While the constituent and dependency grammars are the most mainstream representations in the field of natural language processing, there are a lot of different ways to model the syntactic structure of a sentence. For example, functional grammar (LFG) and head-driven phrase structure grammar (HPSG) both have annotated corpora on multiple languages.

LFG distinguishes two main types of syntactic structure: c-structure, which focuses on word order and how words group together, and f-structure, which centers on grammatical roles like subject and object, aligning with dependency grammar principles. The system also allows for other levels of annotation to store different types of linguistic information (e.g. phonological, morphological, or semantic information) in an interconnected way (Simkó et al., 2014; Rákosi and Laczkó, 2013).

HPSG evolved from the principles of constituent grammar and the name 'head-driven' highlights the critical role of information contained in the lexical heads of syntactic phrases (Müller et al., 2021; Szécsényi, 2011).

2.2 Syntactic Parsing

Early syntactic parsing research relied on rule-based systems designed by linguistic experts. While effective in some cases, the complexity and adaptability of natural language exposed limitations on the ability of manually constructed rules to capture all syntactic phenomena. In recent decades, the field has experienced a shift towards machine learning-based approaches. These methods offer the potential to automatically learn complex syntactic patterns from data, often addressing challenges that proved difficult for rule-based systems. This shift was driven by the availability of large, annotated datasets, which enabled researchers to train and evaluate machine learning models on a scale that was previously infeasible. These datasets, such as the Penn Treebank (Marcus et al., 1993) for English and Szeged Treebank (Csendes et al., 2005) for Hungarian, provide detailed syntactic annotations for large corpora of text, allowing researchers to train models that can learn complex patterns and make generalizations about unseen data.

While the theoretical background of consistent representation came from the late 1950s by Noam Chomsky, the data-driven statistical parsing approaches have become popular since the appearance of the first large-sized English syntactically annotated corpus the Penn Treebank (Marcus et al., 1993).

Dependency treebanks started to appear at the beginning of the 2000s. One of the most well-known datasets is the Prague Dependency Treebank (Hajič, 1998) for the Czech language, which was followed by many other treebanks mostly for languages with rich morphology (Kakkonen, 2005).

2.2.1 Constituent Parsing

The field of statistical constituent parsing was dominated by probabilistic context-free grammars (PCFG) (Charniak, 2000; Charniak and Johnson, 2005; Petrov et al., 2006a) in the last decades. They combine the expressive power of context-free grammars (CFGs) with the ability to assign probabilities to constituent trees. This allows PCFGs to make better predictions about the structure of sentences.

A PCFG consists of a set of productions, each of which specifies a rule for generating a syntactic structure. Each production has a left-hand side, which is a nonterminal symbol, and a right-hand side, which is a sequence of terminals and nonterminals. In addition, each production is assigned a probability, which represents the likelihood of that rule generating the corresponding syntactic structure.

PCFGs can be used to parse sentences, which means assigning a constituent tree to a given sentence. Parsing with PCFGs involves finding the constituent tree that has the highest probability according to the grammar. This can be done using a dynamic programming algorithm called the CYK algorithm that efficiently calculates the probabilities of all possible parse trees for a given sentence.

The Berkeley Parser (Petrov et al., 2006a) has several changes over the standard PCFG parsers but the most important change is the splitting of the nonterminals to latent substates. During the training process, the Berkeley Parser automatically splits and merges these substates to enrich grammar trees with additional information not explicitly present in the original constituent trees.

Recent advancements in natural language processing have led to the development of deep neural parsers. These parsers leverage neural networks to learn complex representations of language, often leading to improved accuracy and the ability to identify more nuanced syntactic relationships compared to traditional PCFG models.

Current state-of-the-art systems apply deep learning with transformer-based architectures (Tian et al., 2020). These parsers leverage neural networks to learn complex representations of language, often leading to improved accuracy and the ability to identify more nuanced syntactic relationships compared to traditional PCFG models.

Despite the recent popularity of dependency parsers, the best paper award at the ACL 2022 conference, won by Kitaev et al. (2022), demonstrates the continued relevance of constituent parsers. They describe an incremental syntactic representation that analyzes sentences word-by-word. Each word receives a unique latent label based on the words that came before it. The constituent tree is calculated by only using these latent labels.

Evaluation Metrics

PARSEVAL score is the most common metric to evaluate the efficiency of constituent parsing systems. PARSEVAL is an F_1 metric where the precision is calculated by determining the number of correct constituents in the parser output divided by the total number of constituents in the parser output, while recall is computed by finding the number of constituents from the gold standard that are present in the parser output divided by the total number of constituents in the gold standard.

In this thesis, the PARSEVAL score and the ratio of exactly matching parse trees are applied to compare the different systems.

2.2.2 Dependency Parsing

Dependency parsing techniques have two main categories, transition-based and graph-based parsers.

Transition-based parsers construct a dependency tree incrementally, making decisions step-by-step. They resemble a machine navigating a sentence, taking actions like shifting words onto a stack or attaching them to form dependencies. Nivre (2010)

Graph-based parsers, on the other hand, contemplate the entire sentence at once, identifying the most coherent dependency structure from a global perspective. They often employ algorithms to search for the highest-scoring dependency tree, considering all possible connections between words. Nivre (2010)

We used the graph-based module of the Bohnet parser (Bohnet, 2010a) in many of our experiments. It uses advanced features (sibling/grandchild info, relation labels), and passive-aggressive perception for

learning while it utilizes hash kernels for quicker processing, achieving high accuracy and efficiency during training and testing.

Nowadays, large language model-based parsers achive the highest accuracy. Section 4.4 will demonstrate HuSpaCy (Orosz et al., 2023), an easy-to-use deep learning system, that achieves one of the best results reported for Hungarian.

Evaluation Metrics

The most common metrics used in dependency parsing are the labeled attachment score (LAS) and the unlabeled attachment score (ULA). It calculates the percentage of tokens that have been assigned the correct head, regardless of whether the dependency label is considered.

2.3 Morphologically Rich Languages

From the viewpoint of syntax, the languages of the world are usually categorized according to their level of morphological richness (which is negatively correlated with configurationality). At one end, there is English, a strongly configurational language while there is Hungarian at the other end of the spectrum with rich morphology and free word order (Fraser et al., 2013). A large part of the methodology for syntactic parsing has been developed for English but many other languages of the world are fundamentally different from English. In particular, morphologically rich languages – the other end of the configurational spectrum – convey most sentence-level syntactic information by morphology (i.e. at the word level), not by configuration. Because of these differences, the parsing of morphologically rich languages requires techniques that differ from or extend on the methodology developed for English (Tsarfaty et al., 2013).

For the constituent parsing of morphologically rich languages, I present techniques that led to state-of-the-art results on 8 languages.

2.4 Syntax Parsing in Hungarian

Hungarian is one of the rare examples with manual annotations for both constituency and dependency syntax on the same bunch of texts which allows for the comparison of these datasets with each other and the creation of suitable machine learning-based solutions for both representations.

2.4.1 Hungarian Syntax Datasets

For Hungarian, the first large, manually annotated corpus for computational linguistics is the Szeged Corpus (Csendes et al., 2004).Although the Szeged Corpus does not consist of syntactic annotation, but most of the currently available Hungarian syntactic corpora use the dataset that was collected then.

Szeged Corpus

The Szeged Corpus contains over 82,000 sentences, roughly 1.2 million words. Its thematic diversity ensures a nuanced representation of the language, spanning across domains such as fiction – by Jenő Rejtő, Antal Szerb, and George Orwell –, student essays, factual news articles, technical manuals, and even legal texts. This breadth allows researchers to analyze Hungarian not only in its various stylistic registers but also across a spectrum of informative and communicative functions. Every word in the text is annotated with its part of speech (POS) tags, morphological description, and lemma. The original version of the Szeged Corpus used a modified version of the MSD code system (Erjavec, 2012) for the POS tags and morphological description.

Szeged Treebank

Extending the Szeged Corpus, the Szeged Treebank (Csendes et al., 2005) overlays manually annotated constituent trees upon its sentences, providing a syntactic representation of the text. The corpus uses a total of 13 syntactic tags, of which the most important are:

- CP, which marks individual clauses
- NP, which delimits noun phrases
- V_, which contains the verb

In addition, the corpus also marks the arguments of the verb, the participle, and the infinitive. These denote syntactic roles such as subject or object, pointing to another word in the clause. I will not discuss these in more detail as they are usually added to the constituency trees as part of a post-processing procedure and were not used in my experiments.

Szeged Dependency Treebank

The Szeged Dependency Treebank (Vincze et al., 2010) is a further expansion of this data. The base structures of the Szeged Treebank constituency trees were automatically converted to a dependency format, which was then reviewed and corrected by linguists. Thus the data of the Szeged Corpus has two different types of parallel syntactic annotation.

Hungarian Universal Dependencies Corpus

The Universal Dependencies Project (Nivre, 2015) aimed to develop a POS tagging, morphology, and dependency grammar framework that could be used to annotate all languages consistently. The Hungarian Universal Dependencies corpus (Vincze et al., 2017) contains 1800 sentences by applying the Universal Dependencies annotations schemes. The annotations in this corpus were automatically converted and manually corrected from the Szeged Dependency Treebank. The creation of this dataset will be described in more detail in the section 4.3.

2.4.2 Hungarian Syntax Parsers

Nowadays the three largest Hungarian text-processing frameworks the magyarlanc (Zsibrita et al., 2013), emtsv (Simon et al., 2020; Indig et al., 2019; Váradi et al., 2018), HuSpaCy(Orosz et al., 2023) consists of dependency parsers and the first two contain constituent parsers also.

The first parsing results on the Szeged Treebank were published by Barta et al. (2005); Iván et al. (2007). Around the same time, a few studies reported on hand-crafted parsers using different, smaller corpora (Babarczy et al., 2005; Prószéky et al., 2004).

After a long break, Farkas et al. (2012) produced results on Hungarian dependency parsing, and we (Szántó and Farkas, 2014) created a constituent parsing system that reached state-of-the-art results on morphologically rich languages like Hungarian. These methods became part of the magyarlanc and the emtsv.

The dependency parsers of the magyarlanc and the emtsv were trained on the tagset of the Szeged Dependency Treebank, meanwhile, in the development of HuSpaCy our main goal was using the modern Universal Dependencies tagset to join the international trends. While the size of the UD dataset is much smaller than the Szeged Dependency Treebank, the deep neural network-based parser of the HuSpaCy learned a comparable model from that.

Part I

Constituent Parsing

Chapter 3

Constituent Parsing of Morphologically Rich Languages

This chapter presents different techniques to improve constituent parsing; especially for handling the challenges of morphologically rich languages. Section 3.3 introduces my proposals that utilize the information from the constituent trees, while section 3.4 demonstrates my approaches that use external information like large amounts of unlabeled data and dependency trees.

3.1 Related Work

Constituent parsing of English is a well researched area. While the constituent parsing of morphologically rich languages is a much less investigated field.

3.1.1 Reranking

The constituent parsing systems usually use two steps. The first step is a PCFG parser which selects the best parses from all possible trees.

The second step is usually a discriminative n-best reranking step. These reranking systems can improve the performance of first-stage PCFG parsing methods. The first-stage parser must be fast to be able to select the best trees from all the possible parses, and the reranker can extract a rich feature set to describe the n best parses of the original parser.

3.1.2 Morhologically Rich Languages

In prior work on data-driven syntactic parsing of morphologically rich languages, it has been shown that parsers developed for English struggle with the complexity introduced by morphologically rich languages.

The Statistical Parsing of Morphologically Rich Languages (SPMRL) Seddah et al. (2013) workshop series aims to foster the development of parsing techniques dedicated to morphologically rich languages.

Before the SPMRL shared tasks there existed constituent treebanks for several languages along with a very limited number of parsing reports on them. For instance, Petrov (2009) trained BerkeleyParser on Arabic, Bulgarian, French, German and Italian and he reported good accuracies, but there has been previous work on Hebrew (Goldberg and Elhadad, 2013), Korean (Choi et al., 1994) and Spanish (Le Roux et al., 2012) etc. The SPMRL shared task series addressed the dependency and constituency parsing of nine morphologically rich languages and provided useful benchmark datasets for these languages.

One of our chief contributions to this area is a procedure to merge preterminal labels. The related work for this line of research includes the studies on manual refinement of preterminal sets such as Marton et al. (2010) and Le Roux et al. (2012). The most closely related approach to our proposal is Dehdari et al. (2011), who defines metaheuristics to incrementally insert or remove morphological features. Their approach uses the parser – training and parsing – as a black box evaluation of a preterminal set. In contrast, our proposal operates as a submodule of the BerkeleyParser, hence does not require the re-training of the parser for every possible preterminal set candidate, thus it is way faster.

The most successful supervised constituent parsers contained a second feature-rich discriminative parsing step (Charniak and Johnson, 2005; Huang, 2008a; Chen and Kit, 2012) as well. At the first stage they apply a PCFG to extract possible parses. The *n-best list parsers* keep just the 50-100 best parses according to the PCFG (Charniak and Johnson, 2005). These methods employ a large feature set (usually a few million features) (Collins, 2000; Charniak and Johnson, 2005). These feature sets are engineered for English. We introduce feature templates for exploiting morphological information and investigate their added value over the standard feature sets.

Our other chief contribution in this chapter is the introduction of three more feature sets for morphologically rich languages in the second stage reranking. Previously, the dependency based features were successfully applied to German (Farkas et al., 2011). Here we experimented with them on five morphologically rich languages. The morphological features were designed especially for morphologically rich languages. To the best of our knowledge, the Brown clustering (Brown et al., 1992) based features had not been previously used in the context of reranking.

3.2 Experimental Setup

3.2.1 The SPMRL Datasets

The Statistical Parsing of Morphologically Rich Languages (SPMRL) workshop series aims to foster the development of parsing techniques dedicated to morphologically rich languages. By providing standard data, evaluation tools, and strong baselines, it empowered researchers and drove advancements in this crucial area. In 2013, the first shared task on parsing morphologically rich languages was organized, which contains challenges in the two most commonly used syntactic frameworks (dependency and constituency) on nine

	Basque	French	German	Hebrew	Hungarian
#sent. in training	7577	14759	40472	5000	8146
#sent. in dev	948	1235	5000	500	1051
#sent. in test	946	2541	5000	716	1009
avg. token/sent.	12.92	30.13	17.51	25.33	21.76
#non-terminal labels	3000	770	994	1196	890
#main POS labels	16	33	54	46	16
unknown token ratio (dev)	18.35%	3.22%	6.34%	9.65%	19.94%

Table 3.1: Basic statistics of the treebanks used.

morphologically rich languages (Arabic, Basque, French, German, Hebrew, Hungarian, Korean, Polish, and Swedish).

The SPMRL 2014 Shared Task (Seddah et al., 2014b) was a direct extension of this and it also involved parsing both dependency and phrase-structure representations. The only difference between the two tasks is that large amounts of unlabeled data were additionally available to participants for the 2014 task.

We conducted experiments on the treebanks of the 2013 and SPMRL shared tasks. We used the train/dev/test splits of the shared task's Basque (Aduriz et al., 2003), French (Abeillé et al., 2003), Hebrew (Sima'an et al., 2001), German (Brants et al., 2002) and Hungarian (Csendes et al., 2005) treebanks. Table 3.1 shows the basic statistics of these treebanks, for a more detailed description of their annotation schemata, domain, preprocessing etc. see Seddah et al. (2013).

The second part of our experiments utilizes information from unlabeled data that we could do because the SPMRL 2014 shared task extended the first dataset with large unlabeled corpora.

Hungarian corpus The newspaper sub-corpus of the Szeged Treebank and the Szeged Dependency Treebank (Vincze et al., 2010) were used as the Hungarian treebanks of the shared task as the organizers collected treebanks only from the newspaper domain for each language. The unlabeled data is made up of 1747239 sentences of newspaper articles from the Hungarian National Corpus (Váradi, 2002). We provided automatic POS-tagging and dependency parsing using magyarlanc (Zsibrita et al., 2013) for the unlabeled data to the shared task organizers.

3.2.2 Evaluation Metrics

Like the shared task, we employed the PARSEVAL score (Abney et al., 1991) along with the exact match accuracy (i.e. the ratio of perfect parse trees). The PARSEVAL score is an F-measure over the syntax tree as parentheses, an internal node is correct if both the covered words and the non-terminal label match the tree in the gold standard treebank. The preterminals themselves do not count in the evaluation metric. We used the evalb implementation of the shared task[1].

[1] Available at http://pauillac.inria.fr/~seddah/evalb_spmrl2013.tar.gz. An important change in this version compared to the original evalb is the penalization of unparsed sentences.

3.3 Special Techniques for Constituent Parsing of Morphologically Rich Languages

We propose answers to the two main challenges of constituent parsing of morphologically rich languages, which are finding the optimal preterminal set and handling the huge number of wordforms.

The size of the preterminal set in the standard context free grammar environment is crucial. If we use only the main POS tags as preterminals, we lose a lot of information encoded in the morphological description of the tokens. On the other hand, using the full morphological description as preterminal yields a set of over a thousand preterminals, which results in data sparsity and performance problems as well. The chief contribution of this work is to propose a novel automatic procedure to find the optimal set of preterminals by **merging morphological feature values**. The main novelties of our approach over previous work are that it is very fast – it operates inside a probabilistic context free grammar (PCFG) instead of using a parser as a black box with re-training for every evaluation of a feature combination – and it can investigate particular morphological feature values instead of removing a feature with all of its values.

Another challenge is that because of the inflectional nature of morphologically rich languages, the number of forms a single word has is much higher than in English. Hence the number of unknown and very rare tokens – i.e. the tokens that do not appear in the training dataset – is higher here, which hurts the performance of PCFG parsers. Following Goldberg and Elhadad (2013), we **enhance the lexical model** by utilizing an external lexicon. We investigate the applicability of fully supervised taggers instead of unsupervised ones for gathering external lexicons.

Lastly, we introduce novel feature templates for an n-best reranker operating on the top of a PCFG parser. These feature templates are **using atomic morphological features** and achieve improvements over the standard feature set engineered for English.

We conducted experiments by the above mentioned three techniques on Basque, French, German, Hebrew and Hungarian, five morphologically rich languages from the SPMRL 13 dataset. The BerkeleyParser enriched with these three techniques achieved state-of-the-art results on each language (Szántó and Farkas, 2014).

3.3.1 Lexical Sparsity

Before introducing our proposal and experiments with preterminal set optimisation, we have to offer a solution for the out-of-vocabulary (OOV) problem, which – because of their inflectional nature – is a crucial problem in morphologically rich languages. We follow here Goldberg and Elhadad (2013) and enhance a lexicon model trained on the training set of the treebank with frequency information about the possible morphological analyses of tokens. We estimate the tagging probability $P(t|w)$ of the tag t given the word w by

$$P(t|w) = \begin{cases} P_{tb}(t|w), & \text{if } c(w) \geq K \\ \frac{c(w)P_{tb}(t|w)+P_{ex}(t|w)}{1+c(w)}, & \text{otherwise} \end{cases}$$

where $c(w)$ is the count of w in the training set, K is predefined constant, $P_{tb}(t|w)$ is the probability estimate from the treebank (the relative frequency with smoothing) and $P_{ex}(t|w)$ is the probability estimate from an external lexicon. We calculate the emission probabilities $P(w|t)$ from the tagging probabilities $P(t|w)$ by applying the Bayesian rule.

The key question here is how to construct the external lexicon. For a baseline, Goldberg and Elhadad (2013) suggest using the uniform distribution over all possible morphological analyses coming from a morphological analyser ('uniform').

Goldberg and Elhadad (2013) also report considerable improvements over the 'uniform' baseline by relative frequencies counted on a large corpus which was automatically annotated in the unsupervised POS tagging paradigm (Goldberg et al., 2008). Here we show that even a supervised morphological tagger without a morphological analyzer can achieve the same level of improvement. We employ MarMot[2] (Mueller et al., 2013) for predicting full morphological analysis (i.e. POS tags and morphological features jointly). MarMot is a Conditional Random Field tagger which incrementally creates forward-backward lattices of increasing order to prune the sizable space of possible morphological analyses. We used MarMoT with the default parameters. This purely data-driven tagger achieves a tagging accuracy of 97.6 evaluated at full morphological analyses on the development set of the Hungarian treebank, which is competitive with the state-of-the-art Hungarian taggers which employ language-specific rules (e.g. magyarlanc (Zsibrita et al., 2013)). The chief advantage of using MarMot instead of an unsupervised tagger is that the former does not require any morphological lexicon/analyser (which can lists the possible tags for a given word). This morphological lexicon/analyser is language-dependent, usually hand-crafted and it has to be compatible with the treebank in question. In contrast, a supervised morphological tagger can build a reasonable tagging model on the training part of the treebanks – especially for morphologically rich languages, where the tag ambiguity is generally low – thus each of these problems is avoided.

Table 3.2 shows the results of various $P_{ex}(t|w)$ estimates on the Hungarian development set. The first row 'BerkeleyParser' is our absolute baseline, i.e. the original implementation of BerkeleyParser[3] defining signatures for OOVs. For the 'uniform' results, we used the morphological analyser module of magyarlanc (Zsibrita et al., 2013). The last two rows show the results achieved by training MarMot on the treebank's training dataset, having tagged the development set plus a huge unlabeled corpus (10M sentences from the Hungarian National Corpus) with it then having counted relative tag frequencies. We report scores on only using the frequencies from the development set (`dev`) and from the concatenation of the development set and the huge corpus (`huge`).

After a few preliminary experiments, we set $K = 7$ and use this value thereafter.

[2] https://code.google.com/p/cistern/
[3] http://code.google.com/p/berkeleyparser/

	PARSEVAL	EX
BerkeleyParser	87.22	12.75
uniform	87.31	14.78
dev	88.29	15.22
huge	89.27	16.97

Table 3.2: The results achieved by using various external lexical models on the Hungarian development set.

Table 3.2 shows that even 'dev' yields a considerable improvement over the baseline parser and 'uniform'. These results are also in line with the findings of Goldberg and Elhadad (2013), i.e. 'uniform' has some added value and using relative frequencies gathered from automatically tagged corpora contributes more. Although we can see another nice improvement by exploiting unlabeled corpora ('huge'), we will use the 'dev' setting in the experiments of the next sections as we did not have access to huge, in-domain unlabeled corpora for each language used in the SPMRL 13 dataset.

3.3.2 Morphological Feature Values as Preterminals

Finding the optimal set of morphological features incorporating into the perterminal labels is crucial for any PCFG parsers. Removing morphological features might reduce data sparsity problems while it might lead to loss of information for the syntactic parser. In this section, we propose a novel method for automatically finding the optimal set of preterminals then we present empirical results with this method and compare it to various baselines.

Merge Procedure for Morphological Feature Values: There have been studies published on the automatic reduction of the set of preterminals for constituent parsing. For instance, Dehdari et al. (2011) proposed a system which iteratively removes morphological features as a unit then evaluates the preterminal sets by running the training and parsing steps of a black-box constituent parser. Our motivation here is two-fold. First, morphological features should not be handled as a unit because different values of a feature might behave differently. Take for instance the degree feature in Hungarian adjectives. Here the values positive and superlative behave similarly (can be merged) while distinguishing comparative and positive+superlative is useful for syntactic parsing because comparative adjectives often have an argument (e.g. *x is more beautiful than y*) while positive and superlative adjectives are not syntactic governors thus have no arguments. Second, keeping a morphological feature can be useful for particular POS tags and useless at other particular POS tags (e.g. the number of possessed in Hungarian for nouns and pronouns).

Based on these observations we propose a procedure which starts from the full morphological description of a treebank then iteratively merges particular morphological feature values and it handles the same feature at the different POS tags separately. The result of this procedure is a clustering of the possible values of each morphological feature. The removal of a morphological feature is a special case of our approach because if the values of the feature in question form one single cluster it does not have any discriminative function

Algorithm 1 The preterminal set merger algorithm.

1. training the standard BerkeleyParser using only main POS tags as preterminals
2. merging each subsymbol at the preterminal level
3. for each POS tag - morphological feature pair
   (a) split the POS tag for the values of the morphological feature[4]
   (b) recalculating the rule probabilities where there are preterminals in the right-hand side by uniformly distributing the probability mass among subsymbols
   (c) set the lexical probabilities according to the relative frequencies of morphological values counted on gold standard morphological tags of the treebank
   (d) running 10 iterations of the Expectation-Maximization procedure on the whole treebank initialized with (b)-(c)
   (e) constructing a fully connected graph whose nodes are the morphological values of the feature in question
   (f) for every edge of the graph, calculate the loss in likehood for the merging the two subsymbols (the same way as for BerkeleyParser's merge procedure)
4. removing edges from the entire set of graphs (controlled by the parameter th)
5. merge the morphological values of the graphs' connected components

anymore. Hence our proposal can be regarded as a generalisation of the previous approaches.

This general approach requires much more evaluation of intermediate candidate preterminal sets, which is not feasible within the external black-box parser evaluation scenario (training and parsing an average sized treebank by the BerkeleyParser takes more than 1 hour). Our idea here is that re-training a parser for the evaluation of each preterminal set candidates is not necessary. They key objective here is to select among preterminal sets based on their usefulness for the syntactic parser. This is the motivation of the merge procedure of the BerkeleyParser. After randomly splitting non-terminals, BerkeleyParser calculates for each split the loss in likelihood incurred when merging the subsymbols back. If this loss is small, the new annotation does not carry enough useful information and can be removed. Our task is the same at the preterminal level. Hence at the preterminal level, – instead of using the automatic subsymbol splits of the BerkeleyParser – we call this merging procedure over the morphological feature values. Algorithm 1 shows our proposal for the preterminal merging procedure.

Baseline Preterminal Set Constructions: The two basic approaches for preterminal set construction are the use of only the main POS tag set ('mainPOS') and the use of the full morphological description as preterminals ('full'). For Hungarian, we also had access to a linguistically motivated, hand-crafted preterminal set ('manual') which was designed for a morphological tagger (Zsibrita et al., 2013). This manual code set keeps different morphological features at different POS tags and merges morphological values instead of fully

removing features hence it inspired our automatic merge procedure introduced in the previous section.

Our last baseline is the repetition of the experiments of Dehdari et al. (2011). For this, we started from the full morphological feature set and completely removed features (from all POS) one-by-one then re-trained our parser. We observed the greatest drop in PARSEVAL score at removing the 'Num' feature and the least severe one at removing 'Form'. 'Num' denotes number for verbs and nominal elements (nouns, adjectives and numerals), and since subject-verb agreement is determined by the number and person features of the predicate (the verb) and the subject (the noun), deleting the feature 'Num' results in a serious decline in performance. On the other hand, 'Form' denotes whether a conjunction is single or compound (which is a lexical feature) or whether a number is spelt with letters, Arabic or Roman numbers (which is an orthographic feature). It is interesting to see that their deletion hardly harms the PARSEVAL scores, moreover, it can even improve the exact match scores, which is probably due to the fact that the distinction between different orthographic versions of the same number (e.g. *6* and *VI*) just confused the parser. On the other hand, members of a compound conjunction are not attached to each other in any way in the parse tree, and behave similar to single compounds, so this distinction might also be problematic for parsing.

Results with Various Preterminal Sets: Table 3.3 summarizes the results achieved by our four baseline methods along with the scores of two preterminal sets output by our merger approach at two different merging threshold th value.

	#pt	PARSEVAL	EX
mainPOS	16	82.36	5.52
manual	72	85.38	9.23
full	680	88.29	15.22
full - Num	479	87.43	14.49
full - Form	635	88.24	15.73
merged ($th = 0.5$)	378	88.36	**15.92**
merged ($th = 0.1$)	642	**88.52**	15.44

Table 3.3: The results achieved by using various preterminal sets on the Hungarian development set.

The difference between mainPOS and full is surprisingly high, which indicates that the morphological information carried in preterminals is extremely important for the constituent parser and the BerkeleyParser can handle preterminal sets of the size of several hundred. For Hungarian, we found that the full removal of any feature cannot increase the results. This finding is contradictory with Dehdari et al. (2011) in Arabic, where removing 'Case' yielded a gain of 1.0 in PARSEVAL. We note that baselines for Arabic and Hungarian are also totally different, Dehdari et al. (2011) reports virtually no difference between mainPOS and full in Arabic.

We report the results of our proposed procedure with two different merging thresholds. The $th = 0.1$ case merges only a few morphological feature values and it can slightly outperform the 'full' setting (statistically

	Basque	French	German	Hebrew	Hungarian
mainPOS	68.8/3.9 16	78.4/13.9 33	82.3/38.7 54	88.3/12.0 46	82.6/7.3 16
full	**81.8/18.4** 2976	78.9/15.0 676	**82.3/40.3** 686	88.9/**15.2** 257	88.3/15.2 680
preterminal merger	81.6/16.9 2791	**79.7/15.6** 480	82.3/39.3 111	**89.0**/14.6 181	**88.5/15.4** 642

Table 3.4: PARSEVAL / exact match scores on the development sets. The third small numbers in cells show the size of the preterminal sets.

significant[5] in exact match.). On the other hand, the $th = 0.5$ setting is competitive with the 'full' setting in terms of parsing accuracy but it uses only the third of the preterminals used by 'full'. Although it is not statistically better than 'full' in accuracy, it almost halves the running time of parsing[6].

Table 3.4 summarizes the results achieved by the most important baselines and our approach along with the size of the particular preterminal sets applied. The 'full' results outperform 'mainPOS' at each language with a striking difference at Basque and Hungarian. These results show that – contradictory to the general belief – the detailed morphological description is definitely useful in constituent parsing as well. The last row of the table contains the result achieved by our merger approach. Here we run experiments with several merging threshold th values and show the highest scores for each language.

Our merging proposal could find a better preterminal set than full on French and Hungarian, it found a competitive tag set in terms of accuracies which are much smaller than full on German and Hebrew and it could not find any useful merge at Basque. The output of the merger procedure consists of one sixth of preterminals compared with full. Manually investigating the clusters, we can see that it basically merged every morphological feature except case at nouns and adjectives (but merged case at personal pronouns). This finding is in line with the experimental results of Fraser et al. (2013).

3.3.3 Morphology-based Features in n-best Reranking

After a PCFG parser, n-best rerankers (Collins, 2000; Charniak and Johnson, 2005) are used as a second stage and they usually achieve considerable improvement over the first stage parser. They extract a large feature set to describe the n best output of a PCFG parser and they select the best parse from this set (i.e. rerank the parses). Here, we define feature templates exploiting morphological information and investigate their added value for the standard feature sets (engineered for English). We reimplemented the feature templates from Charniak and Johnson (2005) and Versley and Rehbein (2009) excluding the features based on external corpora and use them as our baseline feature set (`dflt`).

We used $n = 50$ in our experiment and followed a 5-fold-cross-parsing (a.k.a. jackknifing) approach for generating unseen parse candidates for the training sentences (Charniak and Johnson, 2005). The reranker is trained for the maximum entropy objective function of Charniak and Johnson (2005), i.e. the sum of

[5] According to two sample t-test with p<0.001.

[6] Parsing the 1051 sentences of the Hungarian development set takes 15 and 9 minutes with full and $th = 0.5$ respectively (on an Intel Xeon E7 2GHz).

	PARSEVAL	EX
Reranked $_{dflt}$ - merged morph	89.05	18.45
Reranked $_{dflt}$ - mainPOS	89.33	18.64

Table 3.5: The results achieved by using various feature template sets for 50-best reranking on the Hungarian development set.

posterior probabilities of the oracles. We used a slightly modified version of the Mallet toolkit for reranking (McCallum, 2002) and L2 regularizer with its default value for coefficient.

The feature templates of the baseline feature set frequently incorporate preterminals as atomic features. As a first step, we investigated which preterminal set is the most useful for the baseline feature set. We took the 50 best output from the parser using the merged preterminal set and used its preterminals (`merged`) or only the main POS tag (`mainPOS`) as atomic building blocks for the reranker's feature extractor. Table 3.5 shows that mainPOS outperformed full. This is probably due to data sparsity problems.

Based on this observation, we decided to use mainPOS as preterminal in the atomic building block of the baseline features. We designed new feature templates capturing the information in the morphological analysis (`morph`). We experimented with the following templates:

- For each preterminal of the candidate parse and for each morphological feature value inside the preterminal we add the pair of wordform and morphological feature value as a new feature. For instance, if the preterminal N#Cas=n|Num=s# is assigned to the word *cat*, we add two features cat-N-Cas=n and cat-N-Num=s.

- In a similar way, we define a reranker feature from every morphological feature value of the head word of the constituent. E.g. in the (NP (A#SubPOS=f|Deg=p# black) (N#Cas=n|Num=s# cat)) example we add two features for the N head word of the NP: NP-N-Cas=n, NP-N-Num=s.

- For each head-daughter attachment in the candidate parse we add each pair of the morphological feature values from the head words of the attachment's participants. E.g. in the (NP (A#SubPOS=f|Deg=p# black) (N#Cas=n|Num=s# cat)) example we add four features for the NP-A attachment: NP-Cas=n-A-SubPOS=f, NP-Cas=n-A-Deg=p, NP-Num=s-A-SubPOS=f, NP-Num=s-A-Deg=p.

- Similarly we take each combination of head word's morphological features values from sister constituents.

The first two templates enable the reranker to incorporate information into its learnt model from the rich morphology of the language at the lexical and constituent levels, while the last two templates might capture (dis)agreement at the morphological level. The motivation for using these features is that because of the free(er) word order of morphologically rich languages, morphological (dis)agreement can be a good indicator of attachment.

Table 3.6 shows the added value of these feature templates over mainPOS ('extended'), which is again statistically significant in exact match. Exploiting the morphological agreement in syntactic parsing has been investigated in previous studies, e.g. the Bohnet parser (Bohnet, 2010b) employs morphological feature value pairs similar to our feature templates and Seeker and Kuhn (2013) introduces an integer linear programming framework including constraints for morphological agreement. However, these works focus on dependency parsing and to the best of our knowledge, this is the first study on experimenting with atomic morphological features and their agreement in a constituency parsing.

	PARSEVAL	EX
Reranked $_{dflt}$	89.33	18.64
Reranked $_{dflt+moprh}$	**89.47**	**20.35**

Table 3.6: The results achieved by using various feature template sets for 50-best reranking on the Hungarian development set.

3.3.4 Results of the Full System

After our investigations focusing on building blocks of our system independently from each other on the development set, we parsed the test sets of the treebanks adding steps one-by-one. Table 3.7 summarizes our final results. We start from the BerkeleyParser using the full morphological descriptions as preterminal set, then we enrich the lexical model with tagging frequencies gathered from the automatic parsing of the test sets ('+ lexical model'). In the third step we replace the full preterminal set by the output of our preterminal merger procedure ('+ preterminal merger'). We tuned the merging threshold of our method on the development set for each language. The last two rows contain the results achieved by the 50-best reranker with the standard feature set ('+ reranker') and with the feature set extended by morphological features ('+ morph features').

The enhanced lexical model contributes a lot at Basque and considerable improvements are present at German and Hungarian as well while it harmed the results in French. The advance of the preterminal merger approach over the full setting is clear at French and Hungarian, similarly to the development set. It is interesting that an rationalized preterminal set could compensate the loss suffered by a inadequate lexical model at French.

Although the reranking step could further improve the results at each languages we have to note that the gain (0.5 in average) is much smaller here than the gains reported on English (over 1.5). This might be because of the high number of wordforms at morphologically rich languages i.e. most of feature templates are incorporate the words itself and the huge dictionary can indicate data sparsity problems again. Our morphology-based reranking features yielded a moderate improvement at four languages, but we believe there a lots of space for improvement here.

	Basque	French	German	Hebrew	Hungarian
BerkeleyParser	79.21 / 19.03	79.53 / 18.46	74.77 / 26.56	87.87 / 14.53	88.22 / 26.96
+ Lexical model	82.02 / 25.69	78.91 / 17.87	75.64 / 28.36	88.53 / 13.69	89.09 / 26.76
+ Preterminal merger	83.19 / 24.74	79.53 / 18.58	77.12 / **30.02**	88.07 / 13.83	89.15 / 28.05
+ Reranked $_{dflt}$	83.81 / 25.66	80.31 / 18.91	**77.78** / 29.80	88.38 / 15.12	89.57 / 30.23
+ Reranked $_{dflt+moprh}$	**84.03 / 26.28**	**80.41 / 20.07**	77.74 / 29.23	**88.55 / 15.24**	**89.91 / 30.55**

Table 3.7: PARSEVAL / exact match scores on the test sets.

3.4 Exploitation of External Knowledge

In this chapter, we focus only on the reranking step and we introduce two more new feature templates that can improve the results in the case of these morphologically rich languages by applying external information such as unlabeled data or dependency trees (Szántó and Farkas, 2015). For exploiting unlabeled data, we applied Brown clusters-based features, which groups the words into hierarchical categories based on their context. This method also can help in the case of the out-of-vocabulary issue, which is the consequence of the large number of word forms. We evaluate these feature templates on the SPMRL Shared Task datasets, consisting of training and evaluation datasets for morphologically rich languages. Besides the reranker, we show two techniques that can improve the results. We improved our previous solutions for lexical sparsity and introduce the application of product parsing to the pipeline.

3.4.1 Lexical Sparsity

We use two fundamentally different methods here to handle the out-of-vocabulary issue. One of these two techniques is the usage of extended lexicons, which was described in the Section 3.3.1. In contrast to Section 3.3.1, we here take advantage of the opportunity provided by unlabeled data in the SPMRL 2014 dataset (`Lexical model`). Instead of using the dev set, we calculated the frequency information about the possible morphological analyses of tokens from the unlabeled corpora and the dev sets.

The other method based on Clark and Curran's (Clark and Curran, 2007) work, where they replaced the rare words with their predicted POS tags in CCG grammars. We also use this strategy in our constituent parsing framework (`Replace`).

3.4.2 Product Parser

The Berkeley Parser uses an iterative expectation–maximization approach to calculate the latent variables (Petrov et al., 2006a). This is a local search algorithm and the result of this optimization is highly dependent on the initialization of the variables. In every iteration the Berkeley Parser splits the latent variables in two with a small amount of randomness. Petrov shows (Petrov, 2010a) that the modification of these random values can change the result of the parsing. We can get different parsers when we only change the seed of the random. If we product the probabilities of the same sentence with these different grammars, we can get better

scores than in the case when we just used one grammar. Based on this experience, we trained 8 grammars with different random seeds and we got the product of them.

3.4.3 Reranker for Morphologically Rich Languages

The second step of our constituency pipeline is discriminative reranking. We conduct ranking experiments on the 50-best outputs (see Section 3.3.3 for details) of the product grammars.

We propose here new feature templates exploiting automatic dependency parses of the sentence in question and Brown clusters. Our purpose here is to investigate the efficiency of these new feature templates in morphologically rich languages over the product grammar configuration. Here we present our feature templates in more detail.

Dependency-based Features

The SPMRL 2014 Shared Task had a dependency track. The organizers provided dependency annotations over the same texts. We used dependency prediction (Björkelund et al., 2014) and created features from that. These features are made from heads of constituents and their dependency relations. We used features describing relations between the same head-dependent pairs in both the constituency and dependency parses. The features are described in details in Farkas et al. (2011). The frequency of these relations was also used.

These features are especially interesting for Hungarian because we have two manually annotated corpora in both representations as opposed to the other SPMRL languages (Simkó et al., 2014).

Brown Cluster-based Features

We defined Brown cluster-based features. Brown clustering is a context-based hierarchical clustering over words (Brown et al., 1992). These Brown clusters are useful for syntax because words with similar context may have similar grammatical roles. These features can handle the feature sparsity issue. Utilizing these clusters, we duplicate every feature containing words by replacing words with their Brown clusterID.

Figure 3.1: Result of Brown cluster based feature templates on the Hungarian dataset.

Figure 3.1 investigates the effect of employing different levels of the Brown hierarchical tree evaluated on the Hungarian dataset. We get similar improvement in the case of both methods, `Replace` and `Lexical model`, namely that these features increase the PARSEVAL metric with 0.9 percentage point. We optimized the depth of hierarchy for each language separately.

3.4.4 Results and Discussion

In this section, we investigate the effect of the different lexical presentations, the product parser and the new reranker feature sets. As baselines we applied the better between the main POS and full morphological description for each languages.

Our two lexical methods use the POS tags differently. In the case of `Lexical model` we use the full morphological description at the preterminal level and main POS tags in the Replace method.

	Basque	French	German	Hebrew	Hungarian
Baseline	81.8	78.9	82.3	88.9	88.3
Lexical model	77.57	79.67	81.54	88.24	88.99
Replace	84.27	80.26	82.99	89.73	89.59
Lexical model + Product	79.47	**81.38**	82.94	89.22	90.43
Replace + Product	**85.31**	81.29	**84.55**	**89.87**	**90.72**

Table 3.8: PARSEVAL scores on the development sets for the predicted setting

Table 3.8 shows the results achieved by the two strategies for handling lexical sparsity and the effect of the usage of the product of different grammars. The product of the grammars increased the accuracy in every case. We got the best results with the `Replace Product` system in six languages, but this strategy could not predict the full morphological description. In the case of French the `Lexical model + Product`

	Basque	French	German	Hebrew	Hungarian
Lexical model + Product + Reranked $_{dflt}$	79.16	81.92	83.01	89.39	91.06
Lexical model + Product + Reranked $_{dflt+morph}$	79.41	82.88	83.36	89.63	91.27
Lexical model + Product + Reranked $_{dflt+dep}$	81.89	82.65	84.83	90.28	91.88
Lexical model + Product + Reranked $_{dflt+Brown}$	80.63	82.49	84.33	90.30	91.93
Lexical model + Product + Reranked $_{dflt+morph+Brown+dep}$	82.69	82.62	85.16	90.64	92.05
Replace + Product + Reranked $_{dflt}$	86.11	82.30	84.59	90.02	91.09
Replace + Product + Reranked $_{dflt+dep}$	86.73	82.78	86.05	90.47	91.89
Replace + Product + Reranked $_{dflt+brown}$	86.57	82.65	85.85	90.62	92.06
Replace + Product + Reranked $_{dflt+dep+brown}$	**87.24**	**82.94**	**86.56**	**90.73**	**92.40**

Table 3.9: PARSEVAL scores of the reranker on the development set for the predicted setting.

got slightly better scores.

In the next step we added the reranker stage to our product parsing systems.

Table 3.9 shows the final results of the reranker on the development set. We evaluated the effect of each new feature template and the combinations of all feature sets. `dflt` is the standard feature set from Charniak and Johnson (Charniak and Johnson, 2005) and Collins (Collins, 2000). We applied our morphology-based `(morph)` features from section 3.3.3 and investigated the effect of the dependency-based `(dep)` and Brown cluster-based `(Brown)` features to this baseline. In the case of configurations that contain Brown cluster-based features, we show the best results in Table 3.9.

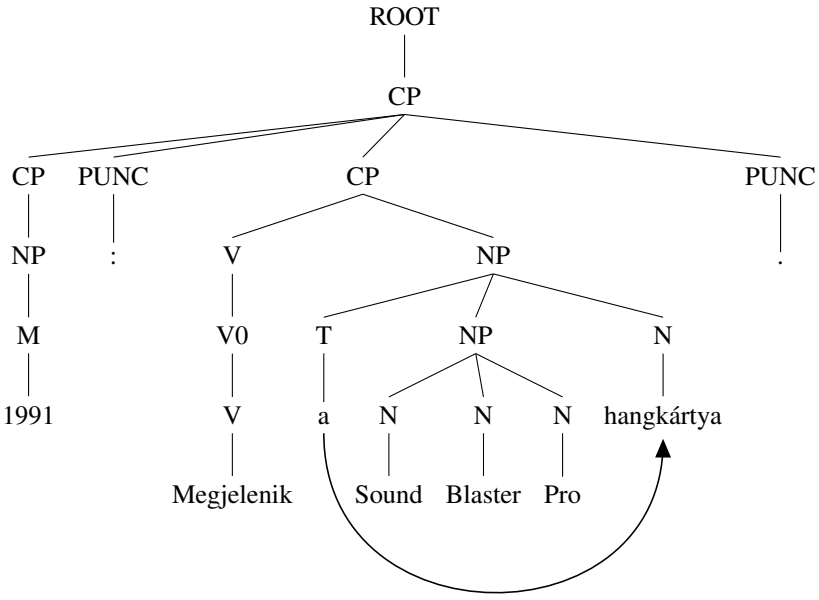


Figure 3.2: The gold standard parse of a Hungarian sentence with a dependency edge. (1991: The Sound Blaster Pro soundcard has appeared.)

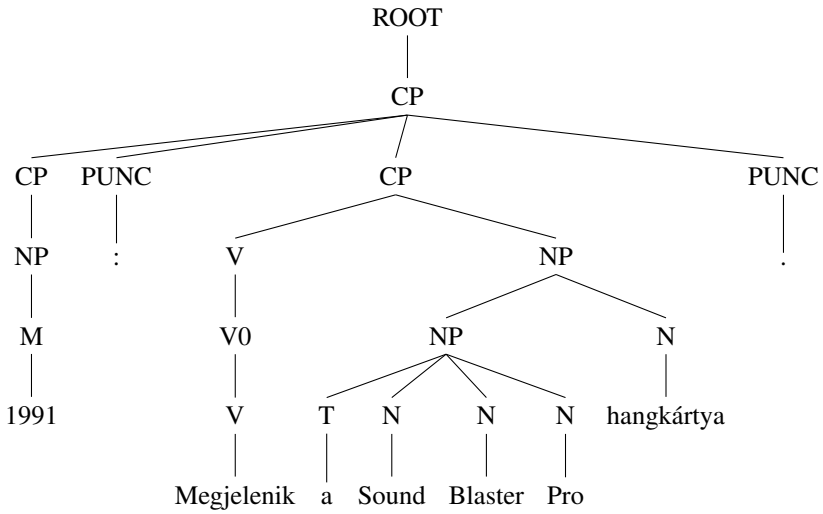


Figure 3.3: The reranked parse of a Hungarian sentence without dependency-based features.

Reranking with default features improved the scores over product grammars both for `Lexical model` and `Replace`. In the case of both representations, the combination of the proposed feature templates further increased our scores except Swedish, where the morphology-based features and Brown cluster-based features also decreased the accuracy of the `Lexical model` method, but when we added each new featureset we got a slight improvement compared to the `dflt+dep` model. Finally, the `Replace` method got higher scores in every language. But in some languages (French, Hebrew, Hungarian) the difference is small and we can keep the morphological information with `Lexical model`. We plan to combine these two fundamentally different methods in the future.

We analyzed manually the effect of new feature sets. Figures 3.2 and 3.3 show an example for the usefulness of the dependency based features. Figure 3.2 contains the correct parse of a Hungarian sentence. Figure 3.3 contains the result of our consituent parsing system to this sentence where we used the baseline (`dflt`) configuration. With dependency-based features (`dep`) we got the correct parse. There is only one difference between the two parses. In the correct parse the article is connected to the noun phrase of *"Sound Blaster Pro hangkártya"* and in the wrong parse this article is connected to the noun phrase of *"Sound Blaster Pro"*. If the computer does not see the dependency parse of this sentence, then the second parse is likely a good choice, because a noun follows the article. But in the dependency parse there is a relation from the article to the noun *hangkártya*, and the dependency-based features encode this relationship, so the correct parse can be yielded

Figure 3.4 and Figure 3.5 contain the parse of a part of a Hungarian sentence. In this phrase "*Tabulating Machine Companyt*" is a proper noun, where "*Companyt*" is in the accusative case and the "a" is the article of the proper noun. In Hungarian a noun phrase can consist of the parts of a multi-word proper noun, like in Figure 3.4. Our parser (without morphology-based features) splits the expression because the parser

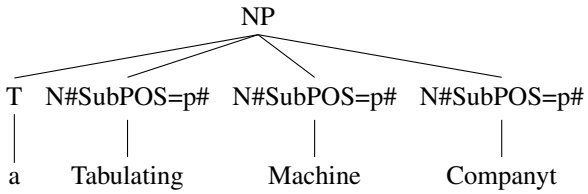


Figure 3.4: The gold standard parse of a part of a Hungarian sentence. (the Tabulating Machine Company in accusative)

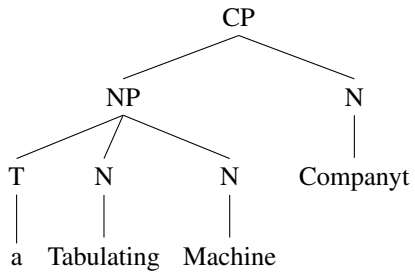


Figure 3.5: The reranked parse of a part of a Hungarian sentence without morphology-based features.

has insufficient information to decide whether it is a multi-word proper noun, or it is just a sequence of independent nouns. In this expression, all wordforms are very rare. Instead of the usage of the wordforms, when we employ the morphology-based features, we have whether features which examine whether the head of the NP (*Companyt*) is a proper noun and the head of the NP also has a sibling which is proper noun. Hence these features can increase the probability of the correct sentence.

In the case of Brown-cluster based features we evaluated the F-score of the non-terminal labels. We found improvement in the case of labels ADJP (2.88), CP (1.39) and NP (0.86). The Brown clusters can group syntactically similar words, for instance, it clusterized the superlative adjectives and the ordinal numerals to one cluster. These words behave similarly at the syntax level, since both groups usually function as the head of an ADJP. These automatically found similarities can help the reranker to choose the best parse from the candidates.

3.5 SPMRL 2014 Shared Task

We participated in the SPMRL 2014 shared tasks in an international team. The IMS-Szeged-CIS Björkelund et al. (2013) submissions – a joint effort of three universities – managed to get the highest scores in every category in 2013.

The SPMRL 2014 shared task was an extension of the first challenge, where every annotated corpora from last year was extended with a large unlabeled data set. The team IMS-Wroclaw-Szeged-CIS achieved

the best scores Björkelund et al. (2014) on all languages in the dependency track and on all languages (except for Polish) in the constituency track in 2014.

For constituent parsing, we applied most of the techniques from the previous sections. In the reranking step, we used all the previously described feature sets. For the lexical sparsity handling methods – the enhanced lexicon model and the replacement of the rare words with their POS tag – we made separate submissions.

	Basque	French	German	Hebrew	Hungarian	Korean	Polish	Swedish
ST Baseline	74.74	80.38	78.3	86.96	85.22	78.56	86.75	80.64
Other	85.35	79.68	77.15	86.19	87.51	79.5	**91.6**	82.72
ExtendLex - Reranked	83.78	**82.53**	79.76	89.75	90.76	-	89.19	82.94
Replace - Reranked	**88.24**	82.52	**81.66**	**89.8**	**91.72**	**83.81**	90.5	**85.5**

Table 3.10: Final PARSEVAL F1 scores for constituents on the SPRML 2014 test sets for the predicted setting. ST Baseline denotes the best baseline provided by the ST organizers. Other denotes the best competitor.

We also compare our results with the baselines provided by the organizers and with the best shared task (ST) competitor. The constituency parsing results at SPMRL 2014 shared task are given in Table 3.10. Our $Replace$ system outperforms the $ExtendLex$ system across all languages with the only exception of 0.01 difference in French. We have the highest accuracies for all languages except Polish.

3.6 Summary

In this chapter, I showed novel techniques that improved the accuracy of constituent parsing systems on morphologically rich languages.

One of the chief contributions here is a novel preterminal merger procedure. This is a more general approach than previous proposals and still much faster thanks to operating on probabilities from a PCFG instead of employing a full train+parse step for evaluating every preterminal set candidate. I found that including the rich morphological description in the preterminal level is crucial for parsing morphologically rich languages. I also experimented with exploiting external corpora in the lexical model. A new scientific result is that automatic tagging of an off-the-shelf supervised morphological tagger can also contribute to the results. My last experiment was carried out with the feature set of an n-best reranker. We showed that incorporating feature templates built on morphological information improves the results. The results were published in the article "Special Techniques for Constituent Parsing of Morphologically Rich Languages" at the *EACL'14 conference* as a long paper (Szántó and Farkas, 2014).

I improved the efficiency of the discriminative reranking step with new feature templates. I proposed novel features from Brown clustering of the words and analyzed the effect of morphology-based, dependency-based, and Brown cluster-based features. Those methods were published in the "Constituency Parse Reranking for Morphologically Rich Languages" paper in the *Acta Polytechnica Hungarica* journal (Szántó and Farkas, 2015).

Part II

Dependency Parsing

Chapter 4

Hungarian Dependency Parsing

4.1 Introduction

Dependency parsing is a widely utilized technique in computational linguistics for representing the syntactic structure of sentences. This chapter delves into the current landscape of Hungarian dependency parsing. It introduces the available datasets, explores the relationship between dependency parsing and constituent parsing in Hungarian (Simkó et al., 2014), and examines the development of the Universal Dependency dataset for Hungarian (Vincze et al., 2017). Additionally, it highlights HuSpaCy, which provides a cutting-edge dependency parser for Hungarian (Orosz et al., 2023).

4.2 Automatic Conversion from Constituency to Dependency

In this section, I introduce a Hungarian constituency to a dependency converter. Based on that system I demonstrate that although the results obtained by training on the constituency treebank and converting the output to dependency format and those obtained by training on the automatically converted dependency treebank are similar in terms of accuracy scores, the typical errors made by different systems differ from each other (Simkó et al., 2014).

There exist constituency-based treebanks for many languages and dependency treebanks for most of these languages are converted automatically from constituent trees with the help of conversion rules, which is the case for e.g. the languages used in the SPMRL-2013 Shared Task (Seddah et al., 2013) with the exception of Basque, where constituency trees are converted from manually annotated dependency trees (Aduriz et al., 2003), and Hungarian, where both treebanks are manually annotated (Csendes et al., 2005; Vincze et al., 2010). However, the quality of automatic dependency conversion is hardly investigated.

Hungarian is one of those rare examples where there exist manual annotations for both constituency and dependency syntax on the same bunch of texts, the Szeged (Dependency) Treebank (Csendes et al., 2005;

Vincze et al., 2010), which makes it possible to evaluate the quality of a rule-based automatic conversion from constituency to dependency trees, to compare the two sets of manual annotations and also the output of constituency and dependency parsers trained on converted and gold standard dependency trees.

We investigate the effect of automatic conversions related to the two parsing paradigms as well. It is well known that for English, the automatic conversion of a constituency parser's output to dependency format can achieve competitive unlabeled attachment scores (ULA) to a dependency parser's output trained on automatically converted trees[1] (cf. Petrov et al. (2010)). One of the possible explanations for this is that English is a configurational language, hence constituency parsers have advantages over dependency parsers here. We check whether this hypothesis holds for Hungarian too, which is the prototype of free word order languages.

This section introduces the comparison of three pairs of dependency analyses in order to evaluate the usefulness of converted trees. First, we examine the errors of the conversion itself by comparing the converted dependency trees with the manually annotated gold standard ones. Second, we argue for the importance of training parsers on gold standard trees by looking at the typical differences between the outputs of dependency parsers trained on converted (silver standard) trees, parsers trained on gold standard trees, and the manual annotation itself. Third, we demonstrate that similar to English, training on a constituency treebank and converting the results to dependency format can achieve similar results in terms of ULA to the dependency parser trained on the automatically converted treebank, but the typical errors they make differ in both cases.

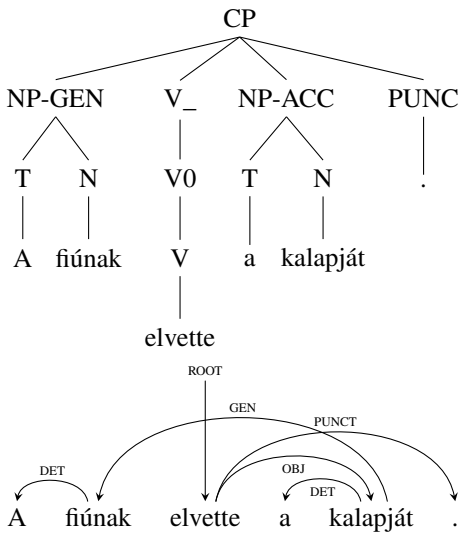


Figure 4.1: Discontinuous structure *A fiúnak elvette a kalapját* (the boy-DAT take-past3SGOBJ the hat-POSS3SG-ACC) "He took the boy's hat" in constituency and dependency analysis.

[1] However, it has been pointed out that errors in the conversion script may significantly influence the results of parsing, see e.g. Petrov and McDonald (2012) and Pitler (2012)

4.2.1 Constituent and Dependency representations in the Szeged Treeebanks

Hungarian is a morphologically rich language, where word order encodes information structure, which makes its syntactic analysis very different from English's as the arguments in a sentence cannot be determined by their position but by their suffixes, cf. É. Kiss (2002). Words' grammatical functions are signified by case suffixes and verbs are marked for the number and person of their subject and the definiteness of their object, thus these arguments may be often omitted from the sentence: *Látlak* (see-1SG2OBJ) "I see you". Due to word order reasons, words that form one syntactic phrase may not be adjacent (long-distance dependencies), which is true for the possessive construction as well: the possessor and the possessed may be situated in two distant positions: *A fiúnak elvette a kalapját* (the boy-DAT take-PAST-3SGOBJ the hat-POSS3SG-ACC) "He took the boy's hat". Verbless clauses are also common in Hungarian, as the copula in third person singular present tense indicative form is phonologically empty, while it is present in all other moods and tenses: *A kalap piros* (the hat red) "The hat is red", but *A kalap piros volt* (the hat red was) "The hat was red".

The Szeged Dependency Treebank (Vincze et al., 2010) contains manual dependency syntax annotations for the same texts as the Szeged Treebank. Certain linguistic phenomena – such as discontinuous structures – are annotated in the dependency treebank, but not in the constituency treebank. In the dependency treebank, the possessor is linked to the possession while this connection is not annotated in the constituency treebank. The two types of trees can be seen in Figure 4.1.

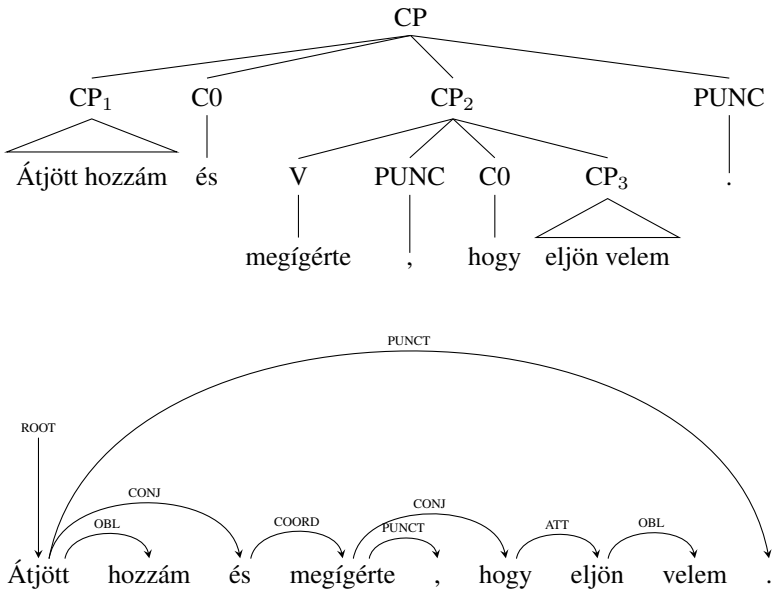


Figure 4.2: Constituency and dependency analysis of coordination and subordination in the sentence *Átjött hozzám és megígérte, hogy eljön velem* (through.come-PAST-3SG to.me and promise-PAST-3SG-OBJ that away.come-3SG with.me) "He came over and promised that he will come with me".

Another difference between the two treebanks is the way they represent different types of complex sentences, as can be seen in Figure 4.2. In the dependency treebank subordinations and coordinations are handled very similarly. The head of one of the clauses (the subordinated clause or the second clause in the case of coordination) is linked to the head of the other clause (the matrix clause of the subordination or the first clause of the coordination), only the type of relation between the two heads differs in the two structures, in the dependency tree in Figure 4.2, the heads of the three clauses (*átjött* "came over", *megígérte* "promised" and *eljön* "come") are linked to one another through their conjunctions with either an ATT relation in the case of subordination or COORD for coordination. In the constituency treebank these sentences are represented very differently: in the case of subordination, the subordinated clause is within the matrix clause: CP_3 is within CP_2 in the constituency tree in Figure 4.2. Coordinated clauses appear at the same level in the structure, in the same figure CP_1 and CP_2 are coordinated clauses.

The parallels of these two manually annotated treebanks make them suitable for testing our hypotheses about automatic dependency conversion. The differences between them originate from the characteristics of constituent and dependency syntax.

4.2.2 Converting Constituency Trees to Dependency Trees

In this section, we present our methods to convert constituency trees to dependency trees and we also discuss the most typical sources of errors during conversion.

Conversion Rules

In order to convert constituency trees to dependency trees, we used a rule based system. Sentences with virtual dependency nodes were omitted, as they are not annotated in the constituent treebank and their treatment in dependency trees is also problematic (Farkas et al., 2012; Seeker et al., 2012). As a result, we worked with 7,372 sentences and 162,960 tokens.

First, we determined the head of each clause (CP) and the relations between CPs in complex sentences. In most cases the head of the CP is a finite verb, if the CP contains no finite verb, the head is the either an infinitive verb or a participle, if none of these are present in the CP, the head can be a nominal expression. The relations between the CP heads make up the base of the dependency structure using ROOT relation for the sentence's main verb, COORD for coordination and ATT for subordination, as well as CONJ in the case of conjunctions between the CPs.

The arguments of verbs, infinitives and participles in the CP were linked to their governor and marked for their grammatical role in the Szeged Treebank. We used this information to construct the appropriate dependency relations between governors and their arguments. The main grammatical roles such as subject, object, dative have their own label in dependency syntax, while minor ones are assigned the oblique (OBL) relation. The argument's modifiers were then linked to the head or other modifiers based on the phrase structure with relations according to their morphological code.

Long distance dependencies, like the connection between a genitive case possessor and the possessed are not annotated in the constituency treebank. In these cases we used morphological information to link these elements together in the dependency tree. Figure 4.3 shows an example of converting a constituency tree to a dependency tree.

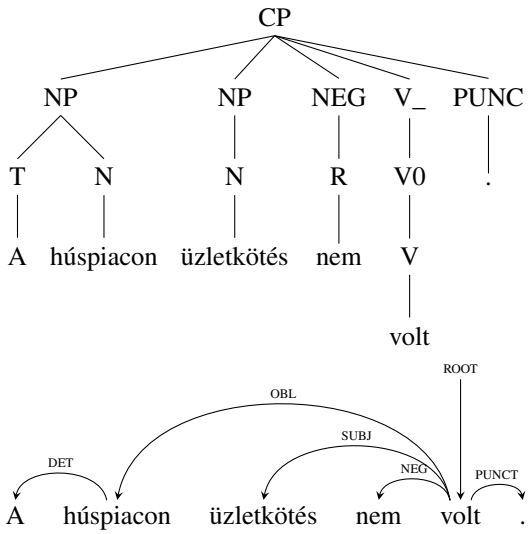


Figure 4.3: Conversion of the sentence *A húspiacon üzletkötés nem volt* (the meat.market-SUP transaction not was) "There were no transactions at the meat market." from constituency to dependency trees.

Error Analysis

We automatically converted the constituency treebank into dependency trees following the principles described above and detailed at our website[2]. For evaluation, we applied the metrics labeled attachment score (LAS) and unlabeled attachment score (ULA), without punctuation marks. The accuracy of the conversion was 96.51 (ULA) and 93.85 (LAS). The errors made during conversion were categorized manually in 200 sentences selected randomly from the short business news subcorpus of the Szeged Dependency Treebank, and the most typical ones are listed in Table 4.1, Column *convError*.

As it is shown, the most common source of error was when more than one modifier was within a phrase as the example in Figure 4.4 shows. In each figure, the gold standard parse can be seen above while the erroneous one can be seen below.

[2] https://rgai.inf.u-szeged.hu/node/113

Error type	convError		goldTrain		silverTrain		BerkeleyConv		convDep	
	#	%	#	%	#	%	#	%	#	%
Coordination	26	13.00	39	13.22	59	14.82	55	16.37	64	19.57
Multiple modifiers	26	13.00	30	10.17	49	12.31	52	15.48	47	14.37
Determiner	7	3.50	28	9.49	25	6.28	31	9.23	31	9.48
Conj./adverb attached	33	16.50	23	7.80	45	11.31	39	11.61	42	12.84
Arg. of verbal element	10	5.00	27	9.15	34	8.54	59	17.56	44	13.46
Sub- vs. coordination	7	3.50	9	3.05	12	3.02	–	–	–	–
Possessor	9	4.50	14	4.75	16	4.02	28	8.33	22	6.73
Wrong root	14	7.00	17	5.76	23	5.78	35	10.42	27	8.26
Consecutive nouns	4	2.00	11	3.73	14	3.52	13	3.87	15	4.59
Multiword NE	8	4.00	25	8.47	33	8.29	8	2.38	19	5.81
Wrong MOD label	25	12.50	26	8.81	34	8.54	–	–	–	–
Wrong other label	17	8.50	33	11.19	30	7.54	–	–	–	–
Other errors	14	7.00	13	4.41	24	6.03	16	4.76	16	4.89
Total	200	100	295	100	398	100	336	100	327	100

Table 4.1: Error Types. `convError`: errors made during converting constituency trees to dependency trees. `goldTrain`: errors in the output got by training the Bohnet parser on the gold standard data. `silverTrain`: errors in the output got by training the Bohnet parser on the silver standard data. **BerkeleyConv**: errors in the output got by training the Berkeley parser on the gold standard constituency data and converting the output into dependency format. `convDep`: errors in the output got by training the Bohnet parser without dependency labels on the silver standard data.

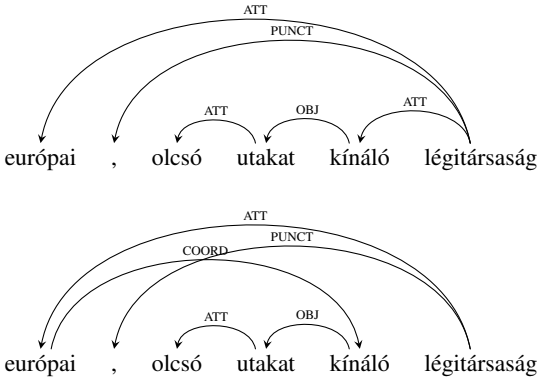


Figure 4.4: Multiple modifier error in *európai, olcsó utakat kínáló légitársaság* (European cheap trips-ACC offering airline) “European airline offering cheap trips”.

Coordination errors occurred when multiple members of a coordination were wrongly connected. On the other hand, the attachment of conjunctions and some adverbs was also problematic, for example in Figure 4.5 the conjunction *is* “also” is connected to the verb in the gold standard and to the noun in the converted version.

Figure 4.5: Conjunction attachment error in *a minisztérium is beszáll* (the ministry also steps.in) "the ministry also steps in".

Also, the constituency treebank did not mark all the grammatical relations (e.g. numerals and determiners were simply parts of an NP but had no distinct labeling, like *[NP az öt [ADJP fekete] kutya]* (the five black dog) "the five black dogs"), but it was necessary to assign them a dependency label and a parent node during conversion. However, in some cases it was not straightforward which modifier modifies which parent node: for instance, in *[NP nem [ADJP megfelelő] módszerek]* (not appropriate methods) "inappropriate methods", the negation word *nem* is erroneously attached to the noun instead of the adjective in the converted phrase. Determiner errors were those where the determiner was attached to the wrong noun in a NP with a noun modifier. In CPs with multiple verbal elements (both a finite verb and an infinitive or a participle in the CP) the arguments were sometimes linked to the wrong verb, as in Figure 4.6.

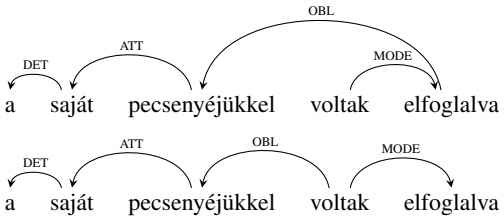


Figure 4.6: Verbal argument error in *a saját pecsenyéjükkel voltak elfoglalva* (the own roast-3PLPOSS-INS were busy) "they were busy with their own thing".

Possessors are sometimes wrongly identified during conversion as long distance dependencies are not marked in the constituency treebank (see Figure 4.7).

Figure 4.7: Possessor attachment error in *a gyártó szárítóüzemében hasznosít* (the manufacturer drying.plant-3SGPOSS-INE utilizes) "the manufacturer utilizes it in its drying plant".

In CPs with more verbal element, sometimes the wrong word is selected as the root, as in Figure 4.8.

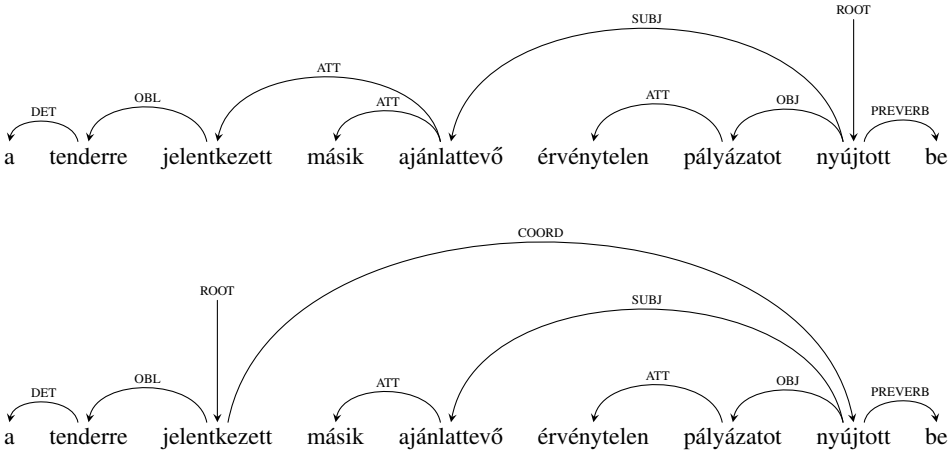


Figure 4.8: Root error in *a tenderre jelentkezett másik ajánlattevő érvénytelen pályázatot nyújtott be* (the tender-SUB applied other bidder invalid application-ACC submit-PAST-3SG) "the other bidder applying to the tender submitted an invalid application".

In some cases, consecutive (but separate) noun phrases were taken as one unit as if one noun modified the other, for example in Figure 4.9.

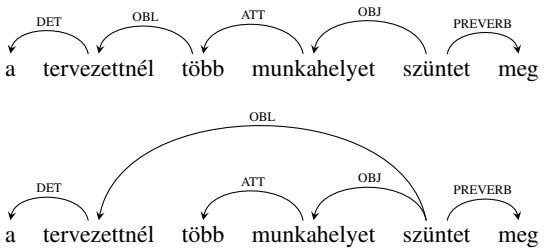


Figure 4.9: Consecutive noun error in *a tervezettnél több munkahelyet szüntet meg* (the planned-ADE more workplace-ACC terminates) "it terminates more workplaces than planned".

Multiword NEs also caused some problems in the conversion, as in Figure 4.10.

Figure 4.10: Multiword NE error in *Beszállítói Befektető Rt.* (a name of a company) .

In other cases, divergences between the gold standard and the converted trees are due to some erroneous annotations either in the constituency treebank or in the dependency treebank. A typical example of this is the wrong MOD (modifier) label. In the treebank, locative and temporal modifiers were classified according to

the tridirectionality typical of Hungarian adverbs and case suffixes: *where*, *from where* and *to where* (or *when*, *from what time and till what time*) the action is taken place. Thus, there are six dependency relations dedicated to these aspects and all the other adverbials are grouped under the relation MOD. However, this distinction is rather semantic in nature and was sometimes erroneously annotated in the constituency treebank, which was later corrected in the dependency one and thus now resulted in conversion errors, as shown in Figure 4.11.

Figure 4.11: MOD label error in *nyár vége felé kezdik* (summer end-3SGPOSS around begin) "they begin around the end of the summer".

There were also some atypical errors that occurred too rarely to categorize them in a different class, like cases when an article or determiner got erroneously attached to a verb and so on, so they were lumped into the category of "other errors" in Table 4.1.

4.2.3 Training on Gold Standard and Silver Standard Trees

We also experimented with training the Bohnet dependency parser on the manually annotated (gold standard) and the converted (silver standard) treebank.

From the corpus, 5,892 sentences (130,211 tokens) were used in the training dataset and the remaining 1,480 sentences (32,749 tokens) in the test dataset. For evaluation, we again applied the metrics LAS and ULA. Results are shown in Table 4.2, Rows `goldTrain` and `silverTrain`.

As the numbers show, better results can be achieved when the gold standard data are used as training database than when the parser is trained on the silver standard data, the differences being 1.6% (ULA) and 3.16% (LAS). Besides evaluation scores, we also compared the outputs of the two scenarios: we used the same set of randomly selected sentences as when investigating conversion errors and carried out a manual error analysis against the gold standard data in each case: see Table 4.1, Columns `goldTrain` and `silverTrain`.

There are some common error types that seem to cause problems for both ways of parsing. For instance, coordination and multiple modifiers are among the most frequent sources of errors in both cases as for the error rates are concerned. However, with regard to the absolute numbers, we can see that both error types are reduced when the gold standard dataset is used for training. On the other hand, finding the parent node of a conjunction or an adverb seems to improve significantly when the parser is trained on gold standard data. This is probably due to the fact that they are not marked in the constituency treebank and thus training data for these grammatical phenomena are very noisy in the silver standard treebank. All in all, we argue that there are some grammatical phenomena – e.g. the attachment of conjunctions or adverbs – that require manual checking even if automatic conversion from constituency to dependency is applied.

Setting	LAS	ULA
Conversion	93.85	96.51
goldTrain	93.48	95.17
silverTrain	90.32	93.57
BerkeleyConv	–	92.78
convDep	–	93.23

Table 4.2: Results of the experiments. Conversion: converting constituency trees to dependency trees. goldTrain: training the Bohnet parser on the gold standard data. silverTrain: training the Bohnet parser on the silver standard data. BerkeleyConv: training the Berkeley parser on the gold standard constituency data and converting the output into dependency format. convDep: training the Bohnet parser without dependency labels on the silver standard data.

4.2.4 Pre- or Post Conversion?

It is well known that for English, converting a constituency parser's output to dependency format (post conversion) can achieve competitive ULA scores to a dependency parser's output trained on automatically converted trees (pre conversion) (Petrov et al., 2010; Farkas and Bohnet, 2012a). One of the possible reasons for this may be that English is a configurational language, hence constituency parsers are expected to perform better here. In this section, we investigate whether this is true for Hungarian, which is the prototype of morphologically rich languages with free word order.

We employed the product-of-grammars procedure (Petrov, 2010b) of the Berkeleyparser, where grammars are trained on the same dataset but with different initialization setups, which leads to different grammars. We trained 8 grammars and used tree-level inference. The output of the parser was then automatically converted to dependency format, based on the rules described in Section 4.2.2 (`BerkeleyConv`). Second, we used the silver standard dependency treebank for training the Bohnet parser (`convDep`). Since our constituency parser did not produce grammatical functions for the nodes, we trained the Bohnet parser on unlabeled dependency trees in order to ensure a fair comparison here (that is the difference between the columns `BerkeleyConv` and `convDep` in Table 4.1).

As the numbers show, competitive results can be obtained with both methods, yielding an ULA score of 92.78 and 93.23, respectively. This means that the same holds for Hungarian as for English and the surprisingly good results of post conversion are not related to the configurational level of the language.

Manually analysing the errors on the same set of sentences as before, there are again some error categories that occur frequently in both cases such as coordination, the attachment of conjunctions, modifiers and determiners. On the other hand, training on constituency trees seems to have some specific sources of errors. First, the possessor in possessive constructions is less frequently attached to its possessed, which may be due to the fact that the genitive possessor is not linked to the possessed in the constituency treebank and thus the parser is not able to learn this relationship. Second, arguments of verbal elements (i.e. verbs, participles and infinitives) are also somewhat more difficult to find when there are at least two verbal elements within the clause, which is especially true for adverbial participles and infinitives. In Figure 4.6, the differences between the two trees are shown. The noun *pecsenyéjükkel* (roast-3PLPOSS-INS) "with their thing" is linked to the

adverbial participle in the correct analysis, but it connects to the main verb in the other. Third, identifying the root node of the sentence may also be problematic for this setting. As Farkas and Bohnet (2012a) reported that preconversion can achieve better results for finding the root node in English, this seems to be a language-specific issue and it represents an interesting difference between English and Hungarian. Nevertheless, training on constituency trees has a beneficial effect on finding multiword named entities. Hence, it can be concluded that although the evaluation scores are similar, the errors the two systems make differ from each other.

4.3 Universal Dependencies and Morphology for Hungarian

In this section, I present how the principles of universal dependencies and morphology have been adapted to Hungarian (Vincze et al., 2017). I also introduce experiments on this manually annotated corpus for evaluating automatic conversion and the added value of language-specific, i.e. non-universal, annotations. These results reveal that converting to universal dependencies is not necessarily trivial, moreover, using language-specific morphological features may have an impact on overall performance.

Morphological tagging and syntactic parsing are key components in most natural language processing (NLP) applications. Linguistic resources and parsers for morphological and syntactic analysis have been developed for several languages, see e.g. the SPMRL shared tasks (Seddah et al., 2013, 2014a). However, the comparison of results achieved for different languages is not straightforward as most languages and databases apply a unique tagset, moreover, they were annotated following different guidelines. In order to overcome these issues, the project Universal Dependencies and Morphology (UD) has recently been initiated within the NLP community (Nivre, 2015). The main goal of the UD project is to develop a "universal", i.e. a language-independent morphological and syntactic representation which can contribute to the implementation of multilingual morphological and syntactic parsers from a computational linguistic point of view. Furthermore, it can enhance studies on linguistic typology and contrastive linguistics.

From the viewpoint of syntactic parsing, the languages of the world are usually categorized according to their level of morphological richness (which is negatively correlated with configurationality). At one end, there is English, a strongly configurational language while there is Hungarian at the other end of the spectrum with rich morphology and free word order (Fraser et al., 2013). This section presents how UD principles were adapted to Hungarian, with special emphasis on Hungarian-specific phenomena.

Hungarian is one of the prototypical morphologically rich languages thus our UD principles can provide important best practices for the universalization of other morphologically rich languages. The UD guidelines for Hungarian were motivated by both linguistic considerations and data-driven observations. We developed a converter from the existing Szeged Dependency Treebank (Vincze et al., 2010) to UD and manually corrected 1,800 sentences from the newspaper domain. The experiences gained during the converter development and during the manual correction could reinforce the linguistic guidelines. Moreover, the manually corrected gold standard corpus provides the opportunity for empirical evaluations like assessing the converter and

comparing dependency parsers employing the original and the universal morphological representations. Thus, we evaluated the quality of the automatic conversion, which reveals that converting to universal dependencies is not necessarily trivial, at least for Hungarian. We also show that using different morphological tagsets may have an impact on overall parsing performance and utilizing language-specific, i.e. non-universal, information has a considerable added value at both the morphological and syntactic layers.

4.3.1 Universalization

Standardized tagsets for both morphological and syntactic annotations have been constantly developed in the international NLP community. For instance, the MSD morphological coding system was developed for a set of Eastern European languages (Erjavec, 2012), within the MULTEXT-EAST project. Interset functions as an interlingua for several morphological coding systems, which can convert different tagsets to the same morphological representation (Zeman, 2008). There have also been some attempts to define a common set of parts-of-speech: Rambow et al. (2006) defined a multilingual tagset for part-of-speech (POS) tagging and parsing, while McDonald and Nivre (2007) identified eight POS tags based on data from the CoNLL-2007 Shared Task (Nivre et al., 2007). Petrov et al. (2012) offered a tagset of 12 POS tags and applied this tagset to 22 languages.

Universal Dependencies (UD) is an international project that aims at developing a unified annotation scheme for dependency syntax and morphology in a language-independent framework (Nivre, 2015). In these datasets, the very same tagsets are applied at the morphological and syntactic levels and texts are annotated on the basis of the same linguistic principles, to the widest extent possible.

The UD tagset encodes morphological information in the form of POS tags and feature–value pairs. As for syntactic information, each word is assigned to its parent word in the dependency tree and the grammatical function of the specific word is encoded in dependency labels. Dependency labels, POS tags and features are universal (i.e. there is a fixed set of them without the possibility of introducing new members), but values and dependency labels can have language-specific additions if needed. Features are divided into the categories lexical features and inflectional features. Lexical features are features that are characteristics of the lemmas rather than the word forms, whereas inflectional features are those that are characteristics of the word forms. Both lexical and inflectional features can have layered features: some features are marked more than once on the same word, e.g. a Hungarian noun may denote its possessor's number as well as its own number. In this case, the Number feature has an added layer, Num[psor].

Several papers have been published on the general principles behind UD (Nivre, 2015; Nivre et al., 2016) or on specific treebanks. For instance, there are UD treebanks available for agglutinative languages such as Finnish (Haverinen et al., 2014; Pyysalo et al., 2015), Estonian (Muischnek et al., 2016) and Japanese (Tanaka et al., 2016), for Slavic languages (Zeman, 2015) and spoken Slovenian (Dobrovoljc and Nivre, 2016) and for Nordic languages such as Norwegian (Øvrelid and Hohle, 2016), Danish (Johannsen et al., 2014) and Swedish (Nivre, 2014), together with several other languages (Persian (Seraji et al., 2016) and Basque (Aranzabe et al., 2014), just to name a few).

Our UD principles introduced in this section follow the central UD guidelines (Nivre, 2015) and we did our best to align with the existing guidelines for other morphologically rich languages as well. On the other hand, there are several Hungarian-specific phenomena that required changes and extensions of the original UD principles.

The only available manually annotated treebank for Hungarian is the Szeged Corpus (Csendes et al., 2004) and Szeged Dependency Treebank (Vincze et al., 2010). We developed an automatic tool that converts the morphological descriptions of the Szeged Corpus to universal morphology tags and the dependency trees of the Szeged Treebank to universal dependencies.

4.3.2 Universal Morphology for Hungarian

When adapting the principles of Universal Morphology to Hungarian, we were able to automatically convert most of the morphological features used in the Szeged Treebank 2.5 (Vincze et al., 2014), which was based on MSD principles (Erjavec, 2012). The details of universal morphological codeset of Hungarian are available on our website[3].

Possessive Constructions

The possessor in Hungarian possessive constructions can have two different surface forms, without any difference in meaning: the possessor can be morphologically marked or not, just like the English constructions *the girl's doll* and *the doll of the girl*. Thus, both of the following possessive constructions are widely used:

(2) a szomszéd kertje
the neighbor garden-3SGPOSS
the neighbor's garden

(3) a szomszédnak a kertje
the neighbor-DAT the garden-3SGPOSS
the neighbor's garden

In Example 2, the possessor is not marked, i.e. it shares its form with the nominative form of the noun, however, in Example 3, the possessor is morphologically marked, sharing its from with the dative form of the noun. Nevertheless, the possessed is morphologically marked in both cases, which was a novelty in the UD project as the languages already included in the data do not mark the possessor on the possessed noun but use determiners for this purpose (cf. *my car* but *az autóm* (the car-1SGPOSS)). Moreover, the number of the possessed can be marked on the noun in elliptical constructions such as:

(4) Láttam az autódat , de a szomszédét nem .
see-PAST-2SGPOSS-ACC the car-2SG-POSS , but the neighbor-POSSD.SG-ACC not
I could see your car but not that of the neighbor.

[3] https://ai.inf.u-szeged.hu/nlp/univmorph/

(5) Láttam a gyerekeidet , de a szomszédéit nem .
see-PAST-2SGPOSS-ACC the child-2SG-PL-POSS , but the neighbor-POSSD.PL-ACC not
I could see your children but not those of the neighbor.

Hence, we had to introduce novel morphological features to mark the person and number features of the possessor on Hungarian nouns. Number denotes the number of the noun, Number[psor] and Person[psor] denote the number and person of the possessor, and Number[psed] denotes the number of the possessed. Below, there is a sample word annotated according to the Universal Morphology principles.

(6) házaimének
house-1SGPOSS-PL-POSSD.SG-DAT
to that of my houses
NOUN
Case=Dat|Number=Plur|Number[psed]=Sing
|Number[psor]=Sing|Person[psor]=1

Object-verb Agreement

Another Hungarian-specific feature was the definiteness of the object. As a special type of agreement, the definiteness of their objects determines which paradigm of the verb is to be chosen. In other words, the form of the verb changes when the definiteness of the object also changes (Törkenczy, 2005). For instance, proper nouns and NPs with a definite article are typical examples of definite objects and trigger the objective form of the verb (see Example 7) while bare nouns and NPs with an indefinite article are indefinite objects (see Example 8) and trigger the subjective form of the verb. Second person objects also trigger a special form of the verb as listed in Example 9:

(7) Látom Pistit .
see-1SGOBJ Steve-ACC .
I can see Steve.

(8) Látok egy gyereket az udvaron .
see-1SGSUBJ a kid-ACC the yard-SUP .
I can see a kid in the yard.

(9) Látlak .
see-1SGOBJ2 .
I can see you.

In this way, the feature Definiteness needs to be applied to verbs in Hungarian, moreover, it has a language-specific feature due to the special form triggered by the second person objects. Thus, Definiteness has three possible values in Hungarian: `Definite`, `Indefinite`, `2`.

Determiners and Pronouns

Determiners, pronouns and ordinal numbers also constituted a peculiarity. According to Hungarian grammatical traditions, ordinal numbers have been treated as numerals but in the universal morphology, they have to be annotated as adjectives. Thus, their POS tags were automatically converted to adjectives.

Demonstrative pronouns were also treated differently in the original annotation used in the Szeged Treebank and in universal morphology. While demonstrative pronouns *ez* and *az* are tagged as pronouns independently of their positions, in universal morphology such words occurring before an article should be tagged as a determiner (see Example 10) but when they are used as an NP, they should be tagged as a pronoun (see Example 11).

(10) Olvastam azt a könyvet .
read-PAST-1SGOBJ that-ACC the book-ACC .
I have read that book.

(11) Olvastam azt .
read-PAST-1SGOBJ that-ACC .
I have read that.

These cases were also automatically converted, following the universal morphology guidelines.

Verbal Prefixes

In our original treebank, verbal particles that were spelt as a separate token had their own part-of-speech, i.e. verbal particle. According to the UD description however, not all function words that are traditionally called particles automatically qualify for the PART tag. They may be adpositions or adverbs by origin, therefore should be tagged ADP or ADV, respectively. Thus, we manually compiled a list that contained the original part-of-speech of words that were tagged as verbal prefixes, for instance, *el* "away" was treated as an adverb and *agyon* brain-SUP as a noun – the latter is usually used in phrases like *agyonüt* "kill someone by hitting on his head". Based on this list, we were able to automatically assign UD POS tags to verbal prefixes.

4.3.3 Universal Dependency in Hungarian

When adapting the universal dependency labels to Hungarian, we could find a one-to-one correspondence between the original labels of the Szeged Treebank and the UD labels only in most of the cases, and these labels could be automatically converted to the UD format, making use of the dependency and morphological annotations found in the original treebank. The details of universal dependency rules of Hungarian are available on our website[4].

[4] https://ai.inf.u-szeged.hu/nlp/univdep/

Non-overt Copulas

Traditionally, it is the verb that functions as the head of the clause in dependency grammars but in certain languages, there are verbless clauses where the predicate consists of a single nominal element (typically a noun or an adjective) at the surface level. The dependency analysis of such sentences may be problematic due to the lack of an overt verb. Some studies such as Polguère and Mel'čuk (2009) argue for a zero copula in such cases, especially when the copula is empty only in certain slots of the verbal paradigm. For instance, in Hungarian, the copula has its zero form only in the present tense, indicative mood, third person forms as shown in Examples 12-15:

(12) Present tense, indicative mood, Sg1:

Én tanár vagyok .
I teacher be-1SG .

I am a teacher.

(13) Present tense, indicative mood, Sg3:

Ő tanár .
he teacher .

He is a teacher.

(14) Past tense, indicative mood, Sg3:

Ő tanár volt .
he teacher be-PAST-3SG.

He was a teacher.

(15) Present tense, imperative mood, Sg3:

Ő legyen tanár !
he be-IMP-3SG teacher !

He should be a teacher.

The original dependency analysis in the Szeged Treebank inserts a zero copula (VAN), i.e. a virtual node in the dependency tree, which functions as the head of the clause and the nominal predicate is attached to it. Figure 4.12 shows such an analysis of the sentence *E gondolat sem új* (this thought not new) "This thought is not novel at all".

Beside the function head analysis (i.e. where function words, e.g. the copula is the head), there is another approach to dependencies, namely, the content head analysis, where the head is a content word instead of a function word. In the latter case, the main grammatical relations can be found among content words and all the other function words are attached to the main structure. UD applies the content head analysis, which

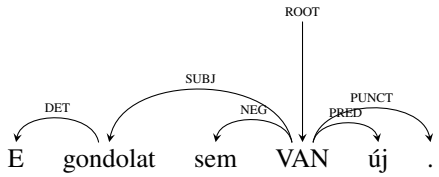


Figure 4.12: A function head analysis in the Szeged Dependency Treebank (*E gondolat sem VAN új* (this thought IS not new) "This thought is not novel at all").

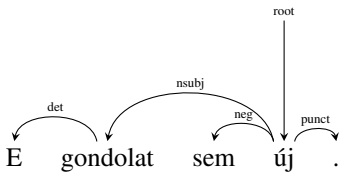


Figure 4.13: A content head analysis in the Hungarian UD treebank (*E gondolat sem új* (this thought not new) "This thought is not novel at all").

means that in copular constructions, the nominal element is the head and the copula (if present) is attached to it with a `cop` relation. In a similar way, the head of adpositional constructions is the noun and the adposition is attached to it.

Sentences with nominal predicates were automatically converted from the original treebank into the UD format: Figure 4.13 shows the UD analysis of the sentence found in Figure 4.12. Likewise, postpositional constructions were converted: the noun was treated as the head and the postposition was attached to it with a `case` label.

Subordinate Clauses

Subordinate clauses proved also to be a problematic issue as UD principles make a sharp distinction among several types of subordinate clauses – e.g. clausal subject, clausal object, adverbial clause – in contract with the Szeged Dependency Treebank, which applies one single label for all types of subordinate clauses. Some types of subordinate clauses had a special label in the constituency version of the treebank hence their conversion was straightforward. In other cases, we could rely on manually constructed conversion rules but the resulting trees had to be corrected manually.

Multiword Named Entities

The UD treatment of multiword named entities required a Hungarian-specific solution. According to the UD principles, the first token of the multiword expressions should be marked as the head. However, in Hungarian, it is always the last element of the multiword expression that is inflected. Examples 16-17 demonstrate that the first element cannot be inflected, only the last one:

(16) Találkoztam Kovács Jánossal .
meet-PAST-1SG Kovács János-INSTR .
I met János Kovács[5].

(17) *Találkoztam Kováccsal János .
meet-PAST-1SG Kovács-INSTR János .
I met János Kovács.

Due to the above morphosyntactic facts, we marked the last token of multiword named entities as the head in the Hungarian UD treebank while all the other UD treebanks mark the first token as the head.

Dative Forms

In Hungarian, nouns that bear the suffix *-nAk* can fulfill several grammatical roles in the sentence such as:

(18) indirect object:

Laci adott a **barátjának** egy almát .
Leslie give-PAST-3SG the friend-3SGPOSS-DAT an apple-ACC .

Leslie gave an apple to his friend.

(19) possessor:

Laci elvette a **barátjának** a könyvét .
Leslie take-PAST-3SGOBJ the friend-3SGPOSS-DAT the book-3SGPOSS-ACC .

Leslie took his friend's book.

(20) dativus ethicus:

Nekem nehogy eladd az autódat !
I-DAT so.as.not.to sell-IMP-2SGOBJ the car-2SGPOSS-ACC !

As for me, you should not sell you car.

(21) experiencer:

Nekem nagyon tetszett az előadás .
I-DAT very like-PAST-3SG the performance .

I really liked the performance.

(22) semantic subject:

Lacinak bocsánatot kellett kérnie a barátjától .
Leslie-DAT apology-ACC must-PAST-3SG ask-INF-3SG the friend-3SGPOSS-ABL .

Leslie had to apologize to his friend.

[5]The standard order of person names is surname + first name in Hungarian.

While these forms do not show any difference at the morphological level, they have very different roles at the syntactic and semantic levels. Thus we decided not to make any distinction in the morphological annotation but they should have different syntactic labels. Indirect objects are marked with the label *iobj*, possessors with the label *nmod:poss* and other occurrences with *nmod:obl*. Obviously, these annotations had to be carried out manually as most of these cases could not be easily and unequivocally converted to the UD format only on the basis of morphology and syntax. Consider the following examples (Example 20 is repeated for convenience):

(23) Nekem nehogy eladd az autódat !
I-DAT so.as.not.to sell-IMP-2SGOBJ the car-2SGPOSS-ACC !
As for me, you should not sell your car.

(24) Nehogy eladd nekem az autódat !
so.as.not.to sell-IMP-2SGOBJ I-DAT the car-2SGPOSS-ACC !
You should not sell your car to me.

Example 23 contains a dativus ethicus whereas Example 24 contains an indirect object. The two sentences only have different word orders thus their automatic distinction would not be straightforward.

Light Verb Constructions

Light verb constructions are verb + noun combinations where most of the semantic content of the whole expression is carried by the noun while the syntactic head is the verb (e.g. *to have a shower*, *to make a decision*). They are not uniformly treated in Version 1.3 of the UD treebanks. Light verb constructions are either not marked at all or if they are marked, they may have a special structure or special labels (Nivre and Vincze, 2015). The Hungarian treebank belongs to the latter group, that is, members of light verb constructions bear a special label. For instance, Figure 4.14 shows that the label `dobj:lvc` can be found between the nominal and verbal component of the light verb construction *döntést hoz* (decision-ACC bring) "to make a decision". In this way, the `dobj` part of the label marks that syntactically it is a verb–object relation but semantically, it is a light verb construction, marked by the `lvc` extension of the label.

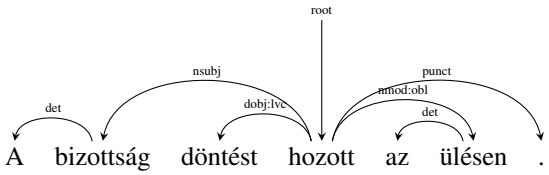


Figure 4.14: Light verb construction in the Hungarian UD treebank (*A bizottság döntést hozott az ülésen* (the committee decision-ACC bring-PAST-3SG the meeting-SUP) "The committee made a decision at the meeting").

4.3.4 Experiments

We developed a converter from the existing Szeged Dependency Treebank (Vincze et al., 2010) to UD and manually corrected 1,800 sentences from the newspaper domain. The manually corrected UD sentences are available in the UD repository v3.0. The experiences gained during the manual correction could reinforce the linguistic conversion rules and the manually corrected gold standard corpus provides the opportunity for empirical evaluations which we introduce in this section.

On the Accuracy of Automatic Converters

Most of the UD treebanks are the result of automatic conversion from a dependency treebank of originally different principles. The accuracy of these automatic converters is unknown, i.e. we do not know how much information was lost or how much noise was introduced by the converters. To empirically investigate this in the case of Hungarian UD, we compared the converted and the manually corrected, i.e. gold standard, trees of the 1800 sentences.

The converter itself is based on linguistic rules (it is available on our website[6]) which were iteratively improved by manually investigating the results of conversion on sentences of the Szeged Dependency Treebank. The final version of the converted achieves an UAS of 87.81 and a LAS of 75.99 on the 1800 sentences compared against the manually corrected UD trees. We believe that this level of accuracy is not sufficient for releasing the rest of the 80,000 sentences of the automatically converted Szeged Dependency Treebank. On the other hand, some of the shortcomings of the automatic conversion could be corrected by exploiting annotation found in other versions of the Szeged Treebank. For instance, the type of certain subordinate clauses is marked in the constituency version of the treebank, which can be transformed into UD labels. Moreover, coreference annotations from the subcorpora annotated for coreference relations could enhance the proper attachment of relative clauses. We intend to add these pieces of information to our converter in the future, hence higher accuracy scores can be provisioned for the automatic conversion process: just with the above mentioned corrections, an additional 6 percentage points could be achieved in terms of LAS as about 20% of the errors are due to subordinate or relative clauses.

On the Price of Universality

We carried out experiments for investigating whether is there any difference between using the original MSD (Vincze et al., 2014) and the new universal morphological (UM) descriptions. We were particularly interested in the utility of the two representations for dependency parsing. We trained two models of the MarMot morphological tagger (Müller et al., 2013) using the two morphological representation in 10-fold cross-tagging on our manually corrected 1800 sentences. Then we trained and evaluated the Bohnet dependency parser (Bohnet, 2010a) on the train/test split of the UD repository v3.0 utilizing the two different predicted morphological descriptions. We used the default parameters for both the MarMot and the Bohnet parser.

[6] http://rgai.inf.u-szeged.hu/dependency

Morph. labels	Dep labels	UAS	LAS (full label)	LAS (main label)
UM	full label	81.94	76.98	78.39
MSD	full label	82.27	77.50	78.75
UM	main label	81.70	–	78.39
MSD	main label	82.17	–	78.58

Table 4.3: Dependency parsing results on the Hungarian Universal Dependency dataset. In the case of $LAS(main\ label)$ we do not check the language specific part of the dependency labels in the evaluations while we compare the universal and language-specific dependency labels at $LAS(full\ label)$.

Table 4.3 presents unlabeled (UAS) and labeled (LAS) attachment scores achieved by the parser on the test set. The first column of the table indicates whether the universal morphology (UM) or the original MSD morphological codes were employed in the experiment. The second column of the table shows which dependency label set was used for training the Bohnet parser. `main label` refers here to the universal dependency labels while `full label` refers to using the concatenation of universal and language-specific labels. The difference between the last two columns of the table is that we checked the full or only the main dependency labels at evaluations.

Table 4.3 shows the MSD outperforms UM consistently at each of the experiments. Although these differences are not high, this suggests that some information encoded in the MSD morphology is not represented in UM, i.e. we have to pay a price to be universal. We can observe the greatest difference when training and evaluating on full dependency labels, i.e. language-specific morphological features contribute to the prediction of language-specific dependency labels.

We made a manual error analysis of the results with regard to attachment (UAS) errors, i.e. we compared the outputs of the dependency parsers trained by using predicted universal codes and predicted MSD morphological codes, respectively. Results are presented in Table 4.4. We found that the benefits of the original language-specific annotation (MSD) mostly manifests in the treatment of subordinate clauses, adverbial modifiers and infinitival complements. These results might be explained by the fact that in certain cases, MSD contains more detailed grammatical information than the UM formalism. For instance, MSD encodes whether a conjunction connects clauses or words/phrases, which information is missing from UM. Also, higher results were achieved for cases when two nouns or adjectives were following each other and one of them modified the other (as in *magas rangú képviselői* "representatives of high standings"). However, sentences containing an overt or covert form of the copula could be parsed more effectively by using universal morphology codes.

The Added Value of Language-specific UD Labels

We also investigated the impact of the language-specific parts of the dependency labels. As the numbers in Table 4.3 show, slightly better results can be achieved both in terms of UAS and LAS when training the model with `full labels` than with `main labels`. This highlights the importance of adding language specific distinctions to the universal ones because they may contain information that can be exploited during the tree

Error type	MSD	%	UM	%
Coordination	100	32.05	98	30.34
Article	44	14.10	44	13.62
Adverbial	35	11.22	43	13.31
Other	37	11.86	31	9.60
Part/adj compl.	31	9.94	32	9.91
Adjacent N/A	15	4.81	20	6.19
Subordination	13	4.17	17	5.26
Copula	14	4.49	11	3.41
Infinitive	9	2.88	15	4.64
Nominal arg.	8	2.56	8	2.48
Possessor	6	1.92	4	1.24
Total	312	100	323	100

Table 4.4: Error analysis: the number and ratio of specific error types.

decoding. They contribute even to unlabeled attachment decisions. To take an example, UD does not make any distinction among different types of nominal modifiers, treating them as `nmod`. However, for Hungarian, we applied extra labels such as `nmod:poss` for possessors (see Section 4.3.2) and `nmod:obl` for nominal arguments of the verb. As for the first, it should always be attached to the possessed noun, whereas the second one is attached to a verb (see also Examples 19 and 20 with the dative morphological case). Thus, the parser can learn these fine-grained distinctions, which might be beneficial for the unlabeled attachment scores as well.

Also, we pointed out that the utilization of language-specific labels does not contradict the UD principles. (Vincze et al., 2017) In UD, each language should select the appropriate labels according to their needs but there is no need to apply all of the labels/features. General labels like `nsubj` or `dobj` will be used in most (maybe all) of the UD languages but there are other labels or feature-value pairs that are applicable for only a handful of languages. These ones are now called as "language-specific" features but in principle, their status is not different from those that are more widely applied. So we believe that introducing "language-specific" additions does not harm the UD principles. Moreover, the chief objective of this section's experiments was to highlight the added value of language-specific features and we were able to show that they can even improve parsing accuracy when evaluated exclusively on the general labels. The main goal of UD is to provide a way where the parsing results over languages are comparable, hence using language specific features during decoding but evaluating only on general labels is in line with this comparison principle. Moreover, it indicates for UD treebank developers that – besides general labels – language-specific ones have to be taken seriously.

4.4 HuSpaCy

In this section, I present improvements to a Hungarian text preprocessing toolkit that achieve competitive accuracies compared to the state-of-the-art results in each text processing step (Orosz et al., 2023). An important industrial concern about large language models is the computational cost, which is usually not

worth the accuracy gain. Transformer-based language models require far more computational resources than static word vectors, and their running costs are typically orders of magnitude higher. Furthermore, practical NLP solutions using large language models often only outperform more lightweight systems by a small margin.

We focus on text processing pipelines that are controllable, resource-efficient and accurate. We train new word embeddings for cost-effective text processing applications and we provide four different sized pipelines, including transformer-based language models, which enable a trade-off between the running costs and accuracy for practical applications. To make our pipelines easily controllable, we implement them in the spaCy[7] framework (Honnibal et al., 2020) by extending HuSpaCy (Orosz et al., 2022) with new models.

4.4.1 Background

Specification for Language Processing Pipelines for Industrial Use

Text processing tools providing representation for hand-crafted rule construction should consist of tokenization, sentence splitting, PoS tagging, lemmatization, dependency parsing, named entity recognition and word embedding representation. These solutions have to be accurate enough for real-world scenarios while they should be resource-efficient at the same time. Last but not least, modern NLP applications are usually multilingual and should quickly transfer to a new language. This can be provided by relying on international annotation standards and by the integration into multilingual toolkits.

Multilingual NLP Toolkits

Thanks to the UD project, it is now possible to easily construct multilingual NLP pipelines. Among the most commonly utilized toolkits are UDPipe (Straka, 2018), Stanza (Qi et al., 2020), UDify (Kondratyuk and Straka, 2019), Trankit (Van Nguyen et al., 2021) and spaCy.

On the one hand, these systems exhibit a high degree of algorithmic diversity. They can be classified into two distinct groups based on their utilization of neural networks. UDPipe, spaCy and Stanza apply older, but faster architectures built on word embeddings employing convolutional and recurrent layers, respectively. On the contrary, UDify and Trankit leverage transformer-based large language models, with the former using multilingual `BERT` (Devlin et al., 2019) while the latter utilizing `XLM-RoBERTa-large` (Conneau et al., 2020).

On the other hand, these frameworks are typically limited by the fact that they rely solely on the Universal Dependencies datasets, which may present a disadvantage in languages such as Hungarian, which have large corpora incompatible with UD. Regarding named entity annotations, Stanza is the only tool supporting NER for Hungarian.

[7] https://spacy.io/

Hungarian Language Processing Tools

The landscape of the Hungarian text processing systems was similar to that of English before the "industrial NLP revolution". There were a number of standalone text analysis tools (Simon et al., 2012) capable of performing individual text processing tasks, but they often did not work well with each other.

There were only two Hungarian pipelines that tried to serve industrial needs. One of them, magyarlanc (Zsibrita et al., 2013), was designed for industrial applications offering several desirable features such as software quality, speed, memory efficiency, and customizability. However, despite being used in commercial applications in the real world, it has not been maintained for several years and lacks integration with the Python ecosystem. The other pipeline, called emtsv (Simon et al., 2020; Indig et al., 2019; Váradi et al., 2018), aimed to integrate existing NLP toolkits into a single application, but neither computational efficiency nor developer ergonomics were the main goals of the project. Additionally, while magyarlanc natively uses the universal morphosyntactic features, emtsv can only do this through conversion. Both pipelines use dependency annotation that is incompatible with Universal Dependencies, furthermore, none of them can utilize word embeddings or large language models, which have become increasingly important in recent years.

In contrast, the development of HuSpaCy placed emphasis not only on accuracy, but also on software ergonomics, while also adhering to the international standards established by Nivre et al. (2020). Moreover, it is built on spaCy, enabling users to access its full functionality with ease. One significant drawback of this tool is the lack of precise annotations for lemmata, entities and dependencies syntax.

To fulfill the industrial requirements of text processing pipelines, this work is built on the Universal Dependencies annotation schema and our models are implemented in spaCy by extending HuSpaCy's text processing model. The detailed documentation, intuitive API, high speed and accuracy of these tools make them an optimal choice for building high-performing NLP models. Additionally, HuSpaCy utilizes non UD compatible corpora as well, which allows for a comprehensive analysis of Hungarian texts.

4.4.2 Methods

HuSpaCy's Internals

HuSpaCy's main strength lies in the clever usage of available Hungarian linguistic resources and its multi-task learning capabilities inherited from spaCy. Its machine learning approach can be summarized as "embed, encode, attend, predict" shown in Figure 4.15 and detailed by (Honnibal et al., 2013; Orosz et al., 2022). Tokens are first embedded through the combination of lexical attributes and word vectors, then context encoding is performed by stacked CNN (Lecun et al., 1998) layers[8]. Finally, task specific layers are used parallelly in a multi-task learning setup.

Orosz et al. (2022) used a three step approach for fully utilizing annotated Hungarian datasets. First, they pre-train the tagger, the lemmatizer, and the sentence boundary detection components on the Universal Morphology version of the Szeged Corpus (cf. (Vincze et al., 2017)). Then, the Tok2Vec layers of this model

[8]These steps are usually referred to as the Tok2Vec layers.

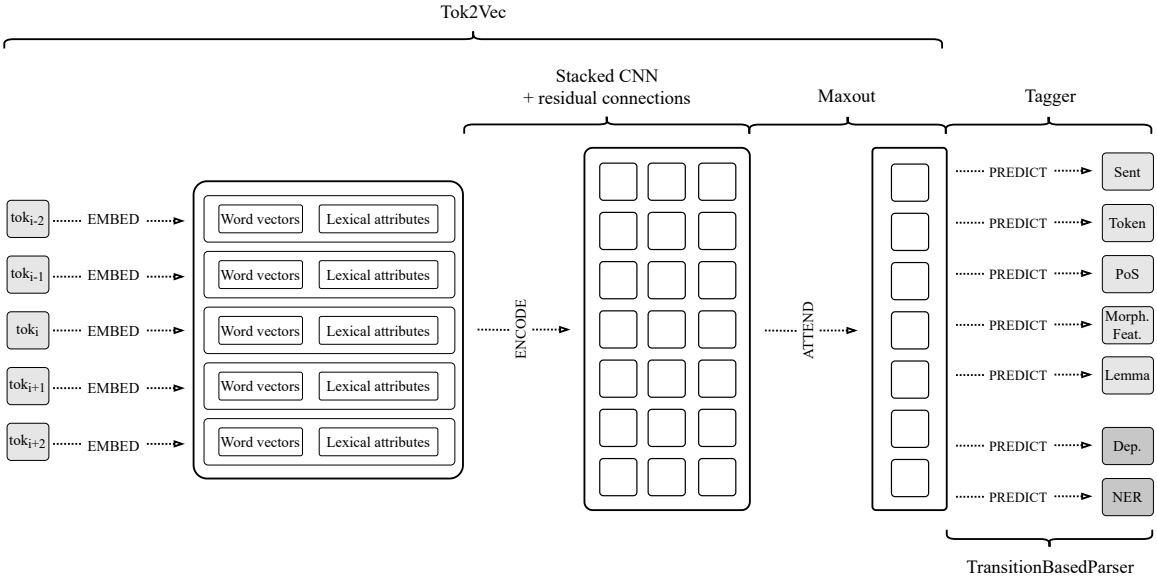


Figure 4.15: The "embed, encode, attend, predict" architecture of spaCy

are reused by both the NER and the parsing components: the dependency parser and the morphosyntactic taggers are fine-tuned on the UD-Hungarian dataset, the lemmatizer is trained on the entire Szeged Corpus, while the entity recognizer is further trained on the combination of the NYTK-NerKor (Simon and Vadász, 2021) and the Szeged NER (Szarvas et al., 2006) datasets.

Improving on the Underlying Language Models

HuSpaCy's model is built on `word2vec` (Mikolov et al., 2013) word embeddings, which are known to have limitations in providing meaningful representations for out-of-vocabulary words. This is particularly problematic for morphology-related tasks in agglutinative languages. To enhance this simple approach, a more fine-grained method that uses sub-word embeddings can be employed. `fastText` (Bojanowski et al., 2017) is a widely-used extension of `word2vec` that learns sub-token embeddings. In this section, we utilized `floret`[9] which is a spaCy-compatible fork of `fastText`. To train new word vectors, we used the Hungarian Webcorpus 2.0 (Nemeskey, 2020b). Two sets of word embeddings were constructed: a 100-dimensional and a 300-dimensional one.

In recent years, there has been a growing interest in transformer-based large language models (LLM), as evidenced by their high performance in text processing models (e.g. (Nemeskey, 2020b; Enevoldsen et al., 2021)). With the advent of spaCy's native support for such architectures and the availability of pre-trained language models for Hungarian, it is now possible to train transformer-based NLP pipelines for Hungarian. Our research is based on two widely used LLMs that provide support for Hungarian. One of these is `huBERT`

[9] https://explosion.ai/blog/floret-vectors

(Nemeskey, 2020b), which has a `BERT`-like architecture and was trained using monolingual data. The other model is `XLM-RoBERTa-large`, which has a much larger capacity compared to the former model and was trained on multilingual corpora.

Pipeline Component Enhancements

In addition to the use of more powerful language models, we propose fundamental changes to the lemmatization and dependency parsing models, as well as minor improvements to the entity recognizer.

HuSpaCy's lemmatizer has been replaced by a new edit-tree-based architecture, recently available in the spaCy framework[10]. This new model builds on the foundations laid out by Müller Müller et al. (2015) (called the Lemming model), but has minor differences from it. On the one hand, this reimplementation fully utilizes the framework's multi-task learning capabilities, which means that the lemmatizer is not only co-trained with PoS and morphological tagging, but also with sentence boundary detection. On the other hand, spaCy's version lacks standard support for morphological lexicons which Lemming benefited from.

We have improved this model in two steps. 1. A simple dictionary learning method is put in place to memorize frequent *(token, tag, lemma)* triplets of the training data which are then used at prediction time to retrieve the roots of words. 2. A common weakness of Hungarian lemmatization methods is addressed. Computing the lemmata of sentence-starting tokens can be challenging for non-proper nouns, as their roots are always lowercase. Thus, we force the model to use the true casing of such words. For example, when computing the root of the sentence starting *Ezzel* 'with this' token, our method checks its PoS tag (that is ideally PRON) first, so that it can use the lowercase wordform for generating and looking up edit-trees.

Moving on, the dependency syntax annotation component is replaced with a model that has higher accuracy for many languages. Although spaCy's built-in transition-based parser (Honnibal et al., 2013) has high throughput, it falls short on providing accurate predictions. Graph-based architectures are known to have good performance for dependency parsing (e.g. (Altıntaş and Tantuğ, 2023)), making such methods good enhancement candidates. Furthermore, a spaCy-compatible implementation of Dozat and Manning's model (Dozat and Manning, 2017) (referred to as the Biaffine parser) has recently been made available, thus we could easily utilize it in our experiments.

Finally, the named entity recognizer has been fine-tuned to provide more accurate entity annotations. This was primarily achieved by using beam-search in addition to the transition-based NER module.

4.4.3 Experiments and Results

This section presents the results of several experiments that demonstrate the improvements of our changes and show competitive results compared to well-established baselines. We evaluated pipelines developed on datasets used by the creators of HuSpaCy: the Hungarian part of the Universal Dependencies corpus[11] was utilized to benchmark the sentence boundary detector, the lemmatizer, the PoS and morphological taggers,

[10] https://explosion.ai/blog/edit-tree-lemmatizer
[11] Experiments are performed at the v2.10 revision.

and the dependency parser, while the entity recognizer is benchmarked on the combination of the NYTK-NerKor and the Szeged NER corpora (similar to (Orosz et al., 2022) and (Simon et al., 2022)). To account for the instability of spaCy's training process we report the maximum result of three independent runs.

4.4.4 Evaluation of Architecture Improvements

The lemmatization accuracy of the original model has been greatly improved through a number of steps discussed in Section 4.4.2. As evidenced in Table 4.5, incorporation of the new neural architecture along with sub-word embeddings produced significant improvements. Furthermore, changing the default behavior of the `edit-tree lemmatizer` by allowing it to evaluate more than one candidate (see the row `topk=3`) also resulted in a slightly better performance. In addition, the integration of true-casing led to a considerable improvement, and the use of lemma dictionaries also significantly improved lemmatization scores.

	Lemma Accuracy
HuSpaCy	95.53%
+ Edit-tree lemmatizer	95.90%
+ `floret` 300d vectors	96.76%
+ `topk=3`	97.01%
+ True-casing	97.30%
+ Learned dictionary	97.58%

Table 4.5: Lemmatization accuracy on the UD-Hungarian test set of different ablation settings. Rows marked with a "+" indicate a new feature added on top of the previous ones. `topk` is a hyperparameter of the lemmatization model controlling the number of edit-trees considered to be evaluated.

Entity recognition tasks often encounter a challenge in the form of a considerable number of out-of-vocabulary tokens, leading to decreased performance. However, the utilization of `floret` vectors has proven to be effective in addressing this issue, as indicated by the results in Table **??**. Additionally, the use of beam search allowed the model to take prediction history into account, which slightly improved its efficiency.

%newcolumntypeR>X

	NER F_1-score
HuSpaCy	83.68
+ `floret` 300d vectors	85.53
+ Beam search	85.99

Table 4.6: Evaluation of the entity recognition model improvements on the combination of the Szeged NER and NYTK-NerKor corpora. The rows starting with "+" signify the inclusion of a new feature in addition to the existing ones.

The results in Table 4.7 indicate that the improved text representations and the new parsing architecture offer substantial improvements over `HuSpaCy`'s outcomes. However, it is worth noting that spaCy's

CNN-based base model is not fully compatible with the Biaffine parser's architecture. Therefore, parsing improvements were benchmarked on top of a transformer-based encoder architecture using `huBERT`. The results show that the use of `floret` vectors is beneficial to predict morphosyntactic characteristics and dependency relations, while the use of `huBERT-based` text representations substantially improves performance across all subtasks. Furthermore, the `Biaffine` parser significantly outperforms its transition-based counterpart, as evidenced by its better attachment scores.

	PoS Acc.	Morph. Acc.	UAS	LAS
HuSpaCy	96.58%	93.23%	79.39	74.22
HuSpaCy + `floret` 300d vectors	96.55%	93.93%	80.36	74.89
HuSpaCy + `huBERT`	98.10%	96.97%	89.95	83.94
+ Biaffine parser	98.10%	96.97%	90.31	87.23

Table 4.7: Evaluation of text parsing improvements on the UD-Hungarian test set. "+" indicate a new feature added on top of the existing ones.

Comparison with the State-of-the-Art

In addition to parsing and tagging correctness, resource consumption is an important consideration for industrial NLP applications. Therefore, following the approach of Orosz et al. (2022) we conducted a benchmark study to compare both the accuracy and memory usage as well as the throughput of our models with text processing tools available for Hungarian.

First of all, an important result of this section is a base model (referred to as `lg`), which achieves a good balance between accuracy and resource usage as seen in Tables 4.8 and 4.9. This pipeline is built on top of the 300d `floret` vectors and incorporates all the enhancements described above, except for the new parser. Evaluation data demonstrates that the `lg` pipeline consistently outperforms `Stanza` in all tasks except syntactic dependency relation prediction, which can be explained by the superior parsing model of the latter tool.

We present the results of a medium-sized model (`md`) as well that is a reduced version of the **`lg`** pipeline utilizing the smaller (100d) word embeddings. Surprisingly, the `md` pipeline delivers performance similar to that of the larger model. Furthermore, the medium-sized model achieves scores comparable to or higher than those of `HuSpaCy`, despite requiring half the memory and exhibiting much higher throughput on CPU.

Transformer-based pipelines using the graph-based dependency parser have the highest scores across all language analysis tasks. Remarkably, despite its smaller capacity, the model based on `huBERT` (`trf`) achieves the highest attachment scores for dependency parsing, while the one using `XLM-RoBERTa-large` (`trf_xl`) provides slightly more accurate PoS tags and named entities.

It is important to consider that not all third-party pipelines in Table 4.8 are directly comparable to our results, due to differences in the versions of the UD-Hungarian dataset used to train and evaluate their models.

[12] All benchmarks are run on the same environment having AMD EPYC 7F72 CPUs and NVIDIA A100 GPUs.

	Sent. F_1-score	PoS Acc.	Morph. Acc.	Lemma Acc.	UAS	LAS	NER F_1-score
emtsv	*98.11*	*89.19%*	*87.95%*	*96.16%*	–	–	**92.99**
Trankit	*98.00*	*97.49%*	*95.23%*	*94.45%*	***91.31***	***87.78***	–
UDify	–	96.15%	90.54%	88.70%	88.03	83.92	–
Stanza	97.77	96.12%	93.58%	94.68%	84.05	78.75	83.75
HuSpaCy	97.54	96.58%	93.23%	95.53%	79.39	74.22	83.68
`md`	97.88	96.26%	93.29%	97.38%	79.25	73.99	85.35
`lg`	98.33	96.91%	93.93%	97.58%	79.75	74.78	85.99
`trf`	99.33	**98.10%**	**96.97%**	98.79%	90.31	87.23	91.35
`trf_xl`	**99.67**	97.79%	96.53%	**98.90%**	90.22	86.67	91.84

Table 4.8: Text parsing accuracy of the novel pipelines compared to HuSpaCy, Stanza, UDify, Trankit and emtsv. Results for non-comparable models are shown in italics.

	Throughput CPU	Throughput GPU	Memory Usage (GB)
emtsv	113	–	3.9
Trankit	434	2119	3.7
UDify	129	475	3.2
Stanza	30	395	5.3
HuSpaCy	1525	6697	3.5
`md`	2652	3195	1.4
`lg`	847	3128	3.2
`trf`	273	2605	4.8
`trf_xl`	82	2353	18.9

Table 4.9: Resource usage[12] of the new models and state-of-the-art of text processing tools available for Hungarian. Throughput is measured as the average number of processed tokens per second, while memory usage columns records the peak value of each tool.

To ensure a fair comparison, `Stanza` and `UDify` have been retrained. On the other hand, we obtained the results of `Trankit` from (Van Nguyen et al., 2021) since it would be a demanding task to fine-tune this model. Furthermore, the results of emtsv's text parsing components (Orosz and Novák, 2013; Novák, 2014; Novák et al., 2016) cannot be deemed reliable either (cf. (Orosz et al., 2022)), since its components use a different train-test split of the Szeged Corpus. However, this tool's entity recognition module (`emBERT` (Nemeskey, 2020a)) was evaluated by Simon Simon et al. (2022) using the same settings as in this section, thus we rely on their assessment. Additionally, state-of-the-art results are also shown in Table 4.8. With regard to highest dependency parsing scores, the results of the multilingual `Trankit` system are produced by a parsing model similar to that of ours. As for named entity recognition, `emBERT` attains the best F_1 scores by utilizing a Viterbi encoder that eliminates invalid label sequences from the outputs of the underlying model.

Regarding computational requirements, Table 4.9 presents findings that demonstrate how `floret` embeddings can effectively decrease the memory usage of models without compromising their accuracy and

throughput. However, it is apparent that enhancing pipeline accuracy frequently results in slower processing speed, as can be observed from the `lg`, `trf` and `trf_xl` models. Additionally, our tests also showed that most of the readily available NLP pipelines are not adequately optimized to handle large workloads, which is evident from their low throughput values.

4.5 Summary

In this chapter, I showed the results of three studies.

Converting constituency trees into dependency trees also made possibility to experiment with a silver standard dependency corpus. The results empirically showed that better results can be achieved on the gold standard corpus, hence manual annotation of dependency trees is desirable. However, when there is no access to manually annotated dependency data, converting the output of a constituency parser into dependency format or training the dependency parser on converted data may also be viable: similar to English, both solutions result in competitive scores but the errors the systems make differ from each other. These results were published in the article *An Empirical Evaluation of Automatic Conversion from Constituency to Dependency in Hungarian* at the COLING 2014 conference (Simkó et al., 2014).

In section 4.3, I presented how the principles of universal dependencies and morphology have been adapted to Hungarian. Experiments were introduced on the new manually annotated corpus for evaluating automatic conversion and the added value of language-specific, i.e. non-universal, annotations. This work was published at the EACL 2017 conference with the title *Universal Dependencies and Morphology for Hungarian - and on the Price of Universality* (Vincze et al., 2017).

In section 4.4, I introduced the HuSpaCy a new industrial-grade text processing pipeline for Hungarian and presented a thorough evaluation showing their (close to) state-of-the-art performance. This work was published in the paper *Advancing Hungarian Text Processing with HuSpaCy: Efficient and Accurate NLP Pipelines* at the TSD 2023 conference (Orosz et al., 2023).

Contribution

In the Simkó et al. (2014) and Vincze et al. (2017) I made the following contributions:

- I implemented the constituency to dependency and the universal dependencies converters.
- I provided statistical support to linguists to develop the rules.
- I did the Machine Learning experiments in both works to compare various representations.

In the Orosz et al. (2023), my contribution was the application of different dependency parser heads.

Part III

Application of Syntax Parsing

This Part focuses on the application of syntactic parsing. It introduces three applications of syntactic parsing in higher-level tasks. In chapter 5 I focus on the exploitation strategies of syntactic structures for in-sentence and sentence-level sentiment analysis (Hangya et al., 2017). The chapter 6 introduces the work of Team Szeged for the *First Shared Task on Extrinsic Parser Evaluation* (EPE 2017) (Szántó and Farkas, 2017), I show three approaches to exploit the opportunities of the general dependency graph representation of the shared task. The last chapter of this Part presents strategies for medication event extraction and classification, revealing how syntax-based information extraction unlocks efficiency gains in the case of pretrained language models too (Szántó et al., 2023).

Chapter 5

Latent Syntactic Structure-Based Sentiment Analysis

People publicly share their opinions using social media on a variety of topics, like products and political issues. The task of sentiment analysis (SA) is to automatically extract opinions from textual content. Most of the SA systems assign polarity labels (e.g. positive, negative, and neutral) to textual elements like documents and sentences. The basic solution for SA is to represent the texts in a bag-of-word model and train supervised classifiers or/and employ polarity lexicons for polarity classification (Ravi and Ravi, 2015).

Previous studies have been investigating the utilization opportunities of the syntactic structure of sentences for enhancing sentiment analyzers. Most of these proposals use hand-crafted rules based on the syntactic parse of the sentence (Vilares et al., 2013). These rules are engineered to address certain restricted sets of in-sentence SA's challenges, like negation and intensification.

In the Stanford Sentiment Treebank (Socher et al., 2013) a polarity label was manually assigned to each constituent of the sentence's phrase structure parse. This treebank can be utilized as a training dataset for statistical structure prediction methods and it introduces the opportunity of exploiting the syntactic structure of sentences without restricting the models to a closed set of language phenomena (like negation and intensifiers), neither demands the direct modeling of those phenomena. It enables the application of supervised machine learning techniques to model how morphosyntactic and lexical structures alter the polarity of a constituent. On the other hand, the supervised approach has the disadvantage of requiring a manually annotated treebank. This treebank is domain-dependent, i.e. sentiment analyzers trained on it work fine only on movie reviews and the annotation of new treebanks for other domains is expensive.

In this chapter, I focus on the exploitation strategies of syntactic structures for in-sentence and sentence-level SA (Hangya et al., 2017). Usually, sentence-level polarity labels can be easily obtained in a huge amount for various domains, take for instance pro/con or bottomline summaries of the product review sites. Hence this chapter proposes a machine learning framework for sentence-level and in-sentence polarity classifiers

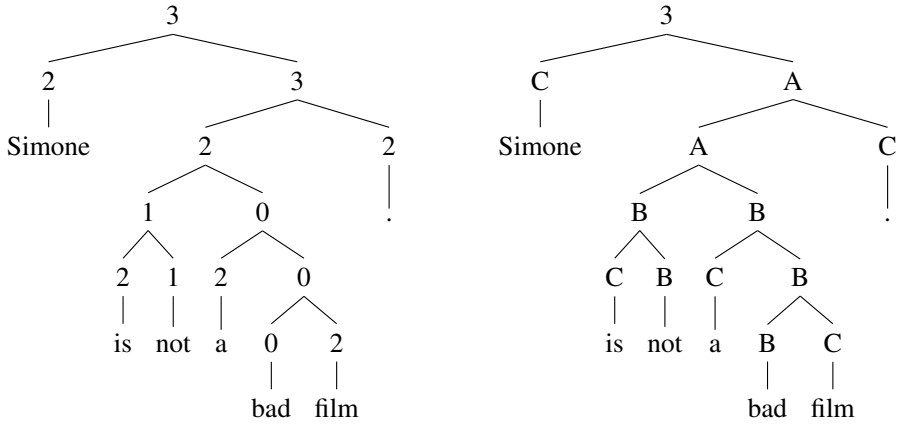


Figure 5.1: Representation of sentiment trees in the Stanford Sentiment Treebank (Socher et al., 2013) (left) contains 5-level polarity annotation {`0=very negative`, `4=very positive`} for each node of the binary syntactic tree. On the other, we assume that we have access only sentence-level polarity annotation, i.e. only the label of the root is given (right). Here, the states of the inner nodes are described by latent discrete variables {`A,B,C`}.

by using exclusively sentence-level polarity annotation for training. This approach can predict the sentiment labels assigned to the constituents of a phrase structure parse tree without an annotated sentiment treebank by handling the polarity labels of internal nodes in parse trees as latent variables. Figure 5.1 exemplifies the difference between a fully annotated sentiment tree and the proposed latent representation.

I shall introduce two experimental setups for the investigation of the proposed approach. The objective of the experiments' first batch (in section 5.3) is to investigate whether the sentence-internal latent structure helps the prediction of sentence-level polarity. The second batch of experiments (in section 5.4) shall show that the sentence-internal latent structures themselves are also meaningful when we extract features from them for a target-oriented sentiment analysis task.

The chief added value of this chapter is to propose a latent syntactic structure-based approach that requires only sentence-level polarity labels for training. The experiments on three domains (movies, IT products, restaurants) support that sentiment analyzers are domain-dependent.

5.1 Related Work

SA is an actively researched area (Liu, 2012) due to the fact that a huge amount of data is available on the Internet. In the early stages most of the systems were based on supervised machine learning techniques using bag-of-words representation, see (Ravi and Ravi, 2015) for a survey on SA. The goal of the *SemEval-2014 Task 9 – Sentiment Analysis in Twitter* (Rosenthal et al., 2014) was to classify short messages into polarity classes. Most of the participating systems were based on supervised machine learning techniques. Besides the

standard bag-of-words representation, various lexical resources (Baccianella et al., 2010; Wilson et al., 2005) were also employed in order to improve the performance of these systems. A drawback of these systems is that they cannot exploit the syntactic and semantic structure of texts, i.e. negations, intensifiers, discourse relations, etc.

Several studies have been published on exploiting syntactic parsers for SA. For instance, in (Vilares et al., 2013) a dependency parser was employed in order to detect intensifications and negations. They used hand-crafted rules over dependency parses and lists of intensifiers and negation words respectively. The relation between text fragments can influence the polarity of a document as well. In (Lazaridou et al., 2013) a joint model for unsupervised induction of sentiment, aspect, and discourse information was proposed. They showed that performance can be improved by incorporating latent discourse relations (*but*, *and*, *when*, etc.) in the model.

In the Stanford Sentiment Treebank (Socher et al., 2013) a polarity label was manually assigned to each constituent of the sentence's phrase structure parse and they introduced a Recursive Neural Tensor Network-based procedure to capture the compositional effects of the sentences. Although this approach provides a more free representation for in-sentence SA, it has the disadvantage of requiring a manually annotated treebank. This treebank is domain-dependent and the annotation of new treebanks for other domains is expensive. Our proposal is related to (Socher et al., 2013) as we also start from syntactically parsed sentences but we are handling the polarity labels of internal nodes as latent variables. This way the inputs for training our system are texts annotated only on the sentence level.

A SA approach which is based on in-sentence structures was also introduced in (Dong et al., 2015). They also propose a system that can learn in-sentence sentiment structures using exclusively sentence-level annotation. On the other hand, their system contains several hand-crafted assumptions and rules (e.g. they handle negations and intensifiers by dedicated rules) while our latent representation introduces the opportunity of exploiting the syntactic structure of sentences without restricting the models to a closed set of language phenomena, neither demands the direct modeling of those phenomena. Another difference is that we use a syntactic parser to provide the in-sentence structure while they use a CYK sentiment parser. Although their approach provides an opportunity of learning also the structure itself the running time is cubic hence it is not feasible to train on several hundred thousand sentences.

Sentiment analysis can be applied at different levels depending on the depth of target information (Feldman, 2013). The aim of the so-called target-oriented SA is to classify sentiments that are related to a given target (Jiang et al., 2011). In this case, extracting bag-of-word features is not enough. The aim of the *SemEval-2014 Task 4 – Aspect Based Sentiment Analysis* (Pontiki et al., 2014) shared task was to compare systems on SA tasks. Many participated systems used syntactic parsers to identify text parts that are related to the target phrase in question. One of the novelties of our paper is that we also experimented with target-level SA and we shall show that the induced latent sentiment trees have a considerable added value in a target-level SA system.

Our approach is also related to semantic parsing. For instance, in (Angeli et al., 2012) latent temporal

types were used in a latent CFG to learn temporal expressions. This semantic parsing problem is very similar to ours, both methods require a sentence level label in the training phase and use latent variables in the non-terminals (except the root). They assigned temporal types to non-terminal nodes, in contrast, we have polarity labels in these nodes. They employed an EM-style bootstrapping approach for training the models.

We used structured perceptron for machine learning which is successfully applied for structured prediction with latent variables in other areas of natural language processing. In (Björkelund and Kuhn, 2014) a system similar to ours was introduced for coreference resolution with latent tree structures of mention clusters. They used the passive-aggressive algorithm for this training task and updated against the highest-scored tree with correct clustering of mentions.

5.2 Latent Syntactic Structure-Based Sentiment Analysis

We propose a procedure for predicting the polarity label for each constituent of a sentence's phrase structure parse and we assume that we have access to exclusively sentence-level polarity labels during training.

Preprocessing Sentences are tokenized by the Stanford CoreNLP toolkit (Manning et al., 2014). The syntactic structure of a sentence is fixed, i.e. we syntactically parse each sentence in a preprocessing step. We employed the BerkeleyParser (Petrov et al., 2006b), with the English 6th iteration model. We used right-branch binarised and unlabeled – both POS tags and internal node labels are deleted – syntactic parse trees for sentiment parsing.

Latent State Representation We assume that the system has access to sentence-level binary polarity (`positive` and `negative`) annotations, which serve as the label of the syntactic parse tree's root node. A latent discrete random variable is assigned to each internal node of the parse tree. In our experiments, we use latent variables with three possible states {A,B,C}. Although the number of possible states can be easily changed, we postpone the investigation on the effect of different state space sizes for future research.

Decoder We use a structured perceptron to decode the labels for the nodes of the syntactic parse tree. The decoder iterates through the tree in a bottom-up order and employs only local features which are described in Section 5.2. Preliminarily we experimented with non-local features along with a beam-search decoder but their improvement was not considerable while running times increased exponentially.

The decoder selects the top-scored derivation with latent variables {A,B,C} at the internal nodes and polarity labels {positive, negative} for the root node. It is an exact search, i.e. the derivation space is complete, we do not filter the possible derivations. The branching factor of the derivation space is 9 as we work with binary parse trees and 3 possible states.

We use the hypergraph representation of the derivation space along with the Viterbi decoder from the Joshua software package[1].

[1] Joshua is a JAVA package available at `https://attic.apache.org/projects/joshua.html`.

Training At training time, only the root label is available for the preprocessed sentences and the in-sentence polarity labels at the nodes of the parse tree is handled to be latent variables. We follow an Expectation-Maximization (EM) approach for training the structured perceptron. In the E-step, we select the top-scoring derivation of the gold standard root label. In this way, we update the weights of the perceptron against a latent sentiment tree which is easily learnable by the structured perceptron (Björkelund and Kuhn, 2014).

In the M-step, the update rule of the averaged perceptron is employed (Collins, 2002). The learning rate parameter was set to 0.1 and the batch update size to 30. These parameters were set based on a grid search metaparameter optimization on a development dataset. We run 15 epochs of training in each of our experiments and we use our in-house implementation for training.

Features We implemented three feature templates to extract information from derivation candidates. We use only local features, i.e. which can be extracted from the 1-level subtree of the derivation (see Figure 5.2). The bottom-up decoder extracts the new features for a new node of the derivation and adds them to the feature vectors of the two daughters' feature vector.

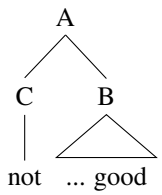


Figure 5.2: The Subtree with latent labels {A, B, C} is the subject of local feature extraction.

We note that our main objective was to investigate whether our latent representation can improve in-sentence SA. There is plenty of space for feature engineering, i.e. introducing other local or non-local features (e.g. the cube pruning approach provides an efficient procedure for incorporating non-local features (Huang, 2008b)).

- The **Word features** extract the cooccurrence of a latent polarity label and the unigrams in the yield of the syntactic parse tree's node. From node A on Figure 5.2 we extract the following features: A-NOT, A-GOOD.

- The **label features** describe the latent structure as they are the rules in the context-free grammar terminology. From our example, we get: A-C-B.

- The **Compositional features** are similar to the label features but we exchange one of the daughters' state label with the head of the particular constituent of the syntactic parse tree. This feature template is designed to capture the lexical dependencies of polarity changing words. In the case of current example we get: A-C-GOOD, A-NOT-B. We experimented with two head finding strategies but the difference between taking the right-most word and the semantic head finding rules of Collins (Collins, 2003) was negligible.

5.3 Sentence-Level Polarity Classification

The goal of the sentence-level SA is to classify the polarity of the sentiment that a given sentence conveys. In our first batch of experiments, we investigated whether the sentence internal latent structure helps the prediction of the sentence-level polarity.

5.3.1 Datasets

For our experiments, we used 3 corpora from the IT products, restaurants and movies domains, all of which were annotated with `positive` or `negative` labels on the sentence level. We split our corpora to train and test sets with sizes of 100,000 and 1,000 text examples.

IT Products We downloaded reviews from the Newegg[2] site. Each review on this site must contain short pro and con summaries of the review in free textual form. We have downloaded the pros and cons of those products which were in the IT category and used them as positive and negative examples, respectively. The downloaded texts were noisy because many of them did not contain the appropriate sentiment (e.g. *PRO: I didn't find any.*). To overcome this problem we used only those texts whose token length is between 6 and 40 tokens and contain only one sentence.

Restaurants In the case of the restaurant review domain, we applied a similar procedure. We used the dataset provided by Yelp[3], which contains reviews about businesses (we only used those which are related to restaurants). Each review is annotated with stars from 1 to 5 by the reviewer. We selected only the ones that were annotated with 1 or 5 as negative and positive examples, respectively. In order to filter out noise, we applied the same method as before.

Movie For the movie review domain we downloaded reviews similarly to (Socher et al., 2013) and (Dong et al., 2015) from `www.rottentomatoes.com`. We filtered this dataset as well and used only the reviews with score 1 and 5 as negative and positive examples.

5.3.2 Experimental Setup

We predict the whole sentiment tree for the test sentences and we considered the label on the resulting trees' root node as the sentence-level polarity.

For comparison reasons we ran the `RNTN` system introduced in (Socher et al., 2013), which – similarly to our system – yields syntactic trees with a polarity level on each node. The difference of this system and ours is that it was pre-trained[4] on fully annotated trees from the Stanford Sentiment Treebank[5]. The system

[2] `www.newegg.com`
[3] `www.yelp.com/dataset_challenge`
[4] We re-trained the system on the train part of the Stanford Sentiment Treebnak.
[5] The Stanford Sentiment Treebank was composed of movie reviews from RottenTomatoes.

	IT products	restaurants	movies
Most frequent class	53.0%	88.0%	50.1%
RNTN	62.1%	79.1%	76.9%
baseline 10k	77.4%	91.8%	67.8%
latent 10k	76.5%	91.9%	67.9%
baseline 100k	82.9%	93.6%	75.7%
latent 100k	83.4%	93.8%	76.6%

Table 5.1: Accuracies achieved on the three domains. RNTN is our reference system (Socher et al., 2013), the baseline is a unigram model and latent refers to the proposed system.

can predict polarity labels along with their probability values on a five level scale (`very negative, negative, neutral, positive, very positive`). Using the probability values we mapped its prediction to `positive` or `negative` labels according to the highest probability value ignoring the neutral label[6].

5.3.3 Results on Sentence-Level SA

Our results can be seen in Table 5.1, which contains the accuracy of the systems of each domain. Our baseline system used only unigram features, so it could not exploit the inner structure of the sentences. We used smaller (10K sentences) and bigger (100K sentences) training sets along with the same test set for evaluating the unigram baseline and the proposed latent representation-based models.

Two conclusions can be made based on Table 5.1. Firstly, in the case of all three domains with 100K train, the `latent` system outperformed the `baseline`. This shows us that by exploiting the latent structure of the sentences, the performance of the SA system could be increased. With the feature templates introduced, our system managed to learn structures, and using this it can classify more sentences correctly than the simple bag-of-words models. It also shows that 10K train sentences are not enough to the latent method, it could even achieve worse results than the baseline in the IT product dataset.

On the other hand, it can be seen that the baseline and our system outperformed the reference system in the case of the IT product and restaurant domains but not in the movie domain. The reason why the `RNTN` system performed well in the movies domain but not in the other two is that it was trained on movie reviews. This confirms the fact that it is important to train an SA model on a domain that is similar to the one on which it will be used. If a fully annotated treebank is available in the given domain, the supervised model is more efficient but competitive results can be achieved with this employing a 10-times bigger training dataset and the proposed latent representation.

[6]We experimented with various mapping strategies from 5 polarity levels to 2 levels but the difference between the achieved accuracies were negligible

5.4 Target-Level Polarity Classification

In the second batch of experiments, we investigated the utility of the latent sentiment annotation for target-level polarity classification. The task of target-oriented SA is to classify sentiments that refer to a given target. The difficulty of this task is that a sentence can contain multiple targets, e.g. *The food was good, but it was too expensive*. In this example, a positive sentiment refers to the food quality but a negative one refers to the prices. Using a SA model that is not aware of the targets can easily misclassify the sentiments. We utilized the sentiment trees for target-oriented SA by inducing the sentiment trees then extract features from them for a target-level polarity classifier.

5.4.1 Target-Level Dataset

For the evaluation of target-level classifiers we used the dataset provided by the organizers of *SemEval-2014 Task 4 – Aspect Based Sentiment Analysis* (Pontiki et al., 2014), which consists of laptop and restaurant review sentences. For each review, aspects of an entity are annotated, such as the *battery life* of a laptop or the *prices* of a restaurant. The aspect mentions are the targets of the sentiment analysis task. For each aspect notation, the polarity level is given depending on the sentiments related to the given aspect in that review. In the database, 4 polarity labels were used which were `positive`, `negative`, `neutral` and `conflict` (when both positive and negative sentiments were referring to a target). We did not use the `conflict` class because of the small number of occurrences in the corpus. The resulting database consists of 2,300 laptop and 3,602 restaurant reviews, which will be referred as *absa-laptop* and *absa-restaurant*. We only used the train sets of the official datasets and ran 10-fold cross-validation to obtain our results. The reason for this decision is that in our early experiments we noticed that the standard deviation of the accuracy among each fold and the test set is high (2.9% and 2.3% for the laptop and restaurant datasets respectively) thus by cross validating we got much more robust results.

5.4.2 Exploitation of Latent Sentiment Trees in Target-Level SA

To solve the target-oriented SA problem we used a bag-of-features model with Naïve Bayes classifier from the MALLET toolkit (McCallum, 2002) with default parameter values. The features describing a sentence consist of word unigrams along with features derived from the predicted latent sentiment tree. We selected a subtree of the whole latent sentiment tree in order to emphasize the part of the text which is related to the target in question. This subtree is the smallest subtree which 1) contains the target mention and 2) has at least as many leaves as the quarter of the number of words in the sentence.

The exact features used by the classifier are the following:

- word unigrams
- label of the sentiment subtree's root
- the label sequence on the path from the root to the target in the subtree
- the number of each polarity label in the above path respectively

	absa-laptops	absa-restaurants
baseline	64.30%	67.42%
baseline + RNTN features	64.81%	66.50%
baseline + latent-tree features	67.47%	69.95%

Table 5.2: Accuracy scores of the target-oriented 3-class classifier whose feature set is enriched by sentiment-tree based features. We calculated the accuracy using 10-fold cross validation on the absa-laptop and absa-restaurant databases using the sentiment tree based features.

- the collapsed label sequence on the path from the root to the target, more precisely we collapsed the consecutive equal labels, e.g. 0_A_A_C_B_B_B $\rightarrow$ 0_A+_C_B+

- the same as the last 4 features but by using the entire tree

5.4.3 Results on Target-Level SA

The accuracy of the target-oriented system can be seen in Table 5.2 for both the absa-laptops and absa-restaurants databases. The baseline for this experiment is a simple bag-of-words model (unigrams without the sentiment tree features). The other rows in the table differ in the model used for predicting the sentiment tree for the sentences. Similar to the sentence-level task, we used the pre-trained fully supervised RNTN system for comparison reasons. In the case of the last row our models trained on 100,000 sentence-level annotated IT product and restaurant datasets were used for the absa-laptops and absa-restaurants respectively.

From the results, it can be seen that the performance of the target-oriented system could be considerably improved by using additional features derived from the sentiment tree. The `RNTN` system was trained on out-domain data, thus it only helped on the laptop dataset but not on the restaurant reviews. Because our model was trained on in-domain data it managed to capture the latent semantics of the given domain more accurately and by using the sentiment tree-based features we managed to increase the accuracy on both target-level corpora.

5.5 Discussion

We manually and statistically investigated the output of the models used in our experiments in order to reveal the reasons for accuracy differences.

The reason why our `latent` model can outperform the supervised `RNTN` system (Socher et al., 2013) lies in the domain differences which were used to train the systems. The RNTN system was trained on movie reviews and it performed better on the Movies test corpus but worse on the other two ones compared to our system which was trained on the same domain as the test domain. The domain difference can be captured at the lexical level. For instance, the word *cheap* has opposite polarity content in the IT and movie domains as it is positive in case we want to buy a device but negative in case of a movie because it implies the poor quality of a film. Similarly *fast* and *quiet* act the same. There are some strongly IT-related terms like *WiFi* or

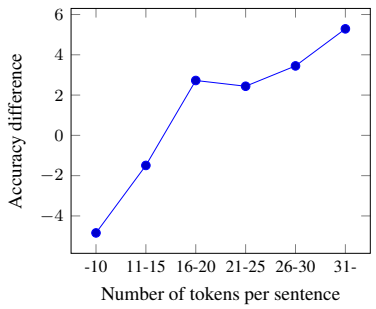


Figure 5.3: Average accuracy improvements in percentage points of the latent system over the baseline system on the movie test dataset in the function of sentence length.

Gigabit which are positive in this domain but neutral in the movies domain thus the RNTN system interprets it incorrectly on IT reviews. The restaurant domain act similarly, there are domain-specific words as well like *Mexican* which bear a different polarity content in the case of cuisines and otherwise.

We investigated the differences between the outputs of the baseline unigram classifier and our latent structure-based model. The only considerable explanation we found is that our in-sentence structure-based method could outperform the baseline with a greater advance at longer sentences. Figure 5.3 depicts the difference between the accuracies achieved by the two system on the Restaurant database in the function of sentence length.

In the case of the target-oriented evaluation, the performance increase was achieved by both the full sentiment tree and the selected subtree. In cases when only one target was presented in a sentence the correct label on the root of the sentiment tree helped the classification. Because our latent model can only predict `positive` or `negative` on the root (due to the fact that it was trained using binary training data) this could not help in the case of the `neutral` label. On the other hand, when multiple targets were in a sentence, the label of the selected subtree helped the classification. We sorted the feature of the Naïve Bayes model by the absolute value of learned feature weights. The top two features from the sentiment subtree-based features were the label `C` which indicated the `neutral` class and the number of each polarity label on the path from the root to the target. On the other hand, the full and the collapsed path-based label sequence features were less effective because their data sparsity. With the additional sentiment tree-based features we managed to improve the classification of the `positive` and `negative` labels on both absa datasets and in the case of the laptop domain we increased the accuracy of the neutral class as well. This latter result is surprising because our latent model was trained only on `positive` and `negative` class labels.

5.6 Summary

In this chapter, I introduced a sentiment analysis framework that uses a latent state representation on the syntactic structure of the sentence in question (Hangya et al., 2017). The main added value of the system is

that it uses only sentence-level polarity annotations for training while it is not required to manually handle in-sentence language phenomena (like negation and intensification). The experimental results introduced support the fact that polarity classification is a highly domain-dependent task as the analysers trained on out-domain sentences failed. They also showed that the currently proposed sentiment analyzer which has access only to sentence-level polarity annotation for in-domain sentences can outperform models that were trained on out-domain parse trees with sentiment annotation for each node of the trees. In practice, millions of sentence-level polarity annotations are usually available for a particular domain thus the currently proposed approach is applicable for training a sentiment analyzer for a new domain and it can exploit the syntactic structure of sentences.

Besides the evaluation of sentence-level polarity classifiers, the internal structure of the sentiment trees in target-level polarity classification was utilized as well. The features extracted from the sentiment trees had a considerable added value for target-level polarity classification and the results also show that the latent sentiment trees predicted by models trained in-domain are more useful than the concrete sentiment trees predicted by RNTN which was trained on an out-domain treebank.

This work was published at the *2nd IEEE International Conference on Computational Intelligence and Applications* conference with the *Latent syntactic structure-based sentiment analysis* title (Hangya et al., 2017).

Contribution

- Proposal of the sentiment tree representation.
- The latent structured decoder and the training algorithm.

The idea of fine-grained analysis with just sentence annotations cannot be distributed between the coauthors.

Chapter 6

Application of Generalized Syntactic Parsing Framework

In this chapter, I introduce the work of Team Szeged for the *First Shared Task on Extrinsic Parser Evaluation* (EPE 2017) (Szántó and Farkas, 2017). I present three approaches to exploit the opportunities of the general dependency graph representation of the shared task.

The goal of the EPE 2017 was to estimate "*the relative utility of different types of dependency representations for a variety of downstream applications that depend heavily on the analysis of grammatical structure*". (Oepen et al., 2017).

To enable different types of dependency representations, the organizers of the shared task introduced a very general graph-based representation of '*relational*' structure reflecting syntactic-semantic analysis. The nodes of this graph correspond to lexical units, and its edges represent labeled directed relations between two nodes. Nodes can be defined in any terms of (in principle arbitrary) sub-strings of the surface form of the input sentence. This representation allows overlapping and empty (i.e. zero-span) node sub-strings as well. Moreover, nodes and edges are labeled by attribute–value maps without any restriction on the attribute set.

This very general graph-based representation opens brand new ways for expressing syntactic or semantic information besides the standard dependency tree formalism. We understood the call of the shared task in a generalized way and came up with ideas that aim to leverage the opportunities of the general representation beyond dependency parse trees. We experimented with a couple of such ideas (instead of trying to achieve high scores in the shared task).

In the first set of experiments (in section 6.2), we start from the classic dependency parsing approach but instead of a single dependency parse, we express the distribution of possible dependency parses given a sentence in the graph-based general representation. In Section 6.3, I introduce a possible solution for enriching the dependency parse by constituent information given by a standard phrase-structure parser. In this way, various syntactic representations can be represented in the graph and information is not lost because

the downstream application can only accept a single dependency parse tree. Furthermore, in the EPE 2017 setting, we can send a blended relational structure to the downstream task, like a parse distribution and blended version of different syntactic approaches, and the downstream application is able to machine learn which type of syntactic structure or phenome or even which combination of syntactic information is useful for itself.

Our last batch of experiments (in section 6.4) is a consequence of this objective, i.e. the relational representation has to be useful for the downstream application. Here, we tried to automatically recognize which dependency parse labels are useful to a downstream task and collapsed the useless ones.

6.1 Downstream tasks

The EPE 2017 shared task consists of three downstream tasks, biological event extraction, Negation scope resolution, and fine-grained opinion analysis. For each downstream task, the organizers applied systems that previously achieved good results and applied syntax-based features. They adapted those systems to enrich information from the dependency graph representation of the shared task.

Biological event extraction The task (Björne et al., 2017) utilizes the GENIA corpus (Kim et al., 2003) and the Turku Event Extraction System (Björne et al., 2009) from the BioNLP'09 shared task (Kim et al., 2009) to extract and classify biological events, specifically focusing on typed, text-bound events involving proteins.

Negation scope resolution The task (Lapponi et al., 2017) aims to find the negations in the texts, determine their scope, and find the negated event in that scope. It applies the dataset of the SEM 2012 shared task (Morante and Blanco, 2012) and employs the Sherlock system (Lapponi et al., 2012), which reached first place in that shared task.

Fine-grained opinion analysis The task (Johansson, 2017) uses the Trento–Gothenburg system (Johansson and Moschitti, 2013) on the MPQA (Wiebe et al., 2005) dataset, which has similar motivations to the Stanford Sentiment Treebank (Socher et al., 2013) from the previous chapter. However, it is built upon three types of linguistic expressions: `direct-subjective expressions` that explicitly mention emotions and opinions, `expressive-subjective elements` that hint at attitudes without explicitly stating them, and `objective statement expressions` that convey factual information without expressing opinions. In addition to the type of expression, each instance is also associated with an opinion holder, who expresses the sentiment. Moreover, `direct-subjective expressions` and `expressive-subjective elements` are assigned a polarity, indicating the sentiment's direction (`positive`, `negative`, or `neutral`) (Johansson, 2017).

6.2 Parse Distribution as Input for Downstream Applications

Standard dependency parsers output a single dependency parse tree. Our hypothesis was that a downstream application could profit from having access to the distribution of possible parses and not just to the most likely parse tree. The distribution over possible parses estimated by the parsing model might be useful for a downstream application because it might reveal that edges or their labels are less confident or also point out relatively highly probable dependencies that are not part of the best single parse tree. The general graph-based representation of EPE 2017 enables to express the distribution over possible edges and labels, i.e. possible parses.

Getting out the density estimation from a particular parser is usually complicated because of both theoretical and practical (software implementation) issues. Hence we decided to use an approximation of edge and label likelihoods based on top-k parses for our first experiments. Our assumption here is that the top k output of a parser model contains most of the useful bi-lexical dependencies and the frequency of a particular dependency counted among the k parses is a good enough approximation for its likelihood (this idea is similar to the constituent-level strategy of the Berkeley product parser (Petrov, 2010c)).

We added each edge from the k-best trees of a parser to the general dependency graph. We also added a new label to all edges whose value is the frequency of the same edge label pairs among the k parses. For these experiments, we used the MSTParser (McDonald et al., 2005) which we trained on Universal Dependencies v2 (Nivre et al., 2015) and we asked for the 10-best trees with default parameters.

6.3 Constituents in the (Bi-Lexical) Relational Representation

Constituency parsers focus on the phrases/constituents and phrase structure of the sentence, i.e. follow a non bi-lexical syntactic representation. Several applications might prefer bi-lexical representations (like the ones based on predicate-argument structures) while others might prefer constituency (like scope detection). Fortunately, the general graph representation of EPE 2017 enables us to put both the dependency and constituency parse output into a blended syntactic graph. Hence we do not have to choose between the two approaches but the downstream application can machine learn which syntactic phenomena are useful for itself or even can learn patterns in the graph consisting of information from both constituency and dependency. A couple of previous works have shown that the two syntactic representation and their parsers can work together efficiently cf. (Farkas and Bohnet, 2012b). We believe that is especially true for using them jointly in downstream applications.

There are many possible ways how we can represent a constituent tree in the general dependency graph format. Although these representations contain the same information because of the feature extractors of the downstream applications they can have different effects in practice.

An interesting opportunity of the EPE 2017 general graph representation is that it enables the creation of virtual nodes. This feature gives the possibility to create a new node for each non-terminal in a constituent tree. Our three proposals differ in how these virtual nodes are linked to the overt nodes in the graph.

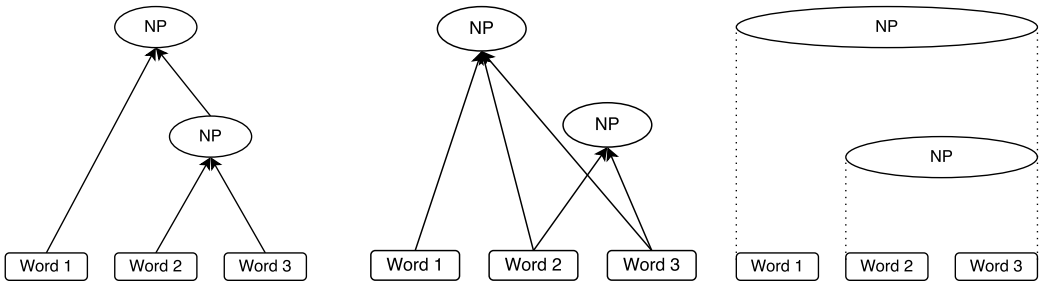


Figure 6.1: Alternatives for representing constituent trees in the general dependency graph format.

1. In the first setup, shown by Figure 6.1a, we connect each of the children to their direct parents. In this way, our graphs will be very similar to a constituent tree. In this example, *Word 2* and *Word 3* are connected to an *NP*, and that *NP* and *Word 1* are connected to another NP.

2. Another possibility is when each of the nodes is connected to all ancestor non-terminals (Figure 6.1b). In that case, there is a direct relation between a constituent and their descendants. In the current example, the higher level *NP* directly contains the children (*Word 2*, *Word 3*) of its child. This representation has the hope that the feature extractor of downstream applications can directly generate features about the ancestors without recursive rules

3. A different approach is where we give the covering area for each new non-terminal (Figure 6.1c). In these cases, like in the previous we have not got direct information about the connection between the nonterminals. On the other hand, it can help for an application that uses the position of a node.

For constituent parsing, we used the Berkeley Parser (Petrov et al., 2006b) with default parameters and pretrained model (eng-sm6). In our submission, we used the second and third methods in the dependency graph format.

6.4 Label Set Adjustment Driven by Downstream Applications

Different downstream applications might utilize different types of grammatical patterns. The simplest case is that a downstream application might extract important features from particular edge labels while features over other edge labels are negligible in its machine-learned model. Moreover, different applications might utilize different types of dependencies, see for example event recognition versus negation scope detection.

We propose a simple procedure to recognize edge labels that can be collapsed into other edge labels because their discrimination does not give any added value to the downstream application in question. We start from the full set of edge labels and systematically check the effect of collapsing two particular labels evaluated through the downstream application.

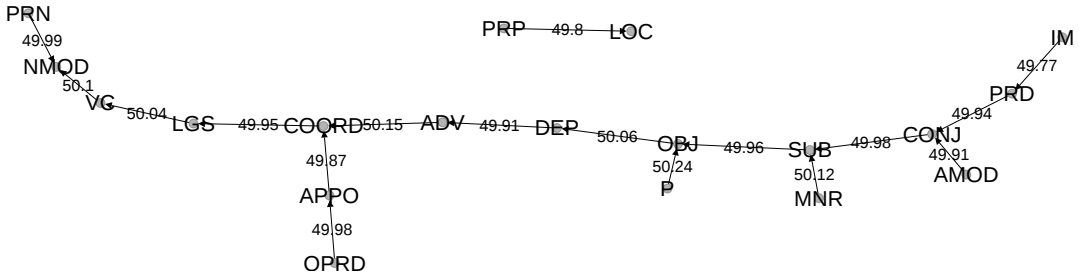


Figure 6.2: Label adjustment graph.

	Event Extraction	Negation Resolution	Opinion Analysis
Bohnet - baseline	47.84	61.98	65.87
Bohnet + label adjustment	47.37	60.53	66.33
Bohnet + constituent	46.71	61.26	63.13
MST - baseline	46.69	59.78	63.25
Bohnet + MST - k-best	45.96	59.05	62.5

Table 6.1: Final result in evalutaion set.

We calculated for all label pairs what will happen if we replace one dependency label with another. For our experiments, we used the TEES system, but because we did not have enough time to retrain the TEES system for all combinations, we trained it once with the full label set and we did the prediction part separately to each dependency label pair. In this prediction part we replaced each of the labels with each of another labels on the full development set. We got a complete directed graph where the nodes are the labels and edges contain the scores from the TEES system with the merged labels. For each node, we kept the outcoming edge with maximum weight i.e. when the replacement was the most efficient. When there were two edges between two nodes we removed the smaller one.

Figure 6.2 contains the graph we got. (When we ran the TEES system with default parameters we got 49.76 with original labels). By using this graph we started replacing the nodes from the highest edge weight to the lowest. We evaluated the new label set in every step and we found the best result after three steps, 50.36, which is slightly better than the best merged pair. After that, we did the three replace steps in the full dataset.

6.5 Results

Table 6.1 shows our official results achieved on the shared task. The `Bohnet - baseline` is one of our baselines where we just run the Bohnet parser (Bohnet, 2010c) with pretrained model. The second and third rows contain the result of `label adjustment` and `constituent parsing` experiments. The fourth row contains another baseline when we applied the `MSTParser` and the last row shows the scores of our `k-best experiment` (we used MSTParser here).

6.5.1 Event Extraction

In the event extraction task, we can not beat our baselines, all of our modifications – including the label adjustment which is optimized for this task – get a negative effect. The dependency label merging mechanism that we directly developed on this task also failed.

6.5.2 Negation Resolution

One of the main motivation of the constituent-based approach was the negation resolution task. The scope of the negations is usually a close what can we identify with constituent parsers. This `constituency-based system` got better results in three out of four scope-focused evaluation metrics than our baseline. Table 6.2 shows the detailed comparison of the baseline and the constituent system.

	baseline		Bohnet + const	
	dev	test	dev	test
Scope Match	78.42	80.00	77.98	81.14
Scope Tokens	86.64	89.17	87.38	89.27
Event Match	75.47	67.90	72.90	65.20
Full Negation	62.15	61.98	59.91	61.26

Table 6.2: Detailed results of the *baseline - Bohnet* and the *Bohnet + constituent* systems in negation resolution task.

The following example shows how the constituent parse helps:

*"I join in it because there is **no** other way in the world by which justice can be gained."*

The scope of the no negation clue starts from the *there* and ends with the *gained* word. Our baseline system marked the negation from the *there* to the *justice*, but the constituent-based method found the correct scope. If we look at the constituent tree we see the full scope is covered by a constituent with *S* label. Instead of scope detection, the constituent-based information can't help in the event detection subtask.

6.5.3 Opinion Analysis

In the opinion analysis task, the `label adjustment` method improved by 0.5 percentage point against the `Bohnet-baseline` and got the best results in the shared task in Holders (In Vitro) metric. It seems the label collapsions that our method found in the event extraction task are more general than we expected. On the other hand, it is still an open question why this label collapsing did not work at the event extraction task's evaluation set.

6.6 Summary

In this chapter, I proposed three techniques for the general representation of syntax beyond the canonical dependency parse tree approach (Szántó and Farkas, 2017). While the application of information from the

top-k dependency tree decreased the accuracy of the system, the constituent-based virtual nodes improved the results in the negation resolution task and *label adjustment* helped in the opinion analysis task.

This system (Szántó and Farkas, 2017) participated in the EPE 2017 shared task (Oepen et al., 2017), where it reached first place in the opinion mining task and second place overall among eight teams, behind the cooperation of Stanford and three Universities from Paris, Diderot, INRIA, and Sorbonne (Schuster et al., 2017).

The results of this chapter were published in the *Proceedings of the 2017 Shared Task on Extrinsic Parser Evaluation at the Fourth International Conference on Dependency Linguistics and the 15th International Conference on Parsing Technologies* with the title *Szeged at EPE 2017: First experiments in a generalized syntactic parsing framework* Szántó and Farkas (2017).

Chapter 7

Enhancing Medication Event Classification with Syntax Parsing

In this chapter, I introduce strategies for medication event extraction and classification, revealing how syntax-based information extraction unlocks efficiency gains in the presence of pretrained large language models too (Szántó et al., 2023).

Understanding the complete medication history is necessary for having a fuller picture of the patient, but in many cases, medication-related information is documented only as unstructured clinical notes. This can make it challenging for healthcare providers to obtain a comprehensive view of a patient's medication history including information on medication changes, dosages, and adverse reactions. The automatic analysis of these notes could help medical providers have a fuller background on the patient, better understand the reasons behind medication changes, and identify the healthcare provider who ordered a medication change, as well as the reason for the change. This would allow for more informed medical decisions and improve patient safety.

The Contextualized Medication Event Dataset (CMED) (Mahajan et al., 2021) and the *National NLP Clinical Challenges 2022 Track 1* aimed at the extraction of these medication events from clinical notes. As well as identifying names of medications, this dataset allows for the detailed analysis of the context of medication-related events. It aims to extract more detailed information from the text about the mentioned medications: like whether the use of the medication was started or stopped, or identifying the person requesting the change. This is a context classification problem where the goal is to find the information that relates to the specific expression, eg. being able to correctly identify if one medicine was started, but another was stopped for a patient within the same note.

Nowadays the most generic approach for this type of problem is using a pre-trained language model and fine-tuning it for our tasks. We applied two main additions to this standardized framework. (Szántó et al., 2023) One of our modifications is aiming to handle the noisy, error-ridden nature of the clinical notes, for this

Task	Label	#train	#test
Event	NoDisposition	5260	1326
	Disposition	1412	335
	Undetermined	557	122
Action	Start	568	131
	Stop	340	67
	Increase	129	22
	Decrease	54	13
	UniqueDose	285	88
	OtherChange	1	0
	Unknown	35	14
Negation	Negated	32	6
	NotNegated	1380	329

Task	Label	#train	#test
Temporality	Past	744	173
	Present	494	132
	Future	145	29
	Unknown	29	1
Certainty	Certain	1176	281
	Hypothetical	134	33
	Conditional	100	15
	Unknown	2	6
Actor	Physician	1278	311
	Patient	106	17
	Unknown	28	7

Table 7.1: Frequency of event and context classes in the training and test data.

problem we applied adversarial attacks throughout the training process. The other was used for the context classification task and motivated by the state-of-the-art algorithms of aspect-based sentiment analysis as they both aim to identify the context relevant to the selected phrase. We also used syntactic relations to help find the more closely related parts of the sentences.

7.1 Dataset

The shared task aimed to find new ways to extract information from raw medical notes using NLP techniques. The full Contextualized Medication Event Dataset is comprised of 500 clinical notes that contain a total of 9012 medication mentions. The annotation of these documents can be divided into three levels, each reliant on the last. These three tasks are the following:

- Medication Extraction: The first task is to identify all the medications mentioned in clinical notes; this is a standard sequence labeling task.

- Event Classification: Once we have extracted these mentions, we classify each of them into one of three categories: `Disposition` (meaning a change in the medication was discussed), `NoDisposition` (meaning no change was discussed), or `Undetermined` (meaning we need more information to make a determination).

- Context Classification: For medication mentions that fall into the Disposition category, we go a step further and classify them according to five different dimensions: `Action` (did the medication start or stop?), `Negation` (was it negated in any way?), `Temporality` (is it a past or a present change?), `Certainty` (was it hypothetical or conditional?), and `Actor` (who initiated the medication change?). This gives us a more complete understanding of the context in which the medication was mentioned.

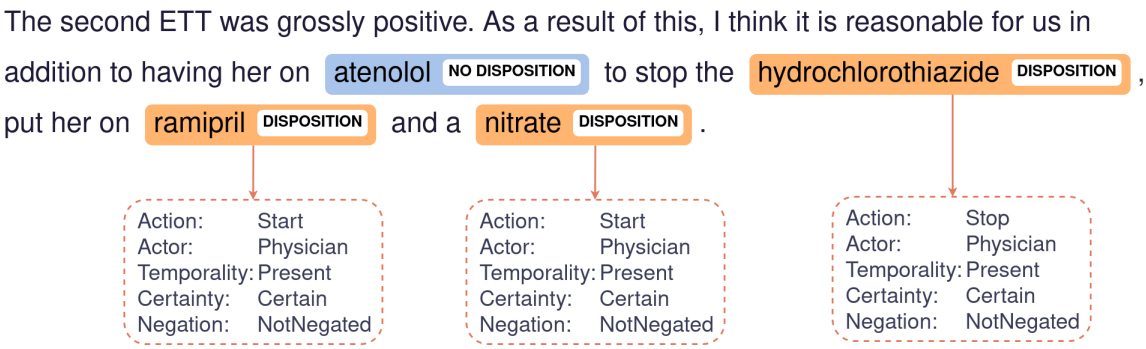


Figure 7.1: The annotation of two sentences in the Contextualized Medication Event Dataset.

As we mentioned earlier there are 9012 examples for event classification (the medication mentions) and only 1747 of them are in the `Disposition` category (ones with change in the medication). The detailed sizes of the different classes are shown in Table 7.1. There are three dimensions - the Actor, the Certainty, and Negation - where one class dominates the dataset: for these their most frequent label makes up more than 80 percent of the data. The distribution of different Actions is more balanced: the `Start`, the `Stop`, and the `UniqueDose` occur in more than 20 percent of the medication event changes.

Figure 7.1. shows an annotation for two example sentences. The colored boxes show the four different medications mentioned in the text. The color of the mention indicates the classification of the event, there are three with a change in disposition (marked orange) and one with no disposition change (marked blue). All three with a changed disposition are further annotated by the five context categories. We can see that the physician ordered the stop of the *hydrochlorothiazide* and the start of the *ramipril* and *nitrate*. All of these changes are in the present and are not negated, hypothetical, or conditional.

7.2 Method

Our system consists of two separable components. First, we solve the medication extraction and the event classification tasks in one step. Therefore in this step, we start from the raw text and find the mentions of medications and assign event classes (`Disposition`, `NoDisposition` and `Undetermined`) to each of them.

In the second step, we work only with the medications tagged as `Disposition`: we use the related parts of the sentence to assign values to all of the predefined categories of Action, Negation, Temporality, Certainty, and Actor.

The full architecture is shown in Figure 7.2.

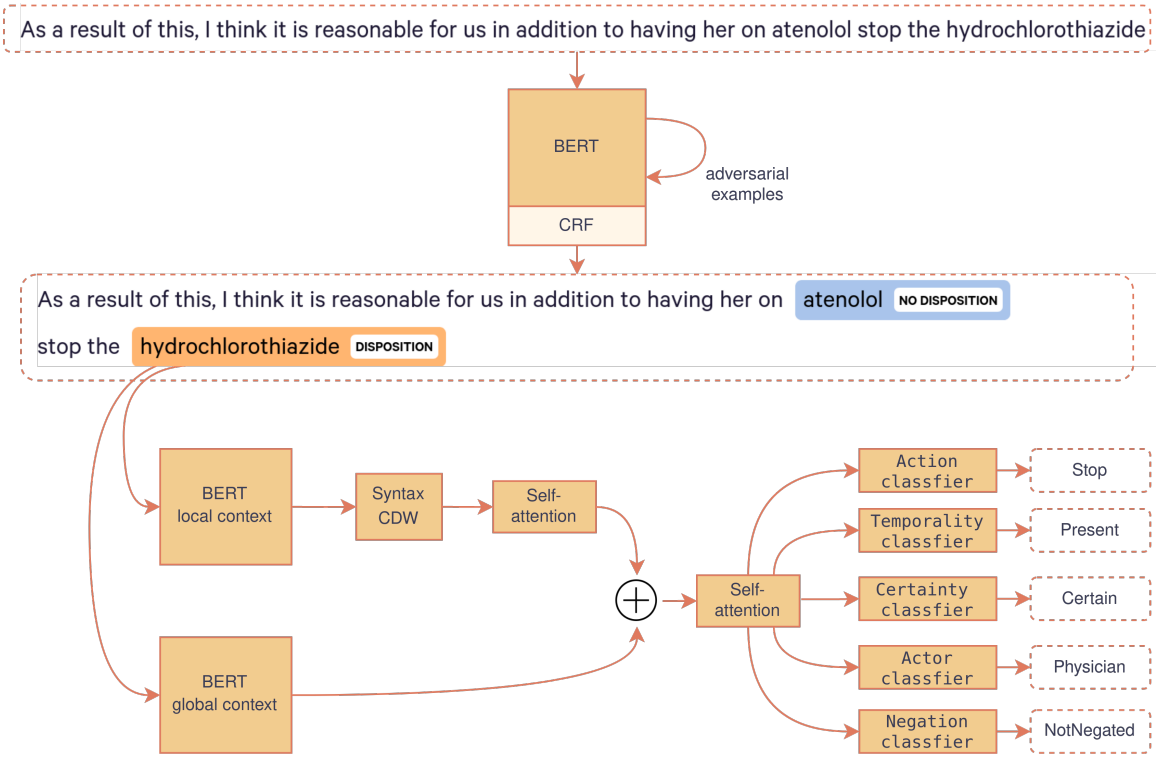


Figure 7.2: The architecture of our system with an example document. The dashed line text boxes contain the example sentences and our outputs.

7.2.1 Medication Event Recognition

First, we devised a solution for the medication extraction and the event classification tasks in the same step. Our basic architecture was similar to a named entity recognition setup. We handled the problem as a token classification task, annotated the texts with IOB encoding, and used the three event labels as entity types.

To solve this token classification problem, we utilized a large language model and extended it with a CRF layer on top of that. This idea involves merging the transfer learning capabilities of pre-trained language models such as BERT with the structured predictions made by CRF. This method was successfully applied to different token classification tasks, like Portuguese named entity recognition (Souza et al., 2019) or text anonymization in medical documents (Mao and Liu, 2019).

Adversarial Learning

Next, we aimed to handle the error-prone nature of the clinical notes. Previous studies showed that adversarial attacks can increase the generalization abilities of natural language processing systems, especially in the case of medical documents (Moradi and Samwald, 2022).

To apply adversarial attacks we used the iterative version of the fast gradient sign method (Goodfellow

et al., 2014; Kurakin et al., 2016) (IFGSM) with text-specific modifications (Gong et al., 2018). These modifications were needed as the traditional attacking methods are more often applied to images where the pixels are continuous values, as opposed to words in a document.

The gradient attack could make changes in the embedding layer of the large language model, but generating documents from the attacked embeddings is not a straightforward task. After a gradient attack, we reconstruct the attacked document by searching for the closest word piece to each embedding. We only accept a new adversarial example where the text is changed compared to the original. The size of the changes is controlled by the ϵ parameter of the IFGSM attack. Larger ϵ makes more changes. We started with a small ϵ value (1) and increased it with a fixed step size (1) to a maximum of 10 iterations until we found a change in the text. Every second epoch we generate 200 adversarial examples that we add to the training set.

7.2.2 Medication Context Classification

For the last subtask, our methods were motivated by the field of aspect-based sentiment analysis. Aspect-based sentiment analysis (Do et al., 2019) is a well-researched area that has similarities to these types of problems. In sentiment analysis, the task is to determine which sentiments are associated with which target. For example, in the sentence "I love the last jedi, but not a fan of the rise of skywalker" there is positive sentiment about The Last Jedi Star Wars movie, but for The Rise of Skywalker the writer shared a negative opinion.

Local Context Focus Mechanism

For aspect-based sentiment analysis the local context focus (Zeng et al., 2019) (LCF) mechanism was efficiently used. This method pays more attention to the words that are more closely related to our target.

For the context classification task, we applied this LCF mechanism to prioritize the local context of the given target expressions: in our case the given medication. The architecture of the LCF is shown at the bottom of Figure 7.2. First, the text is encoded by two language models, one handles the global and one the local context. The local language model has two extra layers, a context features dynamic weighted (CDW) layer and a multi-headed self-attention, these highlight the tokens that are close to the target medication. The outputs of the local and global model are concatenated and there is another multi-headed self-attention over that. The CDW layer weights the tokens by calculating how many tokens were between the target expression and the given token, the tokens closer to the target get higher weights. This method helps to highlight the context of the medication, but a token that is far from the target is not necessarily irrelevant.

Syntax-based Weighting

To describe more precisely the context of the medication, we applied syntax-based dynamic weighting(Phan and Ogunbona, 2020).

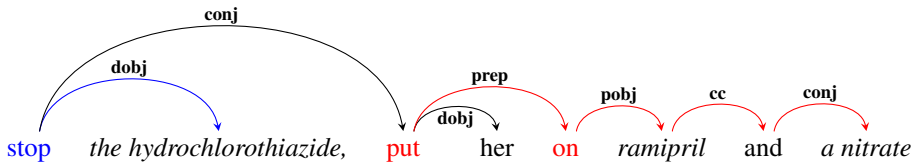


Figure 7.3: Dependency structure for a part of a sentence. Blue and red edges indicate the route between the medication mentions and the verb they are the syntactic dependent of.

Figure 7.3 shows the dependency tree of a part of a sentence that contains three medication mentions. The medication names are marked in italics, the verbs showing the type of disposition change are colored blue for the `stop` and red for the `start` action; these verbs' dependency relations to the medications are marked the same colors also.

The figure shows that while the name of the medication *hydrochlorothiazide* is equally close to the `start` action verb *put* and the `stop` action verb, in the dependency tree it is directly reachable with one step from *stop*, but not from *put*.

Despite this method's drawback that the dependency trees can only possibly consider one sentence at a time and cannot handle a following sentence referring back to a previously mentioned medication, we found that using the syntactic information improves our results for this task.

So far we have described a method for a single context classification task, but in this dataset, we had five separate classification problems for each of the dispositions.

Like in the two previous tasks, we also applied adversarial learning in the same way. For implementation, we used the PyABSA (Yang and Li, 2022) framework that was initially developed for aspect-based sentiment analysis. We used SpaCy's (Honnibal et al., 2020) English transformer model for dependency tree parsing.

7.3 Results

7.3.1 Experimental Setup

We used the Contextualized Medication Event Dataset for each of our experiments. The organizers of the National NLP Clinical Challenges 2022 shared task separated 50 clinical notes from the original training set to create a development set. We used that development set to evaluate our intermediate systems. Therefore the training set contains 350, the development set 50 and the test set 100 clinical notes. The training and development set was provided with labels and the test set was released without them.

In the next subsections, we evaluate our system's performance on the development set, then show the results of our final system on the test set as evaluated by the organizers of the shared task.

For evaluation, we applied the official script of the shared task. This script provides strict and lenient scores based on the matching of the spans in the medication extraction task. To achieve strict matching, the span's offsets must be an exact match. While for lenient matching, it is enough for the spans to have some

	Precision	Recall	F1
CRF-BERT	0.9572	0.9752	0.9661
CRF-BERT + adv	0.9573	0.9772	0.9671

Table 7.2: Lenient F1-scores on the medication extraction task on the development set.

	Precision	Recall	F1
CRF-BERT	0.8671	0.8671	0.8670
CRF-BERT + adv	0.8985	0.8631	0.8791

Table 7.3: Lenient macro F1-scores on the event classification tasks on the development set.

overlap. Like in the official results of the shared task, we always publish our results with lenient matching.

Because of the nondeterministic nature of the training process, we trained all the models three times and selected the best-achieving system from these independent runs. The results of these systems are shown in Table 7.2 and Table 7.4, and these are the system that we have submitted for the shared task.

7.3.2 Event Classification

First, we evaluated the medication extraction and the event classification tasks. We are starting from raw text and annotating the medication mentions and classifying them by the type of events.

We used the BERT large model and applied the following parameters during the training: batch size: 6, learning rate: 1e-05, and trained the model for 20 epochs.

In the medication extraction task, we have already achieved good results with the `CRF-BERT` baseline. Table 7.2. shows that the adversarial examples didn't give us further improvement. This is because the detection of the medication mentions is more dependent on specific word forms.

However, in the event classification task, where the labels depend more on the context, the application of adversarial examples increased the results. As we can see in Table 7.3. there is a 1.2 percentage point increment in the case of the macro F1 score in the development set.

7.3.3 Context Classification

For context classification experiments we used the provided gold annotation for the medication extraction and event classification tasks. We applied the same parameters as we used in the previous step. Like in the shared task, we provide lenient combined F1 score. The combined F1 score only accepts a prediction when the class is correct in each of the five dimensions.

Table 7.4. describes the effect of the local context focus mechanism with syntax-based context weighting. These `context-specific features` increased the performance of the `baseline system` by 1.4 percentage points.

The detailed results of the syntax-enhanced local focus mechanism system are in Table 7.5. It shows lenient macro scores over all of the five classification tasks.

	Precision	Recall	F1
BERT	0.5920	0.5385	0.5640
BERT + S-LCF	0.6070	0.5520	0.5782

Table 7.4: Combined performance of the systems on the event context classification task on the development set.

	Precision	Recall	F1
Action	0.8109	0.7376	0.7725
Temporality	0.8358	0.7602	0.7962
Certainty	0.9502	0.8643	0.9052
Actor	0.8955	0.8145	0.8531
Negation	0.9801	0.8914	0.9336

Table 7.5: Task level lenient macro F1 scores of the BERT + S-LCF system on the development set.

We can see that the action and temporality detection proved to be the most difficult tasks. These tasks have the most classes and the most balanced label distribution. An interesting type of error in the detection of the action type can be seen in the following example, where for finding the correct solution the model would require mathematical knowledge: *Continue T 40 mg b.i.d. - as 20 mg b.i.d. did not give full control with the generic pills.* In this sentence to decide whether the dose is increased or decreased the model should know if *20* or *40* is larger.

7.3.4 Shared Task Results

The shared task was evaluated in three steps on the test set that was originally released unlabelled. In the first release, the raw text had to be analyzed, thus the event and the context classification tasks contained the error of the lower-level tasks as well. In the second release, the organizers provided the gold medication mentions, in the third release the gold event annotation was also provided and the only task was the context classification. Since our system performs the first and second tasks in one step, we focused on the first and third releases.

The best results were reached by the Toyota Technological Institute Nagoya's system. They won all the scenarios where we participated. For the named entity recognition task they applied a RoBERTa-based system that was pretrined in three other clinical named entity datasets. For the other two tasks, they applied an ensemble model and classified medication attributes with a multi-turn question-answering system.

The 7.6 table shows our official results on the test set. Among the 32 participant teams, our system achieved 8th place on the context classification task both in the case of `Release 1` and `Release 3`, also 8th place on the event classification in release 1, and 9th place on the medication mention task.

	Release 1 (raw text)			Release 3 (gold event clf.)
	Medication F1	Event F1	Context F1	Context F1
Max	0.9846	0.8348	0.6647	0.7297
Min	0.0945	0.2666	0.0219	0.0209
Median	0.9586	0.7438	0.4782	0.5752
Mean	0.9176	0.6928	0.4537	0.5249
Szeged	0.9714	0.7682	0.478	0.5982

Table 7.6: Performance of our final system compared to the max/min/median/mean results of the shared task on the test set of Contextualized Medication Event Dataset. `Release 1` only contained raw texts that we ran all of our systems on. The gold event classification in `Release 3` could be used for evaluating the context classification task.

7.4 Related Work

Although the use of a local context focus mechanism and its syntax-based extension was developed in recent years, highlighting part of the context and the application of syntax parsing in context classification tasks is not a new concept. In aspect-based sentiment analysis, both dependency and constituent analysis have been applied for feature extraction (Hangya et al., 2014; Kong et al., 2014) before the spread of large language models. Several different methods have also been developed for the weighting of the features (Hangya and Farkas, 2013).

In the case of large language models, the first solution that comes to mind is the application of sentence pair classification (Song et al., 2019) that doesn't make any change to the structure of the neural network, only modifies the input document. It concatenates the full context and the target word with a separator. This allows the model to identify the target in the context. This simple method improved the results over the previous feature- and word-embedding-based solutions.

In each of our experiments, we applied the large version of BERT, but the application of other large language models such as RoBERTa (Liu et al., 2019) or DeBERTa (He et al., 2021) is a low-hanging fruit for future improvement. As well as utilizing more domain-specific knowledge from biological and clinical text-based pre-trained language models, like the BioBERT(Lee et al., 2020) or the Clinical BERT (Alsentzer et al., 2019).

The distinction between the local and global context was also applied to clinical text analysis. In previous works, the combination of CNN and RNN (Raj et al., 2017) was used for relation classification where the CNN was motivated by their local context extraction capabilities, while RNNs are more suited for long-term dependencies. Earlier work in the Second i2b2 Shared-Task (Uzuner, 2008) about obesity classification used dictionary-based solutions to analyze the context of diseases for negation and uncertainty (Farkas et al., 2009).

7.5 Summary

This chapter proposes solutions for the extraction and analysis of the medication mentions in clinical notes. For our experiments, we used the Contextualized Medication Event Dataset which contains three tasks. We employed a CRF-BERT-based solution enriched with adversarial examples for the medication extraction and event classification tasks.

In the third task of assigning more detailed context to the medication changes, we implemented a pre-trained language model-based solution that I extended with syntax-based highlighting of the relevant part of the documents. The main motivations of this approach come from the field of sentiment analysis. The usage of the local context focus mechanism and syntax parsing-based weighting successfully improved the results of the context classification as well.

This work was published at the *IFIP International Conference on Artificial Intelligence Applications and Innovations* with the title *Enhancing medication event classification with syntax parsing and adversarial learning* (Szántó et al., 2023).

Contribution

In the Szántó et al. (2023) my contribution is:

- The idea of the full system except for the adversarial attack-based solution.
- The idea and the implementation of the syntax-based local context mechanism for context classification.

Chapter 8

Summary

8.1 Summary in English

This thesis focused on syntax parsing especially for Hungarian and other morphologically rich languages. It also showed applications of syntax parsing in different higher level tasks like sentiment analysis or medication event extraction.

8.1.1 Constituent Parsing

Chapter 3 presented different techniques to improve constituent parsing, especially for handling the challenges of morphologically rich languages. Section 3.3 introduced my proposals that utilizes the information from the constituent trees, while section 3.4 demonstrated my approaches that use external information like large amounts of unlabeled data and dependency trees.

One of the chief contributions of Section 3.3 is to propose a novel automatic procedure to find the optimal set of preterminals by merging morphological feature values. The size of the preterminal set in the standard context free grammar environment is crucial. If we use only the main POS tags as preterminals, we lose a lot of information encoded in the morphological description of the tokens. On the other hand, using the full morphological description as preterminals yields a set of over a thousand, which results in data sparsity as well as performance problems. The main novelties of our approach over previous work are that it is very fast – it operates inside a probabilistic context free grammar (PCFG) instead of using a parser as a black box with re-training for every evaluation of a feature combination – and it can investigate particular morphological feature values instead of removing a feature with all of its values. I also experimented with exploiting external corpora in the lexical model. A new scientific result is that automatic tagging of an off-the-shelf supervised morphological tagger can also contribute to the performance. My last experiment was carried out with the feature set of an n-best reranker. We showed that incorporating feature templates built on morphological information improves the results.

Section 3.4 focused on approaches that use external information like large amounts of unlabeled data and dependency trees to improve the accuracy of constituent parsers. In the discriminative reranking step I introduced a new feature template that employs dependency-based information. By using the unlabeled data, I applied Brown clustering which I also applied as features in the final system.

8.1.2 Dependency Parsing

Chapter 4 delved into the current landscape of Hungarian dependency parsing. It introduced the available datasets, explored the relationship between dependency parsing and constituent parsing in Hungarian (Simkó et al., 2014), and examined the development of the Universal Dependency dataset for Hungarian (Vincze et al., 2017). Additionally, it highlighted HuSpaCy, which provides a cutting-edge dependency parser for Hungarian (Orosz et al., 2023).

In Section 4.2, I introduced a Hungarian constituency to a dependency converter. Based on that system I demonstrated that although the results obtained by training on the constituency treebank and converting the output to dependency format and those obtained by training on the automatically converted dependency treebank are similar in terms of accuracy scores, the typical errors made by these two systems differ from each other.

In Section 4.3, I presented how the principles of Universal Dependencies and Morphology have been adapted to Hungarian. Experiments were introduced on the new, manually annotated corpus for evaluating automatic conversion and the added value of language-specific, i.e. non-universal, annotations.

In Section 4.4, I introduced the HuSpaCy a new industrial-grade text processing pipeline for Hungarian and presented a thorough evaluation showing their (close to) state-of-the-art performance.

8.1.3 Application of Syntax Parsing

Latent Syntactic Structure-Based Sentiment Analysis

In Chapter 5, I focused on the exploitation strategies of syntactic structures for in-sentence and sentence-level SA (Hangya et al., 2017). Usually, sentence-level polarity labels can be easily obtained in a huge amount for various domains, take for instance pro/con or bottomline summaries of the product review sites. Hence this chapter proposed a machine learning framework for sentence-level and in-sentence polarity classifiers by using exclusively sentence-level polarity annotation for training. This approach can predict the sentiment labels assigned to the constituents of a phrase structure parse tree without an annotated sentiment treebank by handling the polarity labels of internal nodes in parse trees as latent variables.

I introduced two experimental setups for the investigation of the proposed approach. The objective of the experiments' first batch (in Section 5.3) is to investigate whether the sentence-internal latent structure helps the prediction of sentence-level polarity. The second batch of experiments (in Section 5.4) shows that the sentence-internal latent structures themselves are also meaningful when we extract features from them for a target-oriented sentiment analysis task.

The chief added value of this chapter is to propose a latent syntactic structure-based approach that requires only sentence-level polarity labels for training. The experiments on three domains (movies, IT products, restaurants) support that sentiment analyzers are domain-dependent.

Application of Generalized Syntactic Parsing Framework

In Chapter 6, I introduced the work of Team Szeged for the *First Shared Task on Extrinsic Parser Evaluation* (EPE 2017) (Szántó and Farkas, 2017). I presented three approaches to exploit the opportunities of the general dependency graph representation of the shared task.

The goal of the EPE 2017 was to estimate "*the relative utility of different types of dependency representations for a variety of downstream applications that depend heavily on the analysis of grammatical structure*". (Oepen et al., 2017).

To enable different types of dependency representations, the organizers of the shared task introduced a very general graph-based representation of '*relational*' structure reflecting syntactic-semantic analysis. The nodes of this graph correspond to lexical units, and its edges represent labeled directed relations between two nodes. Nodes can be defined in any terms of (in principle arbitrary) sub-strings of the surface form of the input sentence. This representation allows overlapping and empty (i.e. zero-span) node sub-strings as well. Moreover, nodes and edges are labeled by attribute–value maps without any restriction on the attribute set.

This very general graph-based representation opens brand new ways for expressing syntactic or semantic information besides the standard dependency tree formalism. We understood the call of the shared task in a generalized way and came up with ideas that aim to leverage the opportunities of the general representation beyond dependency parse trees. We experimented with a couple of such ideas (instead of trying to achieve high scores in the shared task).

In the first set of experiments (in Section 6.2), we started from the classic dependency parsing approach but instead of a single dependency parse, we expressed the distribution of possible dependency parses given a sentence in the graph-based general representation. In Section 6.3, I introduced a possible solution for enriching the dependency parse by constituent information given by a standard phrase-structure parser. In this way, various syntactic representations can be represented in the graph and information is not lost because the downstream application can only accept a single dependency parse tree. Furthermore, in the EPE 2017 setting, we can send a blended relational structure to the downstream task, like a parse distribution and blended version of different syntactic approaches, and the downstream application is able to machine learn which type of syntactic structure or phenome or even which combination of syntactic information is useful for itself.

Our last batch of experiments (in Section 6.4) was a consequence of this objective, i.e. the relational representation has to be useful for the downstream application. Here, we tried to automatically recognize which dependency parse labels are useful to a downstream task and collapsed the useless ones.

Enhancing Medication Event Classification with Syntax Parsing

In Chapter 7, I introduced strategies for medication event extraction and classification, revealing how syntax-based information extraction unlocks efficiency gains even in the presence of pretrained large language models (Szántó et al., 2023).

For our experiments, we used the Contextualized Medication Event Dataset which contains three tasks. We employed a CRF-BERT-based solution enriched with adversarial examples for the medication extraction and event classification tasks.

In the task of assigning more detailed context to the medication changes, we implemented a pre-trained language model-based solution that I extended with syntax-based highlighting of the relevant part of the documents. The main motivations for this approach come from the field of sentiment analysis. The usage of the local context focus mechanism and syntax parsing-based weighting successfully improved the results of the context classification as well.

8.2 Magyar Nyelvű Összefoglaló

A dolgozat elsősorban a magyar és más morfológiailag gazdag nyelvek szintaktikai elemzésére koncentrál, emelett bemutatja a szintaktikai elemzés lehetséges alkalmazásait olyan magasabb szintű feladatokban, mint például a szentiment elemzés vagy az orvosi dokumentumokban található gyógyszerszedési események kinyerése.

8.2.1 Konstituens Elemzés

A 3. fejezet a morfológiailag gazdag nyelvek konstituens elemzésének a kihívásaira mutatott be megoldásokat. A konstituens fákban található információk jobb kihasználására ismertetett módszereket a 3.3. rész, míg a 3.4. rész olyan külső forrásból származó információk, mint a nagymennyiségű címkézetlen szövegekes korpuszok felhasználására adott megoldási javaslatokat.

A 3.3. rész egyik fő hozzájárulása, hogy újszerű, automatikus eljárást javasolt a preterminálisok optimális halmazának megtalálására a morfológiai jellemzők értékeinek az összevonásával. A preterminálisok halmazának mérete nagyon fontos a hagyományos környezetfüggetlen nyelvtan alapú megközelítések esetén. Ha csak a főszófajt használjuk preterminálisokként, akkor sok, a szavak morfológiai leírásában kódolt információt veszítünk. Ezzel szemben, ha a teljes morfológiai leírást használjuk a preterminálisok szintjén, akkor több mint ezer különböző elemünk lesz, aminek a következtében kevés példánk lesz az egyes preterminálisokra és teljesítményproblémákba is ütközünk. Az általam javasolt megközelítés legfőbb újításai a korábbi munkákhoz képest, hogy nagyon gyors – egy valószínűségi kontextusfüggetlen nyelvtanon (PCFG) belül működik, ahelyett, hogy egy feketedobozként használt elemzőt kellene minden jellemző kombináció vizsgálatához újratanítani – ezen felül képes páronként vizsgálni bizonyos morfológiai jellemzők értékeit ahelyett, hogy egy jellemzőt az összes értékével együtt dobna el vagy tartana meg.

Ezen felül a kísérleteimben a lexikai modell javítására külső korpuszok felhasználását is megvizsgáltam. Eredményeim alapján a rendszer teljesítménye tovább javítható szófaji elemzési statisztikák felhasználásával. Az utolsó kísérletem bemutatta, hogy a konstituens elemzésben gyakran használt újrarangsorolási lépés hatékonysága javítható morfológiai információkra építő jellemzők készítésével.

A 3.4. rész olyan megközelítésekre összpontosított, amelyek külső információkat, például nagy mennyiségű címkézetlen adatot és függőségi fákat használnak a konstituens elemző pontosságának javítására. Az újrarangolási lépésben új jellemző készletet vezettem be, ami függőségi elemzés alapú információkat használ. A címkézetlen adatok felett Brown klaszterezést készítettem, amit szintén jellemzőként építettem be az újrarangsoroló rendszerbe.

8.2.2 Függőségi Elemzés

A 4. fejezet a magyar nyelvű függőségi elemzés jelenlegi helyzetével foglalkozott. Bemutatta a rendelkezésre álló korpuszokat, megvizsgálta a magyar (Simkó et al., 2014) függőségi és konstituens elemzés közötti kapcsolatot, és bemutatta a Universal Dependencies projekt magyar alkorpuszának az elkészültét (Vincze et al., 2017). Emellett ismertette a (közel) state-of-the-art függőségi elemzővel rendelkező, ipari igényekre optimalizált HuSpaCy magyar nyelvű szövegfeldolgozó keretrendszert (Orosz et al., 2023).

A 4.2. részben ismertettem egy rendszert, ami magyar nyelvű konstituens fákat képes függőségi fákká átalakítani. Ezen rendszer alapján demonstráltam, hogy bár a konstituens elemzővel betanított modellből függőségi formátumba konvertált eredmények és az automatikusan konvertált függőségi elemzővel betanított eredmények pontosság szempontjából hasonlóak, a két rendszer által elkövetett tipikus hibák eltérnek egymástól.

A 4.3. részben bemutattam, hogy a Universal Dependencies and Morphology alapelveit hogyan ültettük át a magyar nyelvre. Az új, manuálisan létrehozott korpuszon kísérleteket végeztünk az automatikus átalakítás értékelésére, valamint a nyelvspecifikus, azaz nem univerzális annotációk hozzáadott értékének vizsgálatára.

A 4.4. részben ismertettem a HuSpaCy-t, egy új, ipari célokra kialakított magyar nyelvű szövegfeldolgozó keretrendszert és kísérleteken keresztül annak (közel) state-of-the-art teljesítményét.

8.2.3 Szintaxis Elemzés Alkalmazásai

Rejtett Szintaktikai Struktúra Alapú Szentiment Elemzés

Az 5. fejezetben szintaktikai struktúrák kiaknázására összpontosítottam a mondatokon belüli és a mondatszintű szentiment elemzésben (Hangya et al., 2017). A mondatszintű szentiment címkék általában könnyen és nagy mennyiségben beszerezhetőek, például a termékismertető oldalakon található értékelések letöltésével. Ezért ebben a fejezetben egy olyan gépi tanulási keretrendszert javasoltam, ami kizárólag mondatszintű szentiment annotációt használ a tanításhoz, de mondaton belüli elemekhez is rendel szentiment címkéket. Ez a megközelítés képes előrejelezni a konstituens fa csomópontjaihoz rendelt szentiment címkéket annotált szentimenteket tartalmazó fa nélkül, a belső csomópontok szentiment címkéit látens változókként kezelve.

Két kísérleti környezetet alakítottam ki a javasolt megközelítés vizsgálatára. Az első kísérletsorozat (5.3. rész) célja annak vizsgálata, hogy a mondaton belüli látens struktúra segíti-e a mondatszintű szentiment előrejelzését. A 5.4. részben szereplő második kísérletsorozat azt mutatja be, hogy maguk a mondaton belüli látens struktúrák is jelentéssel bírnak, amikor jellemzőket készítünk belőlük egy célorientált szentiment elemzés feladathoz.

EPE: Általános Szintaktikai Keretrendszer Alkalmazása

A 6. fejezetben bemutattam a *First Shared Task on Extrinsic Parser Evaluation* (EPE 2017) versenyen a Team Szeged munkáját (Szántó and Farkas, 2017). Ismertettem három megközelítést a verseny céljául szolgáló általános gráfalapú függőségi reprezentáció kihasználására.

Ahhoz, hogy egyszerre alkalmazni lehessen különböző függőségi reprezentációkat a verseny szervezői egy nagyon általános, gráfalapú reprezentációt vezettek be. A gráf csomópontjai lexikai egységeknek felelnek meg, az élei pedig címkézett irányított kapcsolatokat jelentenek két csomópont között. A csomópontokat a bemeneti mondat egy tetszőleges karakterszámú szakaszával lehet meghatározni. Ez a reprezentáció lehetővé teszi az átfedő és az üres (azaz nulla karakter hosszú) csomópontok létrehozását is. Ezenfelül lehetőség van csomópontokat és az éleket tetszóleges attribútum-érték párokkal címkézni.

Ez a nagyon általános gráfalapú reprezentáció teljesen új lehetőségeket nyit a szintaktikai vagy szemantikai információk kifejezésére a hagyományos függőségi leírás felett. Ennek megfelelően a feladatra olyan ötletekkel álltunk elő, amelyek célja, hogy az általános reprezentáció lehetőségeit a függőségi elemzésen túl is kihasználjuk.

Az első kísérletsorozatban (6.2. rész) a klasszikus függőségi elemzési megközelítésből indultunk ki, de egyetlen függőségi elemzés helyett a lehetséges függőségi elemzések eloszlását határoztuk meg egy adott mondatra, aminek a lekódolására lehetőséget adott a verseny gráfalapú általános reprezentációja. A 6.3. részben bemutattam egy lehetséges megoldást a függőségi elemzés kiegészítésére konstituens elemzésből érkező információk segítségével. Az EPE 2017 verseny struktúrájának köszönhetően ezek a kevert információk egyszerre eljutnak a magasabb szintű feladatokhoz, ahol a gépi tanuló rendszer el tudja dönteni, hogy mely adatok, illetve milyen kombinációik a leghasznosabbak számára.

Az utolsó kísérletsorozatunkban (6.4. rész) megpróbáltuk automatikusan felismerni, hogy mely függőségi elemzési címkék hasznosak a magasabb szintú alkalmazásokhoz, és csak azokat megtartani. A haszontalannak bizonyult címkéket pedig összevontuk egymással.

Gyógyszerszedési Események Osztályozásának Javítása Szintaktikai Elemzéssel

A 7. fejezetben különböző stratégiákat mutattam be orvosi dokumentumokból történő gyógyszerszedési események kinyerésére és osztályozására, demonstrálva, hogy a szintaxis alapú információk felhasználása még nagy nyelvi modellek jelenlétében is javítani tud a rendszer hatékonyságán (Szántó et al., 2023).

A kísérleteink során a Contextualized Medication Event Dataset-et használtuk, amely három feladatot

tartalmaz. A gyógyszerkinyerési és eseményosztályozási feladatokhoz egy CRF-BERT alapú megoldást alkalmaztunk, amelyet ellenséges példákkal gazdagítottunk.

Az adatbázisban szereplő harmadik feladat gyógyszerszedési változások osztályozása kontextus alapján. Ehhez szintén egy BERT alapú megoldást valósítottunk meg, amelyet a dokumentumok releváns részének szintaxis alapú súlyozásával bővítettünk. Ennek a megközelítésnek a fő motivációja a szentimentelemzés területéről származik. Az ott már jól bevált *local context focus mechanism* és a szintaktikai elemzésen alapuló súlyozás használata sikeresen javította a gyógyszerszedési változások osztályozásának az eredményeit is.

Bibliography

Abeillé, A., Clément, L., Toussenel, F.: Building a treebank for french. In: Abeillé, A. (ed.) Treebanks. Kluwer, Dordrecht (2003)

Abney, S., Flickenger, S., Gdaniec, C., Grishman, C., Harrison, P., Hindle, D., Ingria, R., Jelinek, F., Klavans, J., Liberman, M., Marcus, M., Roukos, S., Santorini, B., Strzalkowski, T.: Procedure for quantitatively comparing the syntactic coverage of English grammars. In: Black, E. (ed.) Proceedings of the workshop on Speech and Natural Language. pp. 306–311 (1991)

Aduriz, I., Aranzabe, M.J., Arriola, J.M., Atutxa, A., Diaz de Ilarraza, A., Garmendia, A., Oronoz, M.: Construction of a Basque dependency treebank. In: Proceedings of the 2nd Workshop on Treebanks and Linguistic Theories (TLT). pp. 201–204. Växjö, Sweden (2003)

Alsentzer, E., Murphy, J.R., Boag, W., Weng, W.H., Jin, D., Naumann, T., McDermott, M.: Publicly available clinical bert embeddings. arXiv preprint arXiv:1904.03323 (2019)

Altıntaş, M., Tantuğ, A.C.: Improving the performance of graph based dependency parsing by guiding bi-affine layer with augmented global and local features. Intelligent Systems with Applications 18, 200190 (2023)

Angeli, G., Manning, C.D., Jurafsky, D.: Parsing time: Learning to interpret time expressions. In: Proc. NAACL HLT. pp. 446–455. NAACL HLT '12, Stroudsburg, PA, USA (2012)

Aranzabe, M.J., Atutxa, A., Bengoetxea, K., de Illarraza, A.D., Goenaga, I., Gojenola, K., Uria, L.: Automatic Conversion of the Basque Dependency Treebank to Universal Dependencies. In: Proceedings of the Fourteenth International Workshop on Treebanks and Linguistic Theories (TLT 14) (2014)

Babarczy, A., Gábor, B., Hamp, G., Rung, A., Szakadát, I.: Hunpars: a rule-based sentence parser for hungarian. In: Proceedings of the 6th International Symposium on Computational Intelligence (2005)

Baccianella, S., Esuli, A., Sebastiani, F.: SentiWordNet 3.0: An Enhanced Lexical Resource for Sentiment Analysis and Opinion Mining. In: Proc. LREC (2010)

Barta, C., Csendes, D., Csirik, J., Hócza, A., Kocsor, A., Kovács, K.: Learning syntactic tree patterns from a balanced hungarian natural language database, the szeged treebank. In: 2005 International Conference on Natural Language Processing and Knowledge Engineering. pp. 225–231. IEEE (2005)

Björkelund, A., Çetinoğlu, O., Faleńska, A., Farkas, R., Müller, T., Seeker, W., Szántó, Z.: Introducing the IMS-Wroclaw-Szeged-CIS entry at the SPMRL 2014 Shared Task: Reranking and Morpho-syntax meet Unlabeled Data. In: Proceedings of the First Joint Workshop on Statistical Parsing of Morphologically Rich Languages and Syntactic Analysis of Non-Canonical Languages. pp. 97–102 (August 2014), URL http://www.aclweb.org/anthology/W14-6110

Björkelund, A., Çetinoğlu, O., Farkas, R., Müller, T., Seeker, W.: (re)ranking meets morphosyntax: State-of-the-art results from the SPMRL 2013 shared task. In: Proceedings of the Fourth Workshop on Statistical Parsing of Morphologically-Rich Languages. pp. 135–145. Seattle, Washington, USA (October 2013), URL http://www.aclweb.org/anthology/W13-4916

Björkelund, A., Kuhn, J.: Learning structured perceptrons for coreference resolution with latent antecedents and non-local features. In: Proc. ACL. pp. 47–57 (2014)

Björne, J., Ginter, F., Salakoski, T.: Epe 2017: The biomedical event extraction downstream application. Proceedings of the 2017 Shared Task on Extrinsic Parser Evaluation pp. 17–24 (2017)

Björne, J., Heimonen, J., Ginter, F., Airola, A., Pahikkala, T., Salakoski, T.: Extracting complex biological events with rich graph-based feature sets. In: Proceedings of the BioNLP 2009 Workshop Companion Volume for Shared Task. pp. 10–18 (2009)

Bohnet, B.: Top accuracy and fast dependency parsing is not a contradiction. In: Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010). pp. 89–97 (2010a), URL http://www.aclweb.org/anthology/C10-1011

Bohnet, B.: Top accuracy and fast dependency parsing is not a contradiction. In: Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010). pp. 89–97 (2010b), URL http://www.aclweb.org/anthology/C10-1011

Bohnet, B.: Very high accuracy and fast dependency parsing is not a contradiction. In: Proceedings of the 23rd International Conference on Computational Linguistics. pp. 89–97. COLING '10, Association for Computational Linguistics, Stroudsburg, PA, USA (2010c)

Bojanowski, P., Grave, E., Joulin, A., Mikolov, T.: Enriching word vectors with subword information. Transactions of the Association for Computational Linguistics 5, 135–146 (2017)

Brants, S., Dipper, S., Hansen, S., Lezius, W., Smith, G.: The TIGER treebank. In: Hinrichs, E., Simov, K. (eds.) Proceedings of the First Workshop on Treebanks and Linguistic Theories (TLT 2002). pp. 24–41. Sozopol, Bulgaria (2002)

Brown, P.F., Della Pietra, V.J., deSouza, P.V., Lai, J.C., Mercer, R.L.: Class-based n-gram models of natural language. Computational Linguistics 18(4), 467–479 (1992)

Charniak, E.: A maximum-entropy-inspired parser. In: Proceedings of the 1st North American chapter of the Association for Computational Linguistics conference. pp. 132–139 (2000)

Charniak, E., Johnson, M.: Coarse-to-fine n-best parsing and MaxEnt discriminative reranking. In: Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics. pp. 173–180. ACL '05 (2005), URL http://dx.doi.org/10.3115/1219840.1219862

Chen, X., Kit, C.: Higher-order constituent parsing and parser combination. In: Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers). pp. 1–5 (2012), URL http://www.aclweb.org/anthology/P12-2001

Choi, K.S., Han, Y.S., Han, Y.G., Kwon, O.W.: KAIST tree bank project for Korean: Present and future development. In: Proceedings of the International Workshop on Sharable Natural Language Resources. pp. 7–14. Citeseer (1994)

Clark, S., Curran, J.R.: Wide-coverage efficient statistical parsing with ccg and log-linear models. COMPUTATIONAL LINGUISTICS 33 (2007)

Collins, M.: Discriminative Reranking for Natural Language Parsing. In: Proceedings of the Seventeenth International Conference on Machine Learning. pp. 175–182. ICML '00 (2000), URL http://portal.acm.org/citation.cfm?id=645529.658119

Collins, M.: Discriminative training methods for hidden Markov models. In: Proc. EMNLP. vol. 10, pp. 1–8 (2002)

Collins, M.: Head-driven statistical models for natural language parsing. Computational linguistics 29(4), 589–637 (2003)

Conneau, A., Khandelwal, K., Goyal, N., Chaudhary, V., Wenzek, G., Guzmán, F., Grave, É., Ott, M., Zettlemoyer, L., Stoyanov, V.: Unsupervised Cross-lingual Representation Learning at Scale. In: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics. pp. 8440–8451 (2020)

Csendes, D., Csirik, J., Gyimóthy, T.: The Szeged Corpus. A POS Tagged and Syntactically Annotated Hungarian Natural Language Corpus. In: COLING 2004 5th International Workshop on Linguistically Interpreted Corpora. pp. 19–22 (2004)

Csendes, D., Csirik, J., Gyimóthy, T., Kocsor, A.: The Szeged treebank. In: Matoušek, V., Mautner, P., Pavelka, T. (eds.) Text, Speech and Dialogue: Proceedings of TSD 2005. Springer (2005)

Dehdari, J., Tounsi, L., van Genabith, J.: Morphological features for parsing morphologically-rich languages: A case of arabic. In: Proceedings of the Second Workshop on Statistical Parsing of Morphologically Rich Languages. pp. 12–21. Association for Computational Linguistics, Dublin, Ireland (October 2011), URL http://www.aclweb.org/anthology/W11-3802

Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In: Proceedings of NAACL-HLT. pp. 4171–4186 (2019)

Do, H.H., Prasad, P., Maag, A., Alsadoon, A.: Deep learning for aspect-based sentiment analysis: a comparative review. Expert systems with applications 118, 272–299 (2019)

Dobrovoljc, K., Nivre, J.: The Universal Dependencies Treebank of Spoken Slovenian. In: Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016) (2016)

Dong, L., Wei, F., Liu, S., Zhou, M., Xu, K.: A statistical parsing framework for sentiment classification. Computational Linguistics 41(2), 265–308 (2015)

Dozat, T., Manning, C.D.: Deep Biaffine Attention for Neural Dependency Parsing. In: International Conference on Learning Representations (2017)

É. Kiss, K.: The Syntax of Hungarian. Cambridge University Press, Cambridge (2002)

Enevoldsen, K., Hansen, L., Nielbo, K.: DaCy: A Unified Framework for Danish NLP. arXiv preprint arXiv:2107.05295 (2021)

Erjavec, T.: MULTEXT-East: morphosyntactic resources for Central and Eastern European languages. Language Resources and Evaluation 46(1), 131–142 (Mar 2012)

Farkas, R., Bohnet, B.: Stacking of dependency and phrase structure parsers. In: Proceedings of COLING 2012. pp. 849–866. The COLING 2012 Organizing Committee, Mumbai, India (December 2012a), URL http://www.aclweb.org/anthology/C12-1052

Farkas, R., Bohnet, B.: Stacking of dependency and phrase structure parsers. In: Proceedings of COLING 2012. pp. 849–866. The COLING 2012 Organizing Committee, Mumbai, India (December 2012b)

Farkas, R., Bohnet, B., Schmid, H.: Features for phrase-structure reranking from dependency parses. In: Proceedings of the 12th International Conference on Parsing Technologies. pp. 209–214 (2011), URL http://www.aclweb.org/anthology/W11-2924

Farkas, R., Szarvas, G., Hegedűs, I., Almási, A., Vincze, V., Ormándi, R., Busa-Fekete, R.: Semi-automated construction of decision rules to predict morbidities from clinical texts. Journal of the American Medical Informatics Association 16(4), 601–605 (2009)

Farkas, R., Vincze, V., Schmid, H.: Dependency Parsing of Hungarian: Baseline Results and Challenges. In: Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics. pp. 55–65. Association for Computational Linguistics, Avignon, France (April 2012), URL http://www.aclweb.org/anthology/E12-1007

Feldman, R.: Techniques and applications for sentiment analysis. Communications of the ACM 56(4), 82 (apr 2013)

Fraser, A., Schmid, H., Farkas, R., Wang, R., Schütze, H.: Knowledge sources for constituent parsing of german, a morphologically rich and less-configurational language. Computational Linguistics 39(1), 57–85 (2013)

Goldberg, Y., Adler, M., Elhadad, M.: EM can find pretty good HMM POS-taggers (when given a good start). In: Proceedings of ACL-08: HLT. pp. 746–754 (2008), URL http://www.aclweb.org/anthology/P/P08/P08-1085

Goldberg, Y., Elhadad, M.: Word Segmentation, Unknown-word Resolution, and Morphological Agreement in a Hebrew Parsing System. Computational Linguistics 39(1), 121–160 (2013)

Gong, Z., Wang, W., Li, B., Song, D., Ku, W.S.: Adversarial texts with gradient methods. arXiv preprint arXiv:1801.07175 (2018)

Goodfellow, I.J., Shlens, J., Szegedy, C.: Explaining and harnessing adversarial examples. arXiv preprint arXiv:1412.6572 (2014)

Hajič, J.: Building a syntactically annotated corpus: The prague dependency treebank. Issues of Valency and Meaning. Studies in Honour of Jarmila Panevová pp. 106–132 (1998)

Hangya, V., Berend, G., Varga, I., Farkas, R.: Szte-nlp: Aspect level opinion mining exploiting syntactic cues. SemEval 2014 p. 610 (2014)

Hangya, V., Farkas, R.: Filtering and polarity detection for reputation management on tweets. In: CEUR WORKSHOP PROCEEDINGS. szte (2013)

Hangya, V., Szántó, Z., Farkas, R.: Latent syntactic structure-based sentiment analysis. In: 2017 2nd IEEE International Conference on Computational Intelligence and Applications (ICCIA). pp. 248–254. IEEE (2017)

Haverinen, K., Nyblom, J., Viljanen, T., Laippala, V., Kohonen, S., Missilä, A., Ojala, S., Salakoski, T., Ginter, F.: Building the essential resources for finnish: the turku dependency treebank. Language Resources and Evaluation 48(3), 493–531 (2014), URL http://dx.doi.org/10.1007/s10579-013-9244-1

He, P., Gao, J., Chen, W.: Debertav3: Improving deberta using electra-style pre-training with gradient-disentangled embedding sharing. arXiv preprint arXiv:2111.09543 (2021)

Honnibal, M., Goldberg, Y., Johnson, M.: A Non-Monotonic Arc-Eager Transition System for Dependency Parsing. In: Proceedings of the Seventeenth Conference on Computational Natural Language Learning. pp. 163–172. Association for Computational Linguistics, Sofia, Bulgaria (Aug 2013)

Honnibal, M., Montani, I., Van Landeghem, S., Boyd, A.: spacy: Industrial-strength natural language processing in python. arXiv preprint arXiv:1810.04805 (2020)

Huang, L.: Forest reranking: Discriminative parsing with non-local features. In: Proceedings of ACL-08: HLT. pp. 586–594 (2008a), URL http://www.aclweb.org/anthology/P/P08/P08-1067

Huang, L.: Forest reranking: Discriminative parsing with non-local features. In: Proc. ACL HLT. pp. 586–594 (2008b)

Indig, B., Sass, B., Simon, E., Mittelholcz, I., Vadász, N., Makrai, M.: One format to rule them all – the `emtsv` pipeline for Hungarian. In: Proceedings of the 13th Linguistic Annotation Workshop. pp. 155–165. Association for Computational Linguistics, Florence, Italy (aug 2019)

Iván, S., Ormándi, R., Kocsor, A.: Magyar mondatok svm alapú szintaxis elemzése. In: V. Magyar Számítógépes Nyelvészeti Konferencia (MSZNY2007). pp. 281–283. Szeged (2007)

Jiang, L., Yu, M., Zhou, M., Liu, X., Zhao, T.: Target-dependent Twitter Sentiment Classification. In: Proc. ACL. pp. 151–160 (2011)

Johannsen, A., Alonso, H.M., Plank, B.: Universal Dependencies for Danish. In: Proceedings of the Fourteenth International Workshop on Treebanks and Linguistic Theories (TLT 14) (2014)

Johansson, R.: Epe 2017: The trento–gothenburg opinion extraction system. In: Proceedings of the 2017 Shared Task on Extrinsic Parser Evaluation at the Fourth International Conference on Dependency Linguistics and the 15th International Conference on Parsing Technologies. Pisa, Italy. pp. 31–39 (2017)

Johansson, R., Moschitti, A.: Relational features in fine-grained opinion analysis. Computational Linguistics 39(3), 473–509 (2013)

Jurafsky, D., Martin, J.: Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition, vol. 2 (02 2008)

Kakkonen, T.: Dependency treebanks: methods, annotation schemes and tools. In: Proceedings of the 15th NODALIDA conference (2005)

Kim, J.D., Ohta, T., Tateisi, Y., Tsujii, J.: Genia corpus—a semantically annotated corpus for bio-textmining. Bioinformatics 19(suppl_1), i180–i182 (2003)

Kim, J.D., Ohta, T., Pyysalo, S., Kano, Y., Tsujii, J.: Overview of bionlp'09 shared task on event extraction. In: Proceedings of the BioNLP 2009 workshop companion volume for shared task. pp. 1–9 (2009)

Kitaev, N., Lu, T., Klein, D.: Learned incremental representations for parsing. In: Muresan, S., Nakov, P., Villavicencio, A. (eds.) Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). pp. 3086–3095. Association for Computational Linguistics, Dublin, Ireland (May 2022), URL https://aclanthology.org/2022.acl-long.220

Kondratyuk, D., Straka, M.: 75 Languages, 1 Model: Parsing Universal Dependencies Universally. In: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP). pp. 2779–2795 (2019)

Kong, L., Schneider, N., Swayamdipta, S., Bhatia, A., Dyer, C., Smith, N.A.: A dependency parser for tweets. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP). pp. 1001–1012 (2014)

Kurakin, A., Goodfellow, I., Bengio, S.: Adversarial machine learning at scale. arXiv preprint arXiv:1611.01236 (2016)

Lapponi, E., Oepen, S., Øvrelid, L.: Epe 2017: The sherlock negation resolution downstream application. Proceedings of the 2017 Shared Task on Extrinsic Parser Evaluation pp. 25–30 (2017)

Lapponi, E., Velldal, E., Øvrelid, L., Read, J.: Uio 2: sequence-labeling negation using dependency features. In: * SEM 2012: The First Joint Conference on Lexical and Computational Semantics–Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation (SemEval 2012). pp. 319–327 (2012)

Lazaridou, A., Titov, I., Sporleder, C.: A Bayesian Model for Joint Unsupervised Induction of Sentiment, Aspect and Discourse Representations. In: Proc. ACL. pp. 1630–1639 (2013)

Le Roux, J., Sagot, B., Seddah, D.: Statistical Parsing of Spanish and Data Driven Lemmatization. In: Proceedings of the ACL 2012 Joint Workshop on Statistical Parsing and Semantic Processing of Morphologically Rich Languages. pp. 55–61 (2012), URL http://www.aclweb.org/anthology/W12-3408

Lecun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. Proceedings of the IEEE 86(11), 2278–2324 (1998)

Lee, J., Yoon, W., Kim, S., Kim, D., Kim, S., So, C.H., Kang, J.: Biobert: a pre-trained biomedical language representation model for biomedical text mining. Bioinformatics 36(4), 1234–1240 (2020)

Liu, B.: Sentiment analysis and opinion mining. Synthesis Lectures on Human Language Technologies 5(1), 1–167 (2012)

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., Stoyanov, V.: Roberta: A robustly optimized bert pretraining approach. arXiv preprint arXiv:1907.11692 (2019)

Mahajan, D., Liang, J.J., Tsou, C.H.: Toward understanding clinical context of medication change events in clinical narratives. In: AMIA Annual Symposium Proceedings. vol. 2021, p. 833. American Medical Informatics Association (2021)

Manning, C.D., Surdeanu, M., Bauer, J., Finkel, J., Bethard, S.J., McClosky, D.: The Stanford CoreNLP natural language processing toolkit. In: Proc. ACL. pp. 55–60 (2014)

Mao, J., Liu, W.: Hadoken: a bert-crf model for medical document anonymization. In: IberLEF@ SEPLN. pp. 720–726 (2019)

Marcus, M.P., Marcinkiewicz, M.A., Santorini, B.: Building a large annotated corpus of english: The penn treebank. Comput. Linguist. 19(2), 313–330 (jun 1993)

Marton, Y., Habash, N., Rambow, O.: Improving arabic dependency parsing with lexical and inflectional morphological features. In: Proceedings of the NAACL HLT 2010 First Workshop on Statistical Parsing of Morphologically-Rich Languages. pp. 13–21 (2010), URL http://www.aclweb.org/anthology/W10-1402

McCallum, A.K.: Mallet: A machine learning for language toolkit (2002), http://mallet.cs.umass.edu

McDonald, R., Nivre, J.: Characterizing the errors of data-driven dependency parsing models. In: Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL). pp. 122–131 (2007), URL http://www.aclweb.org/anthology/D/D07/D07-1013.pdf

McDonald, R., Pereira, F., Ribarov, K., Hajič, J.: Non-projective dependency parsing using spanning tree algorithms. In: Proceedings of the Conference on Human Language Technology and Empirical Methods in Natural Language Processing. pp. 523–530. HLT '05, Association for Computational Linguistics, Stroudsburg, PA, USA (2005)

Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient Estimation of Word Representations in Vector Space (2013)

Moradi, M., Samwald, M.: Improving the robustness and accuracy of biomedical language models through adversarial training. Journal of Biomedical Informatics 132, 104114 (2022)

Morante, R., Blanco, E.: * sem 2012 shared task: Resolving the scope and focus of negation. In: * SEM 2012: The First Joint Conference on Lexical and Computational Semantics–Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation (SemEval 2012). pp. 265–274 (2012)

Mueller, T., Schmid, H., Schütze, H.: Efficient higher-order CRFs for morphological tagging. In: Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing. pp. 322–332 (2013), URL http://www.aclweb.org/anthology/D13-1032

Muischnek, K., Müürisep, K., Puolakainen, T.: Estonian Dependency Treebank: from Constraint Grammar tagset to Universal Dependencies. In: Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016) (2016)

Müller, S., Abeillé, A., Borsley, R.D., Koenig, J.P.: Head-Driven Phrase Structure Grammar: The handbook (Volume 9). Language Science Press (2021)

Müller, T., Cotterell, R., Fraser, A., Schütze, H.: Joint Lemmatization and Morphological Tagging with Lemming. In: Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing. pp. 2268–2274. Association for Computational Linguistics, Lisbon, Portugal (Sep 2015)

Müller, T., Schmid, H., Schütze, H.: Efficient higher-order CRFs for morphological tagging. In: Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing. pp. 322–332 (2013), URL http://www.aclweb.org/anthology/D13-1032

Nemeskey, D.M.: Egy emBERT próbáló feladat. In: XVI. Magyar Számítógépes Nyelvészeti Konferencia (MSZNY2020). pp. 409–418. Szeged (2020a)

Nemeskey, D.M.: Natural Language Processing Methods for Language Modeling. Ph.D. thesis, Eötvös Loránd University (2020b)

Nivre, J.: Dependency parsing. Language and Linguistics Compass 4(3), 138–152 (2010), URL https://compass.onlinelibrary.wiley.com/doi/abs/10.1111/j.1749-818X.2010.00187.x

Nivre, J.: Universal Dependencies for Swedish. In: Proceedings of SLTC 2014 (2014)

Nivre, J.: Towards a Universal Grammar for Natural Language Processing. In: Computational Linguistics and Intelligent Text Processing, pp. 3–16 (2015), URL http://stp.lingfil.uu.se/~nivre/docs/nivre15cicling.pdf

Nivre, J., Agić, Ž., Aranzabe, M.J., Asahara, M., Atutxa, A., Ballesteros, M., Bauer, J., Bengoetxea, K., Bhat, R.A., Bosco, C., et al.: Universal dependencies 1.2 (2015)

Nivre, J., Hall, J., Kübler, S., McDonald, R., Nilsson, J., Riedel, S., Yuret, D.: The CoNLL 2007 shared task on dependency parsing. In: Proceedings of the CoNLL Shared Task Session of EMNLP-CoNLL 2007. pp. 915–932 (2007), URL http://www.aclweb.org/anthology/D/D07/D07-1096.pdf

Nivre, J., de Marneffe, M.C., Ginter, F., Goldberg, Y., Hajic, J., Manning, C.D., McDonald, R., Petrov, S., Pyysalo, S., Silveira, N., Tsarfaty, R., Zeman, D.: Universal Dependencies v1: A Multilingual Treebank Collection. In: Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016) (2016)

Nivre, J., de Marneffe, M.C., Ginter, F., Hajič, J., Manning, C.D., Pyysalo, S., Schuster, S., Tyers, F., Zeman, D.: Universal Dependencies v2: An Evergrowing Multilingual Treebank Collection. In: Proceedings of the Twelfth Language Resources and Evaluation Conference. pp. 4034–4043. European Language Resources Association, Marseille, France (May 2020)

Nivre, J., Vincze, V.: Light verb constructions in universal dependencies (2015)

Novák, A.: A new form of humor — mapping constraint-based computational morphologies to a finite-state representation. In: Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14). pp. 1068–1073. European Language Resources Association (ELRA), Reykjavik, Iceland (May 2014)

Novák, A., Siklósi, B., Oravecz, C.: A New Integrated Open-source Morphological Analyzer for Hungarian. In: Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16). pp. 1315–1322. European Language Resources Association (ELRA), Portorož, Slovenia (May 2016)

Oepen, S., Øvrelid, L., Björne, J., Johansson, R., Lapponi, E., Ginter, F., Velldal, E.: The 2017 shared task on extrinsic parser evaluation. towards a reusable community infrastructure. In: Proceedings of the 2017 Shared Task on Extrinsic Parser Evaluation at the Fourth International Conference on Dependency Linguistics and the 15th International Conference on Parsing Technologies. Pisa, Italy. pp. 1–16 (2017)

Orosz, G., Novák, A.: PurePos 2.0: a hybrid tool for morphological disambiguation. In: Proceedings of the International Conference Recent Advances in Natural Language Processing RANLP 2013. pp. 539–545. INCOMA Ltd. Shoumen, BULGARIA, Hissar, Bulgaria (Sep 2013)

Orosz, G., Szabó, G., Berkecz, P., Szántó, Z., Farkas, R.: Advancing hungarian text processing with huspacy: Efficient and accurate nlp pipelines. In: International Conference on Text, Speech, and Dialogue. pp. 58–69. Springer (2023)

Orosz, G., Szántó, Z., Berkecz, P., Szabó, G., Farkas, R.: HuSpaCy: an industrial-strength Hungarian natural language processing toolkit. In: XVIII. Magyar Számítógépes Nyelvészeti Konferencia (2022)

Øvrelid, L., Hohle, P.: Universal Dependencies for Norwegian. In: Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016) (2016)

Petrov, S.: Coarse-to-Fine Natural Language Processing. Ph.D. thesis, University of California at Bekeley, Berkeley, CA, USA (2009), URL http://www.petrovi.de/data/dissertation.pdf

Petrov, S.: Products of random latent variable grammars. In: Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics. pp. 19–27. Association for Computational Linguistics (2010a), URL http://aclweb.org/anthology/N10-1003

Petrov, S.: Products of Random Latent Variable Grammars. In: Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics. pp. 19–27. Los Angeles, California (June 2010b), URL http://www.aclweb.org/anthology/N10-1003

Petrov, S.: Products of random latent variable grammars. In: Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics. pp. 19–27. HLT '10, Association for Computational Linguistics, Stroudsburg, PA, USA (2010c)

Petrov, S., Barrett, L., Thibaux, R., Klein, D.: Learning accurate, compact, and interpretable tree annotation. In: Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics. pp. 433–440 (2006a)

Petrov, S., Barrett, L., Thibaux, R., Klein, D.: Learning accurate, compact, and interpretable tree annotation. In: Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics. pp. 433–440 (2006b)

Petrov, S., Chang, P.C., Ringgaard, M., Alshawi, H.: Uptraining for accurate deterministic question parsing. In: Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing. pp. 705–713. Association for Computational Linguistics, Cambridge, MA (October 2010), URL http://www.aclweb.org/anthology/D10-1069

Petrov, S., Das, D., McDonald, R.: A universal part-of-speech tagset. In: Proceedings of LREC (2012)

Petrov, S., McDonald, R.: Overview of the 2012 shared task on parsing the web. Notes of the First Workshop on Syntactic Analysis of Non-Canonical Language (SANCL) (2012)

Phan, M.H., Ogunbona, P.O.: Modelling context and syntactical features for aspect-based sentiment analysis. In: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics. pp. 3211–3220 (2020)

Pitler, E.: Conjunction representation and ease of domain adaptation. Notes of the First Workshop on Syntactic Analysis of Non-Canonical Language (SANCL) (2012)

Polguère, A., Mel'čuk, I.A. (eds.): Dependency in Linguistic Description. Studies in language companion series (2009)

Pontiki, M., Galanis, D., Pavlopoulos, J., Papageorgiou, H., Androutsopoulos, I., Manandhar, S.: Semeval-2014 task 4: Aspect based sentiment analysis. In: Proc. SemEval. pp. 27–35. SemEval '14 (2014)

Prószéky, G., Tihanyi, L., Ugray, G.: Moose: a robust high-performance parser and generator. In: Proceedings of the 9th EAMT Workshop: Broadening horizons of machine translation and its applications (2004)

Pyysalo, S., Kanerva, J., Missilä, A., Laippala, V., Ginter, F.: Universal Dependencies for Finnish. In: Proceedings of Nodalida (2015)

Qi, P., Zhang, Y., Zhang, Y., Bolton, J., Manning, C.D.: Stanza: A Python Natural Language Processing Toolkit for Many Human Languages. In: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations (2020)

Raj, D., Sahu, S., Anand, A.: Learning local and global contexts using a convolutional recurrent network model for relation classification in biomedical text. In: Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017). pp. 311–321. Association for Computational Linguistics, Vancouver, Canada (Aug 2017)

Rákosi, G., Laczkó, T.: Constraining the grammar of aps in english and hungarian: Challenges and solutions in an lfg-based computational project. HUSSE 11 (2013)

Rambow, O., Dorr, B., Farwell, D., Green, R., Habash, N., Helmreich, S., Hovy, E., Levin, L., Miller, K.J., Mitamura, T., Reeder, Florence, Siddharthan, A.: Parallel syntactic annotation of multiple languages. In: Proceedings of LREC (2006)

Ravi, K., Ravi, V.: A survey on opinion mining and sentiment analysis: Tasks, approaches and applications. Knowledge-Based Systems 89, 14–46 (2015)

Rosenthal, S., Nakov, P., Ritter, A., Stoyanov, V.: Semeval-2014 task 9: Sentiment analysis in twitter. In: Proc. SemEval. pp. 73–80. No. SemEval (2014)

Schuster, S., de La Clergerie, É.V., Candito, M.D., Sagot, B., Manning, C.D., Seddah, D.: Paris and stanford at epe 2017: Downstream evaluation of graph-based dependency representations. In: EPE 2017-The First Shared Task on Extrinsic Parser Evaluation. pp. 47–59 (2017)

Seddah, D., Kübler, S., Tsarfaty, R.: Introducing the SPMRL 2014 Shared Task on Parsing Morphologically-rich Languages. In: Proceedings of the First Joint Workshop on Statistical Parsing of Morphologically Rich Languages and Syntactic Analysis of Non-Canonical Languages. pp. 103–109 (2014a), URL http://www.aclweb.org/anthology/W14-6111

Seddah, D., Tsarfaty, R., Kübler, S., Candito, M., Choi, J., Constant, M., Farkas, R., Goenaga, I., Gojenola, K., Goldberg, Y., Green, S., Habash, N., Kuhlmann, M., Maier, W., Nivre, J., Przepiorkowski, A., Roth, R., Seeker, W., Versley, Y., Vincze, V., Woliński, M., Wróblewska, A., Villemonte de la Clérgerie, E.: Overview of the SPMRL 2014 shared task on parsing morphologically rich languages. In: Notes of the SPMRL 2014 Shared Task on Parsing Morphologically-Rich Languages. Dublin, Ireland (2014b)

Seddah, D., Tsarfaty, R., Kübler, S., Candito, M., Choi, J.D., Farkas, R., Foster, J., Goenaga, I., Gojenola Galletebeitia, K., Goldberg, Y., Green, S., Habash, N., Kuhlmann, M., Maier, W., Nivre, J., Przepiórkowski, A., Roth, R., Seeker, W., Versley, Y., Vincze, V., Woliński, M., Wróblewska, A., de la Clergerie, E.V.: Overview of the SPMRL 2013 shared task: A cross-framework evaluation of parsing morphologically rich languages. In: Proceedings of the Fourth Workshop on Statistical Parsing of Morphologically-Rich Languages. pp. 146–182. Seattle, Washington, USA (October 2013), URL http://www.aclweb.org/anthology/W13-4917

Seeker, W., Farkas, R., Bohnet, B., Schmid, H., Kuhn, J.: Data-driven dependency parsing with empty heads. In: Proceedings of COLING 2012: Posters. pp. 1081–1090 (2012), URL http://www.aclweb.org/anthology/C12-2105

Seeker, W., Kuhn, J.: Morphological and syntactic case in statistical dependency parsing. Computational Linguistics 39(1), 23–55 (2013)

Seraji, M., Ginter, F., Nivre, J.: Universal Dependencies for Persian. In: Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016) (2016)

Sima'an, K., Itai, A., Winter, Y., Altman, A., Nativ, N.: Building a Tree-Bank for Modern Hebrew Text. In: Traitement Automatique des Langues (2001)

Simkó, K.I., Vincze, V., Farkas, R.: Többszintű szintaktikai reprezentáció kialakítása a szeged fc treebankben (2014)

Simkó, K.I., Vincze, V., Szántó, Zs., Farkas, R.: An Empirical Evaluation of Automatic Conversion from Constituency to Dependency in Hungarian. In: Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers. pp. 1392–1401 (2014)

Simon, E., Lendvai, P., Németh, G., Olaszy, G., Vicsi, K.: A magyar nyelv a digitális korban – The Hungarian Language in the Digital Age. Georg Rehm and Hans Uszkoreit (Series Editors): META-NET White Paper Series, Springer (2012)

Simon, E., Indig, B., Kalivoda, Á., Mittelholcz Iván, S.B., Vadász, N.: Újabb fejlemények az e-magyar háza táján. In: Berend, G., Gosztolya, G., Vincze, V. (eds.) XVI. Magyar Számítógépes Nyelvészeti Konferencia. pp. 29–42. Szegedi Tudományegyetem Informatikai Tanszékcsoport, Szeged (2020)

Simon, E., Vadász, N.: Introducing NYTK-NerKor, A Gold Standard Hungarian Named Entity Annotated Corpus. In: Ekstein, K., Pártl, F., Konopík, M. (eds.) Text, Speech, and Dialogue - 24th International Conference, TSD 2021, Olomouc, Czech Republic, September 6-9, 2021, Proceedings. Lecture Notes in Computer Science, vol. 12848, pp. 222–234. Springer (2021)

Simon, E., Vadász, N., Lévai, D., Dávid, N., Orosz, G., Szántó, Z.: Az NYTK-NerKor több szempontú kiértékelése. XVIII. Magyar Számítógépes Nyelvészeti Konferencia (2022)

Socher, R., Perelygin, A., Wu, J., Chuang, J., Manning, C.D., Ng, A., Potts, C.: Recursive deep models for semantic compositionality over a sentiment treebank. In: Proc. EMNLP. pp. 1631–1642 (2013)

Song, Y., Wang, J., Jiang, T., Liu, Z., Rao, Y.: Attentional encoder network for targeted sentiment classification. arXiv preprint arXiv:1902.09314 (2019)

Souza, F., Nogueira, R., Lotufo, R.: Portuguese named entity recognition using bert-crf. arXiv preprint arXiv:1909.10649 (2019)

Straka, M.: UDPipe 2.0 prototype at CoNLL 2018 UD shared task. In: Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies. pp. 197–207. Association for Computational Linguistics, Brussels, Belgium (Oct 2018)

Szántó, Z., Bánáti, B., Zombori, T.: Enhancing medication event classification with syntax parsing and adversarial learning. In: Maglogiannis, I., Iliadis, L., MacIntyre, J., Dominguez, M. (eds.) Artificial Intelligence Applications and Innovations. pp. 114–124. Springer Nature Switzerland, Cham (2023)

Szántó, Z., Farkas, R.: Special techniques for constituent parsing of morphologically rich languages. In: Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics. pp. 135–144. Gothenburg, Sweden (April 2014)

Szántó, Z., Farkas, R.: Constituency parse reranking for morphologically rich languages. Acta Polytechnica Hungarica 12(8) (2015)

Szántó, Z., Farkas, R.: Szeged at epe 2017: First experiments in a generalized syntactic parsing framework. In: Proceedings of the 2017 Shared Task on Extrinsic Parser Evaluation at the Fourth International Conference on Dependency Linguistics and the 15th International Conference on Parsing Technologies. Pisa, Italy. pp. 75–79 (2017)

Szarvas, G., Farkas, R., Kocsor, A.: A multilingual named entity recognition system using boosting and C4.5 decision tree learning algorithms. In: International Conference on Discovery Science. pp. 267–278. Springer (2006)

Szécsényi, T.: Magyar mondatszerkezeti jelenségek elemzése hpsg-ben (2011)

Tanaka, T., Miyao, Y., Asahara, M., Uematsu, S., Kanayama, H., Mori, S., Matsumoto, Y.: Universal Dependencies for Japanese. In: Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016) (2016)

Tian, Y., Song, Y., Xia, F., Zhang, T.: Improving constituency parsing with span attention. In: Cohn, T., He, Y., Liu, Y. (eds.) Findings of the Association for Computational Linguistics: EMNLP 2020. pp. 1691–1703. Association for Computational Linguistics, Online (Nov 2020), URL https://aclanthology.org/2020.findings-emnlp.153

Törkenczy, M.: Practical Hungarian Grammar (2005)

Tsarfaty, R., Seddah, D., Kübler, S., Nivre, J.: Parsing morphologically rich languages: Introduction to the special issue. Computational Linguistics 39(1), 15–22 (2013)

Uzuner, O.: Second i2b2 workshop on natural language processing challenges for clinical records. AMIA ... Annual Symposium proceedings / AMIA Symposium. AMIA Symposium 6, 1252–1253 (02 2008)

Van Nguyen, M., Lai, V., Veyseh, A.P.B., Nguyen, T.H.: Trankit: A Light-Weight Transformer-based Toolkit for Multilingual Natural Language Processing. EACL 2021 p. 80 (2021)

Váradi, T.: The Hungarian National Corpus. In: Proceedings of the Second International Conference on Language Resources and Evaluation. pp. 385–389 (2002)

Váradi, T., Simon, E., Sass, B., Mittelholcz, I., Novák, A., Indig, B., Farkas, R., Vincze, V.: E-magyar – a digital language processing system. In: Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018). European Language Resources Association (ELRA), Miyazaki, Japan (May 2018)

Versley, Y., Rehbein, I.: Scalable discriminative parsing for german. In: Proceedings of the 11th International Conference on Parsing Technologies (IWPT'09). pp. 134–137 (2009), URL http://www.aclweb.org/anthology/W09-3820

Vilares, D., Alonso, M.A., Gómez-Rodriguez, C.: A syntactic approach for opinion mining on Spanish reviews. Natural Language Engineering 21(01), 139–163 (2013)

Vincze, V., Simkó, K., Szántó, Z., Farkas, R.: Universal Dependencies and morphology for Hungarian - and on the price of universality. In: Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers. pp. 356–365. Association for Computational Linguistics, Valencia, Spain (Apr 2017)

Vincze, V., Szauter, D., Almási, A., Móra, Gy., Alexin, Z., Csirik, J.: Hungarian Dependency Treebank. In: Proceedings of LREC 2010. ELRA, Valletta, Malta (May 2010)

Vincze, V., Varga, V., Simkó, K.I., Zsibrita, J., Nagy, Á., Farkas, R., Csirik, J.: Szeged Corpus 2.5: Morphological Modifications in a Manually POS-tagged Hungarian Corpus. In: Proceedings of LREC 2014. pp. 1074–1078 (2014), URL http://www.lrec-conf.org/proceedings/lrec2014/pdf/262_Paper.pdf

Wiebe, J., Wilson, T., Cardie, C.: Annotating expressions of opinions and emotions in language. Language resources and evaluation 39, 165–210 (2005)

Wilson, T., Wiebe, J., Hoffmann, P.: Recognizing contextual polarity in phrase-level sentiment analysis. In: Proc. HLT/EMNLP. pp. 347–354 (2005)

Yang, H., Li, K.: Pyabsa: Open framework for aspect-based sentiment analysis. arXiv preprint arXiv:2208.01368 (2022)

Zeman, D.: Reusable tagset conversion using tagset drivers. In: Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC'08) (2008)

Zeman, D.: Slavic Languages in Universal Dependencies. In: Slovko 2015: Natural Language Processing, Corpus Linguistics, E-learning (2015)

Zeng, B., Yang, H., Xu, R., Zhou, W., Han, X.: Lcf: A local context focus mechanism for aspect-based sentiment classification. Applied Sciences 9(16), 3389 (2019)

Zsibrita, J., Vincze, V., Farkas, R.: magyarlanc: A tool for morphological and dependency parsing of hungarian. In: Proceedings of the International Conference Recent Advances in Natural Language Processing RANLP 2013. pp. 763–771. INCOMA Ltd. Shoumen, BULGARIA (2013), URL http://aclweb.org/anthology/R13-1099

Acknowledgments

First of all, I would like to thank my supervisor, Dr. Richárd Farkas, for his support that began before my PhD studies and has continued to this day, even though many years have passed since then.

I would also like to thank Katalin Simkó, my girlfriend, co-author, and the proofreader of this thesis, for the limitless professional and personal support she has given me.

Additionally, I would like to thank my colleagues and friends who helped me to achieve the results presented here and made the period of my studies enjoyable.

Last but not least, I wish to thank my family for their constant help and support throughout these many years.

Some of the works were supported in part by the European Union and the European Social Fund through project FuturICT.hu (grant no.: TÁMOP-4.2.2.C-11/1/KONV-2012-0013).

Some of the works were supported by the European Union project RRF-2.3.1-21-2022-00004 within the framework of the Artificial Intelligence National Laboratory.