Heinrich Heine
Universität Düsseldorf

Frame-Semantic Parsing with Lexicalized Tree Rewriting Grammars

Kumulative Inaugural-Dissertation
zur Erlangung des Doktorgrades der Philosophie (Dr. phil.)
durch die Philosophische Fakultät der
Heinrich-Heine-Universität Düsseldorf

vorgelegt von

TATIANA BLADIER

aus

Lutsk, Ukraine

Erstbetreuerin: Prof. Dr. Laura Kallmeyer
Zweitbetreuerin: Apl.-Prof. Dr. Wiebke Petersen
Drittbetreuerin: Dr. Katalin Balogh
Düsseldorf, April 2024

Summary

This dissertation is a compilation of publications that seek to develop a large-scale data-driven frame-semantic parsing algorithm based on grammar theories with the property of extended domain of locality (EDL), particularly tree rewriting formalisms of Tree-Adjoining Grammar (TAG; Joshi, 1987) and Tree Wrapping Grammar (TWG; Kallmeyer et al., 2013; Kallmeyer and Osswald, 2018). The building blocks of these formalisms (i.e., *elementary trees*) are linguistically motivated and reflect the argument structures of predicates in sentences. We pursue this aim with a particular focus on the syntax-semantics interface, thus working on developing both a syntactic parsing methodology and combining it with frame semantics. This is, to our best knowledge, the first attempt to implement a large-scale deep semantic parser based on such formalisms, although some prototypical small-scale semantic parsers for different flavors of tree rewriting Grammars already exist. We show that tree rewriting grammars are a useful additional source for neural semantic parsing and can improve the general performance of parsing systems, contributing to the performance of the parsing model on difficult linguistic tasks.

The second chapter of the dissertation describes our method for automatically extracting tree-rewriting grammars from constituency treebanks. These grammars can be utilized for both syntactic and semantic parsing. Since these grammars are extracted from large treebanks, they facilitate generalization over elementary trees and the development of parsers relying on probabilities. Our algorithm is developed based on existing extraction procedures for TAG (Xia et al., 2000), with additional special extraction operations tailored for TWG.

The third chapter of the dissertation outlines our method for syntactic parsing using the extracted tree-rewriting grammars. Our parsing algorithm is based on supertagging followed by a parsing step. We experiment with various deep learning architectures for the supertagger and adapt an existing probabilistic TAG parser for integration into the pipeline using predicted supertags. This adaptation reduces the search space of the parser, thereby speeding up parsing. Additionally, we modify this parser to handle the tree combination operations unique to TWG.

The fourth chapter of the dissertation describes our effort to create treebanks based on the typologically inspired grammar theory of Role and Reference Grammar (RRG; Van Valin and LaPolla, 1997; Van Valin, 2005), which is a major framework used in typological language modeling. RRG places semantics and its interfaces with morphosyntax, information structure, and discourse at the center of the theory. We utilize these treebanks to extract large-scale grammars for our syntactic and semantic parsers. While it is theoretically possible to extract a TWG from every constituency treebank and tailor it according to project needs, similar to what is possible with TAG, TWG development has thus far been tightly linked to RRG theory. Therefore, all previous work on TWG has been based on RRG and utilizes its grammatical categories.

Finally, the fifth chapter describes our approach to large-scale frame-semantic parsing with Tree Rewriting Grammars, using linguistic features extracted from the tree-rewriting grammars. We adopt a compositional approach to map syntax and semantics, similar to that outlined in Kallmeyer and Osswald (2013), and utilize deep learning tools to predict components for our parser.

Keywords: Semantic parsing, supertagging, syntactic parsing, deep learning, Frame Semantics, Tree rewriting grammars, Tree Adjoning Grammar, Tree Wrapping Grammar.

Acknowledgments

Working on this thesis was quite a journey and I want to thank everyone who made it possible!

First and foremost, I would like to thank my advisor Laura Kallmeyer for her guidance throughout my dissertation journey and for being a never ending source of support and energy. Her sense of elegant solutions has shaped the way I see research and will hopefully stay ingrained in me throughout my career. I am extremely grateful for the numerous hours she spent guiding me through my thesis. Laura was the best advisor I could have ever asked for.

Of course I would also like to thank Wiebke Petersen and Kata Balogh, my PhD advisors, for their enthusiasm, their availability, and their expertise, for helping me out with scientific problems, for guiding me towards realistic goals, for telling me when to stop, for all the work they put in this thesis themselves and, last but not least, for having made this experience very enjoyable from start to finish.

I would also like to express my gratitude to the jury members for having accepted to participate in the defense of this thesis.

There are many colleagues who have helped me immeasurably during my dissertation journey. I am extremely grateful to Rainer Osswald, Andreas van Cranenburgh, Younes Samih, Kilian Evang, Jakub Waszczuk, Simon Petitjean, Julia Zinova for their help, for the many hours we have discussing the work in this thesis and our other projects and for all the suggestions they have made. I am very grateful to Anika Westburg for proofreeding my dissertation, our long discussions and for her very useful comments.

I want to thank the colleagues from the Institute of Language Valentin Richard, Deniz Ekin Yavas, Long Chen, David Arps, Christian Wurm, Rafael Ehren, Valeria Generalova for their help and suggestions, for our discussions and conversations in the institutes kitchen and during our Stammtische. I hope we will stay in touch and do some projects together!

During my dissertation journey, I had a research stay in Paris, in the Laboratoire de

Linguistique Formelle of the Université Paris-Cité. My thanks go to Marie Candito, Benoît Crabbé, Djamé Seddah, Benoît Sagot who supported me and gave valuable advice and suggestions for my project during this research stay. I am also grateful to the French PhD students Antoine Simoulin and Vincent Segonne for our inspiring discussions and help with the French administration.

I am grateful for a fruitful collaboration with the researchers from the University of Groningen Gosse Minnema, Rik van Noord, Lasha Abzianidze, Johan Bos, Malvina Nissim, with whom we worked together on several projects.

I also thank Sebastian Löbner, Eva Gentes and all the members and fellow PhD students of the interdisciplinary research training program SToRE of the Collaborative Research Center 991 "The Structure of Representations in Language, Cognition, and Science". Thanks to you I gained insights into various fields of linguistics which sparked the ideas for my future research. I am also thankful to PhilGrad research training program and Selma Meyer Mentorship Program for the great workshops and for giving me many skills I use in my scientific career.

I also want to thank my former chefs from the Fraunhofer Institute Volker Knappertsbusch, Markus Hiebel, and Daniel Maga where I used to work as a student researcher before my dissertation project. You learned me how to write scientific texts and showed me how to work in a scientific environment. I owe to them, to a great extent, my decision of starting a PhD.

I am also very grateful to a number of people I met during these years at summer schools, conferences, and colloquiums, as well as to all the anonymous reviewers of the abstracts and papers this work builds upon.

My dear husband Geoffrey, thank you for always believing in me and for your patience, encouragement and support in all my endeavors. Dear little Sophie, you are so brave and strong, you are my true hero! My dear little Antoine, thank you for learning me to never give up on my goals and to try over and over again! Dear Dad, thank you so much for giving me love, support and freedom and for always believing in me! Dear Mom, you have only witnessed the start of my dissertation journey, but I did always feel your presence and love and still do! I hope you are proud of me looking from up there. Dear brother Aljosha, Ruslana, Auntie Natasha, Uncle Sasha, Marina, Masha, Serjozha, Natasha, Auntie Ljuda, Grandma Zoja, Grandpa Tolja, Grandma Dusja and Grandpa Aljosha - you are a great family! I am also very grateful to my family-in-law - Anne-Marie, Jean-Claude, Gaëlle, Guillaume, Marie and the nephews for your support and love! I am very thankful to Lena&Stefan and Nina&Olli for hosting me during my work trips to Germany. I am also very thankful to our lovely neighbours in Germany and France who always lent a helping hand as

we needed support with guarding our kids.

Last, but not least, I thank the European Research Council, for providing the ERC grant TreeGraSP for funding the TreeGraSP project where I have worked for these seven years. I also thank the German Research Foundation to fund the SFB research center which has given me the necessary skills for scientic work.

Contents

Chapter 1

Introduction and motivation

About this thesis

In this thesis, our aim was to broaden the scope of the rich typologically oriented linguistic framework of Role and Reference Grammar (RRG; Van Valin and LaPolla (1997); Van Valin (2005)) by implementing computational methodologies for syntactic and semantic parsing. RRG is a semantic-oriented theory of natural language grammar widely used in comparative linguistics and linguistic typology. RRG provides an inventory to describe syntax-semantics interface different from constructs in other linguistic theories (such as X-bar syntax or grammatical relations), since it assumes that the conceptualization of phrase structure in these theories is not suitable for a large number of languages across the globe (Van Valin, 2023). RRG is commonly used to describe and study non-European languages, particularly those with limited resources for natural language processing[1]. Due to its ability to handle syntax and semantics together across languages from different language families, RRG shows potential as a valuable framework for addressing research problems in natural language processing. Additionally, we were interested in exploring the possibilities of enhancing machine learning techniques for semantics by integrating additional linguistic structure. More specifically, the aim of our dissertation project was to develop a large-scale semantic parsing methodology based on Tree Rewriting Formalisms such as Tree Adjoining Grammar (TAG; Joshi and Schabes, 1997) and Tree Wrapping Grammar (TWG; Kallmeyer et al., 2013; Kallmeyer, 2016). We aimed to develop a parsing approach that could potentially be extended to a wide variety of languages, especially those with limited resources.

Despite recent advances in syntax-agnostic language modeling based on machine learning, we pursued a syntax-mediated approach to semantic parsing based on

[1]See a list of RRG-based studies on the official RRG homepage https://rrg.caset.buffalo.edu.

principles of compositional semantics because of several important advantages for typologically oriented studies in NLP. For one, syntax-aware approaches to parsing provide more transparency and insights into the decisions of statistical language models. Additionally, syntax-aware semantic models have proven to achieve state-of-the-art performance or sufficiently well performance (Xia et al., 2019; Kasai et al., 2019; Lindemann et al., 2019; Poelman et al., 2022). Another important advantage of grammar-based methods is that they are less data-hungry and are therefore better adaptable to low-resource languages. This factor becomes particularly important when dealing with a typologically oriented language theory like the one we are investigating in our work, Role and Reference Grammar. Lastly, syntax-mediated approaches to semantic parsing can contribute to grammar and semantic theory studies and linguistic investigations in different languages, as the generalizations based on the implemented probabilistic language models can enable comparative studies of syntax-semantics interface in cross-lingual studies, as well as investigations of language-specific phenomena.

Research Questions On the theoretical side, this dissertation is concerned with the syntax-semantics interface in Role and Reference Grammar and semantic composition, which is triggered by tree rewriting process. The primary focus of our research centers around the role of Tree Rewriting Formalisms in implementation of an RRG-based semantic parsing methodology. We aim at investigating several tendencies in the use of syntactic information in recent neural semantic parsing architectures. We hypothesize that despite of the success of syntax-free machine-learning-based methods in semantic parsing, syntactic information is not only useful for overall performance but also crucial for modeling several linguistic phenomena, such as non-local dependencies, control, or raising constructions. On the practical side, this work is concerned with developing a large-scale data-driven semantic parser for multiple languages based on Tree Rewriting Formalisms.

We formulate the main research questions of the dissertation as follows:

(i) Can we propose an algorithm for inducing Tree Rewriting Grammars from an RRG-annotated treebank? Can this grammar extraction algorithm be applied across multiple languages?

(ii) Is it feasible to develop a syntactic parser for RRG based on its formalization as a Tree Wrapping Grammar?

(iii) What strategies are effective in building a large linguistic resource for RRG to support NLP development?

(iv) How can Tree Rewriting Formalisms be used for data-driven RRG-based semantic parsing? Can state-of-the-art semantic parsing results be achieved by

combining Extended Domain of Locality (EDL)-based syntax and semantic frames?

Thesis structure This cumulative thesis comprises eight self-contained research articles, which are divided into four chapters. We describe our method to extract Tree Rewriting Grammars from constituency treebanks in Chapter 2 "Automatic Extraction of Tree Rewriting Grammars". We adapted the algorithm for automatic grammar extraction developed by Xia (1999) for Tree Adjoining Grammars and modified it for the extraction of Tree-Wrapping Grammars. We present our method for syntactic parsing and supertagging with the extracted grammars in Chapter 3 "Supertagging and Parsing with Tree Rewriting Grammars". We followed the parsing algorithm for TAGs proposed by Bangalore and Joshi (1999), which includes two steps: supertagging and the subsequent step of actual parsing. We experimented with different neural architectures for supertagging and adapted the A*-based parser ParTAGe developed by Waszczuk (2017) for the actual parsing step. In Chapter 4 "Building the Role and Reference Grammar Treebanks", we outline our efforts to develop RRG-based constituency treebanks, which we used for grammar extraction and for implementing a syntactic and semantic parser. In Chapter 5 "Syntax-enhanced Semantic Parsing", we describe semantic parsing experiments with a multitask neural architecture and evaluate and discuss the results. In the following Chapter 6 "Discussion and Conclusion", we discuss the results of our research and summarize our findings in different research areas, and also give an outline for future research.

Theoretical Background

Tree Rewriting Grammars Tree Rewriting Grammars[2] (TRGs) are tree-generating systems used to describe the syntax of languages. A TRG consists of a finite set of syntactic tree templates (called *elementary trees*) that can be combined using a

[2]Although the mechanism of generating trees and replacing subtrees with another subtree structures was first formulated in detail in Brainerd (1967), the theory of Tree Rewriting Grammars suitable for natural language description emerged several years later in the early 1970s in the works of Aravind Joshi and his colleagues from the University of Pennsylvania. The formative works "Tree Adjunct Grammars" (Joshi et al., 1975) and "Tree Adjoining Grammars: How Much Context-Sensitivity Is Required to Provide Reasonable Structural Descriptions?" (Joshi, 1985) provided the foundation for the development and formalization of TAG. The most comprehensive overview of TAG, its formal properties, linguistic applications, and its computational aspects were given in work "Tree-adjoining grammars" (Joshi and Schabes, 1997) and in the book "Tree Adjoining Grammars: Formalisms, Linguistic Analyses and Processing" (Abeillé and Rambow, 2000).

limited number of different *tree combination* operations[3], such as *substitution*, *adjunction*, *sister-adjunction*, and *wrapping substitution*. In this thesis, we explore the Tree Adjoining Grammar (TAG) and the Tree Wrapping Grammar (TWG) formalisms. Following the linguistic principles for modeling languages with lexicalized TRGs (Abeillé, 2002; Frank, 2002), elementary trees in these formalisms are associated with a lexical item and represent the span over which the lexical item specifies its syntactic or semantic constraints (Kipper et al., 2000). Elementary trees in the TRGs thus capture the span of the governor and its syntactically necessary arguments, making the argument structure of the predicates explicit. Modifiers (such as adverbial or adjective phrases) are captured in *auxiliary trees*, which are added to the inner nodes of other trees via the adjunction operation. An example of the TAG derivation process for the sentence *Mary absolutely likes pizza* within the classical version of TAG is illustrated in Figure 1.1. The initial trees for *Mary* and *pizza* are substituted at the NP nodes of the *likes* tree, and *absolutely* is adjoined to its VP node (on the left in Figure 1.1). The derived tree (in the middle) shows the result of the combination operations, and the derivation tree on the right shows which operation took place on which nodes in the *likes* tree. The derivation tree (on the right in Figure 1.1) with Gorn addresses (i.e. numerical identifiers of nodes in tree structures proposed by Gorn (1965)) indicates the nodes involved in the tree combination operations.

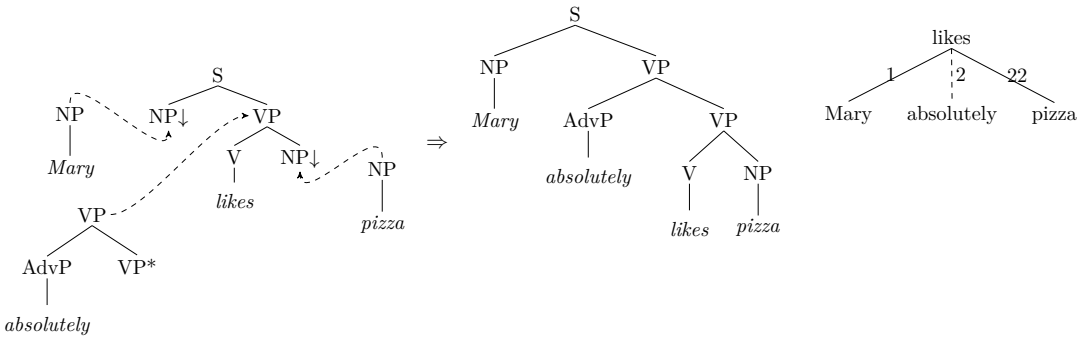


Figure 1.1: TAG elementary trees and a derived tree in LTAG for *Mary absolutely likes pizza*. *Mary*, *likes*, and *pizza* are initial trees, while *absolutely* is an auxiliary tree. The derivation tree on the right includes Gorn addresses, which indicate the target nodes of the tree combinations, while the solid and dotted lines show whether a substitution or adjunction took place.

[3]Hence the name *tree rewriting*, as opposed to *string rewriting* formalisms, such as Context-Free Grammar, or *graph rewriting* formalisms, such as Interaction Nets (Lafont, 1989). Other TRG formalisms include Tree Substitution Grammar (TSG; Rambow et al., 2001), Tree Insertion Grammar (TIG; Schabes and Waters, 1995), and different variants of TAG such as Link-Sharing TAG (LSTAG; Sarkar, 1998), Multicomponent TAG (MCTAG; Kallmeyer, 2005), LTAG-Spinal Liu and Sarkar (2009), Off Spine TAG (osTAG; Swanson et al., 2013), and so on.

Tree Wrapping Grammar (TWG; Kallmeyer et al., 2013; Kallmeyer and Osswald, 2018) is a Tree Rewriting Formalism which is based on TAG principles. TWG was developed as part of formalization of Role and Reference Grammar (RRG; Van Valin and LaPolla, 1997; Van Valin, 2005), a language theory which we discuss in greater detail in the next subsection. Similarly to TAG, TWG grammars consist of elementary trees, which can be combined into larger trees. The tree combination operations in TWG are *substitution*, *sister-adjunction*, and *wrapping substitution* (Kallmeyer et al., 2013). The operation of substitution is defined exactly as the same operation in TAG: the substitution node of a tree X is replaced by the elementary tree which has root node of the same category X. The operation of adjunction is different from the traditional adjunction in TAG. Instead, modifier trees are added via *sister-adjunction* in TWG. Since a modifier can appear on the right or on the left side relative to the position of the constituent head, one distinguishes between *right-* and *left-sister-adjunction*. Auxiliary trees in TWG do not have a foot node, but are marked with an asterisk on the root label (as shown in Figure 1.2), which indicates the node label X in the initial tree to which the auxiliary tree is adjoined. A left-sister-adjoining tree γ can only be adjoined to a node η in the tree τ if the root label of γ is the same as the label of η and the anchor of the elementary tree τ comes in the sentence before the anchor of γ. The children of γ are inserted on the right side of the children in η and become the children of η, as illustrated in Figure 1.2. A *right-sister-adjunction* is defined in a similar way. Figure 1.2 illustrates the operations of substitution and sister-adjunction in TWG.

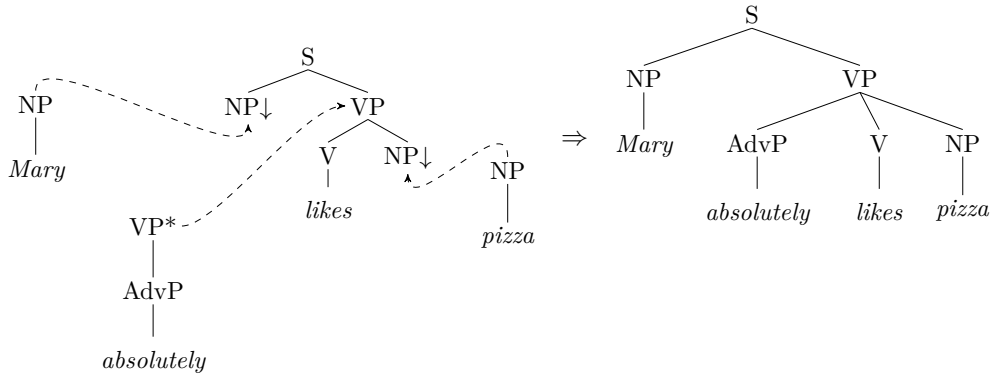


Figure 1.2: Substitution and sister-adjunction in TWG. Compared to the traditional adjunction operation in TAG (see previous Figure 1.1), a sister-adjunction does not introduce additional nodes in the derived tree.

During *wrapping substitution* in TWG, the initial tree y is split at the *d-edge* (dominance link, notated as a dashed edge) and fills a substitution node with the lower part while the upper part is added to the root of the target tree x (see Figure 1.3). The *dominance-edge* or *d-edge* denotes the dominance relation of nodes in a final derived tree (Kallmeyer and Osswald, 2018) (see for example the d-edge elementary

tree for *What you remember* in which the dominance edge is represented with a dashed line). In TWGs designed for natural language modeling, wrapping substitution is used for linguistic phenomena in which an argument is displaced from its canonical position and which cannot be handled by simple substitution or sister-adjunction Kallmeyer et al. (2013); Kallmeyer and Osswald (2018). The subtree for *say* is inserted into the substitution node *CORE*. The upper part of the tree is placed above the root of initial tree for *are you prepared*. The example illustrates how non-local dependencies, here a wh-extraction, across a control construction, can be generated by wrapping substitution from local dependencies in elementary trees.

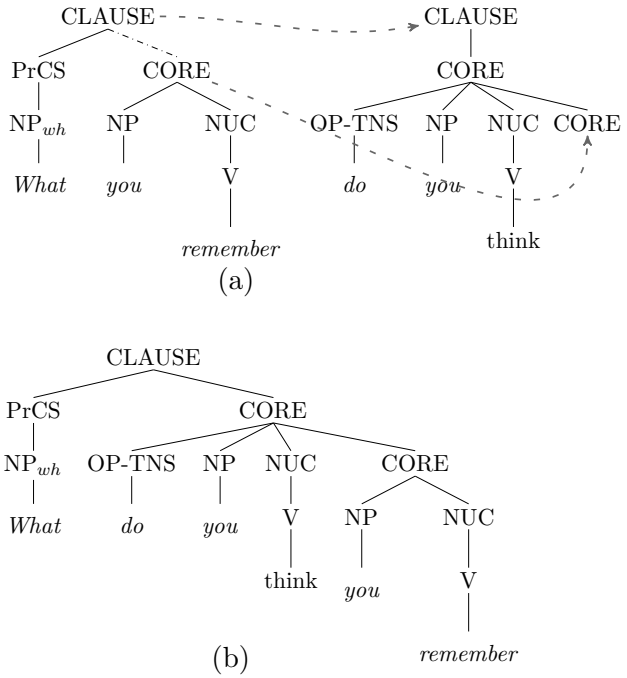


Figure 1.3: Modeling a long distance dependency (wh-movement) in sentence *What do you think you remember?* with wrapping substitution in TWG. Note that wrapping substitution does not introduce extra nodes into the initial tree, as does traditional TAG adjunction in Figure 1.4. Wrapping substitution also simultaneously adds both parts of the non-local dependency to the initial tree.

As with other formal grammars, Tree Rewriting Grammars can find applications in different areas of applied mathematics[4], but they have been mostly used for language modeling in linguistics. The principles to model natural languages with Tree Rewriting Grammars were formulated by Abeillé (2002) and Frank (2002). The first principle postulates that each elementary tree in a TRG must have at least one

[4]Tree Rewriting Formalisms such as TAG were initially developed as a formalism for modeling natural languages, but their applicability has extended to other areas, such as general formal language theory (Rogers, 1999), computational biology (Uemura et al., 1999), and even music composition (Mor et al., 2021).

non-empty lexical item, which is called a *lexical anchor* (elementary trees for multi-word expressions can have more than one lexical anchor). If all elementary trees in a TRG satisfy this condition, the TRG is called a *lexicalized TRG*, or LTRG. This principle reduces parsing time on the computational side by allowing to perform supertagging as a separate step, which reduced the search space considerably. Another important principle for a natural language LTRG is called *elementary tree minimality* (Frank, 1992). It requires that every elementary tree which has a predicate as a lexical anchor must contain slots (i.e., substitution nodes or foot nodes) for all core arguments of this predicate and for nothing else. Non-arguments are realized in auxiliary trees that have one distinguished leaf called a *foot node* which contains a non-terminal and is usually marked with an asterisk. The adjunction operation can also be used to model non-local dependency constructions, such as wh-movement in the sentence *what do you think you remember?*, in which the auxiliary tree for *do you think* "stretches out" the initial tree for *what you remember*, as shown in Figure 1.4 (Kroch, 1989; Frank, 2002). Due to the property of *extended domain of locality (EDL)*[5], the elementary trees in TRGs can collectively represent the lexical and morphosyntactic properties of lexical nodes, which could potentially be distant from each other in the corresponding derived trees. For example, *wh-movement* can be expressed locally in one elementary tree that will be anchored by a verb from which an argument is extracted (see Figure 1.4). Figure 1.4 shows the analysis for a long distance dependency in TAG and Figure 1.3 with TWG.

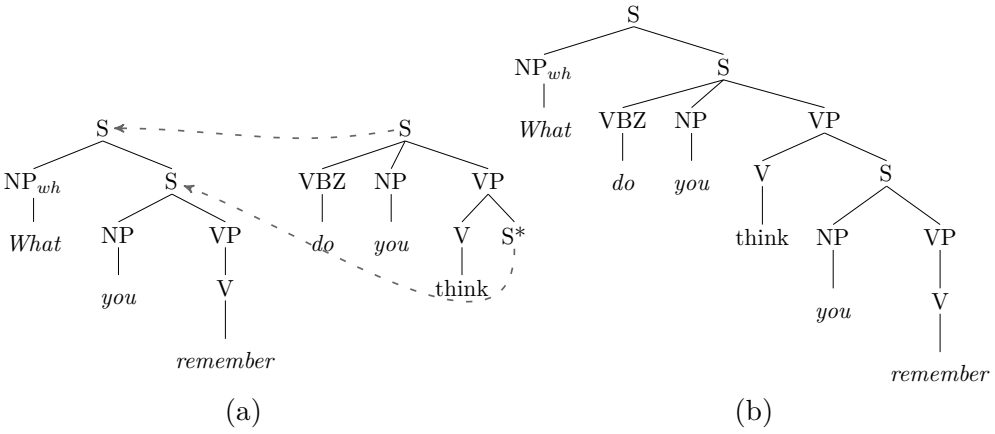


Figure 1.4: Modeling the wh-dependency from sentence *what do you think you remember* with traditional adjunction operation in TAG.

Role and Reference Grammar The starting point of this dissertation is the typologically oriented linguistic theory of Role and Reference Grammar (RRG; Van Valin and Foley, 1980; Van Valin, 2005). RRG started out with the question what a lin-

[5]In the context of grammar formalisms, the *extended domain of locality* refers to a property that allows grammatical rules or constraints to operate beyond the scope of immediately adjacent elements, as required for long-distance dependencies in syntax or non-local agreement patterns.

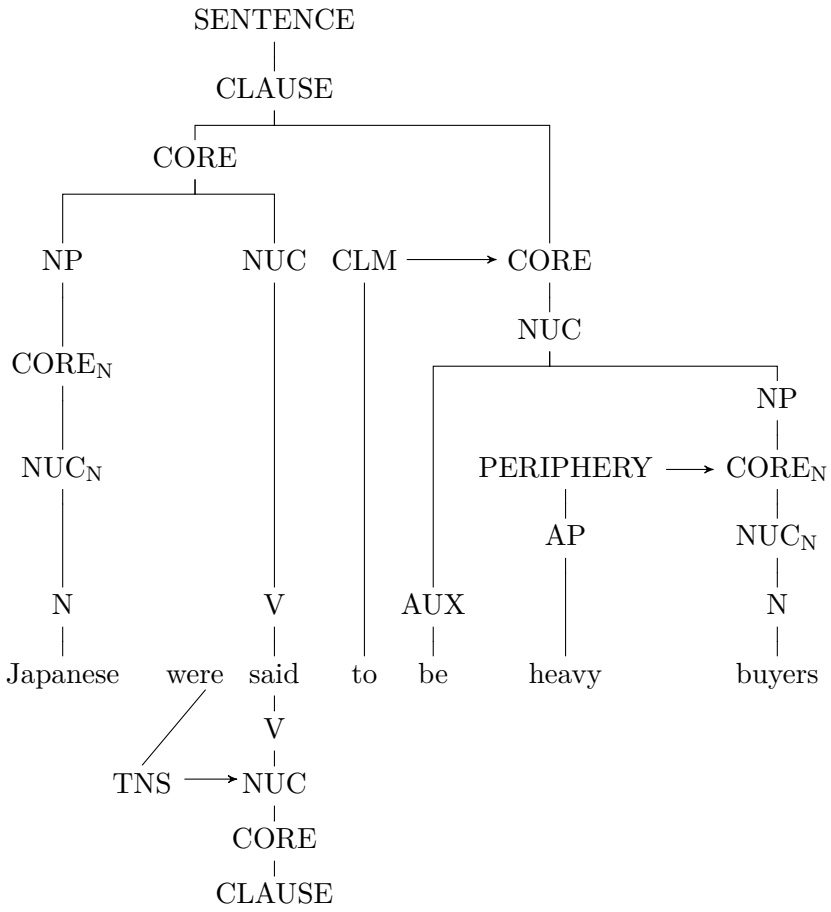


Figure 1.5: RRG-annotated sentence with a constituency projection and a distinct operator projection.

guistic theory would appear if it were based on the description of languages showing varied grammar systems, such as Lakhota, Tagalog, and Dyirbal (Van Valin, 2005). The syntactic structures in RRG are rather flat in order to be applicable to many types of different languages. According to RRG, grammatical relations are not universal, which is why RRG does not use standard formats for representing sentence structure (like X-bar schema), because they might project certain grammatical concepts on languages for which they do not apply (Van Valin, 1993). Instead, RRG assumes sentence structure to be organized in layers motivated by semantic and pragmatic factors. This *layered structure*[6] comprises *nucleus* (containing the predicate), *core* (containing the nucleus and the arguments of the predicate) and *clause* (the core and extracted arguments). Each layer can have modifiers (RRG calls them *periphery elements*) and *operators*, that attach to the layer over which they take semantic scope, as exemplified in Figure 1.5. The set of operators comprises grammatical categories such as tense, aspect or modality. In the representation of the clause structure in RRG, the operators are separated from predicates and ar-

[6]Please refer to Van Valin and LaPolla (1997), Van Valin (2005) and Van Valin (2023) for a more detailed description of *layered structure of the clause* in RRG.

guments and are given in a distinct *operator projection* (see the lower part of the Figure 1.5). RRG also defines a set of *clause linkage markers (CLM)*, which are used for combination of predicator-based units of complex sentences.

RRG assumes an inventory of syntactic templates, which are underspecified templates to build up sentences and which are used to describe the syntactic inventory of every language. For instance, a basic intransitive predicate in English has two placeholders: the nucleus and the singular core argument. Furthermore, due to the Subject-Verb-Object (SVO) structure of English, the single core argument typically precedes the nucleus. Similarly, a basic transitive predicate in English has placeholders for nucleus and two core arguments. Figure 1.6 illustrates two possible core templates for English active declarative sentences. The theory of RRG has been introduced in Van Valin and Foley (1980) and Van Valin and LaPolla (1997), and the most extensive description so far has been provided in (Van Valin, 2005; Van Valin, 2023).

Figure 1.6: Two core templates for English in RRG for an intransitive and a transitive verb without prepositional phrases (Van Valin, 2023).

Formalization of RRG Several approaches for formalization were proposed for RRG: the founders of the theory themselves, Van Valin and LaPolla (1997), suggest two potential strategies for a formalization of the RRG theory. Since RRG structures are basically labeled trees (or can be converted to such), the approaches to formalization of RRG are based on methods developed for formalization of constituency trees. The first approach proposed by Johnson (1987) represents the constituent and the operator projections as two separate context-free grammars which are connected together in a *projection grammar*. The formalization rules for these projections are based on the immediate dominance and linear precedence rules in the RRG structures. This surface-oriented approach is however challenging for a practical implementation, since it does not enforce matching clausal skeletons in both projections. Additionally, assuming the operator projection as a separate tree presents further complexities for cases in which the operators can contribute to multiple layers (Kallmeyer and Osswald, 2023).

The second approach to formalization proposed by Van Valin and LaPolla (1997)

involves decomposition of RRG structures into TAG-like tree templates which can be combined together with tree combination operations. This approach allows for richer encoding of information, facilitating the extraction of sentence semantics, since each template represents a predicate with obligatory syntactic arguments, similar to the elementary trees in TRGs. However, implementing the originally proposed templates in RRG poses challenges in practice, particularly due to the complexity of parsing with templates, where lines can cross and parse trees include detached nodes with modifiers. The works by Kallmeyer et al. (2013); Kallmeyer (2016); Kallmeyer and Osswald (2023) propose a strategy to overcome these challenges by introducing a specialized notation for RRG trees. In this notation original RRG trees are transformed into interconnected trees with crossing branches, and the periphery elements and the operator projection are integrated into the tree. They introduce a TWG formalism, closely related to TAG but accommodating non-local dependencies within trees in a linguistically plausible way. The proposed formalization introduces the concept of the *extended domain of locality*, characteristic of TAG, which proves to be particularly useful for addressing long-distance dependencies resulting from clausal complements in RRG. Moreover, another notion of TRGs, the "factoring recursion from the domain of locality" can be employed to handle instances of multiple coordination in RRG structures. Additionally, since TWG is closely related to TAG, TWG aims to pave the way for using this theory in computational linguistics and for developing NLP applications by adapting the methodologies developed for TAG. Due to the outlined advantages of the formalization approach of Kallmeyer et al. (2013); Kallmeyer (2016); Kallmeyer and Osswald (2023), we decided to investigate it further in the present dissertation.

Nolan (2004) proposes an alternative approach to formalize RRG by exploiting feature-based representations, similar in style to Head-Driven Phrase Structure Grammar (HPSG; Pollard and Sag, 1994). This formalization could potentially be used in various computational applications developed for Lexical Functional Grammar (LFG; Kaplan and Bresnan, 1982). In HPSG, constituent structures are modeled by feature structures, and Nolan (2004) suggests the same for RRG. The disadvantage of this approach is that it requires introducing features and attributes, which can access the subconstituents. This requires a reconstruction of tree structures in RRG by adding feature structures based on formal features (such as FIRST and REST), or by introducing functional notions like SUBJECT, DIRECT-OBJECT etc. Since such configurational syntactic notions do not belong to the basic inventory of the RRG theory, they should rather be avoided in the underlying formalization (Kallmeyer and Osswald, 2023).

Frame Semantics We chose *Frame Semantics* (Fillmore et al., 1976; Barsalou, 1992) as the theory for semantic sentence representations in our implemented parsing sys-

tem. Frame Semantics is a formal linguistic theory of meaning focused on representation of the semantic and conceptual knowledge about a situation. The underlying meaning representations in this theory, *semantic frames*, are contextually sensitive cognitive structures that represent knowledge across diverse concepts, situations, or experiences. They are not fixed templates but rather adaptable structures that adjust to changing contexts and individual experiences. Each frame consists of various coherent elements such as roles, participants, attributes, and relationships, which collectively define characteristic features and functions of the underlying concept. Frame Semantics emphasizes the importance of context in determining meaning instead of focusing on lexems and their meanings in isolation. Linguistic frames are referenced by predicates which describe similar situations (one speaks about *frame occurrences*). The words in sentences "evoke" concepts as well as the perspective from which the situation is viewed. For instance, the word "*sell*" depicts a property transfer from the seller's viewpoint, while "*buy*" describes the same situation from the buyer's perspective. These ideas were implemented within the FrameNet project Fillmore et al. (2003), a large-scale resource available in several languages. FrameNet aims to describe various situations using basic role frames that capture the type of situation and the roles of its participants. For example, the situation of buying or selling goods is captured withing the frame COMMERCE_GOODS-TRANSFER in FrameNet, and involves such *frame elements* as Seller, Buyer, and Goods (those are called *core frame elements*, since they are essential to the meaning of the frame), but also might involve such *non-core frame elements* as Place or Purpose. FrameNet, however, does not address compositional semantics, as the frames are not meant to be combined compositionally. To expand the applicability of frames in general and computational linguistics, several recent studies, among which are those by Petersen (2007); Kallmeyer and Osswald (2012, 2013); Löbner (2014), outlined further formalizations of the frame theory.

Kallmeyer and Osswald (2013) propose a formalization of semantic frames as *base-labelled feature structures with types and relations*. Frames, in their account, are finite relational structures in which attributes correspond to functional relations, meaning that each attribute assigns a unique value to its carrier. A frame can be represented with an attribute-value matrix, as shown in Figure 1.7b. The frames can have a type (e.g., *causation* in Figure 1.7b). They can also be untyped (or have a very general type which can be conjoined with any other frame type), while more than one type is also possible, i.e., types can be combined via conjunction if such combination does not violate an explicitly formulated incompatibility constraint. Feature structures in the account of Kallmeyer and Osswald (2013) also contain base labels, i.e. identifiers which give access to the frame nodes (represented as boxed numbers like 1 in Figure 1.7b). The features (e.g., CAUSE) are depicted as

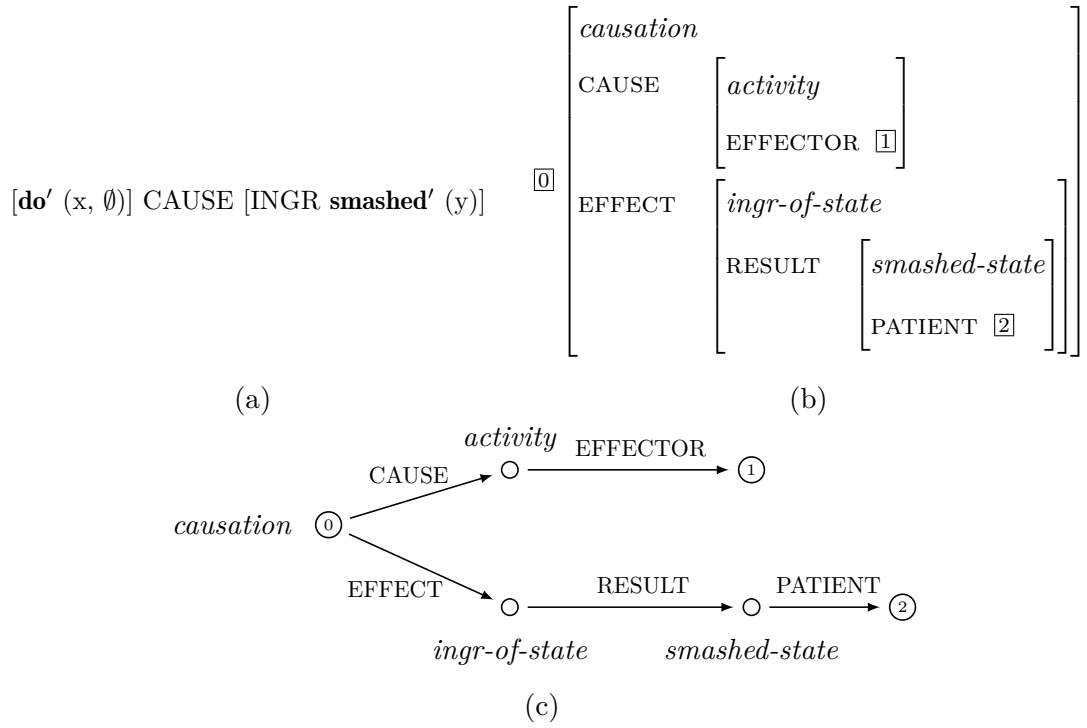


Figure 1.7: Traditional decompositional notation of the causative reading of the verb *'smash'* in RRG in (a) and a corresponding decompositional frame structure as an attribute-value matrix (b) from Kallmeyer and Osswald (2018). The element **do'** in the traditional RRG representation indicates that the effector argument is interpreted as an agent which causes the patient y to undergo an ingressive change of state. The frame in (b) contains all information from the RRG representation, while the agentive role of the effector is encoded in the frame type ACTIVITY. The frame in (b) can also be represented as a graph (c).

a list with their values on the right in this notation. Feature structures can also be represented as a labeled and tagged directed graph, as shown in the example in Figure 1.7c. The nodes in such a graph refer to entities (e.g., individuals, events), the base labels are placed inside the nodes, and types are represented in italics. Edges correspond to (functional or nonfunctional) relations between the entities and are tagged by features in small caps.

Kallmeyer and Osswald (2013) extend the standard definition of feature structures for the formalization of the frames in two regards. First, in addition to features, feature structures in their account can express proper relations between nodes. Second, they introduce a constraint that any frame must have a *functional backbone*. This means that every node has to be accessible via attributes from at least one of the *base nodes*, i.e., nodes that carry *base labels*. Feature structures in their account may have multiple base nodes. In this case, some nodes that are accessible from different base nodes are connected by a relation. Base labels in frames serve as

unique identifiers, meaning that a given base label cannot be assigned to more than one node. Due to the functional backbone requirement, every node of the frame can be accessed from a base label plus a finite sequence of attributes (which can be empty).

Kallmeyer and Osswald (2013) allow the presence of multiple labeled nodes in frames, as long as each node is accessible from at least one labeled node. As a consequence, it is not necessary to identify specific root nodes during frame unification, but the unification process relies on matching nodes with the same label instead. The components within frames, namely event predicates and semantic roles, can be interconnected through explicitly formulated hierarchical and distributional characteristics. This connectivity is supposed to facilitate the computation of textual entailment and semantic similarity and to potentially enable reasoning using frames (Long et al., 2022).

Frame Semantics and RRG Semantics in RRG is traditionally represented as decompositional *logical structures* based on the formalization of Vendler's Aktionsarten (Vendler, 1967) in Dowty et al. (1981). Figure 1.7a illustrates the representation of the causative reading of the verb *'smash'* in the traditional RRG notation[7]. It contains the element *do'* to indicate the activity predicate and also the elements CAUSE and INGR, which were added in Van Valin and LaPolla (1997) and Van Valin (2005) to indicate a causative activity resulting in the ingressive change of state of some underspecified object y, namely that of being *smashed*.

Kallmeyer and Osswald (2018, 2023) adapt the frame-based approach described above to represent semantics in RRG. They propose decompositional frames capable of capturing the same information about the event as logical structures in RRG and can be converted to the traditional RRG notation, as discussed in (Osswald, 2021). The adaptation of frame semantics methodology to RRG by Kallmeyer and Osswald (2018, 2023) allows keeping the key properties of the original decompositional system in RRG without preserving the specific form of the logical structures. An advantage of this approach is that semantic representations can be formalized in terms of types and attribute-value constraints, and semantic composition is reduced to frame unification under constraints. Such formalized frames show good computational properties because their composition relies on the unification of attribute-value structures. Another advantage of this approach is that the theory of feature structures presented in Kallmeyer and Osswald (2013, 2018, 2023) also comes with a well-explored logic and a model-theoretic semantics, which makes their methodology

[7]This methodology of semantic representation in RRG was introduced in (Van Valin and Foley, 1980) and updated with additional elements in (Van Valin and LaPolla, 1997). The most recent comprehensive version of representing semantics in RRG is described in Van Valin (2005) and Van Valin (2023)

well-suited for use in NLP applications.

Before our dissertation project, frames have already been used for semantic parsing (Das et al., 2014; Ringgaard et al., 2017; Swayamdipta, 2017; Kalyanpur et al., 2020; Minnema and Nissim, 2021). However, they have not been employed for a large-scale data-driven approach to semantic parsing in combination with RRG and a syntactic theory based on Tree Rewriting Formalisms with EDL property, which is the main focus of this thesis.

1.1 Automatic Grammar Extraction from Treebanks

Adaptation of the established methods for parsing with TRGs to the formalized RRG theory requires creation of suitable TWG grammars for different languages and tasks. A wide-coverage Tree Rewriting Grammar (which does not rely on a factorization of the elementary tree templates in a metagrammar) for a natural language typically contains up to five-six thousands of elementary tree templates (see for example existing TAG grammars for English (XTAG Research Group, 2001), French (Abeillé et al., 2002), Vietnamese (Nguyen et al., 2006) or Arabic (Habash and Rambow, 2004)). There are two ways to create such linguistically accurate and wide coverage grammars: (1) hand-crafted creation and (2) (semi-)automatic extraction from a manually validated treebank. The hand-crafted grammars are usually built from raw data and follow existing grammar handbooks to create a linguistically plausible language representation. One of the most important manually created grammars is the XTAG project - a large coverage LTAG for English implemented in the 1990s by the XTAG Research Group at the University of Pennsylvania (XTAG Research Group, 2001). This grammar includes over a thousand tree templates covering syntactic contexts for about 317,000 inflected words. Examples of other hand-crafted TRGs include French LTAG (FTAG; Abeillé, 1991; Abeillé et al., 2000), Chinese LTAG (Xia, 2001b), or Old French LTAG (Regnault, 2019), however, these grammars have much smaller coverage.

Although hand-crafted grammars have the advantage of being independent from existing treebanks and thus easier to extend to cover new language phenomena, they also have some drawbacks. The main disadvantage is that building a non-automatically induced TRG for a natural language requires a lot of human effort. Manually crafted grammars also cannot be used for parsing algorithms based on statistics and probabilities, and it becomes difficult to maintain the grammar over time, modify it, and extend it. To make certain changes in the grammar, all the related trees have to be manually checked. This process is inefficient and cannot guarantee consistency (Vijay-Shanker and Schabes, 1992). Additionally, the growing number of large-scale constituency treebanks for different languages (English,

German, French, Old French, Spanish, Vietnamese, Korean, Arabic etc.) has motivated the need for automatic induction of linguistically plausible TRGs from such corpora. This has led to the development of several algorithms for the automatic extraction of TRGs from corpora with syntactically annotated sentences (Xia, 2001a; Chen et al., 2006), traditionally called the treebanks (for example, Penn Treebank (Marcus et al., 1993) or French Treebank (Candito et al., 2010)).

Several algorithms and methods have been proposed for automatic extraction of grammars from treebanks. The extraction algorithms can be divided into rule-based, unsupervised and hybrid approaches. Rule-based approaches involve creation of the hand-crafted heuristics to guide the induction process. Such approaches come closest to a hand-crafted grammar creation and have been broadly used for induction of different grammar types from the linguistic resources (Xia (1999); Chen et al. (2006); Zettlemoyer and Collins (2007); Howell and Bender (2022)). Unsupervised methods let the machine learning algorithms find patterns in syntactically annotated data by applying smoothing techniques for unsupervised probabilistic parsing and decide which information to include in the extracted grammar elements (Headden III et al., 2009; Berg-Kirkpatrick and Klein, 2010; Jin et al., 2018). Hybrid approaches combine syntactic constraints with unsupervised techniques to induce grammars from treebanks while satisfying linguistic constraints or properties (see examples in Boonkwan (2014); Muralidaran et al. (2021); Evang (2019a))

The extraction of TRGs has traditionally been achieved through a rule-based automatic approach Xia et al. (2000); Chen et al. (2006); Liu and Sarkar (2009); Swanson et al. (2013), although some unsupervised methods have also been discussed (Nesson et al. (2006); Cohn et al. (2010)). Xia et al. (2000) and Chen et al. (2006) proposed two different approaches for (semi-)automatic TAG extraction: a top-down and a bottom-up approach.

The top-down TAG induction algorithm proposed by Xia et al. (2000) consists of three steps: (1) adding intermediate nodes to the treebank tree so that at each level, only one of the following relations holds between the head and its siblings: head-argument relation, modification relation, and coordination relation in order to extract elementary trees for traditional TAG adjunction, (2) top-down decomposition of the treebank tree into elementary trees and subsequent filtering out of the invalid elementary trees, and (3) building derivation trees.

On the other hand, Chen et al. (2006) propose a tree-extraction algorithm that is applied recursively and bottom-up. During the application of their method, multiple elementary trees in various stages are generated until the initial tree is completely decomposed into elementary trees and no further options to extract elementary trees are left. Following the bottom-up algorithm, one starts at the leaf node of a tree

and constructs the rest of the tree by extending the trunk of the partial tree by one level, adding complements to the root of the resulting tree as substitution nodes. All elementary trees are extracted in parallel. Since the elementary trees can be in a completed or still incomplete stage of generation, they are called *partial trees*. The full elementary tree is then constructed bottom-up, alternating between two steps: growing the trunk of the partial tree by one level and adding substitution nodes as daughters of the root of the partial tree. Similarly to the top-down extraction procedure, this algorithm also relies on percolation tables, which help to distinguish head nodes from complements and adjuncts for each level of the derived tree.

In this dissertation, we adapted the top-down approach proposed by Xia (1999) for TAG grammar extraction and modified this algorithm to extract TWG grammars for several languages from the multilingual constituency treebank RRGparbank (see paper **(1)** "Automatic Extraction of Tree-Wrapping Grammars for Multiple Languages"). We extended the top-down approach proposed by (Xia et al., 2000) to be able to produce d-edge elementary trees required for the tree-wrapping operation used in Tree Wrapping Grammars. We also modified the extraction procedure for auxiliary trees, since TWGs use sister-adjunction instead of the traditional adjunction operation in TAG.

While extracting TWG grammars from an RRG-annotated treebank, we wanted to study whether our developed algorithm works effectively across various languages. We were also interested in investigating the types of non-local dependencies in different languages and how we can recognize these dependencies within tree structures and extract d-edge trees to use them with TWGs.

Publications

(1) **Bladier, T.**, Kallmeyer, L., Osswald, R., & Waszczuk, J. (2020). Automatic extraction of tree-wrapping grammars for multiple languages. In *Proceedings of the 19th International Workshop on Treebanks and Linguistic Theories* (pp. 55-61).

1.2 Supertagging and Syntactic Parsing

Several methods, both symbolic and statistical, have been proposed for syntactic parsing with Tree Rewriting Grammars (TRGs), including CYK (Vijay-Shanker and Joshi, 1985), Earley (Schabes and Joshi, 1988), and A* (Waszczuk et al., 2016) algorithms. However, TRGs like TAG and TWG, which are the focus of this dissertation, often exhibit high time complexity, making them less suitable for analyzing real-world data within reasonable time frames. To address this challenge and enhance the applicability of TRGs in natural language processing tasks, a pipeline approach

involving *supertagging* has been proposed by Joshi and Srinivas (1994); Bangalore and Joshi (1999). *Supertagging* involves assigning supertags (i.e., enhanced syntactic or semantic labels) to tokens or phrases in sentences. These supertags provide richer information compared to traditional part-of-speech (POS) tags, capturing more nuanced syntactic or semantic properties. For example, the supertags for TRGs are traditionally defined as unanchored elementary trees. Supertagging can be viewed as "almost parsing", as it significantly aids in syntactic disambiguation for the actual TRG parser. Similar pipeline parsing techniques have been employed in other grammar formalisms such as Combinatory Categorial Grammar (Clark, 2002; Lewis and Steedman, 2014; Xu et al., 2015; Vaswani et al., 2016; Yoshikawa et al., 2017) or Head-driven Phrase Structure Grammar (Zhang et al., 2009). In lexicalized grammars like Lexicalized Tree Adjoining Grammar (LTAG) and CCG, supertags are assigned to tokens, which are then used by the parser to construct a parse tree. Supertagging aims to predict a sequence of supertags for each sentence prior to the actual parsing step, thereby constraining the space of possible structures for the parser and facilitating faster parsing. Originally rooted in part-of-speech (POS) tagging, supertagging is essentially a sequence labeling problem, for which various algorithms have been proposed, including the Maximum Entropy model (Ratnaparkhi, 1996; Xu et al., 2015), Conditional Random Fields (CRF; Lafferty et al., 2001), Support Vector Machines (SVM; Cortes and Vapnik, 1995), the perceptron algorithm as used by Collins and Duffy (2002), and neural network models like Bidirectional Long Short-Term Memory (BiLSTM; Hochreiter and Schmidhuber, 1997; Schuster and Paliwal, 1997) and Transformers (Vaswani et al., 2017).

In addition to the supertagging-based pipeline for TRG parsing, recent advancements in neural techniques have also demonstrated their capability to learn TRG-based supertags and the bilexical dependencies directly from data. An example in Figure 1.8 illustrates the tasks learned by a neural TRG parser necessary to produce a full parse, which includes combining predicted supertags into a derived syntactic tree. Thus, a TRG parser must learn the sequence of supertags, the types of combination operations between the supertags, and the bilexical dependencies between the tokens in an utterance. Another example in Figure 1.9 provides concrete details on the input required for a TRG parser developed in this dissertation, as well as the parsing result in bracketed notation. Kasai et al. (2018) propose an end-to-end supertagging approach for TAG employing deep learning. Their method models the entire supertagging process using a BiLSTM neural network, thereby eliminating the need for traditional parsing algorithms and generating supertag sequences and their bilexical dependencies directly from input sentences. However, it should be noted that the work of Kasai et al. (2018) does not address the step of combining predicted supertags to derived TAG trees, thus excluding the full parsing step.

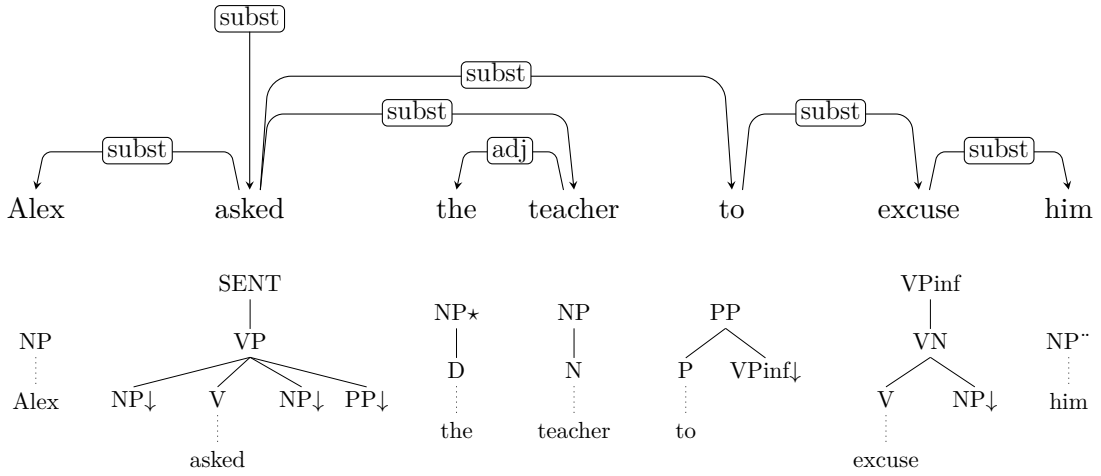


Figure 1.8: Jointly predicted bilexical dependency arcs and dependency labels (upper part of the Figure) and supertags (lower part) required for TRG parsing. Bilexical dependencies in TRG parsing do not represent linguistic categories, rather, they identify the head-dependent relation between the supertags and the type of tree combination operation required.

#	Token	Head	Supertag
1	Alex	2	(NP ⋄)
2	asked	0	(SENT (VP (NP↓) (V ⋄) (NP↓) (PP↓)))
3	the	4	(NP* (D ⋄))
4	teacher	2	(NP (N ⋄))
5	to	2	(PP (P ⋄) (VPinf↓))
6	excuse	5	(VPinf (VN (V ⋄) (NP↓)))
7	him	6	(NP ⋄)

```
(SENT
  (VP
    (NP Alex)
    (V asked)
    (NP (D the) (N teacher))
    (PP (P to)
      (VPinf (VN
               (V excuse)
               (NP him))))))
```

Figure 1.9: Example of predictions made by our TRG parser (in the table on the left) and the resulting derived trees in bracketed notation (on the right).

In this dissertation, we were interested in developing a parsing method for TWG/RRG, especially with low-resource languages in mind, which could greatly benefit from employing transparent syntax-aware parsing approaches. Thus, we decided to implement a traditional supertag-based pipeline approach for parsing. In paper **(3)** "From Partial Neural Graph-Based LTAG Parsing Towards Full Parsing" we combined the probabilistic A*-based parser ParTAGe developed by (Waszczuk, 2017) with a neural supertagger to perform syntactic parsing with the extracted TAG grammars with less computational cost. In paper (**4**) "Statistical Parsing of Tree Wrapping Grammars" we adapted our LTAG for parsing with Tree-Wrapping Grammar. For this, we extended it to handle the operation of tree wrapping in LTWG formalism. Since the quality of the supertagger has been shown to be the bottleneck for the performance of the parser in such a pipeline parsing architecture, we

experimented with different neural supertagging architectures and also investigated if predicting bilexical dependencies, similarly to CCG parsing reported by Yoshikawa et al. (2017), would improve the performance of the parser. In paper **(2)** "German and French Neural Supertagging Experiments for LTAG Parsing", we explored an RNN-based neural architecture for supertagging. We also studied supertagging for German and French to explore if the prediction of supertag sequences would face any language-specific challenges compared to previous work for English resources. Furthermore, we investigated whether improving supertagging results would indeed lead to a better performance in the subsequent parsing step. Given that one of the strengths of TWG lies in modeling non-local dependencies (NLDs), we were also interested in assessing how a TWG-based parser would perform in parsing NLDs compared to other similar parsers.

Publications

(2) **Bladier, T.**, van Cranenburgh, A., Samih, Y., & Kallmeyer, L. (2018). German and French neural supertagging experiments for LTAG parsing. In *Proceedings of ACL 2018, Student Research Workshop* (pp. 59-66).

(3) **Bladier, T.**, Waszczuk, J., Kallmeyer, L., & Janke, J. H. (2019). From partial neural graph-based LTAG parsing towards full parsing. *Computational Linguistics in the Netherlands Journal*, 9, 3-26.

(4) **Bladier, T.**, Waszczuk, J., & Kallmeyer, L. (2020). Statistical parsing of tree wrapping grammars. In *Proceedings of the 28th International Conference on Computational Linguistics* (pp. 6759-6766).

1.3 Treebanking for Role and Reference Grammar

Before we began our dissertation project, there were not sufficiently large annotated linguistic resources for RRG, apart from the corpus created by Chiarcos and Fäth (2019). Chiarcos and Fäth (2019) constructed an RRG corpus through rule-based transformations of data in the Universal Dependencies subcorpus for English (UD; Nivre et al., 2016) and corresponding semantic annotations in PropBank (PB; Palmer et al., 2005). They then used a graph-based parsing algorithm to merge these UD annotations with PropBank, improving the syntactic annotations. While Chiarcos and Fäth (2019) evaluated the conversion algorithm against constituent patterns found in RRG textbooks, they did not manually validate the resulting annotations. Consequently, although their algorithm could potentially convert a large number of English sentences to RRG annotations, these annotations lack the necessary manual linguistic validation to serve as gold standard data. The resulting RRG corpus comprises only a few hundred manually validated English examples,

and further expansion was not pursued. As a result, we needed to develop our own resource containing both syntactic and semantic RRG-based annotations for experimental purposes.

Similar to the approach described by Chiarcos and Fäth (2019) and following the methodology of creating the CCGbank (Hockenmaier and Steedman, 2007) for Combinatory Categorial Grammar, we also initiated our annotation project by converting annotations from the Penn Treebank and UD English corpus to RRG annotations. We developed a semi-automatic approach, augmenting our conversion scripts with additional rules after manually validating the results for further script adaptation. Once we accumulated a sufficient number of annotations, we employed them to train our statistical parser based on RRG/TWG to generate new annotations. We iteratively retrained the statistical parser multiple times as we obtained more manually validated annotations. Throughout our project, we developed two resources: RRGbank (**(5)** "RRGbank: a Role and Reference Grammar Corpus of Syntactic Structures Extracted from the Penn Treebank") and RRGparbank (**(6)** "RRGparbank: A Parallel Role and Reference Grammar Treebank"). RRGbank comprises sentences from the Penn Treebank (Marcus et al., 1993), which we semi-automatically converted into RRG structures using a set of hand-written conversion rules. RRGbank contains only English sentences, while RRGparbank is a parallel treebank consisting of sentences annotated with RRG categories across multiple languages. RRGparbank includes RRG-annotated sentences from George Orwell's novel "1984" and its translations into German, French, and Russian. For semantic annotations, we parsed the sentences using the InVeRo-XL multilingual semantic parser (Conia et al., 2021) based on VerbAtlas (Di Fabio et al., 2019), followed by manual correction of the suggested annotations.

A different route to creating treebanks was taken by the LinGO Redwoods (Oepen et al., 2004) and ParGram (Flickinger et al., 2012; Sulger et al., 2013) approaches to dynamic treebanking for HPSG and LFG, respectively. These projects made use of manually developed grammars and parsers for the grammar formalisms in question, and then manually checked and selected the best output among all possible outputs. We could not follow the same strategy in our dissertation project due to the lack of suitable grammars to train a parser.

While building RRGbank and RRGparbank, we had several purposes in mind. Firstly, it should have been large enough to serve for training of NLP applications based on machine learning. Secondly, we needed a resource which would comprise different linguistic phenomena for corpus-based investigations, potentially disclosing the phenomena not yet covered by the RRG theory and which would thus contribute to the further formalization of RRG. Thirdly, our resource was supposed to facilitate

supervised data-driven approaches to RRG parsing, including grammar induction, probabilistic syntactic and semantic parsing. Lastly, we wanted to create a parallel multilingual resource which would provide new insights into the RRG analyses of the syntax of different languages and which would allow comparison between different languages.

Publications

(5) **Bladier, T.**, van Cranenburgh, A., Evang, K., Kallmeyer, L., Möllemann, R., & Osswald, R. (2018). RRGbank: a role and reference grammar corpus of syntactic structures extracted from the penn treebank. In *17th International Workshop on Treebanks and Linguistic Theories (TLT)*.

(6) **Bladier, T.**, Evang, K., Generalova, V., Ghane, Z., Kallmeyer, L., Möllemann, R., Moors, N., Osswald, R., & Petitjean, S. (2022). RRGparbank: A parallel role and reference grammar treebank. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference* (pp. 4833-4841).

1.4 Semantic Parsing with Tree Rewriting Grammars

Since the 1990s, two main approaches have been proposed for modeling semantics with Tree Rewriting Grammars. The first approach, known as *synchronous TAG* (Shieber and Schabes, 1990), aims to separate the syntactic and semantic contributions of individual lexical items (such as predicates and corresponding argument structures) into parallel syntactic and semantic elementary trees. These trees are linked together through shared nodes, allowing for the simultaneous derivation of both structures. This means that the application of syntactic rules in the derivation process also affects the corresponding semantic structures, ensuring that they remain synchronized throughout (Gardent, 2008). Figure 1.10 provides an illustration of a paired set of elementary trees for the transitive verb *hate*. The tree on the left denotes the source language (English), while the tree on the right shows the target representation, i.e., the corresponding logical form.

The second approach involves Feature Structures Based Tree Adjoining Grammar (Vijay-Shanker and Joshi, 1988) and feature unification. Here, the goal is to map elementary trees to segments of semantic representations, which can be combined through semantic unification as tree combination operations are applied (Kallmeyer and Osswald, 2014). We adopt this approach for semantics with TRGs because it aligns more intuitively with the nature of tree compositionality in TRGs (Kallmeyer and Romero, 2004). Additionally, similar approaches have been developed for other grammar formalisms, such as Combinatory Categorial Grammar (CCG; Steedman, 2000), facilitating transfer and comparison between these formalisms. In this method,

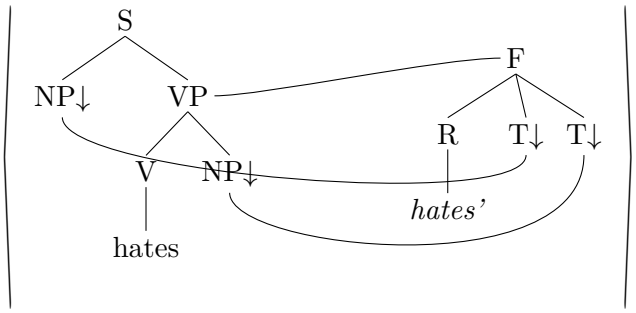


Figure 1.10: Tree pair in synchronous TAG Shieber and Schabes (1990). Thick lines map the linked nodes in source and target representations.

each semantic representation within a tree template is paired with a feature structure. Unifying interface feature structures in the linked frames triggers the unification of feature values across frames as elementary trees are combined during parsing. The concept of combining LTRG elementary trees mapped to individual frames into a unified frame representation is illustrated in Figure 1.11.

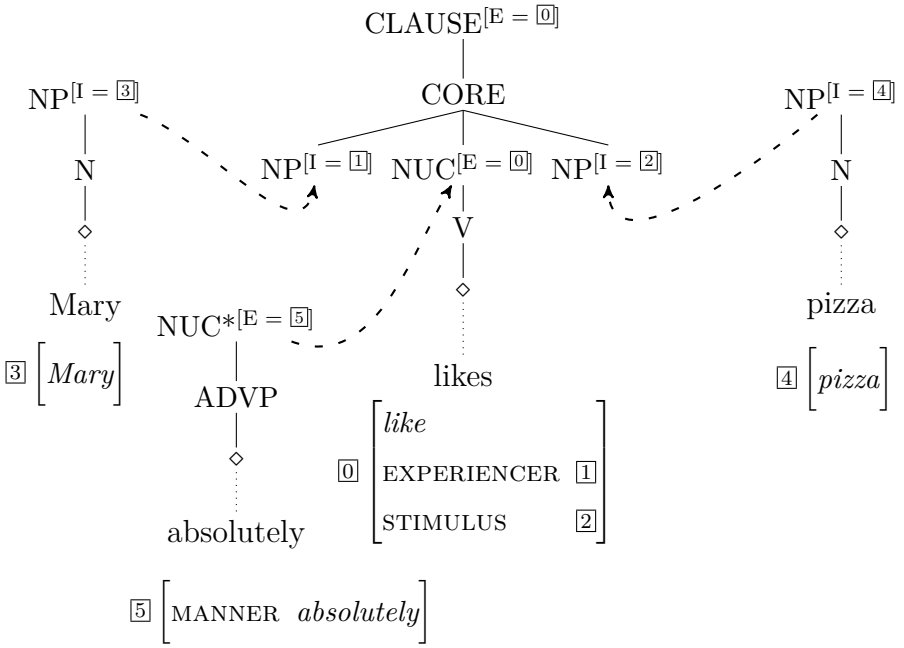
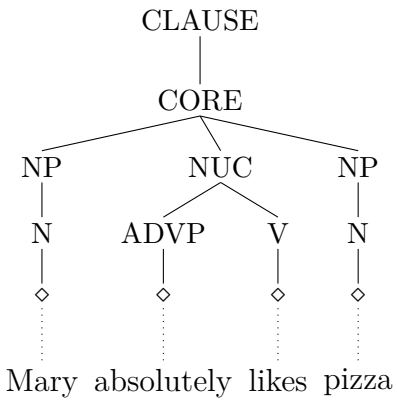


Figure 1.11: Frame-semantic derivation with TWG for *Mary absolutely likes pizza.*

During parsing, as syntactic trees are combined, the associated semantic representations mapped to those elementary trees are also combined. The unification of interface feature structures triggers the unification of feature values within the frames. In our example in Figure 1.11, when the substitution of the subject NP occurs (merging the elementary trees of *likes* and *Mary*), the respective values associated with the attribute I in the interface feature structures are unified. This leads to the unification of the feature structures 3 and 1, resulting in the frame



$$\begin{bmatrix} \textit{like} & \\ \textsc{experiencer} & \textit{Mary} \\ \textsc{stimulus} & \textit{pizza} \\ \textsc{manner} & \textit{absolutely} \end{bmatrix}$$

Figure 1.12: Result of the derivation in Figure 1.11 .

for *Mary* becoming the EXPERIENCER of the event *likes*. A similar process occurs when the tree for *pizza* is substituted at the second NP node of the *likes* tree: 4 and 2 unify, allowing the frame for *pizza* to become the value of the STIMULUS attribute in frame 0 . In this example, the event is modified by a *modifier role* MANNER, which is added to the *like* frame during the adjunction of the adverbial phrase *absolutely* to the *likes* tree. Please note that the frame for *absolutely* does not have a specific type. This is because type specifications are not required in the formal framework developed by Kallmeyer and Osswald (2013), which we are using in this dissertation. Additionally, since the conjunction of frame types is possible, in the case of the modifier *absolutely*, one can implicitly consider the most general frame type that encompasses everything, which is then conjoined to the type of the main frame. The resulting frame before and after the final feature unifications is depicted in Figure 1.13.

$$\begin{bmatrix} \textit{like} & \\ \textsc{experiencer} & \boxed{1} \triangleq \boxed{3}\ \textit{Mary} \\ \textsc{stimulus} & \boxed{2} \triangleq \boxed{4}\ \textit{pizza} \\ \textsc{manner} & \boxed{0} \triangleq \boxed{5}\ \textit{absolutely} \end{bmatrix} \rightsquigarrow \begin{bmatrix} \textit{like} & \\ \textsc{experiencer} & \textit{Mary} \\ \textsc{stimulus} & \textit{pizza} \\ \textsc{manner} & \textit{absolutely} \end{bmatrix}$$

Figure 1.13: Frame-semantic representation for *Mary absolutely likes pizza, before and after the feature unifications.*

The information contained within a frame aligns with the structure provided by the corresponding elementary tree. This means that the specific scope of an elementary tree corresponds to the scope defined by the frame associated with the lexical an-

chor of the tree. To match elementary trees with frames, we adopted a supervised approach, meaning that grammar induction relies on data containing both syntactic structure and frame annotations.

The first large-scale shallow semantic parser with LTAG was implemented by Chen and Rambow (2003). They demonstrated how to predict semantic roles in PropBank by extracting various LTAGs from the PropBank and incorporating elementary tree templates from the grammars as additional features for training a probabilistic dependency-based parser. Liu and Sarkar (2007) proposed utilizing LTAG-based features to enhance the standard features used for semantic role labeling in machine learning approaches, thereby improving the performance of semantic role labeling systems. Their approach involved transforming constituency trees into LTAG derivation trees through rule-based pruning and subsequent decomposition into LTAG elementary trees. These derived trees were then used to extract LTAG-based features for each constituent, which were employed to train a machine learning algorithm. Liu (2009) introduced a novel variant of LTAG known as *LTAG-spinal*, in which elementary trees extracted from the Penn Treebank were combined with PropBank annotations, enabling the integration of syntax and semantic role information in LTAG-spinal supertags. In LTAG-spinal, both initial and auxiliary trees only have a spine and lack substitution nodes. This modification allowed Liu (2009) to maintain a manageable set of features suitable for efficient training of machine learning algorithms, while also making the predicate-argument relationships in PropBank explicit. Another approach to do semantic parsing with LTAG was suggested by Kasai et al. (2019) who extracted traditional LTAG elementary trees from the CoNLL 2009 dataset Hajic et al. (2009) and used elementary tree templates to train several neural network models for semantic role labeling. A different methodology was proposed by Arps and Petitjean (2018) who developed a small-scale purely symbolic deep semantic parser using Tree Adjoining Grammar and a TAG-based metagrammar.

Other semantic parsing approaches, not reliant on LTRGs but similar to our implementation, include those based on CCG. Lewis and Steedman (2013) use a grammar-based strategy that involves CCG-based supertags paired with entries from a distributionally induced lexicon of lambda logical expressions. The choice of correct lexical entries for polysemous words in their system depends on the vector-based predicate clustering. The semantics of the sentence is then constructed compositionally via lambda calculus, guided by the syntactic derivation. A similar approach was used previously by Curran et al. (2007) and Bos (2008, 2015) implemented a CCG-based semantic parser Boxer. This parser produces Discourse Representation Structures (DRSs), semantic representations used in the Discourse Representation Theory (Kamp and Reyle, 1993).

While our dissertation project involves grammar-based feature extraction combined with a machine learning algorithm, our work differs from the approaches to semantic parsing described above. Our methodology is outlined in paper **(8)** titled "Data-Driven Frame-Semantic Parsing with Tree Wrapping Grammar". In our approach, we focused on *deep semantic representations*, which entail structured and logically interpretable meaning representations, as opposed to the *shallow semantic parsing* tasks such as *semantic role labeling* or *word sense disambiguation*. Our objective was to construct a semantic parsing system capable of generating complete, structured, and logically interpretable meaning representations of sentences, facilitating logical inference and reasoning in downstream applications. Our approach to semantic parsing with LTRGs relies on transformers and contextual embeddings, and we do not use a metagrammar. Instead, we adopt a strategy based on supertagging.

Previous research on semantics with LTRGs has presented various approaches for modeling natural language phenomena such as questions (Romero et al., 2004), negations (Banik, 2004), coordination of verbal phrases (Kallmeyer, 2003), relative clauses (Romero and Kallmeyer, 2005), focus (Babko-Malaya, 2004), and universally or existentially quantified sentences (Joshi et al., 2007; Romero, 2002). This demonstrates the suitability of TRGs for comprehensive semantic modeling of natural languages. In this thesis, we developed a large-scale and broad-coverage semantic parsing system that can be expanded to encompass the above-mentioned phenomena, such as incorporating scope information or negation. However, extending the system in this manner is beyond the scope of the current dissertation and is reserved for future investigation.

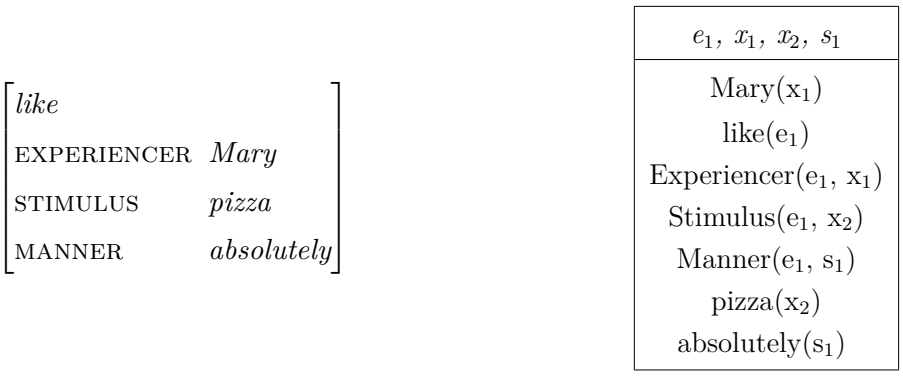


Figure 1.14: Frame-based semantic representation (a) and a DRS (b) for the sentence *Mary absolutely likes pizza*.

Since at the start of our dissertation project, we did not have enough semantically annotated data for parsing experiments, we conducted a semantic parsing experiment based on Discourse Representation Theory (DRT; (Kamp and Reyle, 1993)).

DRT is a formal semantic framework designed to deal with the dynamics of discourse. The basic idea of DRT is that a sentence in natural language discourse must be interpreted in the context provided by the preceding sentences (Kamp and Reyle, 1993). The semantic representations in DRT consist of a set of discourse referents (i.e., participants of the evolving discourse, which can serve as anchors for anaphoric expressions) and a separate set of conditions or statements. Discourse Representation Structures (DRSs) are usually represented in a box notation (see an example of a DRS for the sentence *Mary absolutely likes pizza* in Figure 1.14b and a semantic frame for the same sentence in the form of an attribute-value matrix in Figure 1.14a). The standard large linguistic resource for DRS parsing experiments is the Parallel Meaning Bank (PMB; (Abzianidze et al., 2017)). The DRSs in PMB contain semantic role annotations based on VerbNet (Schuler, 2005), while the predicates are annotated with word senses from WordNet (Fellbaum, 2000). DRS annotations in PMB are complex constructions and contain time points, quantifiers, modal markers, and probability markers. Therefore, in the paper **(7)** "Improving DRS Parsing with Separately Predicted Semantic Roles", we investigated whether semantic parsing with DRSs can be improved by dividing the neural-based parsing approach into two tasks: separately predicting semantic roles and predicates along with predicting the rest of the DRS. In this paper, we experimented with the most recent neural DRS parsers, such as the character-level sequence-to-sequence model by van Noord et al. (2018), an extension of this model that uses linguistic features (van Noord et al., 2019), the BERT-based model by van Noord et al. (2020), and the transition-based parser by Evang (2019b). Since we observed that a multi-task approach was beneficial for most parsing models in our experiment, we then implemented a multi-task neural model in our final LTRG-based semantic parser.

Publications

(7) **Bladier, T.**, Minnema, G., van Noord, R., & Evang, K. (2021). Improving DRS Parsing with Separately Predicted Semantic Roles. In *Proceedings of the ESSLLI 2021 Workshop on Computing Semantics with Types, Frames and Related Structures* (pp. 25-34).

(8) **Bladier, T.**, Kallmeyer, L., & Evang, K. (2023). Data-Driven Frame-Semantic Parsing with Tree Wrapping Grammar. In *Proceedings of the 15th International Conference on Computational Semantics (IWCS)*.

Chapter 2

Automatic Extraction of Tree Rewriting Grammars

Automatic Extraction of Tree-Wrapping Grammars for Multiple Languages

Tatiana Bladier, Laura Kallmeyer, Rainer Osswald, Jakub Waszczuk
Heinrich Heine University Düsseldorf, Germany
{bladier,kallmeyer,osswald,jakub.waszczuk}@phil.hhu.de

Abstract

We present an algorithm for extracting Tree-Wrapping Grammars (TWGs) for multiple languages from constituency treebanks. The TWG formalism, which is inspired by Tree Adjoining Grammar (TAG), has been developed for the formalization of Role and Reference Grammar (RRG). We describe the extraction of TWGs for English, German, French and Russian from the multilingual RRG corpus RRGparbank. A special focus is given to how non-local dependencies are treated by the extraction algorithm. In TWGs, non-local dependencies are considered as arising from local dependencies in elementary trees by the operation of 'wrapping substitution'. The extracted grammars are validated by using them in a subsequent parsing step.

1 Background: Tree-Wrapping Grammars

The Tree Wrapping Grammar (TWG) formalism (Kallmeyer et al., 2013; Kallmeyer, 2016; Osswald and Kallmeyer, 2018) is a tree-rewriting formalism much in the spirit of Tree Adjoining Grammar (TAG) (Joshi and Schabes, 1997) that has been developed for the formalization of Role and Reference Grammar (RRG) (Van Valin, 2005; Van Valin, 2010), a theory of grammar with a strong emphasis on typological concerns. A TWG consists of a finite set of elementary trees which can be combined by the following three operations: a) *(simple) substitution* (replacing a leaf by a new tree), b) *sister adjunction* (adding a new tree as a subtree to an internal node), and c) *wrapping substitution* (splitting the new tree at a d(ominance)-edge, filling a substitution node with the lower part and adding the upper part to the root of the target tree). As in (lexicalized) TAG, the elementary trees of a TWG are assumed to encode the argument projection of their lexical anchors. Figure 1 shows an application of wrapping substitution for generating the German sentence in (1) (the dashed line indicates a d-edge).[1]

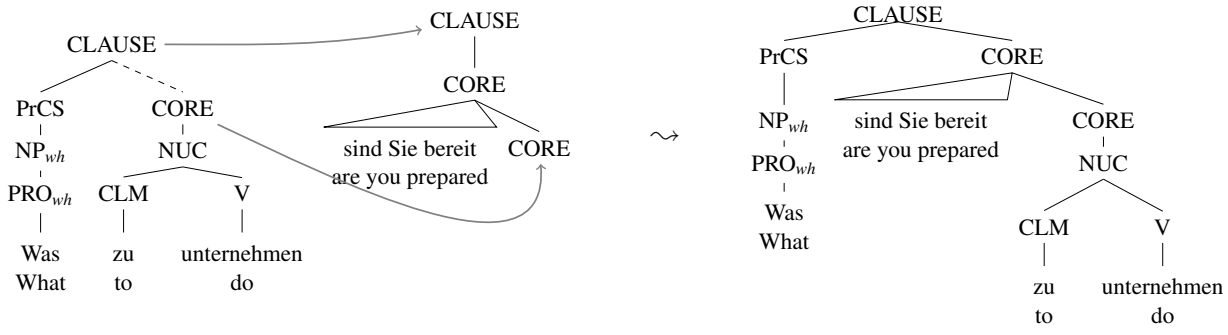


Figure 1: Wrapping substitution for the construction in (1).

(1) *Was sind Sie bereit zu unternehmen ?*
What are you prepared to do ?

[1] Abbreviations: NUC = Nucleus, PrCS = Precore slot. All examples are taken from George Orwell's novel '1984' or its published translations.

The example illustrates how non-local dependencies, here a wh-extraction across a control construction, can be generated by wrapping substitution from local dependencies in elementary trees.

TWG are more powerful than TAG (Kallmeyer, 2016). The reason is that a) TWG allows for more than one wrapping substitution stretching across specific nodes in the derived tree and b) the two target nodes of a wrapping substitution (the substitution node and the root node) need not come from the same elementary tree, which makes wrapping non-local compared to adjunction in TAG. The latter property is in particular important for modeling extraposed relative clauses (see example (3) for a deeper embedded antecedent NP, which requires a non-local wrapping substitution).

In this paper, we adopt a slightly generalized version of wrapping substitution which allows the upper part of the split tree, provided that the upper node of the d-edge is the root, to attach at an inner node of the target tree. For instance, in Figure 1 an additional SENTENCE node above the CLAUSE node in the tree of *bereit* ('prepared') would be possible. A further example for this generalized wrapping will be discussed in Figure 2 below.

By using TWG as a formalization of RRG and applying it to multilingual RRG treebanks, we aim at extracting corpus-based RRG grammars for different languages, thereby obtaining in particular a cross-linguistically valid "core" RRG grammar and, furthermore, providing a cross-lingual proof of concept for TWG in general with respect to its ability to model non-local dependencies. The work presented in this paper is a first step towards these goals.

2 Non-local dependencies in RRGparbank

RRGparbank is part of an ongoing project to create annotated treebanks for RRG (Bladier et al., 2018; Bladier et al., 2019).[2] RRGparbank provides parallel RRG treebanks for multiple languages. At present, RRGparbank contains George Orwell's novel '1984' and its translations in several languages.[3]

RRGparbank provides annotations of non-local dependencies (NLDs) including those given by long-distance wh-extraction (2a), relativization (2b), topicalization (2c), and extraposed relative clauses (2d).

(2) a. *What do you think you remember?*
b. *[. . .] two great problems, which the Party is concerned to solve.*
c. *By such methods it was found possible to bring about an enormous diminution of vocabulary.*
d. *Nothing has happened that you did not foresee.*

In the present context, 'non-local' means that the dependency is not represented within a single elementary tree. We refer to non-local wh-extraction, relativization and topicalization as long-distance dependencies (LDDs).

In RRGparbank, LDDs are annotated in the following way: The fronted phrase node carries a feature PRED-ID whose (numerical) value coincides with the value of the feature NUC-ID of the NUCLEUS the fronted phrase semantically belongs to. For instance, in the annotation of sentence (1), the NP_{wh} node in the tree shown on the right of Figure 1 is marked by [PRED-ID 1] while the NUC node above *unternehmen* is marked by [NUC-ID 1]. See Figure 3 for another example of the annotation convention. In the case of extraposed relative clauses, the relative pronoun and the NP modified by the relative clause both carry the feature REF with identical values (cf. Figure 4).

3 Deriving non-local dependencies by wrapping substitution

Similar to TAG, (simple) substitution in TWG represents the mode of tree composition for expanding argument nodes by the syntactic representations of specific argument realizations, while sister adjunction is mainly used for adding peripheral structures (i.e., modifiers) to syntactic representations. Wrapping substitution, on the other hand, is used for linguistic phenomena in which an argument is displaced from its canonical position and which cannot be handled by simple substitution or sister adjunction

[2] https://rrgparbank.phil.hhu.de/

[3] The data are partly taken from the MULTEXT-East resource (Erjavec, 2012).

(Kallmeyer et al., 2013; Osswald and Kallmeyer, 2018). This holds in particular for the cases of non-local dependencies (NLDs) listed in Section 2. The TWG derivation of LDDs by means of wrapping substitution follows basically the pattern illustrated by the example in Figure 1.

Extraposed relative clauses (ERCs), as in (2d), represent a different type of NLD, namely the extraction of a modifier (the relative clause), typically to a position to the right of the CORE, which leads to a non-local coreference link between the relative clause and its antecedent NP. Example (2d) can be analyzed using wrapping as shown in Figure 2. The extraposed relative clause is associated with a tree that contributes a periphery CLAUSE below a CLAUSE node while requiring that an NP node (which serves to locate the antecedent NP) is substituted into an NP node somewhere below the CLAUSE, modeled by a d-edge between the upper CLAUSE node and a single NP node without daughters. This NP is a substitution node that gets filled with the actual antecedent NP tree. Put differently, the antecedent NP merges with this single NP node, which establishes the link to its modifying relative clause.

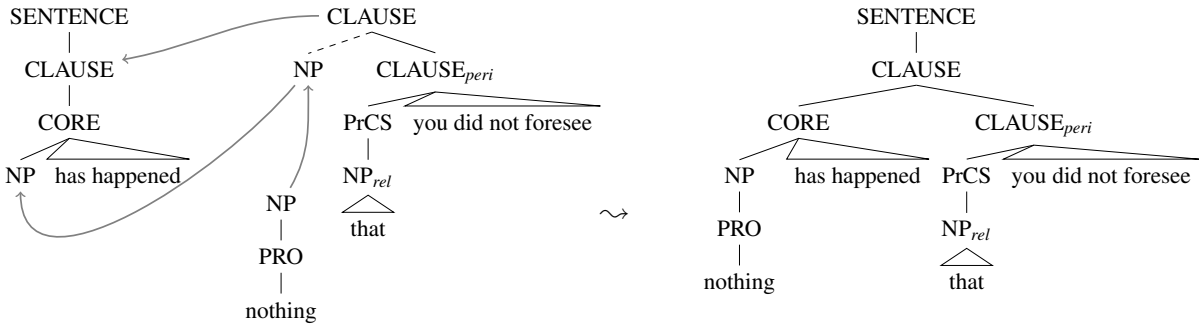


Figure 2: Wrapping substitution for the extraposed relative clause from (2d)

In RRGparbank, we encountered cases where the antecedent NP is further embedded and also cases with more than one relative clause modifying the same antecedent. (3) is an example where we have both: The antecedent NP *Menschen* ('people') is embedded in the direct object NP, and we have two extraposed relative clauses, both modifying the same antecedent.

(3) *Unzählige Male hatte sie [...] [die Hinrichtung von Menschen]$_{NP}$ gefordert , [deren Namen sie nie*
Numerous times had she [...] the execution of people demanded , whose names she never
zu vor gehört hatte] [und an deren angebliche Verbrechen sie nicht im entferntesten glaubte] .
before heard had and in whose alleged crimes she not in the least believed .
'On numerous occasions, she had [...] demanded the execution of people whose names she had never heard before and in whose alleged crimes she did not even remotely believe.'

Another interesting phenomenon is illustrated by the Russian example in (4), which shows both wh-extraction (*čto*) and topicalization (*ja*).

(4) *Ja vot čto xoču skazat'.*
I here what want to.say
'What I'm trying to say is this.'

The current annotation in RRGparbank presumes a scrambling analysis of this topicalization, which gives rise to an RRG tree with crossing branches not generated by sister adjunction. This case is not yet covered by the extraction algorithm presented in Section 4.

4 TWG Extraction

To extract TWGs from treebanks, we adapt the top-down algorithm from (Xia, 1999) for TAG. While substituting and sister-adjoining trees can be extracted following the procedure described in (Xia, 1999), we developed a new algorithm to extract d-edge trees which we describe in more detail below.[4] Since TWGs do not allow for trees to have crossing branches, but the RRG trees often contain them, such edges need to

[4] Additional information on the extraction algorithm can be found in (Bladier et al., 2020).

be removed following a rule-based algorithm for re-attaching certain subtrees in the original tree in a pre-processing step. The process of decrossing tree branches concerns only local re-attaching of peripheral constituents and operator projections and can be reverted applying a rule-based back-transformation algorithm after the parsing step. We extract lexically unanchored elementary tree templates (i.e. *supertags*) for the TWGs. The lexical anchoring happens in the subsequent parsing step.

1. **Decross tree branches.** First, for local discontinuous constituents (for instance NUCs consisting of a verb and a particle in German), we split the constituent into two components (e.g., NUC1 and NUC2), both attached to the mother of the original discontinuous node.
   Second, if a tree τ still has crossing branches, the tree is traversed top-down from left to right and among its subtrees those trees are identified whose root labels contain one of the following strings: OP-, -PERI, -TNS, CDP, or VOC. For each such subtree γ in question with r being its root, we choose the highest node v below the next left[5] sibling of r such that the rightmost leaf dominated by v immediately precedes the leftmost leaf dominated by r. If r and v are not yet siblings, γ is reattached to the parent of v. If the subtree in question has no left siblings, it is reattached to the right in a corresponding way. After this step, it should be checked if the tree τ still contains crossing branches. If yes, the process of decrossing branches is continued by applying the steps above to the next subtree in question.

2. **Extract NLDs.** Then we traverse each tree τ in a top-down left-to-right fashion and check for each subtree of τ whether it contains the following special markings for NLDs in its root label: PREDID=, NUCID= or REF=. The indexes identify the parts of the NLD which belong together. In case of an LDD, the parts of the minimal subtree which contain both parts of the LDD are extracted within a single tree with a *d-edge* (see the multicomponent NUC and CORE in Figure 3). The substitution site and the mother node are added to the remaining subtree in order to mark the nodes on which the wrapping substitution takes place (see Figure 3). A similar process is applied to extract ERCs.

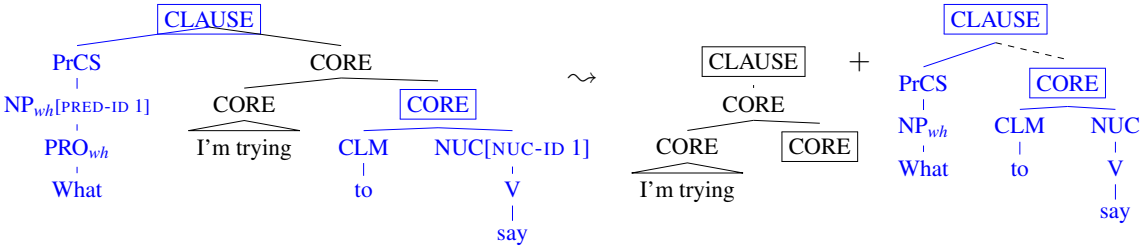


Figure 3: Extraction of tree with a d-edge for an LDD

   The antecedent and the following relative clause (marked with feature REF) are extracted to form a single d-edge tree. The antecedent of the extraposed relative clause is then removed from this d-edge tree and replaced by a substitution slot, as represented in Figure 4.
   After this step, an empty agenda is created and the extracted tree chunks and the pruned tree τ with the remaining nodes are placed into the agenda.

3. **Extract initial and sister-adjoining trees.** If no agenda with tree chunks was created in the previous step, an empty agenda is created in this step and the entire tree τ is placed into it. Each tree chunk in the agenda is traversed and the percolation tables are used to decide for each subtree $\tau_1 \ldots \tau_n$ in the tree chunk whether it is a head, a complement or a modifier with respect to its parent. Initial trees for identified complements and sister-adjoining trees for identified modifiers are extracted recursively in the top-down fashion until each elementary tree has exactly one anchor site.

5 Evaluation of extracted TWGs

We extracted four TWGs for English, German, French, and Russian from the subcorpora of RRGparbank. We used silver and gold annotated data for our experiments, which means that each sentence was

[5] A node v_1 is left to another node v_2 if the leftmost leaf dominated by v_1 is left of the leftmost leaf dominated by v_2.

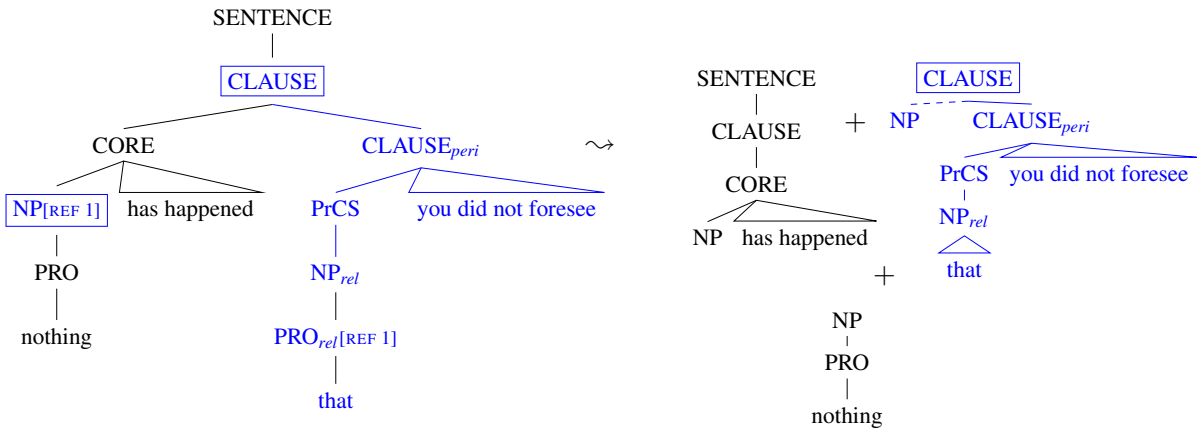


Figure 4: Extraction of a tree with a d-edge for an ERC

annotated and verified manually by at least one linguist. Table 1 provides statistics on the used annotated subcorpora from RRGparbank[6] and the occurrences of non-local dependencies (LDDs and ERCs) in subcorpora. NLDs are generally a relatively rare linguistic phenomenon (Candito and Seddah, 2012; Bouma, 2018). Compared to the other three languages, German shows a fairly large number of ERCs due to its dominant verb-final word order which does not allow putting heavy NPs at the end of the sentence.

Parameters	**English TWG**	**German TWG**	**French TWG**	**Russian TWG**
# word tokens	76893	41324	10550	35975
# word types	7193	7372	2571	9996
Avg. sentence length	14.12	13.5	12.4	10.03
# sentences	5445	3062	851	3586
# LDDs	58	13	36	27
# ERCs	8	110	4	0

Table 1: Statistics on annotated subcorpora in RRGparbank.

The extracted TWGs show a relatively large amount of supertags, more than a half of which occur only once in the corpus. Table 2 shows some statistics on the extracted grammars. The number of supertags with d-edges (which are used for wrapping substitution) is relatively low since the cases of NLDs are not frequent in the data.

Parameters	**English TWG**	**German TWG**	**French TWG**	**Russian TWG**
# supertags	3340	2591	947	2272
# supertags occuring once	1994	1689	584	1503
# initial trees	1727	1490	483	1350
# sister-adjoining trees	1571	1031	431	898
# d-edge trees	42	70	33	22
# nominal supertags	366	299	99	290
# verbal supertags	1382	1164	395	957

Table 2: Statistics on extracted TWG grammars.

We measured the similarity of the extracted TWGs for each language pair. In Table 3 we show the proportions of supertags in one grammar contained in the other grammar[7] (for example, the cell with the row name 'English TWG' and the column name 'German TWG' shows how many supertags from the German TWG are contained in the English grammar). The numbers show that the extracted grammars

[6]The annotation process of the subcorpora in RRGparbank is still in progress and the coverage of annotated sentences differs across the languages. Currently, around 81% of English data, 47% of German, 12% of French, 54% of Russian, and 15% of Farsi sentences are annotated.

[7]Please note that the annotation for different languages in RRGparbank is still in progress, and the proportion of common supertags can change in future.

tend to have a large number of supertags in common. For example, the smallest grammar French TWG (947 supertags) has around 55% supertags in common with the largest grammar for English (3340 supertags). There are 263 supertags common to all four grammars. In future work, we plan to explore the extent to which common supertags in grammars of different languages can be beneficial for multilingual parsing.

Common supertags	**English TWG**	**German TWG**	**French TWG**	**Russian TWG**
English TWG	–	24.97 (834)	15.45 (516)	21.8 (728)
German TWG	32.19 (834)	–	15.51 (402)	24.9 (645)
French TWG	54.49 (516)	42.45 (402)	–	37.80 (358)
Russian TWG	32.04 (728)	28.4 (645)	15.76 (358)	–

Table 3: Ratio of common supertags across language pairs in percents and in numbers (in brackets).

We used the TWG parser ParTAGe (Waszczuk, 2017; Bladier et al., 2020) in a symbolic way in order to validate our grammars and to check that the elementary trees in the extracted TWGs can be combined to produce the original trees.[8] While the majority of sentences could be processed by the parser (see Table 4), some complex sentences which contain an ERC resulting from the free-order placement of predicate arguments as in (4) above could not be parsed. We address these cases in our future work.

	English TWG	**German TWG**	**French TWG**	**Russian TWG**
% exactly matching parses	81	79.07	78.86	80.68
# not parsed sentences	13	8	5	10

Table 4: Validation of extracted TWGs on symbolic parsing with TWG parser ParTAGe.

6 Summary and future work

We presented work in progress on the extraction of TWGs for several languages from the multilingual treebank corpus RRGparbank. TWG is a tree-rewriting system developed for the formalization of Role and Reference Grammar (RRG). TWG is related to TAG and allows, among others, the adequate representation of non-local dependencies (NLDs) in sentences using the operation of wrapping substitution. We showed how wrapping substitution can be used to model various cases of NLDs, including long-distance relativization, long-distance wh-movement, long-distance topicalization, and extraposed relative clauses. We noticed cross-linguistic differences concerning the frequency of NLDs and the corresponding applications of wrapping substitution. At the same time, we observed a considerable overlap of supertags in the TWG grammars extracted for different languages. We validated the extracted grammars using a revised version of the TWG parser ParTAGe.

In future work, we plan to extract larger grammars from the RRG corpora (as the annotation of these projects progresses) and to use them in probabilistic parsing experiments. We also intend to include other languages from RRGparbank into parsing experiments, for example Hungarian and Farsi, depending on the availability of annotated data. Moreover, we will explore how wrapping substitution can be applied to model further linguistic phenomena, such as the variable placement of predicate arguments in languages with a relatively free word order. Finally, we plan to perform multilingual TWG parsing experiments, hopefully benefiting from the considerable number of common supertags across the extracted grammars.

Acknowledgements

We would like to thank three anonymous reviewers for their valuable comments. The work presented in this paper has been partially funded by the European Research Council, within the ERC grant TreeGraSP.

References

Tatiana Bladier, Andreas van Cranenburgh, Kilian Evang, Laura Kallmeyer, Robin Möllemann, and Rainer Oswald. 2018. RRGbank: a Role and Reference Grammar Corpus of Syntactic Structures Extracted from the

[8] For statistical TWG-parsing for English see (Bladier et al., 2020).

Penn Treebank. In *Proceedings of International Workshop on Treebanks and Linguistic Theory TLT17*, Oslo, December.

Tatiana Bladier, Kilian Evang, Laura Kallmeyer, Robin Möllemann, and Rainer Osswald. 2019. Creating RRG treebanks through semi-automatic conversion of annotated corpora. Abstract presented at the International Conference on Role and Reference Grammar 2019, Buffalo, USA, August 19-21, 2019.

Tatiana Bladier, Jakub Waszczuk, and Laura Kallmeyer. 2020. Statistical Parsing of Tree Wrapping Grammars. In *Proceedings of COLING*, December. To appear.

Gosse Bouma. 2018. Corpus-evidence for true long-distance dependencies in Dutch. *Grammar and Corpora*, pages 337–356.

Marie Candito and Djamé Seddah. 2012. Effectively long-distance dependencies in French: Annotation and parsing evaluation.

Tomaž Erjavec. 2012. MULTEXT-East: Morphosyntactic resources for central and eastern european languages. *Language Resources and Evaluation*, 46(1):131–142.

Aravind K Joshi and Yves Schabes. 1997. Tree-adjoining grammars. In *Handbook of formal languages*, pages 69–123. Springer.

Laura Kallmeyer, Rainer Osswald, and Robert D. Van Valin, Jr. 2013. Tree Wrapping for Role and Reference Grammar. In G. Morrill and M.-J. Nederhof, editors, *Formal Grammar 2012/2013*, volume 8036 of *LNCS*, pages 175–190. Springer.

Laura Kallmeyer. 2016. On the mild context-sensitivity of k-Tree Wrapping Grammar. In Annie Foret, Glyn Morrill, Reinhard Muskens, Rainer Osswald, and Sylvain Pogodalla, editors, *Formal Grammar: 20th and 21st International Conferences, FG 2015, Barcelona, Spain, August 2015, Revised Selected Papers. FG 2016, Bozen, Italy, August 2016, Proceedings*, number 9804 in Lecture Notes in Computer Science, pages 77–93, Berlin. Springer.

Rainer Osswald and Laura Kallmeyer. 2018. Towards a formalization of Role and Reference Grammar. In Rolf Kailuweit, Lisann Künkel, and Eva Staudinger, editors, *Applying and Expanding Role and Reference Grammar.*, pages 355–378. Albert-Ludwigs-Universität, Universitätsbibliothek. [NIHIN studies], Freiburg.

Robert D. Van Valin, Jr. 2005. *Exploring the syntax-semantics interface*. Cambridge University Press.

Robert D. Van Valin, Jr. 2010. Role and Reference Grammar as a framework for linguistic analysis. In Bernd Heine and Heiko Narrog, editors, *The Oxford Handbook of Linguistic Analysis*, pages 703–738. Oxford University Press, Oxford.

Jakub Waszczuk. 2017. *Leveraging MWEs in practical TAG parsing: towards the best of the two worlds*. Ph.D. thesis.

Fei Xia. 1999. Extracting tree adjoining grammars from bracketed corpora. In *Proceedings of the 5th Natural Language Processing Pacific Rim Symposium (NLPRS-99)*, pages 398–403.

Chapter 3

Supertagging and Parsing with Tree Rewriting Grammars

German and French Neural Supertagging Experiments for LTAG Parsing

Tatiana Bladier **Andreas van Cranenburgh** **Younes Samih** **Laura Kallmeyer**
Heinrich Heine University of Düsseldorf
Universitätsstraße 1, 40225 Düsseldorf, Germany
{bladier,cranenburgh,samih,kallmeyer}@phil.hhu.de

Abstract

We present ongoing work on data-driven parsing of German and French with Lexicalized Tree Adjoining Grammars. We use a supertagging approach combined with deep learning. We show the challenges of extracting LTAG supertags from the French Treebank, introduce the use of *left-* and *right-sister-adjunction*, present a neural architecture for the supertagger, and report experiments of n-best supertagging for French and German.

1 Introduction

Lexicalized Tree Adjoining Grammar (LTAG; Joshi and Schabes, 1997) is a linguistically motivated grammar formalism. Productions in an LTAG support an extended domain of locality (EDL). This allows them to express linguistic generalizations that are not captured by typical statistical parsers based on context-free grammars or dependency parsing. Each derivation step is triggered by a lexical element and a principled distinction is made between its arguments and modifiers, which is reflected in richer derivations. This has applications in the context of other tasks which can make use of linguistically rich analyses, such as frame semantic parsing or semantic role labeling (Sarkar, 2007). On the other hand, the increased expressiveness of LTAG makes efficient parsing and statistical estimations more challenging.

Previous work (Bangalore and Joshi, 1999; Sarkar, 2007) has shown that the task of parsing with LTAGs can be facilitated through the intermediate step of *supertagging*—a task of assigning possible *supertags* (i.e. elementary trees) for each word in a given sentence (Chen, 2010). Supertagging has been referred to as "almost parsing" (Bangalore and Joshi, 1999), since supertagging performs a large part of the task of syntactic disambiguation and increases the parsing efficiency by lexicalizing syntactic decisions before moving on to the more expensive polynomial parsing algorithm (Sarkar, 2007).

Recently, several papers proposed neural architectures for supertagging with Combinatory Categorial Grammar (CCG; Lewis et al., 2016; Vaswani et al., 2016) and LTAG (Kasai et al., 2017). Supertagging with LTAG is more challenging than with CCG due to a higher number of supertags (counting on average 4000 distinct supertags for LTAGs). Also, almost half of the LTAG supertags occur only once. Nevertheless, the reported neural supertagging approach for LTAG (Kasai et al., 2017) reaches an accuracy of 88-90 % for English (compared to over 95 % for CCG). In this paper we apply a similar recurrent neural architecture to supertagging with LTAGs based on Samih (2017) and Kasai et al. (2017) to German and French data and compare against previously reported results. For the German data, we compare our results to the LTAG supertaggers reported in Bäcker and Harbusch (2002) and Westburg (2016). To our knowledge, no results for French supertagging based on LTAG or CCG have been reported so far.

2 Neural Supertagging with LTAGs

2.1 Lexicalized Tree Adjoining Grammar

A *Tree Adjoining Grammar* (TAG; Joshi and Schabes, 1997) is a linguistically and psychologically motivated tree rewriting formalism (Sarkar, 2007). A TAG consists of a finite set of *elementary trees*, which can be combined to form larger trees via the operations of *substitution* (replacing a leaf node marked with $\downarrow$ with an initial tree) or *adjunction* (replacing an internal node with an auxiliary tree).

Proceedings of ACL 2018, Student Research Workshop, pages 59–66
Melbourne, Australia, July 15 - 20, 2018.

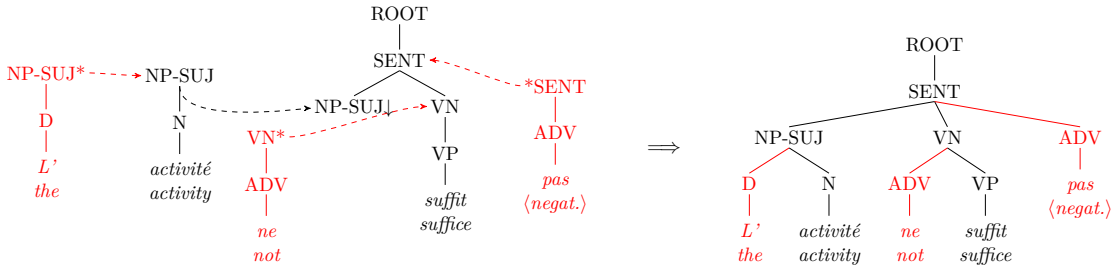


Figure 1: Supertagging with French LTAG for *L'activité ne suffit pas* ("The activity does not suffice")

An auxiliary tree has a *foot node* (marked with $*$) with the same label as the root node. When adjoining an *auxiliary tree* to some node n, the daughter nodes of n become daughters of the foot node. A sample TAG derivation is shown in Figure 2, in which the elementary trees for *Mary* and *pizza* are substituted to the subject and object slots of the *likes* tree and the auxiliary tree for *absolutely* is adjoined at the VP-node.

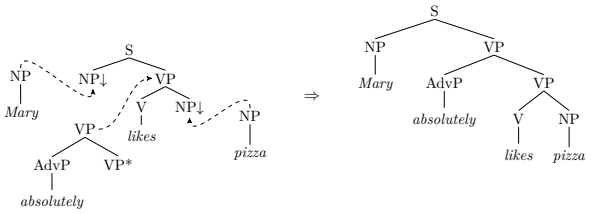


Figure 2: Elementary trees and a derived tree in LTAG

In a lexicalized version of TAG (LTAG) every tree is associated with a lexical item and represents the span over which this item can specify its syntactic or semantic constraints (for example, subject-verb number agreement or semantic roles) capturing also long-distance dependencies between the sentence tokens (Kipper et al., 2000).

2.2 RNN-based TAG supertagging

A supertagger is a partial parsing model which is used to assign a sequence of LTAG elementary trees to the sequence of words in a sentence (Sarkar, 2007). Supertagging can thus be seen as preparation for further syntactic parsing which improves the efficiency of the TAG parser through reducing syntactic lexical ambiguity and sentence complexity. Figure 1 provides an example of supertagging with an LTAG for French.

Several techniques were proposed for supertagging over the years, among which are HMM-based (Bäcker and Harbusch, 2002), n-gram-based (Chen et al., 2002), and Lightweight Dependency Analysis models (Srinivas, 2000). Recent advances show the applicability of recurrent neural networks (RNNs) for supertagging (Lewis et al., 2016; Vaswani et al., 2016; Kasai et al., 2017).

RNN-based supertagging with LTAGs can be seen as a standard sequence labeling task, albeit with a large set of labels (i.e., several thousand classes as supertags). Our deep learning pipeline is shown in Figure 3. A similar architecture showed good results for POS tagging across many languages (Plank et al., 2016).

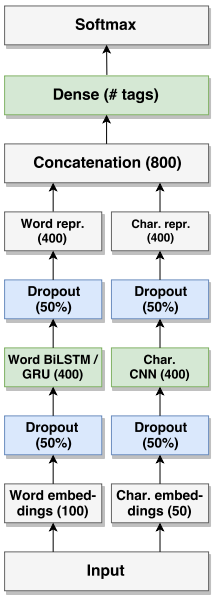


Figure 3: Supertagging architecture based on Samih (2017); dimensions shown in parentheses.

We use two kinds of embeddings: pre-trained word embeddings from the Sketch Engine collection of language models (Jakubíček et al., 2013; Bojanowski et al., 2016), and character embeddings based on the training set data. The pre-trained word embeddings encode distributional information from large corpora. The advantage of the character embeddings is that they can additionally encode subtoken information such as morphological features and help in dealing with unseen words, without doing any feature engineering on

Parameters	**French** (this work)	**German, reduced set** (Kaeshammer, 2012)	**German, full set** (Kaeshammer, 2012)	**English** (Kasai et al., 2017)
Supertags	5145	2516	3426	4727
Supertags occur. once	2693	1123	1562	2165
POS tags	13	53	53	36
Sentences	21550	28879	50000	44168
Avg. sentence length	31.34	17.51	17.71	appr. 20
Accuracy	**78.54**	85.91	**88.51**	**89.32**

Table 1: Supertagging experiments

morphological features.

The embeddings go through a recurrent layer to capture the influence of tokens in the preceding and subsequent context for each token. For the recurrent layer we use either bidirectional Long Short Term Memory (LSTM) or Gated Recurrent Units (GRU). We use a Convolutional Neural Network (CNN) layer for character embeddings, since it was proved to be one of the best options for extracting morphological information from word tokens (Ma and Hovy, 2016). The results for the word and character models are concatenated and fed through a softmax layer that gives a probability distribution for possible supertags. Dropout layers are added to counter overfitting. We replaced words without an entry in the word embeddings with a randomly instantiated vector of the same dimension (100). Table 2 provides an overview of the hyper-parameters we used for the supertagger architecture.

Layer	**Hyper-parameters**	**Value**
Characters CNN	numb. of filters	40
	state size	400
Bi-GRU	state size	400
	initial state	0.0
Words embedding	vector dim.	100
	window size	5
Char. embedding	dimension	50
	batch size	128
Dropout	dropout rate	0.5

Table 2: Hyper-parameters of the supertagger.

3 LTAG induction from the French Treebank

Inducing a grammar from a treebank entails identifying a set of productions that could have produced its parse trees. In the case of LTAG this means decomposing the trees into a sequence of elementary trees, one for each word in the sentence.

In order to extract a TAG from the French Treebank (FTB; Abeillé et al., 2003), we applied the heuristic procedure described by Xia (1999). The main idea of this approach is to consider the trees in the treebank as derived trees from an LTAG. Elementary trees are extracted in top-down fashion using percolation tables to identify grammatically obligatory elements (i.e., complements), grammatically optional elements (i.e., modifiers), as well as a head child for each constituent. All sub-trees corresponding to modifiers and complements are extracted in a further step forming auxiliary trees and initial trees, respectively, while the head child and its lexical anchor are kept in the tree. When extracted in this way, elementary trees contain the corresponding lexical anchor and the branches represent a particular syntactic context of a construction with slots for its complements.

3.1 LTAG induction: pre-processing steps

Before induction of different LTAGs for French, we carried out pre-processing steps described in Candito et al. (2010) and Crabbé and Candito (2008) including extension of the original POS tag set in FTB from 13 to 26 POS tags and undoing multi-word expressions (MWEs) with regular syntactic patterns (e.g. *(MWN (A ancien) (N élève))* → *(NP (AP (A ancien)) (N élève))*). About 14 % of the word tokens (79,466 out of the total of 557,095 tokens) in FTB belong to flat MWEs. After rewriting compounds with regular syntactic patterns, the number of MWEs is reduced to approximately 5 %.

We also restructured some trees in order to bring the complements on a higher level in the tree. In particular, we shifted the initial prepositional phrase of the VPinf constituents to a higher level and raised the subordinating conjunction (*C-S*) of the final clause constituents (*Ssub*) (see Figure 4).

After the preprocessing we extracted the following LTAGs from FTB for our supertagging experiments: including 13 or 26 POS tags, with and without compounds, including and excluding

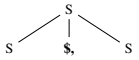
Gold supertag	Predicted supertag	Example
(PP (APPR < >)(NP↓))	(PP* (APPR < >)(NP↓))	**zu** einem Eigenheim zu verhelfen
(NP (DP↓)(NN < >))	(NP (NN < >))	das heutige und künftige **Kreditvolumen**
(S (S*)($, < >)(S↓))	($,* < >)	
(S (NP↓)(VVFIN < >)(NP↓)(PTKVZ↓))	(S (NP↓)(VVFIN < >)(NP↓))	der Umsatzminus **geht** auf 125 Millionen [...] zurück

Table 3: Most common error classes for German TAG supertagging with TiGer treebank

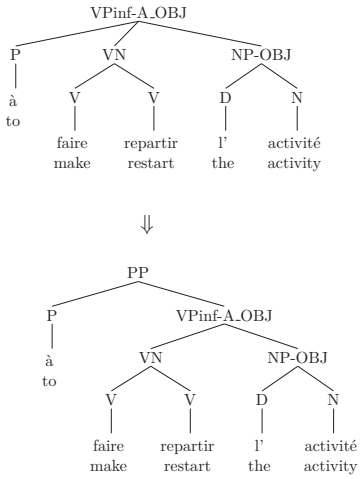


Figure 4: FTB preprocessing: complement raising

punctuation marks. Table 1 provides some statistics on the extracted LTAG which led to the most accurate supertagging results (13 POS tags, without compounds, including punctuation marks).

3.2 Left- and right-sister-adjunction

Extraction of an LTAG from FTB is challenging due to the flat structure of the trees, which allows any combination of arguments and modifiers. In order to preserve the original flat structures in the FTB as far as possible and to facilitate the extraction of the elementary trees we decided against the traditional notion of adjunction in TAG which relies on nested structures and apply *sister-adjunction*; i.e., the root of a sister-adjoining tree can be attached as a daughter of any node of another tree with the same node label.

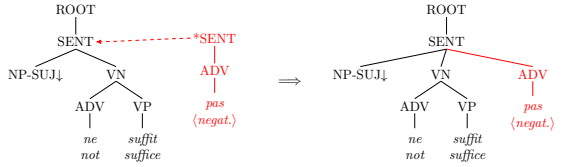


Figure 5: Left-sister-adjunction

Since a modifier can appear on the right or on the left side relative to the position of the constituent head, we distinguish between *right-* and *left-sister-adjoining trees* (marked with * on the left or the right side of the root label as shown in Figure 5).

A left-sister-adjoining tree γ can only be adjoined to a node η in the tree τ if the root label of γ is the same as the label of η and the anchor of the elementary tree τ comes in the sentence before the anchor of γ. The children of γ are inserted on the right side of the children in η and become the children of η. A *right-sister-adjunction* is defined in a similar way.

The resulting LTAGs with sister-adjunction are basically LTIGs (*Lexicalized Tree Insertion Grammar*; Schabes and Waters, 1995) in the way that the auxiliary trees do not allow wrapping adjunction or adjunction on the root node but permit multiple simultaneous adjunction on a single node of initial trees. However, since LTIG is a special variant of LTAG, we refer to the extracted grammar as LTAG in the remainder of the paper.

4 Experiments and error analysis

4.1 Experimental setups for German and French

In order to compare the performance of our supertagger with previous work of Kasai et al. (2017) and LTAG-based supertaggers for German (Bäcker and Harbusch, 2002; Westburg, 2016), we experimented with the supertags extracted by Kaeshammer (2012) from the German TiGer treebank (Brants et al., 2004). The set of supertags for German has the following train, test, and dev. split: 39,925, 5035, and 5040 sentences. We ran a supertagging experiment with this number of sentences, since it is compatible with the experimental setup described in Kasai et al. (2017). Since the number of sentences in FTB is smaller than in TiGer, we created a sample of the train set of the TiGer treebank with a comparable number of sentences in the train set (18,809). For the supertagging experiments with the French LTAG, we di-

vided FTB in the standard train, development and test sets (19,080, 1235, and 1235 sentences), making our test and dev. sets comparable to the dev. and test set reported in Candito et al. (2009).

Tables 3 and 5 show the most frequent erroneous supertags for German and French. The symbol < > in the supertags signifies the spot for the lexical anchor, while * marks the foot node of auxiliary trees and ↓ represents a substitution site.

4.2 German TAG supertagging with TiGer

Generally, results for supertagging with German LTAGs appear to be slightly lower than for English. Westburg (2016) reports an accuracy of 82.92 % for German TAG with a supertagger based on perceptron training algorithm, while Bäcker and Harbusch (2002) reached 78.3 % with a HMM-based TAG supertagger.

Supertagging for German is more challenging than for English due to a higher number of word order variations and the resulting sparseness of the data (Bäcker and Harbusch, 2002). However, our experiments show that the proposed neural supertagging architecture reaches the best performance among the previously described supertaggers for German (88.51 %) and gets comparable results to the supertagging model for English described in Kasai et al. (2017) (see Tables 1 and 4).

System	Accuracy
Bäcker and Harbusch (2002) (HMM-based)	78.3
Westburg (2016)	82.92
This work, full training set (Bi-LSTM)	87.67
This work, full training set (GRU)	**88.51**
This work, reduced training set (Bi-LSTM)	85.26
This work, reduced training set (GRU)	**85.91**

Table 4: Supertagging experiments with German TiGer treebank.

The biggest class of errors for German supertagging contains wrong predictions concerning the type of the elementary tree (e.g. the supertagger predicts an auxiliary tree instead of an initial tree or vice versa). The main reason for this kind of error is the particularity of German which allows dependent elements in a sentence being divided by a big number of other tokens. For example, a determiner and the determined word or the separable verb prefix and the verb stem can be separated by a dozen other tokens, as in the sentence *Der Umsatzminus* ***geht*** *auf 125 Millionen [..]* ***zurück*** (Engl. "The sales drop goes down to 125 millions"), the verb *geht* and its prefix *zurück* are separated by 11 tokens (see Table 3).

Since the window size of tokens presented to the supertagger is limited, the connection between the tokens can be overlooked by the supertagger. However, increasing the window size leads to greater noise in the data. We experimented with window sizes of 5, 9, and 13 for German and got the best results with a window size of 5 (two words before and after the token).

Another source of mistakes for German is the intersentential punctuation in large complex sentences containing several subordinated clauses. This error can also be explained by the window size of tokens presented to the supertagger—the supertagger does not capture the complex structure of the sentence and classifies the punctuation mark as a one-child auxiliary tree (see Table 3).

Another big class of errors comes from PPs which can be either optional (modifiers) or obligatory elements. For example, the supertagger did not recognize that the verb *verhelfen* (Engl. "to help") requires a prepositional phrase as an argument (e.g. *zu einem Eigenheim zu verhelfen*; Engl. "to help someone to buy a property") and erroneously classified this complement as a modifier PP.

4.3 French TAG supertagging with FTB

Supertagging with French LTAGs appears to be more challenging compared to German or English. There are several general reasons for the performance drop of the supertagger, one of which is a higher average sentence length in FTB (31.34 tokens per sentence, compared to 17.51 in TiGer). Sentences in FTB more frequently have a complex syntactic structure including explicative elements separated with brackets or commas.

The large number of supertags lead to higher data sparsity and make the sequence labeling problem more difficult for the supertagger. One explanation for the larger number of supertags, besides the longer and more complex sentence structures in FTB, is the large number of flat multiword expressions in FTB. Our experiments show that rewriting MWEs with regular compounds improves the supertagging performance.

A large number of supertagging errors for French occur due to different sites of attachment of the intersentential punctuation marks in FTB. The punctuation marks in FTB are attached to the corresponding constituents and not consistently to the

Gold supertag	Predicted supertag	Example
(NP* (PP (P <>) (NP↓)))	(PP (P <>) (NP↓))	32 % **par** an
(NP* (PONCT < >))	(SENT* (PONCT <>))	**-LRB-** 66,7 % **-RRB-**
(NP* (N < >))	(N < >)	Mme Dominique **Alduy**
(ROOT (SENT (NP↓) (VN (V < >))))	(VN (V < >))	le droit est officiellement **transgressé**

Table 5: Most common error classes for LTAG supertagging with French Treebank

System	Accuracy
This work (GRU), 13 POS, undone comp.	**78.54**
This work (GRU), 13 POS, no punct. marks	74.44
This work (GRU), 13 POS, with compounds	76.78
This work (GRU), 26 POS, with compounds	74.84
This work (Bi-LSTM), 13 POS, undone comp.	77.67

Table 6: Supertagging experiments with French Treebank (FTB).

root node of the whole sentence. However, since punctuation marks also help to identify possible constituents, omitting them does not improve supertagging.

Similar to supertagging with German LTAGs, PP attachments are also a major source of errors with French LTAGs. In addition to difficulties with classifying PPs as modifiers or complements (as with German data), the supertagger for French more frequently encounters problems with identifying the correct site for attaching the PPs to a node in the syntactic tree. The reason for these errors could be that FTB—in comparison to TiGer—does not offer additional function marks to distinguish PPs as modifiers from prepositional complements of the support verbs.

4.4 N-best supertagging experiments

The softmax layer of the supertagging model we described in section 2.2 provides a distribution of probabilities of the supertags when classifying words in a sentence, and we used this distribution to enable our supertagger to predict n-best supertags.

n-best	Accuracy German (full set)	Accuracy German (red. set)	Accuracy French
1-best	88.51	85.91	78.54
2-best	94.37	93.04	87.34
3-best	96.08	95.00	90.85
5-best	97.45	96.66	94.38
7-best	98.03	97.40	96.00
10-best	98.52	97.97	97.08

Table 7: N-best supertagging experiments.

We experimented with different numbers of n-best supertags for every word, counting the number of accurately predicted supertags each time when at least one of the n-best supertags was predicted correctly. The experiments show a quick growth in accuracy prediction up to 5-best supertags, while for ranks $n > 5$ the improvement of accuracy is not as big (see Table 7).

5 Conclusion and Future Work

We proposed a neural architecture for supertagging with TAG for German and French and carried out experiments to measure the performance of the supertagging model for these languages. We induced several different LTAGs from FTB in order to compare the supertagging performance. The results with German LTAG show that the neural supertagging model achieves comparable results to the state-of-the art TAG supertagging model described in Kasai et al. (2017) for English, even though German is more difficult for supertagging due to the free word order and the data sparseness. Supertagging for French appears to be more difficult due to the larger average length of sentences and a big number of multiword expressions.

In future work we plan to increase performance of the supertagger for French by dividing the supertagging algorithm in two steps: factorization of the extracted supertags in tree families and deciding afterwards on the correct supertag within the predicted tree family. We plan to use the improved supertagger for graph-based parsing. In particular, we aim at adapting the A*-based PARTAGE parser for LTAGs developed by Waszczuk (2017) for parsing with extracted supertags. We also intend to add deep syntactic features and information on semantic roles to the supertags in order to test whether the proposed supertagging architecture can be used for semantic role labeling.

Acknowledgements

This work was carried out as a part of the research project TREEGRASP (http://treegrasp.phil.hhu.de) funded by a Consolidator Grant of the European Research Council (ERC). We thank three anonymous reviewers for their careful reading, valuable suggestions and constructive comments.

References

Anne Abeillé, Lionel Clément, and François Toussenel. 2003. Building a treebank for french. In *Treebanks*, pages 165–187. Springer.

Jens Bäcker and Karin Harbusch. 2002. Hidden markov model-based supertagging in a user-initiative dialogue system. In *Proceedings of TAG+ 6*, pages 269–278.

Srinivas Bangalore and Aravind K Joshi. 1999. Supertagging: An approach to almost parsing. *Computational linguistics*, 25(2):237–265.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2016. Enriching word vectors with subword information. *arXiv preprint arXiv:1607.04606*.

Sabine Brants, Stefanie Dipper, Peter Eisenberg, Silvia Hansen-Schirra, Esther König, Wolfgang Lezius, Christian Rohrer, George Smith, and Hans Uszkoreit. 2004. Tiger: Linguistic interpretation of a german corpus. *Research on language and computation*, 2(4):597–620.

Marie Candito, Benoît Crabbé, and Pascal Denis. 2010. Statistical french dependency parsing: treebank conversion and first results. In *Seventh International Conference on Language Resources and Evaluation-LREC 2010*, pages 1840–1847. European Language Resources Association (ELRA).

Marie Candito, Benoît Crabbé, and Djamé Seddah. 2009. On statistical parsing of french with supervised and semi-supervised strategies. In *Proceedings of the EACL 2009 Workshop on Computational Linguistic Aspects of Grammatical Inference*, CLAGI '09, pages 49–57, Stroudsburg, PA, USA. Association for Computational Linguistics.

John Chen. 2010. Semantic labeling and parsing via tree-adjoining grammars. *Supertagging: Using Complex Lexical Descriptions in Natural Language Processing*, pages 431–448.

John Chen, Srinivas Bangalore, Michael Collins, and Owen Rambow. 2002. Reranking an n-gram supertagger. In *Proceedings of the Sixth International Workshop on Tree Adjoining Grammar and Related Frameworks (TAG+ 6)*, pages 259–268.

Benoit Crabbé and Marie Candito. 2008. Expériences d'analyse syntaxique statistique du français. In *15ème conférence sur le Traitement Automatique des Langues Naturelles-TALN'08*, pages pp–44.

Miloš Jakubíček, Adam Kilgarriff, Vojtěch Kovář, Pavel Rychlỳ, and Vít Suchomel. 2013. The tenten corpus family. In *7th International Corpus Linguistics Conference CL*, pages 125–127.

Aravind K Joshi and Yves Schabes. 1997. Tree-adjoining grammars. In *Handbook of formal languages*, pages 69–123. Springer.

Miriam Kaeshammer. 2012. *A German treebank and lexicon for tree-adjoining grammars*. Ph.D. thesis, MasterâĂŹs thesis, Universitat des Saarlandes, Saarlandes, Germany.

Jungo Kasai, Bob Frank, Tom McCoy, Owen Rambow, and Alexis Nasr. 2017. Tag parsing with neural networks and vector representations of supertags. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1712–1722.

Karin Kipper, Hoa Trang Dang, William Schuler, and Martha Palmer. 2000. Building a class-based verb lexicon using tags. In *Proceedings of the Fifth International Workshop on Tree Adjoining Grammar and Related Frameworks (TAG+ 5)*, pages 147–154.

Mike Lewis, Kenton Lee, and Luke Zettlemoyer. 2016. LSTM CCG parsing. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 221–231.

Xuezhe Ma and Eduard Hovy. 2016. End-to-end sequence labeling via bi-directional lstm-cnns-crf. *arXiv preprint arXiv:1603.01354*.

Barbara Plank, Anders Søgaard, and Yoav Goldberg. 2016. Multilingual part-of-speech tagging with bidirectional long short-term memory models and auxiliary loss. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 412–418, Berlin, Germany. Association for Computational Linguistics.

Younes Samih. 2017. *Dialectal Arabic processing Using Deep Learning*. Ph.D. thesis, Düsseldorf, Germany.

Anoop Sarkar. 2007. Combining supertagging and lexicalized tree-adjoining grammar parsing. *Complexity of Lexical Descriptions and its Relevance to Natural Language Processing: A Supertagging Approach*, page 113.

Yves Schabes and Richard C Waters. 1995. Tree insertion grammar: a cubic-time, parsable formalism that lexicalizes context-free grammar without changing the trees produced. *Computational Linguistics*, 21(4).

Bangalore Srinivas. 2000. A lightweight dependency analyzer for partial parsing. *Natural Language Engineering*, 6(2):113–138.

Ashish Vaswani, Yonatan Bisk, Kenji Sagae, and Ryan Musa. 2016. Supertagging with LSTMs. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 232–237.

Jakub Waszczuk. 2017. *Leveraging MWEs in practical TAG parsing: towards the best of the two world*. Ph.D. thesis.

Anika Westburg. 2016. Supertagging for german - an implementation based on tree adjoining grammars. Master's thesis, Heinrich Heine University of Düsseldorf.

Fei Xia. 1999. Extracting tree adjoining grammars from bracketed corpora. In *Proceedings of the 5th Natural Language Processing Pacific Rim Symposium (NLPRS-99)*, pages 398–403.

Computational Linguistics in the Netherlands Journal 9 (2019) 3–26 Submitted 09/2019; Published 12/2019

From partial neural graph-based LTAG parsing towards full parsing

Tatiana Bladier* BLADIER@PHIL.HHU.DE
Jakub Waszczuk* WASZCZUK@PHIL.HHU.DE
Laura Kallmeyer* KALLMEYER@PHIL.HHU.DE
Jörg Hendrik Janke* JOERG.JANKE@HHU.DE

** Heinrich Heine University of Düsseldorf, Germany*

Abstract

In this paper, we extend recent approaches to Lexicalized Tree Adjoining Grammar (LTAG) parsing that combine supertagging with dependency parsing. In other words, we assign supertags (= unanchored elementary trees) to lexical items and we compute substitution/adjunction arcs between them. Kasai et al. (2017, 2018) jointly predict these structures with a neural graph-based parser. Predicting 1-best supertags and dependency arcs (as in Kasai et al. (2017, 2018)) however leads only to partial parsing due to incompatibilities between elementary trees and derivation trees. We therefore extend the approach described in Kasai et al. (2017, 2018) to n-best supertags and k-best dependency arcs and combine it with a subsequent A*-parsing step that extends the TAG parser from Waszczuk (2017). We show that this architecture allows for efficient full TAG parsing while being sufficiently accurate. We test our architecture on an LTAG extracted from the French Treebank (FTB).

1. Introduction

Lexicalized Tree-Adjoining Grammar (LTAG; Joshi and Schabes (1997)) is a linguistically motivated grammar formalism which supports an *extended domain of locality* (EDL). The building blocks in LTAG – called *elementary trees* – capture a domain of locality which is large enough to cover co-occurrence relationships between a lexical item (the *anchor* of the elementary tree) and the nodes it posits syntactic or semantic constraints on (Carroll et al., 1999). LTAG elementary trees are capable of expressing linguistic generalizations (for example, the *wh*-movement or multi-word-expressions) which are not captured by typical statistical parsers based on context-free grammars or dependency parsing. The linguistically rich analyses of LTAG can be used for the downstream tasks like semantic role labeling or semantic parsing (Liu and Sarkar, 2007; Kallmeyer and Osswald, 2013).

Several parsing approaches have been proposed for LTAG, including symbolic and statistical ones (Joshi and Srinivas, 1994; Bäcker and Harbusch, 2002; Chiang, 2000; Sarkar, 2000; Kallmeyer and Satta, 2009). Recent advances in LTAG parsing include graph-based and transition-based architectures based on Recurrent Neural Networks (RNN) (Kasai et al., 2017, 2018). While many of the developed approaches for LTAG parsing are computationally costly due to the large number of elementary trees per lexical item, previous work (Bangalore and Joshi, 1999; Sarkar, 2007) has shown that LTAG parsing can be facilitated through an intermediate step of *supertagging*. Supertagging is the task of assigning a supertag, i.e., an LTAG elementary tree template, to each word in a given sentence. This step can be seen as "almost parsing", since it performs a considerable amount of syntactic disambiguation before applying a computationally more costly algorithm for actual parsing. For example, linguistic properties of supertags in LTAG grammars proved to enhance the performance of transition-based LTAG parsers (Chung et al., 2016).

Kasai et al. (2017) proposed to extend the supertagging step of LTAG parsing by jointly predicting the supertags and arcs of the LTAG derivation trees using a deep learning architecture. The idea is to not only predict the elementary tree templates but also predict how to combine them (hence, the

dependency arcs) to form a derived tree. Predicting dependency relations along with the supertags should facilitate the subsequent parsing step even further. The authors claim their approach to be sufficient for LTAG parsing. However, in this paper we show that predicting only 1-best supertags and dependency relations between the supertags is not sufficient for the full LTAG parsing step (i.e. step in which the derived trees are predicted) but only allows for partial parsing. The reason is that the system of Kasai et al. (2017) outputs only sequences of the one most probable supertag per word in a sentence along with the single most probable dependency arc for this supertag. The supertags and dependency arcs must be mutually compatible in order to build the derived tree of a sentence, which means that in many cases predicted 1-best supertags and dependency arcs cannot be combined to a complete derived tree.

In the present paper we extend the architecture proposed by Kasai et al. (2018) by combining it with the bottom-up A* parsing system ParTAGe developed by Waszczuk (2017) in order to allow for the full parsing step. Our parsing architecture uses the output from the parser by Kasai et al. (2018) which is fed into ParTAGe in the subsequent step. For this, we changed the architecture developed by Kasai et al. (2018) to predict n-best supertags and k-best dependency arcs (for some arbitrary n and k). This output is then used as the input to the A* parser which computes the most probable combination of supertags and arcs to predict a full derived tree. The original architecture of ParTAGe, based on weighted deduction rules (see e.g. Nederhof (2003)), was changed in order to take n-best supertags and k-best dependency arcs as input, since the parser no longer works with the whole extracted grammar, but has to deal with the supertags and arcs sentence-wise.

Our approach to LTAG parsing takes up on the idea of imposing constraints on the available dependencies between the LTAG elementary trees for a more efficient parsing, similar to the hypothesis stated in Carroll et al. (1999). In section 5 we show that lowering the number of potential dependencies between the LTAG supertags improves the speed for parsing of longer sentences (i.e. sentences with 40 tokens or longer) and reduces the size of the hypergraph created while processing the parsing chart items.

For our experiments, we extract an LTAG for French along the lines of Bladier et al. (2018c) from the French Treebank (FTB; Abeillé et al. (2003)). This grammar contains more than 4500 distinct supertags, half of which appear only once in the corpus. The grammar we use was extracted from 21550 sentences in the current version of the FTB (version 1.0 2016) using the top-down LTAG extraction algorithm proposed by Xia (1999). A peculiarity of this French LTAG is that it only requires *sister-adjunction* and not regular TAG adjunction (i.e., it does not contain TAG's standard auxiliary trees where one of the frontier nodes is marked as a *foot node*). The reason for this decision is the fact that the FTB trees have a flat structure and do not allow to extract regular TAG auxiliary trees.

In this paper we show that sufficiently high numbers n of supertags and k of dependency arcs allow for full parsing for every sentence in the FTB. We also show that our architecture achieves state of the art labeled EVALB F1 score results of 84.36 % on parsing with the test set of the SPMRL (2013) version of the French Treebank (Seddah et al., 2013). The approach to LTAG parsing presented in this paper can be extended to different kinds of LTAGs and other grammars consisting of sets of elementary tree templates, such as for example formalized Role and Reference Grammar (Osswald and Kallmeyer, 2018).

The paper is structured as follows: Section 2 gives a brief overview of the LTAG theory; section 3 provides a general overview of our architecture and explains the architecture proposed by Kasai et al. (2018) and our modifications to it. Section 4 describes in detail the changes we made upon the architecture of ParTAGe (Waszczuk, 2017). We present our experiments and the extracted French LTAG in section 5. Section 6 provides some error analysis, and we conclude with plans for future work in section 8.

2. Lexicalized Tree-Adjoining Grammar

Lexicalized Tree-Adjoining Grammar is a linguistically motivated grammar formalism which supports an *extended domain of locality* (Joshi and Schabes, 1997). A TAG consists of a finite set of elementary trees, which can be combined into larger trees via the operations *substitution* for filling the argument slots and *adjunction* for modifier insertion (see an example in Fig. 1a and 1b). In case of an adjunction an internal node in a tree is replaced with an *auxiliary tree*, which has a special leaf node marked with an asterisk (called *foot node*). When adjoining an auxiliary tree to a node μ, the subtree with root μ in the old tree is put below the foot node of the auxiliary tree in the resulting tree. Non-auxiliary elementary trees in LTAG are called *initial trees*.

Each elementary tree contains (at least) one leaf labeled with a lexical item, its lexical *anchor*. For a given derivation, the derivation tree (see Fig. 1c) describes the way the elementary trees are combined: It contains a node for each elementary tree that has been used and an edge for each substitution or adjunction that has been performed. Edges are labeled with Gorn addresses of the target nodes of the respective operation. Because of the property of *extended domain of locality* in LTAG, it is possible to state linguistic dependencies between nodes which are further apart in the final derived tree. For example, the relation between a topicalized constituent and its governor can be stated locally in a single elementary tree.

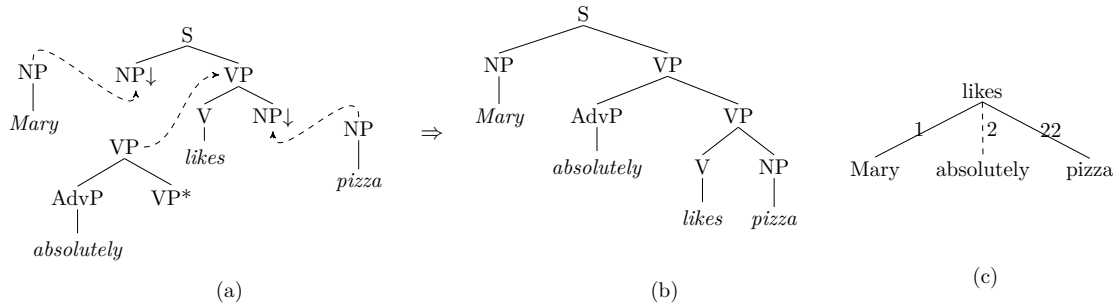


Figure 1: Elementary tree operations (a), a derived tree (b), and a derivation tree (c) in LTAG.

Besides regular adjunction as in Fig. 1a, a simpler type of adjunction for adding modifiers has also been proposed in the literature, namely *sister-adjunction* (Rambow et al., 1995), see Fig. 2 for an example. A modifier tree can be added by sister adjunction if its root category matches the category of the target node. In this case, it adds a new daughter to this node. The result of sister-adjunction are flatter trees, compared to regular adjunction, since no extra nodes are being added to the original tree. Regular adjunction is more powerful concerning generative capacity since it can add two substrings in different places (the spans to the left and the right of the foot node), while sister adjunction adds only one substring. LTAGs using only substitution and sister-adjunction are weakly equivalent to CFGs while LTAG in general can generate a larger class of string languages.

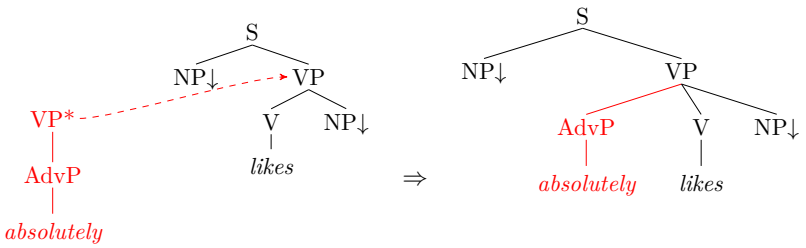


Figure 2: Sister-adjunction operation and the resulting tree. A sister-adjoining elementary tree is marked red.

LTAG grammars for natural languages can be written manually (English XTAG; XTAG Research Group (1998)), also using a metagrammar (French TAG; Crabbé (2005)) or they can be induced automatically from a treebank using top-down (Xia, 1999) or bottom-up (Chen et al., 2002) approaches. An LTAG typically includes several thousands of elementary tree templates, i.e. of unanchored elementary trees (around 4000 on average). About half of them appear only once for the used treebank. The large number of such supertags makes the task of predicting the correct supertag sequence difficult and leads to a large difference in parsing performance with gold and predicted supertags (Chung et al., 2016).

3. LTAG parsing architecture as a pipeline of supertagging, dependency parsing, and A* parsing

LTAG parsing systems based on supertagging consist of a two-step pipeline including a supertagger and a subsequent actual parser (Bangalore and Joshi, 1999; Sarkar, 2007). Supertagging is a *sequence labeling task* which takes as input a sequence of tokens $s_{input} = (w_1, \ldots, w_n)$ for each sentence and outputs a sequence of supertags $s_{output} = (t_1, \ldots, t_n)$ for this sentence. This sequence of supertags is used as input for the subsequent actual parsing step, during which the supertags are combined to a complete derived LTAG tree. In the present paper we use a similar pipeline consisting of the supertagger and dependency parser developed by Kasai et al. (2018) and an A* parser developed by Waszczuk (2017) for the actual parsing step. Fig. 3 shows the pipeline architecture of our parser. Note that we modified both the supertagging architecture provided by Kasai et al. (2018) as well as the A* parser by Waszczuk (2017).

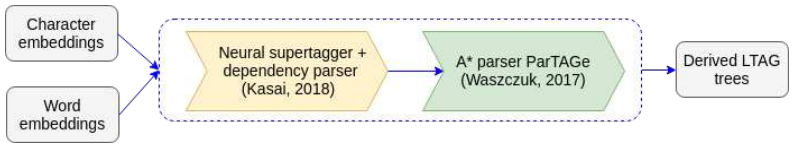


Figure 3: Pipeline neural LTAG parsing architecture.

The term "dependency parser" might be misleading here. The goal of this component is to predict LTAG derivation trees, which formally are dependency structures (see the example in Fig. 1c). It means that the parser does not yield syntactic dependencies in the standard sense but edges for adjunctions and substitutions relating those words whose supertags get combined. Fig. 4 gives an example of what the output of the supertagger and dependency parser should look like.

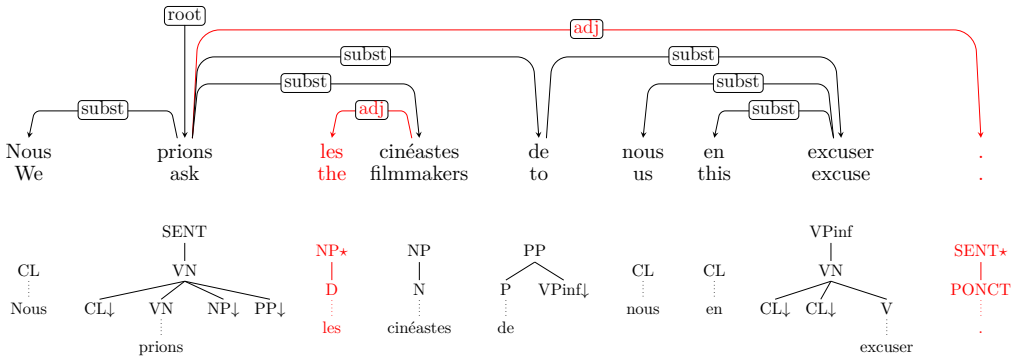


Figure 4: A sentence with supertags and labeled dependency arcs. Auxiliary trees are marked red to distinguish them from initial trees.

Since supertagging is a sequence labeling problem, a neural network is a good choice for such a task, since Recurrent Neural Networks (RNN) with their variants such as Long-Short Term Memory LSTM (Lewis et al., 2016) or Bi-LSTM plus Conditional Random Fields BiLSTM-CRF models (Le and Haralambous, 2019; Ma and Hovy, 2016) have been proven to deliver state of the art performance for sequence labeling tasks in the recent years. The graph-based supertagger and dependency parser developed by Kasai et al. (2018) uses a BiLSTM-based architecture with highway connections between the layers. The highway connections use gating units to regulate the flow of information through the neural network in order to reduce the risk of overfitting and improve the results Srivastava et al. (2015). The neural network model takes as features character embeddings and pre-trained word embeddings and jointly predicts POS tags, dependency arcs, dependency labels, and supertags. The supertagger and dependency parser is based on the graph-based parsing archictecture with deep biaffine attention proposed by Dozat and Manning (2016) (see Fig. 5).

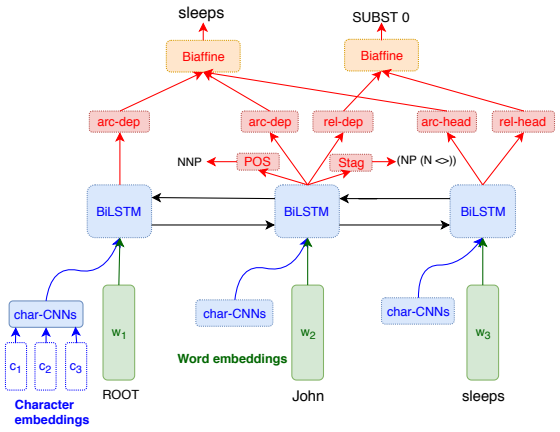


Figure 5: Neural dependency parser for jointly predicting supertags, dependency relations between supertags, POS-tags, and dependency arc labels (Kasai et al., 2018).

Kasai's (2018) architecture predicts 1-best supertag for every word in a sentence and 1-best dependency arc. This is not yet full parsing, since the supertags are not yet combined to a derived tree. The supertags and arcs have to be mutually compatible in order to produce a full derived tree, which is, however, not always the case. The left column in Fig. 6 shows an example of a sequence of predicted supertags which cannot be combined to form a full derived tree, because the supertag for the word *"prions"* (line 2) is lacking the substitution slot for the supertags in lines 5 to 9.

We used Kasai's (2018) parser for our extracted French LTAG and used pre-trained 200-dimensional word embeddings for French (Fauconnier, 2015). We use the n-best output of Kasai's architecture as the input for the ParTAGe parser. Fig. 7 shows a strongly simplified example of an output of Kasai's architecture displaying token number, tokens, dependency arcs with probabilities, and supertags with probabilities (the probabilities are given in float format after the colon). For example, the verb '*eats*' is assigned a dependency arc to the root node (id 0) with probability 1.0 and the two supertags (SENT(NP)(VP(V $\diamond$))) and (SENT(NP)(VP(V $\diamond$)(NP))) with probabilities 0.6 and 0.4 respectively. Among the given n-best supertags and k-best arcs in the input data ParTAGe picks the most probable combination leading to a full derived tree (as represented in the output section in the lower part of Fig. 7).

The information about the dependency arcs on the one hand facilitates the actual parsing leading to a strong decrease in possible parses – as compared to supertagging with no information about the dependency arcs. But on the other hand it can also be a source of parsing errors. For example, it can be the case that all supertags are predicted correctly, but cannot be combined due to erroneously

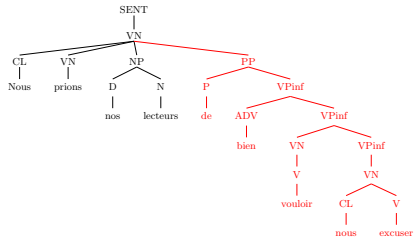
		1-best output (Kasai's (2018) parser)		**A* parser output (10 best)**	
line	token	arc	LTAG supertag	arc	LTAG supertag
1	Nous	2	(CL ⋄)	2	(CL ⋄)
2	prions	0	(SENT (VN (CL) (V ⋄)) (NP))	0	(SENT (VN (CL) (V ⋄)) (NP) (PP))
3	nos	4	(NP* (D ⋄))	4	(NP* (D ⋄))
4	lecteurs	2	(NP (N ⋄))	2	(NP (N ⋄))
5	de	2	(PP (P ⋄) (VPinf))	2	(PP (P ⋄) (VPinf))
6	bien	7	(VPinf* (ADV ⋄))	7	(VPinf* (ADV ⋄))
7	vouloir	5	(VPinf (VN (V ⋄)) (VPinf))	5	(VPinf (VN (V ⋄)) (VPinf))
8	nous	9	(CL ⋄)	9	(CL ⋄)
9	excuser	7	(VPinf (VN (CL) (V ⋄)))	7	(VPinf (VN (CL) (V ⋄)))
		NO PARSE			

Figure 6: The left column shows the output from Kasai et al. (2018) architecture, and the right column provides the output from our model for the sentence *We ask our readers to kindly forgive us*. The 1-best output does not guarantee a full parse. Due to an error in supertag prediction in the highlighted line 2, the supertags in lines 5 to 9 do not have an attachment site and cannot form a full derived tree. Our parsing model solves this problem by using 10-best supertags and arcs.

predicted dependency arcs. In section 5 we show that for French LTAG predicting only 1-best supertag and 1-best arc leads to a fully derived tree in only around 50 % of sentences.

Input

```
1    John    2:1.0          (NP (N <>)):1.0
2    eats    0:1.0          (SENT (NP) (VP (V <>))):0.6
                            (SENT (NP) (VP (V <>) (NP))):0.4
3    an      4:0.5|1:0.5    (NP* (D <>)):1.0
4    apple   2:0.5|0:0.5    (NP (N <>)):1.0
```

Output

```
1    John    2              (NP (N <>))
2    eats    0              (SENT (NP) (VP (V <>) (NP)))
3    an      4              (NP* (D <>))
4    apple   2              (NP (N <>))

(SENT (NP (N John)) (VP (V eats) (NP (D an) (N apple))))
```

Figure 7: Input and output (simplified) of the ParTAGe parser.

We changed the architecture of Kasai et al. (2018) by making it predict n-best supertags and k-best dependency arcs ($n, k \geq 1$) and modified the A* parsing architecture developed by Waszczuk (2017) to combine the supertags according to information about dependency arcs to produce a full derived tree. We experimented with several values for n and k, and it turned out that $n = 10$ and $k = 10$ were the best choices on the development data in order to produce full parses for all sentences (see a summary of our experiments in Fig. 8).

n-best arcs \ k-best stags (FTB, dev set)	**1**	**2**	**4**	**6**	**8**	**10**
1	968	696	509	448	410	385
5	841	296	53	14	5	2
10	838	276	38	7	2	0

Figure 8: Number of sentences in FTB-dev with `no parse` as a function of the number of `n`-best supertags (columns) and `k`-best arcs (rows). The values `n` = 10 and `k` = 10 are sufficiently high to obtain full parses for every sentence in the dev set of the French Treebank. These values can vary depending on the used treebank and the size of the extracted LTAG grammar.

We describe the A* parsing step and the changes which have been made upon the original ParTAGe architecture as described in Waszczuk (2017) in more detail in the next section.

4. A* TAG Parser

Our point of departure is ParTAGe, an A* bottom-up, left-to-right LTAG parser introduced in Waszczuk et al. (2016b); Waszczuk (2017). The parser relies on the A* algorithm, which allows to find a most probable derivation – based on the underlying set of *weighted deduction rules* (Shieber et al., 1995; Nederhof, 2003) – without exploring the entire parsing chart/hypergraph (Klein and Manning, 2001). This is possible provided that an appropriate *heuristic* function is defined, which serves to estimate the cost of parsing the remaining part of the input sentence at any given point of the parsing process. The use of the heuristic and the A* algorithm significantly improves the parser's efficiency in comparison with the corresponding, purely symbolic parser, which requires creating the entire parsing hypergraph before a most probable derivation can be extracted.

ParTAGe provides support for weighted TAGs, i.e., TAGs where a non-negative weight is assigned to each elementary tree (ET). Such a weighting scheme is suboptimal in that it does not allow to benefit from the potential knowledge about the *bilexical dependencies* (i.e. asymmetrical syntactic relations between head tokens and their dependents). The early formalization of probabilistic TAGs admits their importance – each composition of two (lexicalized) ETs, be it via adjunction or substitution, incurs the corresponding probability cost (Resnik, 1992). In the context of CCGs, extending the supertagging-based architecture of Lewis and Steedman (2014); Lewis et al. (2016) with bilexical probabilities leads to significant improvements in terms of the parsing results (Yoshikawa et al., 2017).

In this work, we extend ParTAGe[1] so as to account for the bilexical affinities within the context of lexicalized TAGs, which allows to apply it to the output of the neural LTAG parser (Kasai et al., 2018).

This section is structured as follows. Sec. 4.1 and Sec. 4.2 describe the properties of the TAG grammar required by the revised parser. In Sec. 4.3, the link between the output of the neural parser and the A* parser's input is established. The subsequent sections describe the internals of the parser. In particular, in Sec. 4.12 the parser's deduction rules are specified, and in Sec. 4.13 the A* heuristic tailored to the adopted grammar representation is defined.

4.1 Grammar restrictions

In the new version of ParTAGe, we adopt additional restrictions on the form of the grammar:

- The grammar must be lexicalized, i.e., each ET must contain some terminals in its leaves.
- More specifically, each ET has to have exactly one terminal. Given an ET t, we denote this terminal as $term(t)$.

1. The code of the revised parser can be found at https://github.com/kawu/partage.

The point of this limitation is to adapt the parser to handle dependency relations which hold between terminals (tokens) in the input sentence. In this parsing architecture, tokens and terminals are synonymous and interchangeable.

4.2 Weighting scheme

In the extended version of ParTAGe, both ETs and dependency links are weighted. More precisely:

- A non-negative weight $\omega(t) \in \mathbb{R}_{\geq 0}$ is assigned to each ET t in the grammar.
- A non-negative weight $\omega(x, y) \in \mathbb{R}_{\geq 0}$ is assigned to each pair x, y of grammar terminals, which represents the cost of making y the head of x. This cost applies each time substitution, adjunction, or sister adjunction is performed.

The weight of a particular derivation is defined as the sum of the weights of the participating ETs, as in Waszczuk (2017), plus the sum of the weights of the arcs in the derivation tree, i.e., the weights incurred by the ET combinations. Derivations with lower weights are preferable to those with higher weights. Parsing thus boils down to finding a lowest-weight derivation among all the derivations that can be constructed based on the underlying grammar.

4.3 Interface with supertagging and dependency parsing

As described in Sec. 3, the steps preceding A* parsing include (neural) supertagging and dependency parsing. This means that, for each word in the input sentence, the probability distributions of (i) the potential supertags, and (ii) the potential dependency heads are provided. These can be easily transformed to the weighting scheme required by the parser by taking the negative logarithm of the individual probability values given on input. Figure 9 shows the weighted input grammar presented in Figure 7 after applying this transformation.

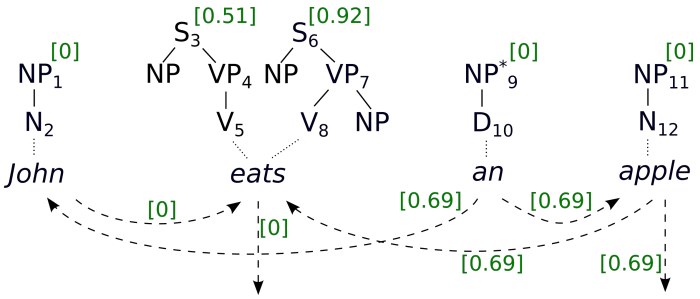


Figure 9: Input grammar from Figure 7 after converting probabilities to weights (enclosed between square brackets and marked in green). Internal nodes are additionally decorated with a unique IDs.

4.3.1 Input

We define the input sentence as a sequence $(s_i)_{i=1}^n$, where n is the length of the sentence and each s_i is a distinct terminal symbol (token). Each terminal must be considered as distinct because, even if the same word form is used at two positions of the input sentence, the two positions will receive two distinct distributions of supertags and dependency heads.

4.4 Grammar representation

ParTAGe adopts a two-layered grammar representation in which (a) the set of the grammar ETs is first transformed to the equivalent directed acyclic graph (DAG), and (b) an automaton-based

representation is then used to compress the traversals of the individual nodes and their children in the DAG (from left to right).

In this work, we only assume the first layer of this representation, i.e., the grammar DAG. The traversals of the individual nodes and their children are represented explicitly via dotted rules. For simplicity, we assume no subtree sharing (Waszczuk et al., 2016a) in the grammar DAG in our formalization.[2]

We define the following auxiliary functions:

- $\ell(N)$ – the label (non-terminal, terminal, or ϵ) assigned to node N
- $root(N)$ – predicate which tells if N is a root node
- $leaf(N)$ – predicate which tells if N is a leaf node
- $foot(N)$ – predicate which tells if N is a foot node
- $sister(N)$ – predicate which tells if N is the root of a sister adjunction tree

4.5 Architecture

ParTAGe is based on *weighted deduction rules* (Shieber et al., 1995; Nederhof, 2003), which serve to infer *chart items*. Each chart item is a pair (x, r), where x is a *configuration* and r is a *span*. Each configuration x represents a position in the traversal of the corresponding ET t_x, and each span r represents a fragment of the input sentence. Informally, (x, r) asserts that the already traversed part of t_x can be matched against the words in r.

Additionally, to each item (x, r) a *weight* is assigned, which is also calculated via deduction rules and which represents the cumulative weight of the already traversed part of t_x. We get back to the topic of weights in Sec. 4.9.

As mentioned before, ParTAGe is an A* parser, which means that apart from the weights calculated via the deduction rules, an estimate of the weight remaining to parse the entire input sentence has to be determined for each chart item (see Sec. 4.13). This allows the parser to explore more plausible derivations first and find an optimal parse without creating the entire chart.

4.6 Configuration

A parsing configuration x represents a position in the traversal of the corresponding ET t_x. It takes the form of a *dotted rule*[3] $N \to \alpha \bullet \beta$, where N is a non-leaf node of an ET and $\alpha\beta$ is the sequence of N's children nodes (from left to right). The interpretation of the rule $N \to \alpha \bullet \beta$ is that the nodes in α (on the left of $\bullet$) are already parsed, i.e., matched against some input words, while the nodes in β still need to be matched.

We also use N to denote a parsing configuration where all the children nodes α have been matched against input. While N can be seen as syntactic sugar for $N \to \alpha\bullet$ (for some α), the parser makes a distinction between the two types of parsing configurations and it includes a rule which explicitly transforms $N \to \alpha\bullet$ to N.

4.7 Span

Let $pos(n) = \{0, \ldots, n\}$ be the set of positions between the words of the input sentence. A *span* is a 4-tuple (i, j, k, l) where $i, l \in pos(n)$ and $j, k \in pos(n) \cup \{-\}$. If $j, k \neq -$, $i \leq j \leq k \leq l$. Otherwise, $i \leq l$. We also write (i, l) to denote $(i, -, -, l)$. The interpretation of the span r depends on the parsing configuration x accompanying r within a chart item.

2. However, we used subtree sharing among the ETs attached to the same input position in our implementation and experiments.
3. Not to be confused with a dotted rewriting rule. A dotted rule in our architecture represents a point in the traversal of a fragment of an ET.

4.8 Chart item

The presence of the item $\eta = (x, (i, j, k, l))$ in the chart asserts that:

- If $j = k = -$, then x is matched against all the input words between positions i and l.
- Otherwise, x corresponds to an auxiliary ET t_x with a foot node. In this case, t_x's foot is matched against the span (j, k), the processed part of t_x on the left of the foot is matched against (i, j), and the processed part of t_x on the right of the foot is matched against (k, l).

If x is a dotted rule, we call η *active*. Otherwise, if x is a node of an ET, we call it *passive*.

4.9 Weight pair

To each item $\eta = (x, (i, j, k, l))$ in the chart a pair of weights (w, w') is assigned,[4] where w is the *inside weight*, i.e., the weight of the inside derivation of η, and w' is the *prediction weight*, i.e., the total weight of the partial derivations used for prediction in η's inside derivation.

Both w and w' are calculated directly via deduction rules. The inside weight w corresponds to the weight of the part of the derivation that is already determined – each full derivation based on η will have to contain it as its part. In our deduction system, the prediction weight w' corresponds to the cost of scanning the words in the gap (j, k) while providing the non-terminal necessary to match the foot of the auxiliary ET t_x. The purpose of w' is to facilitate the calculation of the A* heuristic (see Sec. 4.13), which serves to estimate the *outside weight*, i.e., the weight remaining to parse the entire input sentence. While the words in the gap are accounted for in the prediction weight w', the heuristic has to additionally account for the words on the left of i and on the right of l.

4.10 Scanning cost

To estimate the cost of parsing the remaining part of the sentence, we assume that each word outside of the current item's span will be analyzed with the lowest-weight ET and that it will get assigned the lowest-weight dependency head. This estimation strategy guarantees that the resulting A* heuristic is admissible, i.e., it never overestimates the cost of parsing the remaining words. We thus define, for a given token x, the lower-bound cost $\mathcal{C}(x)$ as:

$$\mathcal{C}(x) = \min\{\omega(t) : t \in T, \mathit{term}(t) = x\} \; + \; \min\{\omega(x, y) : y \in s\} \tag{1}$$

where T is the set of grammar ETs, restricted to supertagging results. The lower-bound cost of parsing the remaining set of tokens X can be then represented as:

$$\mathcal{C}(X) = \sum\nolimits_{x \in X} \mathcal{C}(x) \tag{2}$$

4.11 Amortized weight

Given a chart item $\eta = (x, r)$, we define the *amortized weight* $A(x)$ of x as:

$$A(x) = \omega(t_x) + \omega(t_x, \cdot) - \mathcal{C}(\mathit{sup}(x)),$$

where $\omega(t_x)$ is the weight of the ET t_x corresponding to x, $\omega(t_x, \cdot) = \min\{\omega(\mathit{term}(t_x), y) : y \in s\}$ is the minimal cost of attaching $\mathit{term}(t_x)$ as a dependent to another token in the input sentence, and $\mathcal{C}(\mathit{sup}(x))$ is the cost of scanning the terminals in t_x that remain to be matched ($\mathit{sup}(x)$). In practice, $\mathit{sup}(x)$ is either empty (if t_x's anchor is in the already traversed part of the tree) or contains the t_x's single anchor (we assume that each ET contains precisely one terminal, see Sec. 4.1).

4. We slightly diverge from the terminology used in Nederhof (2003), whose *inner* weight roughly corresponds to our *inside* weight, and *forward* weight to the sum of our *inside* and *prediction* weights.

Intuitively, $A(x)$ can be understood as the weight of the already parsed part of t_x, augmented with the minimal dependency weight of attaching t_x to another tree. Both $\omega(t_x)$ and (at least) $\omega(t_x, \cdot)$ will have to be accounted for in the total weight of any full derivation based on η. $\mathcal{C}(sup(x))$, on the other hand, gets discounted because the anchor $term(t_x)$ may still be outside of the η's item span (i.e., $sup(x) = \{term(t_x)\}$) and we assume the minimal scanning cost for each remaining token in the calculation of the heuristic (see Sec. 4.13 below).

4.12 Deduction rules

Table 1 shows the deduction rules of the extended version of ParTAGe, simplified in comparison with the rules presented previously (Waszczuk et al., 2017) in that no automaton-based grammar compression is assumed, and extended with two new rules (`SA` for handling sister adjunction and `EM` for empty terminals) as well as support for bilexical dependency weights. `AX` (axiom) posits that each subtree can be matched starting from any non-final position in the sentence, which corresponds to the bottom-up nature of the parser. `SC` and `EM` handle scanning input and empty terminals, respectively, while `DE` (deactivate) handles the situation where a full ET subtree has been matched. `PS` is responsible for combining two adjacent fragments of the same ET. `SU` models regular substitution, i.e., matching the leaf node of an ET with another, fully matched ET. `FA` and `RA` both model regular adjunction: `FA` performs predictions in order to identify the spans over which adjunction can be potentially performed, while `RA` performs the actual adjunction, i.e., attaching the auxiliary tree to an internal node of another tree. Finally, `SA` models sister adjunction, where a fully matched sister ET is attached to a non-leaf node of another tree.

AX:	$\frac{}{(0,0) : (N \rightarrow \bullet\alpha, (i,i))}$	$\substack{i \in \{0,\ldots,n-1\} \\ N \rightarrow \alpha \text{ is a rule}}$
SC:	$\frac{(w,w') : (N \rightarrow \alpha \bullet M\beta, (i,j,k,l))}{(w,w') : (N \rightarrow \alpha M \bullet \beta, (i,j,k,l+1))}$	$\ell(M) = s_{l+1}$
EM:	$\frac{(w,w') : (N \rightarrow \alpha \bullet M\beta, (i,j,k,l))}{(w,w') : (N \rightarrow \alpha M \bullet \beta, (i,j,k,l))}$	$\ell(M) = \epsilon$
DE:	$\frac{(w,w') : (N \rightarrow \alpha \bullet, (i,j,k,l))}{(w,w') : (N, (i,j,k,l))}$	
PS:	$\frac{(w_1,w_1') : (N \rightarrow \alpha \bullet M\beta, (i,j,k,l)) \quad (w_2,w_2') : (M, (l,j',k',l'))}{(w_1+w_2, w_1'+w_2') : (N \rightarrow \alpha M \bullet \beta, (i, j \oplus j', k \oplus k', l'))}$	
SU:	$\frac{(w_1,w_1') : (N \rightarrow \alpha \bullet M\beta, (i,j,k,l)) \quad (w_2,0) : (R, (l,l'))}{(w_1+w_2+\omega(R,N), w_1') : (N \rightarrow \alpha M \bullet \beta, (i,j,k,l'))}$	$\substack{leaf(M) \wedge \neg foot(M) \\ root(R) \wedge \neg sister(R) \\ \ell(M) = \ell(R)}$
FA:	$\frac{(w_1,0) : (N \rightarrow \alpha \bullet F\beta, (i,l)) \quad (w_2,w_2') : (M, (l,j',k',l'))}{(w_1, w_2+w_2'+A(M)) : (N \rightarrow \alpha F \bullet \beta, (i,l,l',l'))}$	$\substack{foot(F) \wedge \ell(M) = \ell(F) \\ root(M) \implies (j',k') = (-,-) \\ \neg sister(M)}$
RA:	$\frac{(w_1,w_1') : (R, (i,j,k,l)) \quad (w_2,w_2') : (M, (j,j',k',k))}{(w_1+w_2+\omega(R,M), w_2') : (M, (i,j',k',l))}$	$\substack{root(R) \wedge \ell(R) = \ell(M) \\ root(M) \implies (j',k') = (-,-) \\ \neg sister(M)}$
SA:	$\frac{(w_1,w_1') : (N \rightarrow \alpha \bullet \beta, (i,j,k,l)) \quad (w_2,0) : (M, (l,l'))}{(w_1+w_2+\omega(M,N), w_1') : (N \rightarrow \alpha \bullet \beta, (i,j,k,l'))}$	$\ell(M) = \ell(N) \wedge sister(M)$

Table 1: Deduction rules of the revised parser, where (in `PS`) $i \oplus j$ is equal to i if $j = -$ and j otherwise. To simplify notation, we write $\omega(M, N)$ to denote $\omega(t_M)$ plus the weight $\omega(term(t_M), term(t_N))$ of the dependency arc combining the corresponding ETs.

Note that the weight $\omega(t_x)$ of an ET t_x is transferred to the inside weight only when t_x gets attached as dependent to another tree (via one of the `SU`, `RA`, or `SA` rules). Firstly, while it could be more intuitive to transfer $\omega(t_x)$ to the inside weight already in the axiom rule (`AX`), the current

solution composes better with subtree sharing.[5] Secondly, the situation where an ET t_x becomes the root of the derivation has to be handled as a special case at the end of parsing. More precisely, we have to make sure that both $\omega(t_x)$ and the weight of t_x becoming a dependent of the dummy root position get transferred to the inside weight of the corresponding chart item. For simplicity, these details are not reflected in the deduction rules in Tab. 1.

4.13 A* heuristic

Given a chart item $\eta = (x, r)$ such that $r = (i, j, k, l)$ and the corresponding weight pair (w, w'), the heuristic h (which provides a lower-bound estimate on the cost of parsing the remaining part of the sentence) is defined as follows:

$$h(x, r) = A(x) + \mathcal{C}(\mathit{rest}(r)) + w', \tag{3}$$

where $\mathit{rest}(r)$ is the set of tokens remaining on the left and right of i and l, respectively, and $\mathcal{C}(\mathit{rest}(r))$ is the total minimal cost of scanning each remaining word in $\mathit{rest}(r)$. Note that the minimal possible cost of scanning the words in the gap (j, k) (provided that $j, k \neq -$) is accounted for in w' (see Sec. 4.9). $A(x)$, on the other hand, represents the weight of the part of t_x that has already been processed, which is not yet transferred to the inside weight (see Sec. 4.11).

The total weight of item η, i.e., the sum of its inside and (estimated) outside weights, is equal to $w + h(x, r)$. This total weight determines the order in which the created chart items are removed from the agenda.

4.14 Example

Fig. 10 shows a fragment of the hypergraph constructed when processing the sentence and grammar presented in Fig. 7 and Fig. 9. Each node represents a chart item and each (hyper)arc represents an application of a deduction rule (see Tab. 1). Additionally, to each chart item a pair of weights $[w; w_h]$ is assigned (linked to it via a dotted line), where w is the corresponding inside weight (calculated as prescribed by the deduction rules), and w_h – the corresponding value of the heuristic (see Sec. 4.13). Prediction weights are not shown because, in this example, they are all equal to 0.

4.15 Admissibility and monotonicity

The A* heuristic is *admissible* if it never overestimates the remaining parsing cost. It is *monotonic* if the total parsing cost $(w + h(\eta))$ never decreases as new chart items are created. The two properties guarantee correctness, i.e., that the first *final* chart item $\eta = (N, (0, n)) : \mathit{root}(N) \wedge \ell(N) \in S$ reached by the parser corresponds to a lowest-weight derivation, where S is the set of start symbols, and that the inside weight w accompanying η is the corresponding lowest weight.

The heuristic defined in Eq. 3 is both admissible and monotonic within the context of the parser specified in Tab. 1. Admissibility is virtually by definition – it stems from the assumption adopted in the heuristic that each remaining word will be matched with the lowest-weight ET and the lowest-weight dependency head. Monotonicity, on the other hand, can be proved by induction on the parser's deduction rules. We provide a formal monotonicity proof, written in Coq, in the ParTAGe's code repository. Additionally, the tool provides an optional runtime check, which allows to verify the monotonicity of the actual implementation each time a new chart item is created.

5. When subtree sharing is used, the parsing configuration x can correspond to several different ETs t_x and, therefore, $\omega(t_x)$ cannot be uniquely determined.

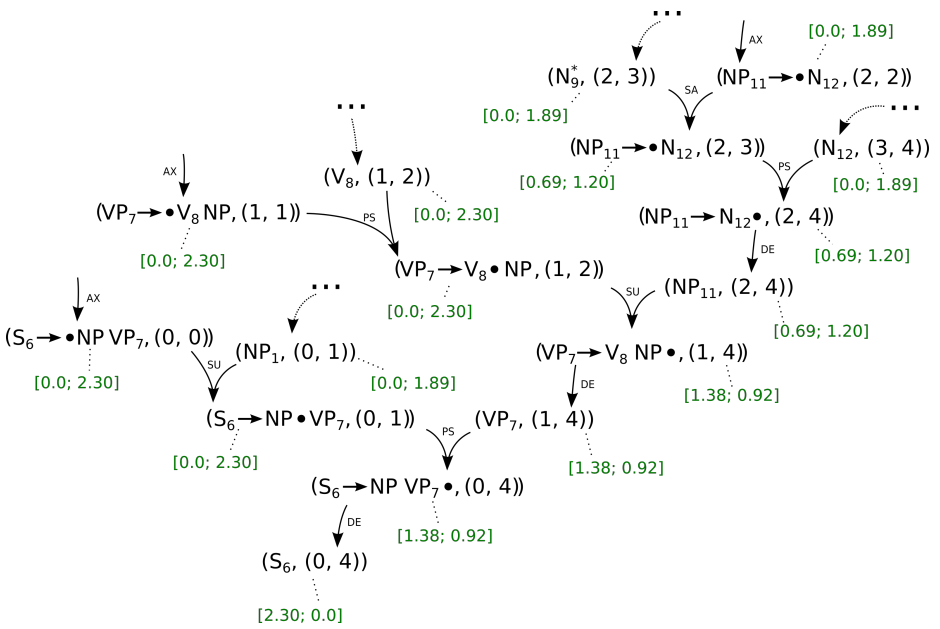


Figure 10: Fragment of the hypergraph decorated with the inside and the estimated outside (heuristic values, see Sec. 4.13) weights, generated when processing the input from Fig. 7 and Fig. 9. The prediction weights (calculated via the deduction rules along the inside weights) are not shown because, in this example, they are all equal to 0.

5. Evaluation/Experiments

5.1 Dataset: French LTAG and Word Embeddings

In our experiments we use an LTAG for French extracted from the French Treebank (FTB; Abeillé et al. (2003)). We followed the approach to treebank-based LTAG induction developed by Xia (1999) and largely adopted parameters reported in Bladier et al. (2018c) for extracting an LTAG from FTB. Inducing a grammar from a treebank means identifying a set of productions that could have produced its parse trees. In our case, this amounts to decomposing each treebank tree into a sequence of elementary trees together with a derivation tree that specifies how the elementary trees have to be combined.

We use the top-down approach to LTAG induction described in Xia (1999). The main idea of this approach is to cast the constituency trees in the treebank as derived trees in LTAG. Elementary trees are extracted in a top-down fashion using percolation tables to identify grammatically obligatory elements (i.e., *complements*), optional elements (i.e., *modifiers*), as well as a head child for each constituent. Upon marking the nodes in trees as being heads, complements or modifiers, all subtrees corresponding to modifiers and complements are extracted forming auxiliary trees and initial trees, respectively. The head child and its lexical anchor are kept in the tree. When extracted in this way, elementary trees contain the corresponding lexical anchor and the branches represent a particular syntactic context of a construction with slots for its complements (Bladier et al., 2018c).

Since several different LTAGs can be extracted from the same treebank depending on the number of POS-tag and node labels, head and modifier percolation tables as well as the research question, we adopted the parameters for the French LTAG induction with the most accurate supertagging results described in Bladier et al. (2018a). We reduced the number of POS tags to 13 and kept most of the multi-word-expressions (MWEs) in the grammar. We only transformed some of the nominal MWEs with regular syntactic patterns to regular NP constituents (for example *(MWN (A ancien)*

(N élève)) → (NP (AP (A ancien)) (N élève))) in order to keep the size of the grammar low. Table 2 provides some statistics on the extracted LTAG grammar (13 POS tags, including compounds and including punctuation marks).

Since the trees in FTB have a rather flat structure, the LTAG grammar we use for the experiments does not have regular adjunction, but sister-adjunction. The resulting LTAG with sister-adjunction is basically an LTIG *Lexicalized Tree Insertion Grammar*; (Schabes and Waters, 1995). Auxiliary trees in an LTIG do not allow wrapping adjunction or adjunction on the root node but permit multiple simultaneous adjunction on a single node of initial trees. However, since LTIG is a special variant of LTAG, we refer to the extracted grammar as LTAG in the remainder of the paper.

Parameters	**French LTAG**
Supertags	5103
Supertags occuring once	2611
POS tags	13
Sentences	21550
Avg. sentence length	29.81
# initial trees	1953
# auxiliary trees	3150

Table 2: Statistics on the extracted French LTAG.

For our experiments, we used the train, dev., and test sets from the French Treebank (version SPMRL 2013, described in Seddah et al. (2013)) in order to compare with previous work. We also included additional sentences to the train set from the latest version of the French Treebank (version 1.0 2016). Thus, our experiment data include 19 080, 1235, and 2541 sentences in the train, dev., and test set, respectively). We use pre-trained 200-dimensional word embeddings for French (Fauconnier, 2015) trained on 1.6 billion words in frWaC corpus to transform tokens in our corpus into numeric vectors. For out-of-vocabulary words, we assign embeddings by random vector.

	English LTAG from PTB (Kasai et al., 2018)			**French LTAG from FTB (our work, dev set)**		
Parsing Model	Stag acc.	UAS	LAS	Stag acc.	UAS	LAS
BiLSTM3	–	91.75	90.22	–	87.74	82.88
BiLSTM3-CNN	–	92.27	90.76	–	88.76	84.68
BiLSTM3-HW-CNN	–	92.29	90.71	–	88.52	84.30
BiLSTM4-CNN	–	92.11	90.66	–	88.73	84.43
BiLSTM4-HW-CNN	–	92.78	91.26	–	88.82	84.62
BiLSTM5-CNN	–	92.34	90.77	–	31.94	17.43
BiLSTM5-HW-CNN	–	92.64	91.11	–	–	–
BiLSTM4-CNN-POS	–	92.07	90.53	–	89.15	85.15
BiLSTM4-CNN-Stag	–	92.15	90.65	–	88.31	84.07
Joint (Stag)	90.51	92.97	91.48	84.05	88.97	84.70
Joint (POS + Stag)	**90.67**	**93.22**	**91.80**	**84.91**	**89.61**	**85.60**

Table 3: Supertagging and dependency parsing experiments and comparison with previous work. HW and CNN stand for the models with highway connections and the CNN-layer, while the numbers 3, 4 and 5 indicate the number of BiLSTM-layers. Both joint models use 4 BiLSTM layers, CNN, and the highway connections to predict dependencies along with supertags or POS tags + supertags, respectively. Our French results show a similar pattern to the reported results for English (Kasai et al., 2018).

5.2 Supertagging and Dependency Parsing Experiments

The pipeline of the tree parsing models based on supertagging consists of the step of choosing the sequences of supertags and dependency arcs and a subsequent actual parsing step. Supertagging is beneficial for parsing, since it disambiguates many choices before applying the costly actual parsing step. The problem of choosing the correct supertags, however, remains the bottleneck for such parsing models, meaning that the accuracy of the pipeline is strongly dependent on this step (Lewis et al., 2016). Thus, following the experiments reported in Kasai et al. (2018), we run several experiments with different parameters on the French LTAG to make sure we get the best results on predicting the supertags and dependency arcs. We used BiLSTM-models including 3, 4, and 5 hidden layers, with and without the CNN layer, and highway connections. We compare our results on French with previous work on English to prove that the parsing models show similar behaviour for both languages with the Joint (POS + Stag) model showing the best results (see Table 3).

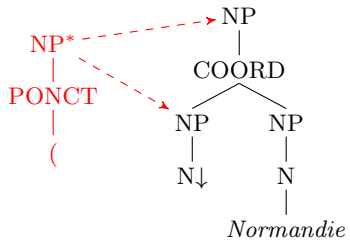


Figure 11: Attachment ambiguity: the tree on the left can be attached to both NP nodes in the tree for *Normandie*.

The parsing models BiLSTM4-CNN-POS and BiLSTM4-CNN-Stag are pipeline models which use predicted POS and supertags as features for the system. The two joint models (Join Stag and Joint POS + Stag) predict either only stags or also POS tags and supertags together with dependency arcs and labels using only word and character features as input. The results of our supertagging and dependency parsing experiments are summarized in Table 3. The BiLSTM-models in this table predict only dependency arcs, while the joint models predict dependency arcs, labels of dependency arcs, supertags and also POS tags. We clustered 27 original dependency labels provided in the FTB data to 6 more general labels (*'suj'*, *'obj'*, *'oblique_arg'*, *'adj'*, *'subst'*, and *'root'*). We used the best parameters from the previous experiments (4 layers, CNN layer for character embeddings, highway connections) for the joint model, which proved to provide the most accurate predictions.

In addition, we also run an experiment with gold data using our modified A* parsing algorithm for LTAGs. For this experiment we used gold supertags and dependency arcs from the extracted French LTAG. The experiments show the parsing accuracy of 99,4 % (see Table 4). The accuracy is not 100 % even with the gold supertags and gold dependency arcs data. The reason for the lacking 0,6 % are the attachment ambiguities: Kasai's parser does not predict the Gorn addresses of the nodes in the elementary trees where the substitution or adjunction takes place. Lack of this information leads to an attachment ambiguity in cases where an initial tree has two different nodes with the same label on which the adjunction can take place (see an example in Fig. 11).

	gold supertags and arcs	
	dev	dev $\leq$ 25
Exactly matching sentences	88.50	95.61
POS accuracy	100.00	100.00
Labeled Recall	99.40	99.60
Labeled Precision	99.40	99.60
Labeled F1	**99.40**	**99.60**
# sentences	1235	501

Table 4: Parsing results with ParTAGe on gold data on the full set of sentences and on the sentences with less than 25 tokens.

5.3 A* LTAG Parsing Experiments

Finally, we carried out experiments with 10-best supertags and 10-best arcs on the FTB SPMRL (2013) Seddah et al. (2013) test set, which showed that ParTAGe is able to find a combination of suitable supertags and dependency arcs to produce full derived trees for every sentence (see Table 5). In comparison, taking the 1-best output allowed to produce a complete derivation only in around 50% of sentences in this dataset. We also measured the labeled evalb F1-score on the resulting constituency trees, excluding punctuation, which showed that our system achieves close to state-of-the-art results on FTB, although lower than the LSTM-based parser with self-attention developed by Kitaev and Klein (2018).

Note that the labeled F1 only evaluates the resulting derived trees with respect to the treebank trees. Our parser, however, outputs also information on which tree fragments constitute elementary trees and how they combine with each other. This additional syntactic information has been claimed to be useful for semantic tasks that require knowledge about predicate-argument relations such as semantic role labeling (Liu and Sarkar, 2009).

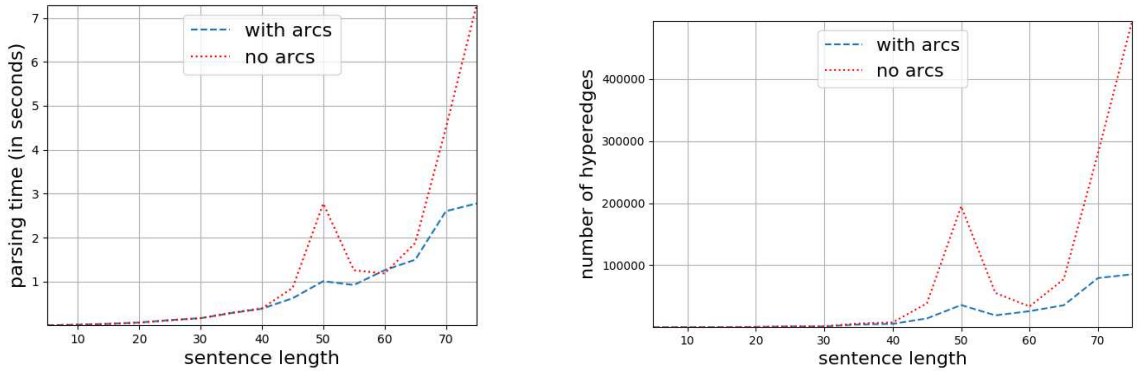
	1-best output (supertagger output)	**10-best output** (with arcs)		**10-best output** (no arcs)	
	test	test	test $\leq$ 25	test	test $\leq$ 25
Unlabeled Attachment Score (UAS)	88.54	**88.76**	**91.35**	84.17	89.07
Supertagging accuracy	81.22	**81.37**	82.93	81.35	**83.04**
Sentences 100% correct stags + arcs:	13.81	**14.44**	**30.16**	12.08	26.26
# sentences without parse	1215 (47.82%)	**0 (0%)**			
Exactly matching parses	–	**21.68**	**41.77**	17.83	36.40
Labeled F1 (our parser)	–	**84.36**	**88.02**	73.97	82.51
# sentences	2541	2541	1154	2541	1154
F1 Best Neural Parser (Cross and Huang, 2016), no punc.		83.11			
F1 Best Top-Down Parser (Stern et al., 2017), no punc.		82.23			
F1 LSTM Self-Attention (Kitaev and Klein, 2018), with punc.		**84.06**			
F1 Multiling. BiLSTM (Coavoux and Crabbé, 2017), with punc.		82.49			

Table 5: Results with ParTAGe parsing on predicted data (10-best columns) compared to the supertagger output (1-best column) and other parsing systems. We measured the results on parsing with 10-best predicted LTAG supertags including and excluding predicted dependency arcs. The results are provided for the full set of sentences and for the sentences with less than 25 tokens on the test set of SPMRL French Treebank (Seddah et al., 2013). We measured our results without counting punctuation marks. We use the coarse-grained POS tags provided in the SPMRL data and do not include function labels into our evaluations.

In the 10-best output (no arcs) column in Tab. 5, we additionally present the results when ParTAGe uses the distributions of supertags but ignores information about dependencies provided on input. In this case, the system achieves comparable supertagging accuracy, but at the cost of significantly lower UAS and F1 results. In particular, the almost 10% drop in F1 shows how important the dependency-related information is for the results of constituency parsing in this architecture. Besides, dependency-related information can also reduce the parsing time and the size of the resulting hypergraph, as shown in Fig. 12a and Fig. 12b, respectively.

6. Error Analysis

In the previous section we have shown that using 1-best supertags and 1-best dependency arcs is not sufficient for full LTAG parsing, while a 10-best input to the A* parsing model yields a parse for every sentence in the test and development set. Although every sentence now gets a full



(a) (b)

Figure 12: Reduction of the parsing time and the number of hypergraph edges due to dependency information (dev set, 10-best output for sentences with length < 80.)

parse, the unlabeled attachment score (UAS) and labeled attachment score (LAS) results as well as results on supertagging improved only slightly compared to the (1-best) output of the original neural architecture of the supertagger and dependency parser by Kasai et al. (2018), while the supertagging results using dependency arcs got slightly worse. The difference in both metrics is small because the algorithm of ParTAGe searches for mutually compatible supertags and arcs to combine them to a full derived tree, and thus might pick not the correct supertags for the sentence but the compatible ones depending on their weights. More precisely, due to this compatibility requirement, an error in one place (incorrect supertag or incorrect dependency arc) oftentimes leads to further errors in order to retrieve a correct derivation tree. Thus, prediction of the supertags and arcs remains the bottleneck of the parsing architecture.

As we have seen, the results for English are better than the ones for French (see Table 3). One reason for this is probably the size of the training data for both languages (39 561 PTB sentences for English versus 19 080 FTB sentences for French). Furthermore, the average sentence length in the PTB (23.90 tokens) is lower than the one in the FTB (29.81 tokens), which makes the prediction of dependency arcs slightly harder for the French data.

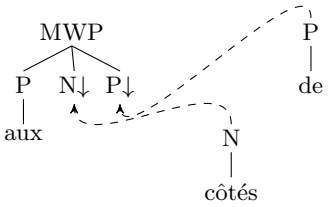


Figure 13: Flat supertags composing the multiword preposition *aux côtés de*.

A large number of supertagging errors for French are due to different possible attachment sites for punctuation marks. Punctuation marks are attached to the corresponding constituents and not to the root node of the whole sentence. However, punctuation marks help to identify possible constituents in sentences and omitting them does not substantially improve supertagging (Bladier et al., 2018c). PP attachments are another major source of errors while predicting supertags with French LTAGs. The supertagger encounters difficulties with classifying PPs as modifiers or complements, since FTB in the majority of cases does not offer additional function marks to distinguish these two. The supertagger also encounters problems with identifying the correct site for attaching the PPs to a node in the syntactic tree. Another major source of errors are the flat multi-word expressions encountered in the FTB, which lead to a high number of flat elementary trees and produce noise in the training data. See an example for the multi-word preposition *aux côtés de* ("alongside (with)") in Table 6 and the corresponding supertags in Fig. 13.

Gold supertag	**Predicted supertag**	**Example**
(NP* (PP (P ⋄) (NP↓)))	(PP* (PP (P ⋄) (NP↓)))	apprentissage **des** enfants
(NP* (PONCT ⋄))	(SENT* (PONCT ⋄))	**-LRB-** 66,7 % **-RRB-**
(N ⋄)	(NP (N ⋄))	aux **côtés** de Volvo
(NP (N ⋄))	(NP (N ⋄) (NP↓))	aux **partenaires** du montage
(SENT (NP↓) (VN (V ⋄)))	(AP (A ⋄))	des chômeurs sont **inscrits**

Table 6: Most common error classes for LTAG supertagging with French Treebank (SENT and PONCT stand for *sentence* and *punctuation*).

Table 7 shows the F1-scores for parsing of the 10 most frequent types of constituents in FTB. NP is the most frequent constituent in the FTB. Parsing NPs shows a relatively low F1 score, the reason for which is a diverse inner structure of NPs in French Treebank, which allows NPs to consist of differently structured subtrees including the problematic PP-subtrees. Thus, NPs can consist of a big variety of different supertags. This makes the prediction of the correct supertags hard, and supertagging errors are then oftentimes subsequently reflected in the resulting constituency tree.

Structural diversity of daughter nodes is also the reason of parsing errors for *Sint* and *Srel* constituents (i.e. final clauses in FTB). Note that parsing errors in NP constituents percolate to other constituents which contain them, for example *PP*, *COORD* (coordinating constituents), and *Sint* or *Ssub* (i.e. internal or subordinate clauses). Prediction of supertags for NPs can be potentially facilitated by using heuristic rules to make the inner structures of the subtrees constituting NPs more regular. In addition to this, the errors in parsing PP constituents result from the attachment to a wrong node in the tree, which is a result of errors in supertag prediction.

label	frequency	recall	precision	F1
(any)	100.00	84.82	83.90	84.36
NP	34.97	87.61	83.23	85.36
PP	20.39	88.54	79.19	83.60
VN	11.67	96.66	97.49	97.07
AP	5.74	93.60	71.81	81.27
SENT	5.02	100.00	100.00	100.00
VP	4.87	83.41	84.61	84.00
COORD	3.56	89.12	89.12	89.12
MWN	3.01	80.43	82.33	81.37
Sint	2.41	78.13	78.91	78.52
Srel	1.56	88.97	87.31	88.14

Table 7: Results of evaluating the pipeline parsing system on the test set, overall and for the 10 most frequent constituent labels. The scores are labeled EVALB scores.

7. Conclusions

We present a novel architecture for LTAG parsing based on the pipeline of a neural supertagger and dependency parser proposed by Kasai et al. (2018) and a modified A* based LTAG parsing algorithm ParTAGe implemented by Waszczuk (2017). We modified the supertagging model to produce n-best supertags and k-best arcs instead of 1-best output. This output is used as the input for the second step of the parsing pipeline. We also modified the A* based parser ParTAGe (Waszczuk, 2017) to be able to process n-best dependency arcs.

We have shown that the 1-best predicted supertags and 1-best predicted dependency relations between supertags are not sufficient do produce a full parse (i.e. full LTAG derived tree) for a large number of sentence. However, a sufficient large number of n and k has proven to be enough for obtaining full parsing trees on our data with our architecture.

We tested our architecture on an LTAG extracted automatically from the French Treebank (FTB). Our architecture shows comparable results to other parsers for French. Adding information on dependency arcs along with the information on supertags considerably reduces ambiguity for the actual parsing step. We have shown that adding information on dependency arcs to the original architecture of the A* based parser ParTAGe greatly decreases the parsing time and the number of possible parses for every tree.

8. Future Work

The parsing approach presented in this paper can be used for several kinds of LTAGs for different languages. In our follow up work we plan to extract different LTAG grammars for English, Polish and Dutch and to test our architecture on these grammars. We also intend to induce our own LTAG from the Penn Treebank (allowing both sister adjunction and regular adjunction), since the existing LTAGs (Chen et al., 2006; Xia, 1999) do not produce the original derived trees from the treebanks, but introduce extra nodes due to the binarization arising from adjoining regular auxiliary trees in LTAG, which contain a footnode. We will extract our grammars for Polish and Dutch from the existing treebanks Składnica (Woliński et al., 2011) and LASSY (Van Noord et al., 2013).

Besides using LTAG, we also plan to apply the approach presented in this paper to statistical parsing with Role and Reference Grammar (RRG; Van Valin and LaPolla (1997); Van Valin Jr (2005)). RRG is a grammar theory that has recently been formlized as a tree-rewriting formalism in the style of LTAG (Kallmeyer et al., 2013; Osswald and Kallmeyer, 2018), except that it has slightly different composition operation, namely an additional *wrapping* operation besides substitution and sister adjunction. For this work we will use RRGbank (Bladier et al., 2018b), an RRG-based version of the PTB, and we have to adapt ParTAGe in order to support wrapping.

Acknowledgments

This work was carried out as a part of the research projects TreeGraSP (treegrasp.phil.hhu.de) funded by a Consolidator Grant of the European Research Council (ERC) and the project "Parsing beyond CFG" funded by the German Research Foundation (DFG). The authors would like to thank Jungo Kasai for his help with the adaptation of the supertagger and the dependency parser, Marie Candito and Benoît Crabbé for consultations on the use of the French linguistic resources and David Arps for his comments on the paper. We also thank the anonymous reviewers for their careful reading, valuable suggestions and constructive comments.

References

Abeillé, Anne, Lionel Clément, and François Toussenel (2003), Building a treebank for French, *Treebanks*, Springer, pp. 165–187.

Bäcker, Jens and Karin Harbusch (2002), Hidden markov model-based supertagging in a user-initiative dialogue system, *Proceedings of the Sixth International Workshop on Tree Adjoining Grammar andRelated Frameworks (TAG+ 6)*, pp. 269–278.

Bangalore, Srinivas and Aravind K Joshi (1999), Supertagging: An approach to almost parsing, *Computational linguistics* **25** (2), pp. 237–265, MIT Press.

Bladier, Tatiana, Andreas van Cranenburgh, and Laura Kallmeyer (2018a), Extraction of LTAG-based supertags from the French treebank: challenges and possible solutions. Abstract presented at Grammar and Corpora 2018 (GAC 2018), November 15–17, 2018, University of Paris-Diderot, France. http://drehu.linguist.univ-paris-diderot.fr/gac-2018/abstracts/bladier_etal.pdf.

Bladier, Tatiana, Andreas van Cranenburgh, Kilian Evang, Laura Kallmeyer, Robin Möllemann, and Rainer Osswald (2018b), RRGbank: a Role and Reference Grammar Corpus of Syntactic Structures Extracted from the Penn Treebank, *Proceedings of the 17th International Workshop on Treebanks and Linguistic Theories (TLT 2018), December 13–14, 2018, Oslo University, Norway*, number 155, Linköping University Electronic Press, pp. 5–16.

Bladier, Tatiana, Andreas van Cranenburgh, Younes Samih, and Laura Kallmeyer (2018c), German and French neural supertagging experiments for LTAG parsing, *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, Student Research Workshop, Melbourne, Australia*. https://www.aclweb.org/anthology/P18-3009/.

Carroll, John, Nicolas Nicolov, Olga Shaumyan, Martine Smets, and David Weir (1999), Parsing with an extended domain of locality, *Ninth Conference of the European Chapter of the Association for Computational Linguistics*, Association for Computational Linguistics, Bergen, Norway. https://www.aclweb.org/anthology/E99-1029.

Chen, John, Srinivas Bangalore, and K Vijay-Shanker (2006), Automated extraction of tree-adjoining grammars from treebanks, *Natural Language Engineering* **12** (3), pp. 251–299, Cambridge University Press.

Chen, John, Srinivas Bangalore, Michael Collins, and Owen Rambow (2002), Reranking an n-gram supertagger, *Proceedings of the Sixth International Workshop on Tree Adjoining Grammar and Related Frameworks (TAG+ 6)*, pp. 259–268.

Chiang, David (2000), Statistical parsing with an automatically-extracted tree adjoining grammar, *Proceedings of the 38th annual meeting of the Association for Computational Linguistics*, pp. 456–463.

Chung, Wonchang, Siddhesh Suhas Mhatre, Alexis Nasr, Owen Rambow, and Srinivas Bangalore (2016), Revisiting supertagging and parsing: How to use supertags in transition-based parsing, *Proceedings of the 12th International Workshop on Tree Adjoining Grammars and Related Formalisms (TAG+12)*. https://www.aclweb.org/anthology/W16-3309.pdf.

Coavoux, Maximin and Benoît Crabbé (2017), Multilingual lexicalized constituency parsing with word-level auxiliary tasks, *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, Association for Computational Linguistics, Valencia, Spain, pp. 331–336. https://www.aclweb.org/anthology/E17-2053.

Crabbé, Benoit (2005), Représentation informatique de grammaires fortement lexicalisées: Applicationa la grammaire darbres adjoints, *These de Doctorat, Université Nancy*.

Cross, James and Liang Huang (2016), Span-based constituency parsing with a structure-label system and provably optimal dynamic oracles, *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, Association for Computational Linguistics, Austin, Texas, pp. 1–11. https://www.aclweb.org/anthology/D16-1001.

Dozat, Timothy and Christopher D. Manning (2016), Deep biaffine attention for neural dependency parsing, *CoRR*. http://arxiv.org/abs/1611.01734.

Fauconnier, Jean-Philippe (2015), French word embeddings. http://fauconnier.github.io.

Joshi, Aravind K and Bangalore Srinivas (1994), Disambiguation of super parts of speech (or supertags): Almost parsing, *Proceedings of the 15th conference on Computational linguistics-Volume 1*, Association for Computational Linguistics, pp. 154–160.

Joshi, Aravind K and Yves Schabes (1997), Tree-adjoining grammars, *Handbook of formal languages*, Springer, pp. 69–123.

Kallmeyer, Laura and Giorgio Satta (2009), A polynomial-time parsing algorithm for tt-mctag, *Proceedings of the Joint Conference of the 47th Annual Meeting of the Association for Computational Linguistics and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2 - Volume 2*, ACL '09, Association for Computational Linguistics, Stroudsburg, PA, USA, pp. 994–1002. http://dl.acm.org/citation.cfm?id=1690219.1690285.

Kallmeyer, Laura and Rainer Osswald (2013), Syntax-driven semantic frame composition in lexicalized tree adjoining grammars, *Journal of Language Modelling* **1** (2), pp. 267–330.

Kallmeyer, Laura, Rainer Osswald, and Robert D. Van Valin, Jr. (2013), Tree wrapping for Role and Reference Grammar, *in* Morrill, G. and M.-J. Nederhof, editors, *Formal Grammar 2012/2013*, Vol. 8036 of *LNCS*, Springer, pp. 175–190.

Kasai, Jungo, Bob Frank, Tom McCoy, Owen Rambow, and Alexis Nasr (2017), TAG parsing with neural networks and vector representations of supertags, *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pp. 1712–1722.

Kasai, Jungo, Robert Frank, Pauli Xu, William Merrill, and Owen Rambow (2018), End-to-end graph-based TAG parsing with neural networks, *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, Association for Computational Linguistics, New Orleans, Louisiana, pp. 1181–1194. https://www.aclweb.org/anthology/N18-1107.

Kitaev, Nikita and Dan Klein (2018), Constituency parsing with a self-attentive encoder, *arXiv preprint arXiv:1805.01052*.

Klein, Dan and Christopher D. Manning (2001), Parsing and Hypergraphs, *Proceedings of the Seventh International Workshop on Parsing Technologies (IWPT-2001), 17-19 October 2001, Beijing, China*, Tsinghua University Press.

Le, Ngoc Luyen and Yannis Haralambous (2019), CCG Supertagging Using Morphological and Dependency Syntax Information, *International Conference on Computational Linguistics and Intelligent Text Processing*, Springer.

Lewis, Mike and Mark Steedman (2014), A* CCG Parsing with a Supertag-factored Model, *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Association for Computational Linguistics, pp. 990–1000. http://aclweb.org/anthology/D14-1107.

Lewis, Mike, Kenton Lee, and Luke Zettlemoyer (2016), LSTM CCG parsing, *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 221–231.

Liu, Yudong and Anoop Sarkar (2007), Experimental evaluation of ltag-based features for semantic role labeling, *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pp. 590–599.

Liu, Yudong and Anoop Sarkar (2009), Exploration of the ltag-spinal formalism and treebank for semantic role labeling, *Proceedings of the 2009 Workshop on Grammar Engineering Across Frameworks, ACL-IJCNLP 2009*, pp. 1–9.

Ma, Xuezhe and Eduard Hovy (2016), End-to-end sequence labeling via bi-directional LSTM-CNNS-CRF, *arXiv preprint arXiv:1603.01354*.

Nederhof, Mark-Jan (2003), Weighted deductive parsing and knuth's algorithm, *Computational Linguistics* **29** (1), pp. 135–143, MIT Press.

Osswald, Rainer and Laura Kallmeyer (2018), Towards a formalization of role and reference grammar, *in* Kailuweit, Rolf, Lisann Knkel, and Eva Staudinger, editors, *Applying and Expanding Role and Reference Grammar.*, Albert-Ludwigs-Universitt, Universittsbibliothek. [NIHIN studies], Freiburg, pp. 355–378.

Rambow, Owen, K. Vijay-Shanker, and David Weir (1995), D-tree grammars, *33rd Annual Meeting of the Association for Computational Linguistics*, Association for Computational Linguistics, Cambridge, Massachusetts, USA, pp. 151–158. https://www.aclweb.org/anthology/P95-1021.

Resnik, Philip (1992), Probabilistic tree-adjoining grammar as a framework for statistical natural language processing, *COLING 1992 Volume 2: The 15th International Conference on Computational Linguistics*. https://www.aclweb.org/anthology/C92-2065.

Sarkar, Anoop (2000), Practical experiments in parsing using tree adjoining grammars, *Proceedings of the Fifth International Workshop on Tree Adjoining Grammar and Related Frameworks (TAG+ 5)*, pp. 193–198.

Sarkar, Anoop (2007), Combining supertagging and lexicalized tree-adjoining grammar parsing, *Complexity of Lexical Descriptions and its Relevance to Natural Language Processing: A Supertagging Approach* p. 113, MIT Press Boston, MA, USA.

Schabes, Yves and Richard C Waters (1995), Tree insertion grammar: a cubic-time, parsable formalism that lexicalizes context-free grammar without changing the trees produced, *Computational Linguistics* **21** (4), pp. 479–513.

Seddah, Djamé, Reut Tsarfaty, Sandra Kübler, Marie Candito, Jinho D. Choi, Richárd Farkas, Jennifer Foster, Iakes Goenaga, Koldo Gojenola Galletebeitia, Yoav Goldberg, Spence Green, Nizar Habash, Marco Kuhlmann, Wolfgang Maier, Joakim Nivre, Adam Przepiórkowski, Ryan Roth, Wolfgang Seeker, Yannick Versley, Veronika Vincze, Marcin Woliński, Alina Wróblewska, and Eric Villemonte de la Clergerie (2013), Overview of the SPMRL 2013 shared task: A cross-framework evaluation of parsing morphologically rich languages, *Proceedings of the Fourth Workshop on Statistical Parsing of Morphologically-Rich Languages*, Association for Computational Linguistics, Seattle, Washington, USA, pp. 146–182. https://www.aclweb.org/anthology/W13-4917.

Shieber, Stuart M, Yves Schabes, and Fernando CN Pereira (1995), Principles and implementation of deductive parsing, *The Journal of logic programming* **24** (1), pp. 3–36, North-Holland.

Srivastava, Rupesh Kumar, Klaus Greff, and Jürgen Schmidhuber (2015), Highway networks, *arXiv preprint arXiv:1505.00387*.

Stern, Mitchell, Jacob Andreas, and Dan Klein (2017), A minimal span-based neural constituency parser, *arXiv preprint arXiv:1705.03919*.

Van Noord, Gertjan, Gosse Bouma, Frank Van Eynde, Daniel De Kok, Jelmer Van der Linde, Ineke Schuurman, Erik Tjong Kim Sang, and Vincent Vandeghinste (2013), Large scale syntactic annotation of written Dutch: Lassy, *Essential speech and language technology for Dutch*, Springer, pp. 147–164.

Van Valin Jr, Robert D. (2005), *Exploring the syntax-semantics interface*, Cambridge University Press.

Van Valin, Robert D., Jr. and Randy LaPolla (1997), *Syntax: Structure, meaning and function*, Cambridge University Press.

Waszczuk, Jakub (2017), *Leveraging MWEs in practical TAG parsing: towards the best of the two worlds*, PhD thesis.

Waszczuk, Jakub, Agata Savary, and Yannick Parmentier (2016a), Enhancing practical TAG parsing efficiency by capturing redundancy, *21st International Conference on Implementation and Application of Automata (CIAA 2016)*, Proceedings of the 21st International Conference on Implementation and Application of Automata (CIAA 2016), Séoul, South Korea. https://hal.archives-ouvertes.fr/hal-01309598.

Waszczuk, Jakub, Agata Savary, and Yannick Parmentier (2016b), Promoting multiword expressions in A* TAG parsing, *COLING 2016, 26th International Conference on Computational Linguistics, Proceedings of the Conference: Technical Papers, December 11-16, 2016, Osaka, Japan*, pp. 429–439. http://aclweb.org/anthology/C/C16/C16-1042.pdf.

Waszczuk, Jakub, Agata Savary, and Yannick Parmentier (2017), Multiword Expression-Aware A* TAG Parsing Revisited, *Proceedings of the 13th International Workshop on Tree Adjoining Grammars and Related Formalisms*, Association for Computational Linguistics, pp. 84–93. http://aclweb.org/anthology/W17-6209.

Woliński, Marcin, Katarzyna Głowińska, and Marek Świdziński (2011), A preliminary version of składnicaa treebank of polish, *Proceedings of the 5th Language & Technology Conference, Poznań*, pp. 299–303.

Xia, Fei (1999), Extracting tree adjoining grammars from bracketed corpora, *Proceedings of the 5th Natural Language Processing Pacific Rim Symposium (NLPRS-99)*, pp. 398–403.

XTAG Research Group (1998), A lexicalized tree adjoining grammar for english, *arXiv preprint cs/9809024*.

Yoshikawa, Masashi, Hiroshi Noji, and Yuji Matsumoto (2017), A* CCG parsing with a supertag and dependency factored model, *CoRR*. http://arxiv.org/abs/1704.06936.

Statistical Parsing of Tree Wrapping Grammars

Tatiana Bladier Jakub Waszczuk Laura Kallmeyer
Heinrich Heine University of Düsseldorf
Universitätsstraße 1, 40225 Düsseldorf, Germany
{bladier,waszczuk,kallmeyer}@phil.hhu.de

Abstract

We describe an approach to statistical parsing with Tree-Wrapping Grammars (TWG). TWG is a tree-rewriting formalism which includes the tree-combination operations of substitution, sister-adjunction and tree-wrapping substitution. TWGs can be extracted from constituency treebanks and aim at representing long distance dependencies (LDDs) in a linguistically adequate way. We present a parsing algorithm for TWGs based on neural supertagging and A* parsing. We extract a TWG for English from the treebanks for Role and Reference Grammar and discuss first parsing results with this grammar.

1 Introduction

We present a statistical parsing approach for Tree-Wrapping Grammar (TWG) (Kallmeyer et al., 2013). TWG is a grammar formalism closely related to Tree-Adjoining Grammar (TAG) (Joshi and Schabes, 1997), which was originally developed with regard to the formalization of the typologically oriented Role and Reference Grammar (RRG) (Van Valin and LaPolla, 1997; Van Valin Jr, 2005). TWG allows for, among others, a more linguistically adequate representation of *long distance dependencies (LDDs)* in sentences, such as topicalization or long distance wh-movement. In the present paper we show a grammar extraction algorithm for TWG, propose a TWG parser, and discuss parsing results for the grammar extracted from the RRG treebanks RRGbank and RRGparbank[1] (Bladier et al., 2018).

Similarly to TAG, TWG has the elementary tree combination operations of *substitution* and *sister-adjunction*. Additionally, TWG includes the operation of *tree-wrapping substitution*, which accounts for preserving the connection between the parts of the discontinuous constituents. Operations similar to tree-wrapping substitution were proposed by (Rambow et al., 1995) as *subsertion* in D-Tree Grammars (DTG) and by (Rambow et al., 2001) as *generalized substitution* in D-Tree substitution grammar (DSG). To our best knowledge, no statistical parsing approach was proposed for DTG or DSG. An approach to symbolic parsing for TWGs with edge features was proposed in (Arps et al., 2019). In this work, we propose a statistical parsing approach for TWG and extend the pipeline based on supertagging and A* algorithm (Waszczuk, 2017; Bladier et al., 2019) originally developed for TAG to be applied to TWG.

The contributions of the paper are the following: 1) We present the first approach to statistical parsing for Tree-Wrapping Grammars. 2) We propose an extraction algorithm for TWGs based on the algorithm developed for TAG by (Xia, 1999). 3) We extend and modify the neural A* TAG-parser (Waszczuk, 2017; Kasai et al., 2018; Bladier et al., 2019) to handle the operation of tree-wrapping substitution.

2 Long distance dependencies and wrapping substitution in TWG

TWGs consist of elementary trees which can be combined using the operations a) *substitution* (replacing a leaf node with a tree), b) *sister adjunction* (adding a new daughter to an internal node) and c) *tree-wrapping substitution* (adding a tree with a d(ominance)-edge by substituting the lower part of the d-edge for a leaf node and merging the upper node of the d-edge with the root of the target tree, see Fig. 1). The latter is

[1] https://rrgbank.phil.hhu.de, https://rrgparbank.phil.hhu.de

Proceedings of the 28th International Conference on Computational Linguistics, pages 6759–6766
Barcelona, Spain (Online), December 8-13, 2020

used to capture long distance dependencies (LDDs), see the wh-movement in Fig. 1. Here, the left tree with the *d-edge* (depicted as a dashed edge) gets split; the lower part fills a substitution slot while the upper part merges with the root of the target tree. TWG is more powerful than TAG (Kallmeyer, 2016). The reason is that a) TWG allows for more than one wrapping substitution stretching across specific nodes in the derived tree and b) the two target nodes of a wrapping substitution (the substitution node and the root node) need not come from the same elementary tree, which makes wrapping non-local compared to adjunction in TAG.

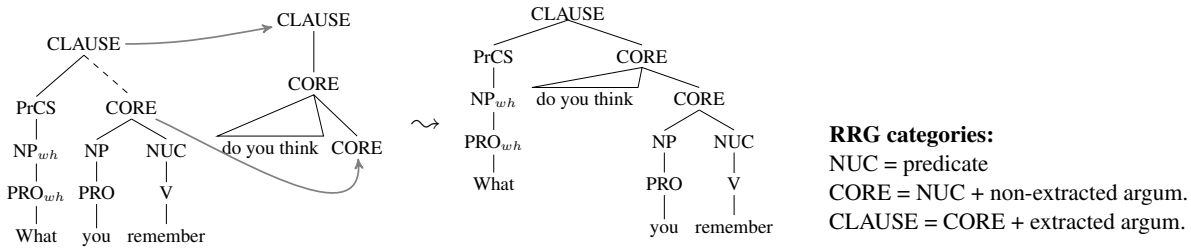


Figure 1: Tree-wrapping substitution for the sentence *"What do you think you remember"* with long-distance wh-movement.

Linguistic phenomena leading to LDD differ across languages. Among LDDs in English are some cases of extraction of a phrase to a non-canonical position with respect to its head, which is typically fronting in English (Candito and Seddah, 2012). We identified the following LDD variants in our data which can be captured with tree-wrapping substitution: long-distance relativization, long-distance wh-movement, and long-distance topicalization, which we discuss in Section 6.[2] LDD cases are rather rare in the data, which is partly due to the RRG analysis of operators such as modals, which do not embed CORE constituents (in contrast to, for example, the analyses in the Penn Treebank). Only 0,11 % of tokens in our experiment data (including punctuation) are dislocated from their canonical position in sentence to form an LDD. This number is on a par with 0,16 % of tokens reported by (Candito and Seddah, 2012) for French data.

3 Statistical Parsing with TWGs

The proposed A* TWG parser[3] is a direct extension of the simpler A* TAG parser described in (Waszczuk, 2017). The parser is specified in terms of weighted deduction rules (Shieber et al., 1995; Nederhof, 2003) and can be also seen as a weighted variant of the symbolic TWG parser (Arps et al., 2019). As in (Bladier et al., 2019), both TWG elementary trees (supertags) and dependency links are weighted, a schema also used in A* CCG parsing (Yoshikawa et al., 2017). These weights come directly from a neural supertagger and dependency parser, similar to the one proposed by (Kasai et al., 2018). Parsing consists then in finding a best-weight derivation among the derivations that can be constructed based on the deduction rules for a given sentence.

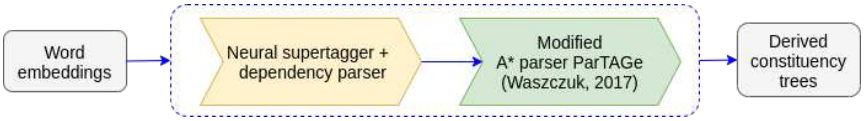


Figure 2: Pipeline of our neural statistical TWG parsing architecture.

The supertagger takes on input a sequence of word embeddings[4] $(x_i)_{i=1}^n$, to which a 2-layer BiLSTM transducer is applied to provide the contextualized word representations $(h_i)_{i=1}^n$, common to all subsequent tasks: POS tagging, TWG supertagging, and dependency parsing. On top of that, we apply two additional

[2] Another potential LDD cases in English are *it-clefts* (for example "*It was the uncertainty that Mr Lorin **feared***"). Although we have not found this LDD variant in our data, our parsing method will work for these cases as well.

[3] The parser, the TWG extraction code and the recipes to reproduce the experiments described in this paper are available at https://github.com/TaniaBladier/Statistical_TWG_Parsing.

[4] In our experiments (see Sec. 5), we used `fastText` (Bojanowski et al., 2016) to obtain the word vector representation.

2-layer BiLSTM transducers in order to obtain the supertag- and dependency-specific word representations:

$$(h_1^{(sup)}, \ldots, h_n^{(sup)}) = \texttt{BiLSTM}_s(h_1, \ldots, h_n) \quad (1)$$

$$(h_1^{(dep)}, \ldots, h_n^{(dep)}) = \texttt{BiLSTM}_d(h_1, \ldots, h_n) \quad (2)$$

The supertag-specific representations are used to predict both supertags and POS tags (POS tagging is a purely auxiliary task, since POS tags are fully determined by the supertags):

$$\Pr(sup(i)) = \texttt{softmax}(\texttt{Linear}_s(h_i^{(sup)})) \quad (3)$$

$$\Pr(pos(i)) = \texttt{softmax}(\texttt{Linear}_p(h_i^{(sup)})) \quad (4)$$

Finally, the dependency parsing component is based on biaffine scoring (Dozat and Manning, 2017), in which the head and dependent representations are obtained by applying two feed-forward networks to the dependency-specific word representations, $\texttt{hd}_i = \texttt{FF}_{hd}(h_i^{(dep)})$ and $\texttt{dp}_i = \texttt{FF}_{dp}(h_i^{(dep)})$. The score of word j becoming the head of word i is then defined as:

$$\phi(i, j) = \texttt{dp}_i^T M\ \texttt{hd}_j + b^T \texttt{hd}_j, \quad (5)$$

where M is a matrix and b is a bias vector.[5]

Extending the TAG parser to TWG involved adapting the weighted deduction rules to handle wrapping substitution as well as updating the corresponding implementation with appropriate index structures to speed up querying the chart. The A* heuristic is practically unchanged and it is both admissible (by construction) and monotonic (checked at run time), which guarantees that the first derivation found by the parser is the one with the best weight. The scheme of our parsing architecture is shown in Fig. 2. In Appendix A we provide details on modifications we have applied to the A* parser to handle the tree-wrapping substitution.

4 TWG extraction

To extract a TWG from RRGbank and RRGparbank, we adapt the top-down grammar extraction algorithm developed by (Xia, 1999) for TAG. While inital and sister-adjoining trees can be extracted following this algorithm, we added a new procedure to extract d-edge trees for wrapping substitution operation. Extraction of initial and sister-adjoining elementary trees requires manually defined percolation tables for marking head and modifier nodes. In order to extract d-edge elementary trees for LDDs, dependent constituents need to be marked prior to TWG extraction. In RRGbank and RRGparbank the constituents belonging to LDDs are indicated with features PRED-ID and NUC-ID and an index. These indicated parts alongside with the mother node are extracted to form a single tree with a *dominance link (d-edge)* (see for instance the elementary tree for "*What to say*" in Fig. 3). The remaining nodes plus the duplicated mother node and a substitution slot form the target tree, for example the tree for "*I'm trying*" in Fig. 3. Please find a more detailed formal description of our extraction algorithm along with a link to the percolation tables in Appendix B.

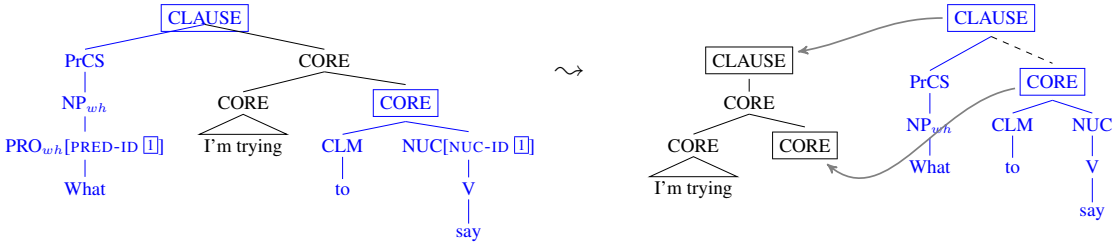


Figure 3: Extraction of a target tree and an elementary tree with a *dominance edge* (marked with dotted line). The nodes with PRED-ID and NUC-ID in the left tree identify the components of the LDD.

[5]The head representation $\texttt{hd}_0$ of the dummy root node is a parameter in this architecture.

5 Data and Parsing Experiments

We have taken the gold and silver data from RRGbank and the English part of RRGparbank[6]. The data is split in a train and a test set. We have taken 10 % of the sentences from the train set to create a dev. set. Thus, our train, dev. and test sets include 4960, 551, and 2145 trees, respectively. There are 46 constituents with LDDs in the train set, 5 in the dev. set and 27 in the test set. We extracted a TWG from this data and present in Table 1 statistics on the elementary tree templates (*supertags*) in the TWG.

Parameters	**TWG**
Supertags	3125
Supertags occuring once	1858
Avg. sentence length	10.97
Sentences	7656
# initial trees	1527
# sister-adjoining trees	1549
# d-edge trees	49

Table 1: Statistics on the extracted TWG.

We compare the parsing results with the parser DiscoDOP (van Cranenburgh and Bod, 2013) which is based on the discontinuous data-oriented parsing model. We also compare our results with the state-of-the-art transition-based parser Discoparset (Coavoux and Cohen, 2019). We evaluated[7] the overall performance of the parsers and also analyzed how well all three systems predict LDDs (see Tables 2 and 3). Unrelated to LDDs, the treebanks contain crossing branches (e.g., for operators and modifiers). Prior to TWG extraction, we decross these while keeping track of the transformation in order to be able to reverse it. For parsing with DiscoDOP and Discoparset, we added crossing branches for all LDDs. To evaluate LDD prediction with DiscoDOP and Discoparset we counted how many crossing branches were established in parsed trees. For ParTAGe we counted the LDD predictions as correct whenever the predicted supertags and dependencies indicated that the long distance element would be substituted to the elementary tree of the corresponding predicate. We counted partially correct LDDs in both parsing architectures as correctly predicted as long as the connection between the predicate and the fronted element was predicted.

	DiscoDOP		**Discoparset**		**ParTAGe**		**ParTAGe** gold POS	
	dev	test	dev	test	dev	test	dev	test
Unlabeled Attachment Score	–	–	–	–	87.74	87.67	85.13	84.64
Supertagging accuracy	–	–	–	–	74.25	75.81	77.50	77.52
POS-tagging accuracy	92.02	93.25	$94.24_{(+3.04)}$	$94.92_{(+2.69)}$	**94.63**	**95.07**	100.00	100.00
Exactly matching parses	29.04	32.87	$28.68_{(+8.89)}$	$28.30_{(+13.19)}$	**36.12**	**38.32**	38.64	38.73
Labeled F1	79.26	80.96	$83.57_{(+6.83)}$	$84.56_{(+6.39)}$	**85.26**	**85.26**	85.54	85.84
# sentences with parses	551	2145	551	2145	551	2145	546(−5)	2120(−25)

Table 2: Parsing results compared with DiscoDOP (van Cranenburgh et al., 2016) and Discoparset (Coavoux and Cohen, 2019). In case of Discoparset, the numbers in subscript represent the relative gain provided by BERT (Devlin et al., 2019) used in neither DiscoDOP nor ParTAGe experiments.

6 Error analysis for LDD prediction

We evaluated the performance of our parsing architecture with regard to the labeled F1-score and we also focused on prediction of the LDDs (see Tables 2 and 3). The results show that ParTAGe predicted the LDDs in the test data more accurately than the compared parsers. Please note that LDDs are generally rare in the corpus data and that we also had only about 5000 sentences in the training data.

Predicted LDDs	**DiscoDOP**	**Discoparset**	**ParTAGe**	
	test	test	test	test (gold POS)
# true positives	13	14	22	18
# false positives	7	0	0	0
# false negatives	14	13	5	9

Table 3: Prediction of LDDs on test data.

Some mistakes resulted from the wrong prediction of a POS tag which

[6] *Gold annotated data* means that data were annotated and approved by at least two annotators of RRGbank or RRGparbank and *silver data* means an annotation by one linguist.

[7] We use the evaluation parameters distributed together with DiscoDOP for constituency parsing evaluation. Our evaluation file is available at https://github.com/TaniaBladier/Statistical_TWG_Parsing/blob/main/experiments/eval.prm.

leads to the parser confusing an LDD constituent with a construction without LDD. For example, in (1), the word *"is"* should have POS tag V, but the parsing system erroneously labels it as AUX (= auxiliary) and thus interprets the wh-element as a predicate. In order to check our assumption about POS tags as a source of error, we have run an experiment in which we presented the parser with gold POS tags. Although this additional information helped to rule out the LDD errors in (1), restriction of the available supertags introduced new errors in LDD predictions (see Table 3) and also was the reason why some sentences could not be parsed (as shown in Table 2).

(1) a. *What is one* ***to think*** *of all this?* (*is* tagged AUX instead of V)
b. *[. . .] which he told her* ***to place*** *on her tongue* (*which* tagged CLM instead of PRO-REL)

In some cases where the relative or wh phrase of the LDD is an adjunct, as in (2), the parser incorrectly attaches it higher, taking it to be a modifier of the embedding verb.

(2) *And why do you imagine that we* ***bring*** *people to this place?*

Cases where the embedding verb also has a strong tendency to take a wh-element as argument sometimes get parsed incorrectly: In (3), *which* is analysed as an argument of *said*.

(3) *[. . .] slip of paper which they said* ***was the bill***

7 Conclusions and Outlook

We have presented a statistical parsing algorithm for parsing Tree-Wrapping Grammar - a grammar formalism inspired by TAG which aims at linguistically better representations of long distance dependencies. The LDDs in TWG are represented in a single elementary tree called *d-edge tree* which is combined with the target tree using *tree-wrapping substitution*. This operation allows to simultaneously put both parts of a discontinuous constituent to the corresponding slots of the target tree. We have extracted a TWG for English from two RRG treebanks and have compared our parsing experiments with the parser DiscoDOP based on the DOP parsing model and with the transition-based parser Discoparset. We have evaluated our parser on prediction of LDDs and could achieve more accurate results than the compared parsers.

In our future work we plan to explore TWG extraction and parsing for different languages, since the linguistic phenomena leading to LDDs vary across the languages. In particular, we have already started to work on extraction of TWGs for German and French. We plan to apply our TWG extraction and parsing algorithm to other constituency treebanks, for example French Treebank (Abeillé et al., 2003). We also plan to implement a slightly extended version of tree wrapping substitution which would allow to place the parts of discontinuous constituents in various slots between the nodes of the target tree.

Acknowledgements

We thank three anonymous reviewers for their insightful comments, as well as Rainer Osswald and Robin Möllemann for their help with collecting the experimental data and fruitful discussions. This work was carried out as a part of the research project TREEGRASP (treegrasp.phil.hhu.de) funded by a Consolidator Grant of the European Research Council (ERC).

References

Anne Abeillé, Lionel Clément, and François Toussenel. 2003. Building a treebank for french. In *Treebanks*, pages 165–187. Springer.

David Arps, Tatiana Bladier, and Laura Kallmeyer. 2019. Chart-based RRG parsing for automatically extracted and hand-crafted RRG grammars. University at Buffalo, Role and Reference Grammar RRG Conference 2019.

Tatiana Bladier, Andreas van Cranenburgh, Kilian Evang, Laura Kallmeyer, Robin Möllemann, and Rainer Osswald. 2018. RRGbank: a Role and Reference Grammar Corpus of Syntactic Structures Extracted from the Penn Treebank. In *Proceedings of the 17th International Workshop on Treebanks and Linguistic Theories (TLT 2018),*

December 13–14, 2018, Oslo University, Norway, number 155, pages 5–16. Linköping University Electronic Press.

Tatiana Bladier, Jakub Waszczuk, Laura Kallmeyer, and Jörg Janke. 2019. From partial neural graph-based LTAG parsing towards full parsing. *Computational Linguistics in the Netherlands Journal*, 9:3–26, Dec.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2016. Enriching word vectors with subword information. *arXiv preprint arXiv:1607.04606*.

Marie Candito and Djamé Seddah. 2012. Effectively long-distance dependencies in French: Annotation and parsing evaluation.

Maximin Coavoux and Shay B. Cohen. 2019. Discontinuous constituency parsing with a stack-free transition system and a dynamic oracle. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 204–217, Minneapolis, Minnesota, June. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June. Association for Computational Linguistics.

Timothy Dozat and Christopher D. Manning. 2017. Deep biaffine attention for neural dependency parsing.

Aravind K Joshi and Yves Schabes. 1997. Tree-adjoining grammars. In *Handbook of formal languages*, pages 69–123. Springer.

Laura Kallmeyer, Rainer Osswald, and Robert D. Van Valin, Jr. 2013. Tree Wrapping for Role and Reference Grammar. In G. Morrill and M.-J. Nederhof, editors, *Formal Grammar 2012/2013*, volume 8036 of *LNCS*, pages 175–190. Springer.

Laura Kallmeyer. 2016. On the mild context-sensitivity of k-Tree Wrapping Grammar. In Annie Foret, Glyn Morrill, Reinhard Muskens, Rainer Osswald, and Sylvain Pogodalla, editors, *Formal Grammar: 20th and 21st International Conferences, FG 2015, Barcelona, Spain, August 2015, Revised Selected Papers. FG 2016, Italy, August 2016, Proceedings*, number 9804 in Lecture Notes in Computer Science, pages 77–93, Berlin. Springer.

Jungo Kasai, Robert Frank, Pauli Xu, William Merrill, and Owen Rambow. 2018. End-to-end graph-based TAG parsing with neural networks. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1181–1194, New Orleans, Louisiana, June. Association for Computational Linguistics.

Mark-Jan Nederhof. 2003. Squibs and Discussions: Weighted Deductive Parsing and Knuth's Algorithm. *Computational Linguistics*, 29(1).

Owen Rambow, K. Vijay-Shanker, and David Weir. 1995. D-Tree Grammars. In *Proceedings of ACL*.

Owen Rambow, K Vijay-Shanker, and David Weir. 2001. D-tree substitution grammars. *Computational Linguistics*, 27(1):87–121.

Stuart M Shieber, Yves Schabes, and Fernando CN Pereira. 1995. Principles and implementation of deductive parsing. *The Journal of logic programming*, 24(1):3–36.

Andreas van Cranenburgh and Rens Bod. 2013. Discontinuous parsing with an efficient and accurate dop model. In *Proceedings of the International Conference on Parsing Technologies (IWPT 2013)*. Citeseer.

Andreas van Cranenburgh, Remko Scha, and Rens Bod. 2016. Data-oriented parsing with discontinuous constituents and function tags. *Journal of Language Modelling*, 4(1):57–111.

Robert D. Van Valin, Jr. and Randy LaPolla. 1997. *Syntax: Structure, meaning and function*. Cambridge University Press.

Robert D. Van Valin Jr. 2005. *Exploring the syntax-semantics interface*. Cambridge University Press.

Jakub Waszczuk. 2017. *Leveraging MWEs in practical TAG parsing: towards the best of the two worlds*. Ph.D. thesis.

Fei Xia. 1999. Extracting tree adjoining grammars from bracketed corpora. In *Proceedings of the 5th Natural Language Processing Pacific Rim Symposium (NLPRS-99)*, pages 398–403.

Masashi Yoshikawa, Hiroshi Noji, and Yuji Matsumoto. 2017. A* CCG parsing with a supertag and dependency factored model. *CoRR*, abs/1704.06936.

Appendix A. Specification of the TWG parser

AX:	$\frac{}{(0,\emptyset) : (N{\rightarrow}\bullet\alpha,i,i,[])}$	$i{\in}\{0,\ldots,n{-}1\}$ $N{\rightarrow}\alpha$ is a rule
SC:	$\frac{(w,m) : (N{\rightarrow}\alpha\bullet M\beta,i,j,\Gamma)}{(w,m) : (N{\rightarrow}\alpha M\bullet\beta,i,j{+}1,\Gamma)}$	$\ell(M){=}s_{j+1}$
DE:	$\frac{(w,m) : (N{\rightarrow}\alpha\bullet,i,j,\Gamma)}{(w,m) : (N,i,j,\Gamma,ws?)}$	$ws?{=}yes \iff dnode(N)$
CS:	$\frac{(w_1,m_1) : (N{\rightarrow}\alpha\bullet M\beta,i,j,\Gamma_1) \quad (w_2,m_2) : (M,j,k,\Gamma_2,no)}{(w_1{+}w_2,m_1{\oplus}m_2) : (N{\rightarrow}\alpha M\bullet\beta,i,k,\Gamma_1{\oplus}\Gamma_2)}$	
SU:	$\frac{(w_1,m_1) : (N{\rightarrow}\alpha\bullet M\beta,i,j,\Gamma_1) \quad (w_2,m_2) : (R,j,k,\Gamma_2,no)}{(w_1{+}w_2{+}\omega(R,N),m_1{\oplus}m_2) : (N{\rightarrow}\alpha M\bullet\beta,i,k,\Gamma_1{\oplus}\Gamma_2)}$	$leaf(M)$ $root(R) \wedge \neg sister(R)$ $\ell(M){=}\ell(R)$
SA:	$\frac{(w_1,m_1) : (N{\rightarrow}\alpha\bullet\beta,i,j,\Gamma_1) \quad (w_2,m_2) : (M,j,k,\Gamma_2,no)}{(w_1{+}w_2{+}\omega(M,N),m_1{\oplus}m_2) : (N{\rightarrow}\alpha\bullet\beta,i,k,\Gamma_1{\oplus}\Gamma_2)}$	$\ell(M){=}\ell(N) \wedge sister(M)$ $\neg sister(N)$
PW:	$\frac{(w_1,m_1) : (N{\rightarrow}\alpha\bullet M\beta,i,j,\Gamma_1) \quad (w_2,m_2) : (D,j,k,\Gamma_2,yes)}{(w_1,m_1[j{\Rightarrow}w_2{+}sum(m_2){+}A(D)]) : (N{\rightarrow}\alpha M\bullet\beta,i,k,\Gamma_1{\oplus}[(j,k,\ell(D))])}$	$leaf(M)$ $\ell(M){=}\ell(D)$
CW:	$\frac{(w_1,m_1) : (R,i,j,\Gamma_1{\oplus}[(f_1,f_2,y)]{\oplus}\Gamma_2,ws?) \quad (w_2,m_2) : (D,f_1,f_2,\Gamma_3,yes)}{(w_1{+}w_2{+}\omega(R,D),m_1[f_1{\Rightarrow}\bot]{\oplus}m_2) : (D,i,j,\Gamma_1{\oplus}\Gamma_3{\oplus}\Gamma_2,no)}$	$root(R) \wedge y{=}\ell(D)$ $\ell(parent(D)){=}\ell(R)$ $\neg sister(R)$

Table 4: Weighted deduction rules of the TWG parser

Our TWG parser is specified in terms of weighted deduction rules (Shieber et al., 1995; Nederhof, 2003). Each deduction rule (see Table 4) takes the form of a set of antecedent items, presented above the horizontal line, from which the consequent item (below the horizontal line) can be deduced, provided that the corresponding conditions (on the right) are satisfied. The specification of the TWG parser consists of 8 deduction rules which constitute a blend of the TAG parser (Bladier et al., 2019) with the symbolic TWG parser (Arps et al., 2019). Here, we assume familiarity with both these parsers and limit ourselves to explaining the features specific to the statistical TWG parser.

Weights. A pair (w, m) is assigned to each chart item via deduction rules, where w is the inside weight, i.e., the weight of the inside derivation, and m is a map assigning weights to the individual gaps in the corresponding gap list Γ. Since each gap in Γ can be uniquely identified by its starting position, we use the starting positions as keys in m. The need to use a map (dictionary) data structure instead of a single scalar value, as in the TAG parser, stems from the `CW` rule (*complete wrapping*), in which the calculation of the resulting weight map requires removing the weight corresponding to the gap (f_1, f_2, y).

We use $\emptyset$ to denote an empty map, $m[x \Rightarrow y]$ to denote m with y assigned to x, $m[x \Rightarrow \bot]$ to denote m with x removed from the set of keys (together with the corresponding value), and $sum(m)$ to denote the sum of values (weights) in the map m. We also re-use the concatenation operator $\oplus$ to represent map union. Whenever map union is used ($m_1 \oplus m_2$), the sets of keys of the two map arguments (m_1 and m_2) are guaranteed to be disjoint (an invariant which can be proved by induction over the deduction rules).

Heuristic. Given a chart item $\eta = (x, i, j, \Gamma)$ with the corresponding weights (w, m), the TWG A* heuristic (which provides a lower-bound estimate on the cost of parsing the remaining part of the sentence) is a straightforward generalization of the TAG A* heuristic used by (Bladier et al., 2019). In particular, it accounts for the total minimal cost of scanning each word outside the span (i, j), as well as the words remaining in the gaps in Γ. Thus, in constrast with the TAG heuristic, since there can be many gaps in Γ, the sum of the weights in the map m has to be accounted for.

Appendix B. TWG extraction algorithm

1. **Decross tree branches.** First, for local discontinuous constituents (for instance NUCs consisting of a verb and a particle in German), we split the constituent into two components (e.g., NUC1 and NUC2), both attached to the mother of the original discontinuous node.

   Second, if a tree τ still has crossing branches, the tree is traversed top-down from left to right and among its subtrees those trees are identified whose root labels contain one of the following strings: OP-, -PERI, -TNS, CDP, or VOC. For each such subtree γ in question with r being its root, we choose the highest node v below the next left[8] sibling of r such that the rightmost leaf dominated by v immediately precedes the leftmost leaf dominated by r. If r and v are not yet siblings, γ is reattached to the parent of v. If the subtree in question has no left siblings, it is reattached to the right in a corresponding way. After this step, it should be checked if the tree τ still contains crossing branches. If yes, the process of decrossing branches is continued by applying the steps above to the next subtree in question.

2. **Extract LDDs.** Then we traverse each tree τ in a top-down left-to-right fashion and check for each subtree of τ whether it contains the following special markings for LDDs in its root label: PREDID=, NUCID= or REF=. The indexes identify the parts of the NLD which belong together. In case of an LDD, the parts of the minimal subtree which contain both parts of the LDD are extracted within a single tree with a *d-edge* (see the multicomponent NUC and CORE in Figure 3). The substitution site and the mother node are added to the remaining subtree in order to mark the nodes on which the wrapping substitution takes place (see Figure 3).

   After this step, an empty agenda is created and the extracted tree chunks and the pruned tree τ with the remaining nodes are placed into the agenda.

3. **Extract initial and sister-adjoining trees.** If no agenda with tree chunks was created in the previous step, an empty agenda is created in this step and the entire tree τ is placed into it. Each tree chunk in the agenda is traversed and the percolation tables[9] are used to decide for each subtree $\tau_1 \ldots \tau_n$ in the tree chunk whether it is a head, a complement or a modifier with respect to its parent. Initial trees for identified complements and sister-adjoining trees for identified modifiers are extracted recursively in the top-down fashion until each elementary tree has exactly one anchor site.

   Initial trees are extracted as follows: If a node of a subtree is identified as a complement, it is removed from the parent tree and the parent node is marked as a substitution slot. In order to extract sister-adjoining trees for identified modifier subtrees, the parent node of the subtree is copied and added as the new root node of the elementary tree with a special marking * on the root label.

[8] A node v_1 is left to another node v_2 if the leftmost leaf dominated by v_1 is left of the leftmost leaf dominated by v_2.

[9] Please find the code for our TWG extraction algorithm along with the percolation tables for head and modifier distinction in this repository: https://github.com/TaniaBladier/Statistical_TWG_Parsing.

Chapter 4

Building the Role and Reference Grammar Treebanks

This chapter presents publications as originally published, reprinted with permission from the corresponding publishers. The copyright of the original publications is held by the respective copyright holders, see the following copyright notices.

RRGbank: a Role and Reference Grammar Corpus of Syntactic Structures Extracted from the Penn Treebank

Tatiana Bladier[1], *Andreas van Cranenburgh*[2], *Kilian Evang*[1], *Laura Kallmeyer*[1], *Robin Möllemann*[1], *Rainer Osswald*[1]

(1) University of Düsseldorf, Germany
(2) University of Groningen, the Netherlands

{bladier,evang,kallmeyer,moellemann,osswald}@phil.hhu.de,
a.w.van.cranenburgh@rug.nl

ABSTRACT
This paper presents RRGbank, a corpus of syntactic trees from the Penn Treebank automatically converted to syntactic structures following Role and Reference Grammar (RRG). RRGbank is the first large linguistic resource in the RRG community and can be used in data-driven and data-oriented downstream linguistic applications. We show challenges encountered while converting PTB trees to RRG structures, introduce our annotation tool, and evaluate the automatic conversion process.

KEYWORDS: Role and Reference Grammar, RRG, treebank conversion, Penn Treebank.

1 Introduction

Wide empirical coverage is a touchstone for every grammatical theory. Treebanks have been widely used as training material for data-driven parsing approaches, data-oriented language processing, statistical linguistic studies, or machine learning throughout the last decades. However, no large linguistic resource exists for the framework of Role and Reference Grammar (RRG; Van Valin and LaPolla, 1997; Van Valin, 2005) so far. In this paper we describe the development of the first annotated corpus of RRG structures[1] created through (semi-)automatic conversion of the Penn Treebank.

Providing a treebank resource to the RRG community will be useful for several reasons: (i) it will be a valuable resource for corpus-based investigations in the context of linguistic modeling using RRG and in the context of formalizing RRG, which is needed for a precise understanding of the theory and for using it in NLP contexts. Efforts towards a formalization of RRG as a tree-rewriting grammar have already been made recently (Kallmeyer et al., 2013; Kallmeyer, 2016; Kallmeyer and Osswald, 2017). (ii) In the context of implementing precision grammars, at least for English, an RRG treebank is useful for testing the grammar and evaluating its coverage. (iii) It will enable supervised data-driven approaches to RRG parsing (grammar induction and probabilistic parsing). (iv) Finally, and more immediately, the specification of the treebank transformation yields valuable new insights into RRG analyses of English syntax — since, even though RRG has covered a large range of typologically different languages, compared to other theories, English has not been considered much.

Since manual annotation is very time-consuming, we decided to (semi-)automatically derive RRGbank from an existing treebank. For this, we chose the Penn Treebank (PTB; Marcus et al., 1993) because of its large size and and availability of additional layers such as OntoNotes (Hovy et al., 2006) which may be used to enrich RRGbank in the future. The PTB has been used in the past, among others, for deriving CCGbank, a corpus of Combinatory Categorial Grammar derivations (Hockenmaier and Steedman, 2007). We decided to start from the original PTB rather than CCGbank because its phrase structure trees are more similar to RRG than CCG derivations, and to avoid possible compounding of errors in automatic conversion. A different route to creating treebanks is taken by the LinGO Redwoods and ParGram approaches to dynamic treebanking for HPSG and LFG, respectively (Oepen et al., 2004; Flickinger et al., 2012; Sulger et al., 2013). These projects made use of manually developed grammars and parsers for the grammar formalisms in question, and then manually checked and selected the best output among all possible outputs. This is not an option for RRGbank at the moment because no wide-coverage computational grammar for RRG is available yet, but it may be a possible avenue in the future, after such a grammar has been extracted from RRGbank.

2 Syntactic Structures in Role and Reference Grammar

2.1 Brief Overview of RRG

RRG is intended to serve as an explanatory theory of grammar as well as a descriptive framework for field researchers. It is a functional theory of grammar which is strongly inspired by typological concerns and which aims at integrating syntactic, semantic and pragmatic levels of description (Van Valin, 2005, 2010). In RRG, there is a direct mapping between the semantic and syntactic representations of a sentence, unmediated by any kind of abstract syntactic representations. In particular, RRG is a strictly non-transformational theory and therefore does not make use of

[1] A demo version of the treebank is awailable at rrgbank.phil.hhu.de.

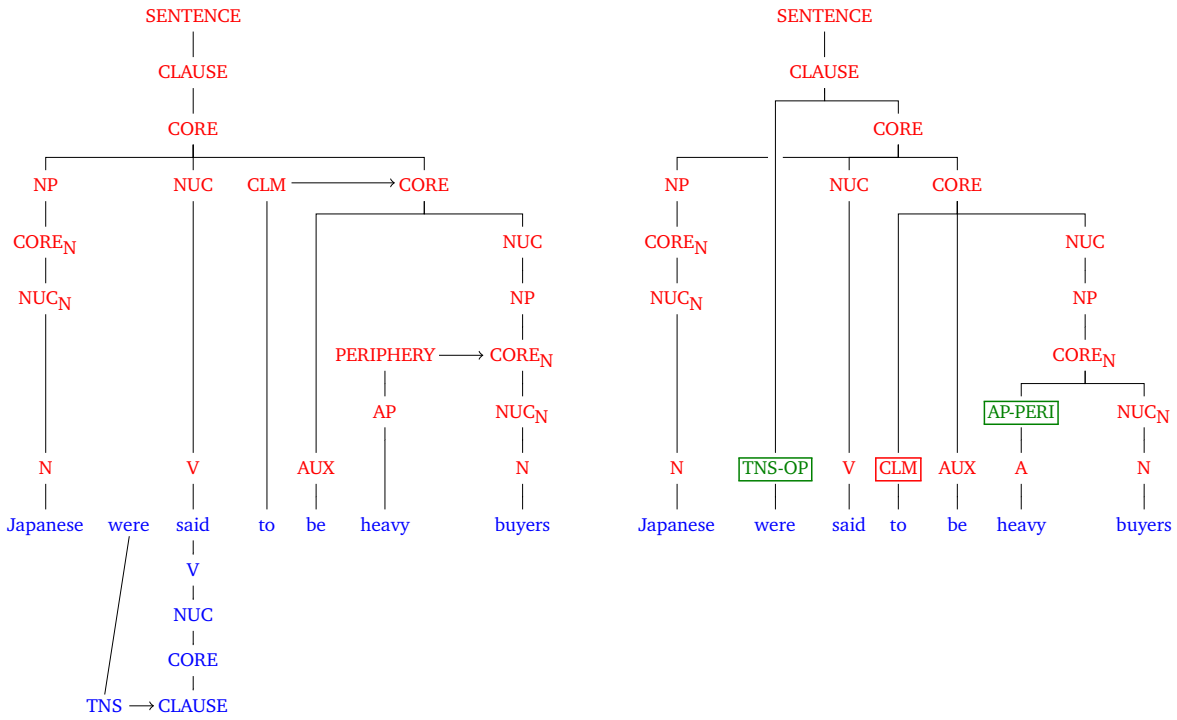


Figure 1: Representation of periphery, operator projection and clause-linkage-markers (CLMs) in standard RRG structures (left-hand side) and our notational variant (right-hand side).

traces and the like; there is only a single syntactic representation for a sentence that corresponds to its actual form. The mapping between the syntactic and semantic representations is subject to an elaborate system of linking constraints. For the purposes of the present paper, only the syntactic side of the representations is taken into account.

A key assumption of the RRG approach to syntactic analysis is a *layered structure* of the clause: The *core* layer consists of the *nucleus*, which specifies the (verbal) predicate, and its arguments. The *clause* layer contains the *core* plus extracted arguments, and each of the layers can have a *periphery* for attaching adjuncts (as shown for example in Figure 1). Another important feature of RRG is the separate representation of *operators*, which are closed-class morphosyntactic elements for encoding tense, modality, aspect, etc. Operators attach to those layers over which they take semantic scope. Since the surface order of the operators relative to arguments and adjuncts is much less transparent and often requires crossing branches, RRG represents the constituent structure and the operator structure as different *projections* of the clause (usually drawn above and below the sentence, respectively).

2.2 Tree Annotation Format for RRG Syntactic Structures

The standard data structure for constituent treebank annotations is trees, specifically, a single tree per sentence whose leaves are the tokens and whose structure and constituent and edge labels depend on the concrete annotation scheme. Many computational tools that process and use treebanks, such as query engines and parsers, rely on this format. By contrast, the usual notation for RRG syntactic structures departs from it in two ways (cf. Van Valin, 2005, 2010). Firstly, there are *two* trees per sentence, the constituent projection and the operator projection. A second idiosyncratic element is the use of arrows (instead of edges) for attaching peripheral

constituents (adjuncts) and clause linkage markers (CLMs), as well as the operators in the operator projection.

To resolve this discrepancy, we adopt a notational variant in which each RRG structure is represented as a single tree, exemplified in the right half of Figure 1. Firstly, note that the spine of the operator projection always mirrors that of the constituent projection. We thus simply identify the corresponding nodes (such as the CLAUSE, CORE, NUC and V nodes in the example) and attach operators in the same tree as other constituents. Secondly, we represent arrows as ordinary edges (and eliminate PERIPHERY nodes), whereby the roots of operators, peripheries and clause linkage markers become daughters of the nodes they attach to (see the TNS, CLM and AP nodes in the example). In order to still distinguish operators and peripheries, we decorate the labels of their roots with -OP and -PERI, respectively. Clause linkage markers are already distinguished by the root label CLM. As a result, we obtain trees that sometimes have crossing branches, resulting from operator scope (see Figure 1 on the right) or from adjunct scope (see Figure 2).

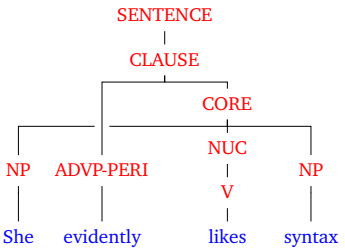


Figure 2: Periphery with crossing branches in RRG.

3 From Penn Treebank to RRGbank

We transform PTB annotations into RRG annotations by iteratively combining automatic conversion with manual correction. The process is sketched in Figure 3. We started with a small sample of sentences from the PTB ($n = 16$). Annotators with RRG expertise annotated these sentences from scratch with RRG trees, without looking at the PTB annotation, resulting in a small validation treebank. We then developed a conversion algorithm which transforms PTB trees into RRG trees. This development was *error-driven*, that is, the algorithm was improved step by step until its output was identical to the gold standard annotation.

We then used the developed algorithm to convert a larger sample ($n = 100$) of PTB trees to RRG.[2] The resulting "silver-standard" annotation was checked and corrected by annotators, using a click/drag/drop-based interface we developed, shown in Figure 7.[3] Correcting silver-standard data is less time-consuming than annotating from scratch; thus in this way we were able to increase the size of our validation treebank iteratively. After this step the set of conversion rules was updated again in order to correctly convert the entire new set of sentences. We plan to repeat the process of manual tree correction and updating the set of conversion rules to increase it further.

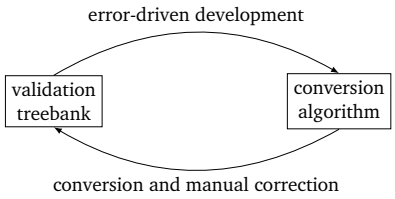


Figure 3: Annotation through iterative conversion and correction.

In the following subsections, we motivate and describe the conversion algorithm in more detail.

[2]The sentences were selected randomly from Sections 02–21 of the PTB, but we excluded sentences that contained fragmentary constituents (marked FRAG) or were longer than 25 tokens.

[3]See rrgbank.phil.hhu.de for a set of demo sentences.

3.1 Differences between PTB Trees and RRG Structures

We illustrate some important differences between PTB and RRG syntactic structures in Figure 4: First, the PTB assumes a separate VP projection inside clauses which does not include the subject, whereas RRG groups the subject together with other arguments in the *core*. This is due to RRG's semantic approach to argument realization. Second, while the PTB treats auxiliaries similarly to other verbs, RRG treats them as operators and attaches them according to their semantic scope. Copulas are the exception to this, as RRG attaches them within the *core*, signalling the following element to be the *nucleus*.

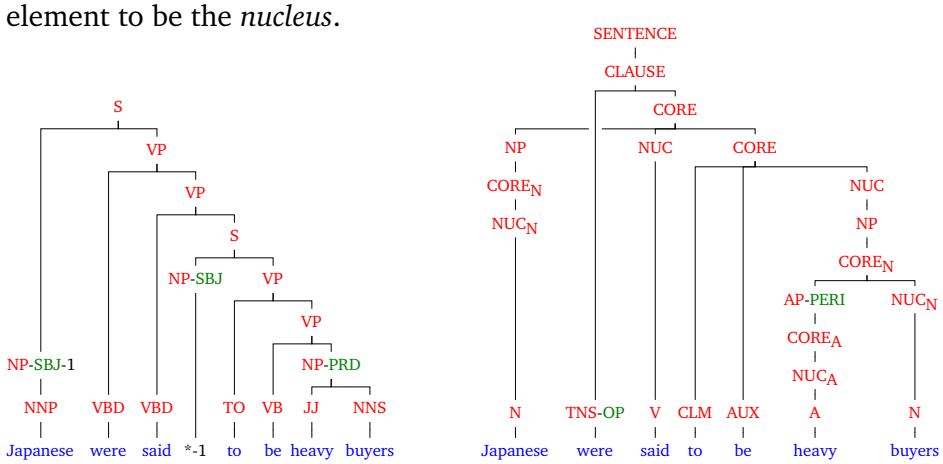


Figure 4: An example of a sentence from PTB (left tree) converted to RRG (right tree).

Third, the PTB uses *traces* to mark non-local dependencies whereas RRG has no such notion (the trace and the corresponding constituent in the PTB are marked with numbers, as shown in Figure 4 on the left-hand side). Fourth, adjuncts and other non-arguments like the adjective *heavy* in the example are analyzed as peripheries in RRG. Note that attachment of operators (as in Figure 4) and peripheries (as in Figure 2) according to their semantic scope can lead to crossing branches in RRG structures, which never occur in the PTB. Figure 5 shows the rules which were used for the conversion.

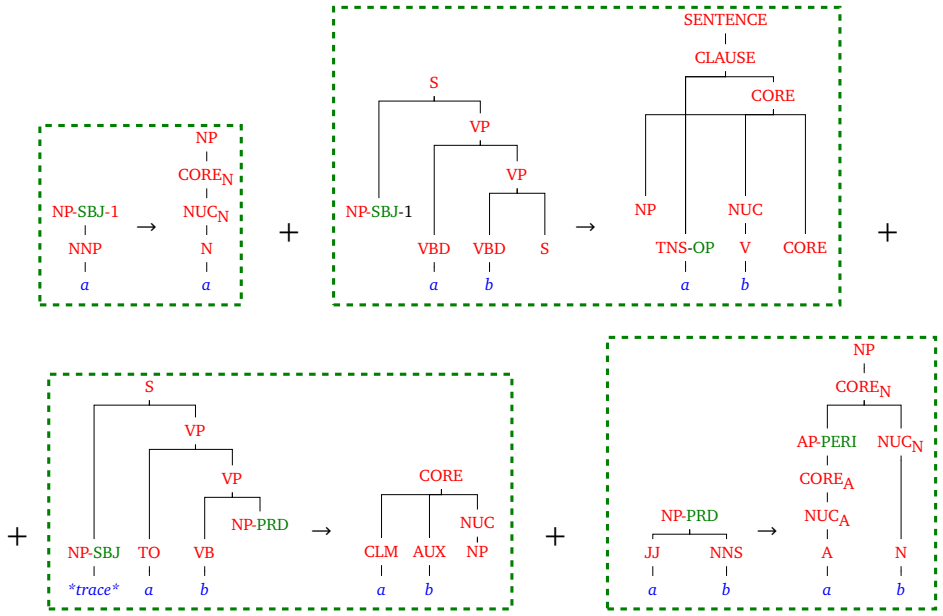


Figure 5: Conversion rules used for the sentence from Figure 4.

3.2 Outline of the Conversion Algorithm

The conversion algorithm was developed in an error-driven way, as outlined above. To each tree, the algorithm applies a series of rules. Each rule applies to specific constituents and may introduce, remove and relabel nodes. We started this conversion process by defining rules for the most frequent constituent types, with the aim of covering the whole treebank.

3.2.1 Conversion Algorithm: Regular Transformation Rules

In order to convert the PTB trees to RRG structures we created a relatively small set of general transformation rules applicable to all constituents of the same type throughout the PTB corpus. Some of these rules convert constituents with exactly one child node (Figure 6a). Other rules are used to convert larger constituents. For example, the rule in Figure 6b rewrites a basic sentence with a transitive verb to an RRG structure. Figure 6c shows one of the rules for transforming topicalized constituents to a left-detached position (LDP) in RRG.

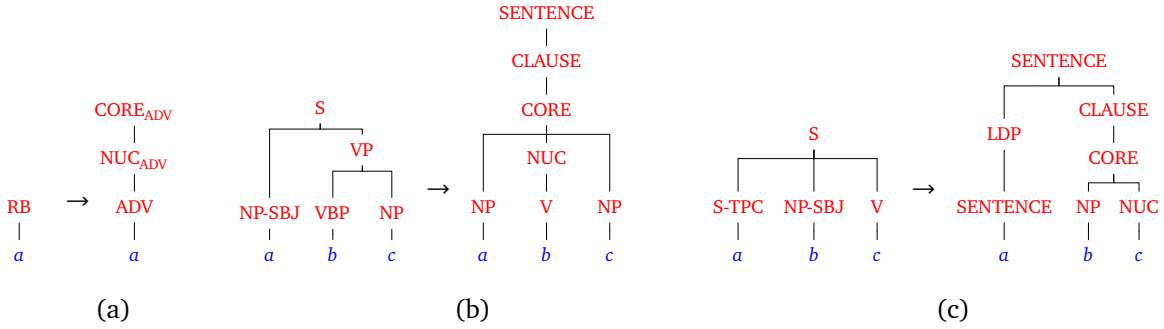


Figure 6: Three examples of conversion rules for PTB trees.

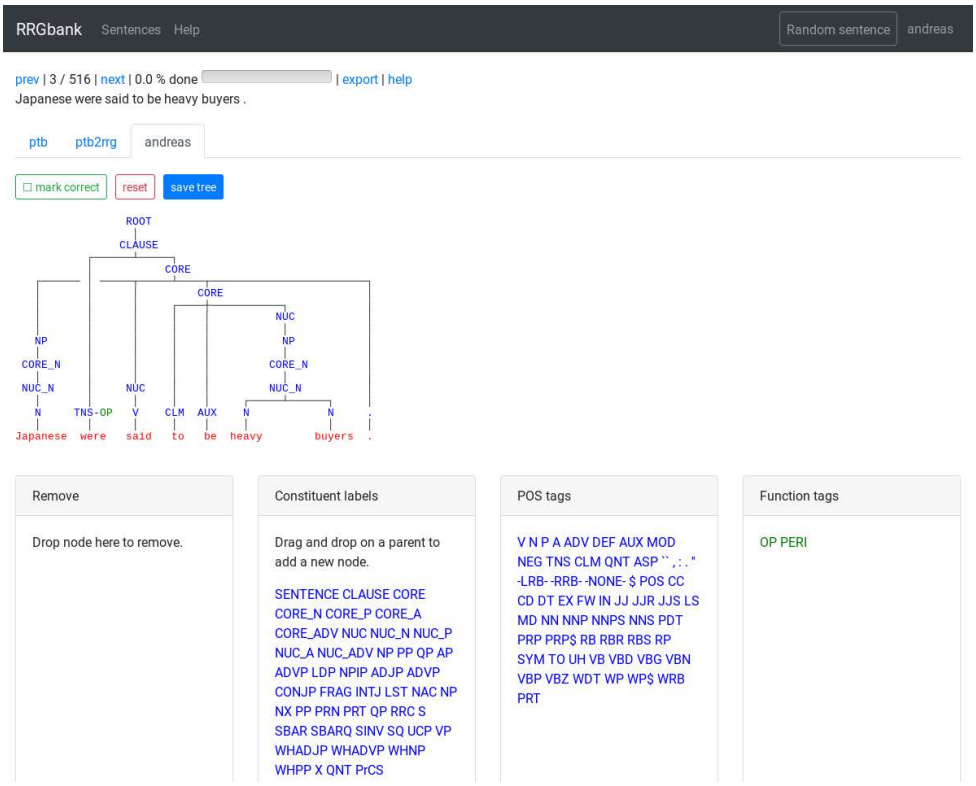


Figure 7: The annotation interface.

3.2.2 Problematic Cases for Conversion

The majority of the constituents in the PTB can be transformed with a small set of transformation rules, described in the previous section. However, the conversion process also revealed some systematic sources of conversion mistakes, among which are the following.

Annotation inconsistencies or errors in the PTB. In the example in Figure 8, a noun *network* is erroneously annotated as a verb. In such cases of annotation inconsistencies in the PTB, we do not introduce special conversions rules, since they would become too specific and only applicable for this particular sentence.

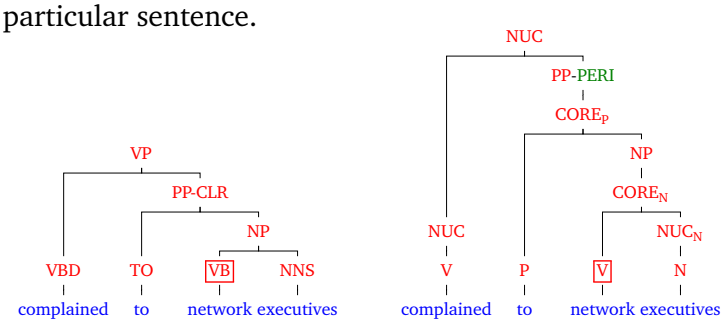


Figure 8: Errors in the PTB annotation.

Underspecific annotation in the PTB. In some cases, a deterministic conversion from PTB to RRG annotations is not possible because RRG makes distinctions that the PTB does not (always) make. One case in point is the negation operator *not*, which is always attached as an adverb inside a VP in the PTB, but can be attached to different layers in RRG depending on its semantic scope (see Figures 9). The RRG analysis provided in the middle tree on Figure 9 displays the case of internal negation with the possible readings "Japan is not a political country (but Belgium is)" or also "Japan is not a political country (it is a cultural one)". External negation however, negates the proposition as a whole, so the sentence displayed in the right tree in Figure 9 can be read as "It is not the case, that Japan is a political country".

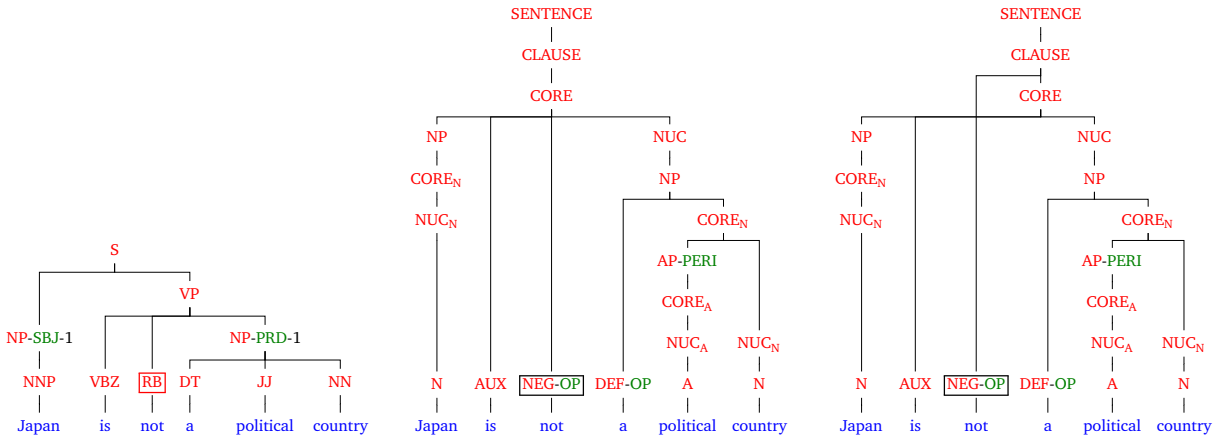


Figure 9: Difficult constructions in RRG: scope of negation in the PTB and in RRG.

Moreover, the trees in Penn Treebank and RRG structures are not deterministically related. That is, similar tree structures in the PTB might require different analyses in RRG. Figures 10 and 11 display the difference between two juncture types in RRG. Figure 10 shows the case of *core cosubordination*, in which the cores share their operators, while operator sharing is not required for *coordinated cores* (Figure 11).

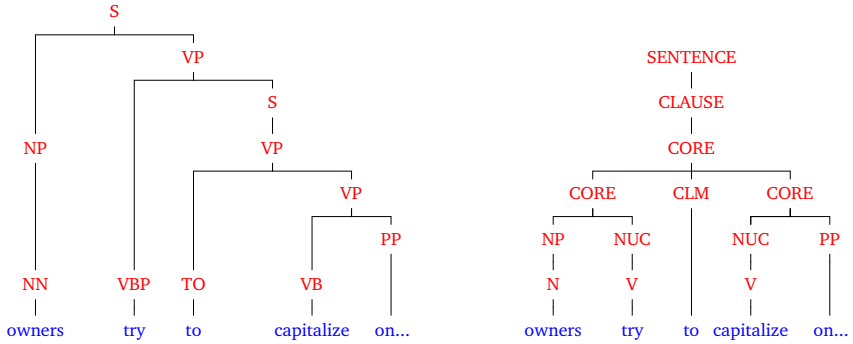


Figure 10: Core cosubordination.

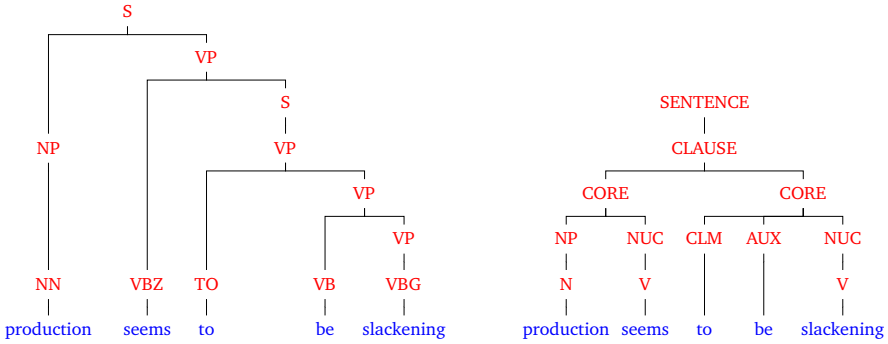


Figure 11: Core coordination.

RRG also differentiates between restrictive and non-restrictive relative clauses (see Figures 12 and 13). Restrictive relative clauses restrict the possible referents of the modified nominal expression by specifying information about them.

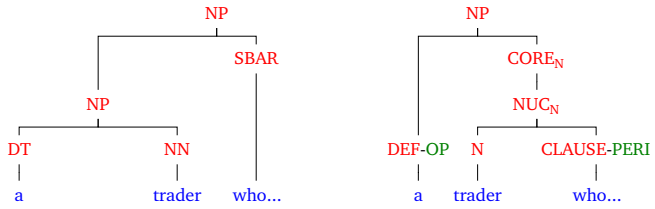


Figure 12: Restrictive relative clause.

Non-restrictive relative clauses, usually separated by a comma, encode additional information about a referent which is already unambiguously identifiable.

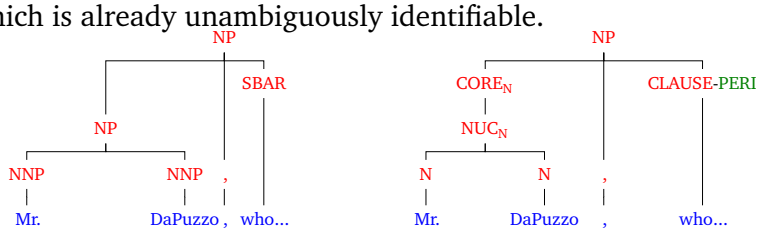


Figure 13: Non-restrictive relative clause.

Another example of underspecification in the Penn Treebank is the distinction between argument (non-pheripheral) PPs, which are to be labeled PP, and adjunct (peripheral) PPs, which are to be labeled PP-PERI. In some cases, functional labels in the PTB (for example, PP-TMP for temporal PPs or PP-DIR for directional PPs) indicate adjuncthood, while in other cases, the PTB provides

no such marking (compare, for example, the PP attachments in Figures 8 and 14).

Open questions in the theory of RRG. The process of converting PTB trees to RRG structures also reveals a number of under-investigated issues within RRG. An example is treatment of quantifier phrases (QPs). In particular, the PTB treats various kinds of constituents as QPs which can be headed by different lexical categories. The analysis of quantifiers differs in RRG, where some elements are analyzed as operators and others as peripheries. In such cases, we decided to leave problematic constituents unchanged until sufficient linguistic analysis is provided (see Figure 14).

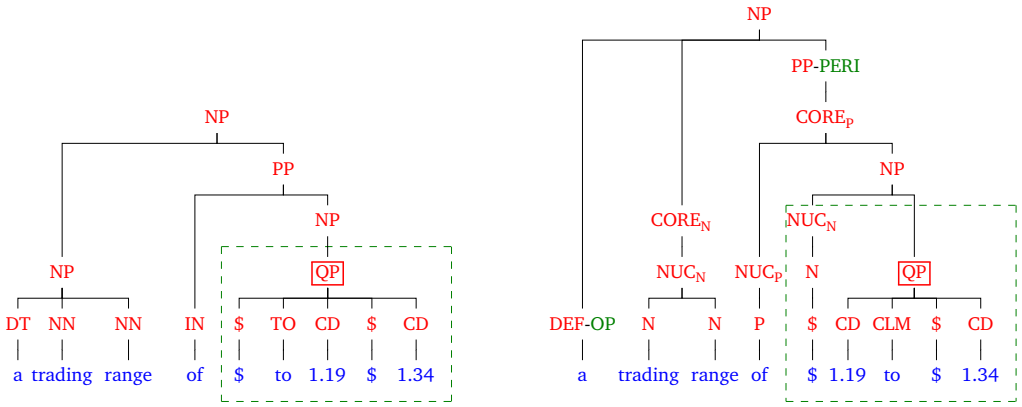


Figure 14: An open question in RRG: Quantifier phrases (marked with dashed lines).

4 Evaluation

We evaluate our conversion algorithm in terms of *completeness* and *correctness*.

Our algorithm finds an output tree for every input tree from the Penn Treebank. We measure the *completeness* of conversion as the ratio of nodes in a tree that have a label in the RRG label set. Because the PTB and RRG share some labels (e.g., NP, PP), this measure is nonzero even before conversion. Applied to WSJ Sections 02–21 of the Penn Treebank, completeness is currently 25.0% before conversion and 97.1% after conversion.

To measure *correctness*, we apply the algorithm to our validation treebank. This currently contains 100 RRG structures that have been manually corrected by one annotator. We are in the process of increasing this number to at least 500 and repeating the correction process with a second annotator to compute inter-annotator agreement and perform arbitration. In Table 1, we provide a preliminary evaluation of our conversion algorithm by comparing its output to the 100 corrected structures. We measure correctness in terms of shared labeled bracketings (the EVALB measure) of the automatic output and the annotated test set.

We also evaluated our conversion algorithm on different constituents since some of them turned out to be more problematic for the automatic conversion than the others. Table 1 provides an overview of the conversion scores for different constituents. Among the most problematic rewriting rules are those which are used to convert the constituents to highly complex structures in the framework of RRG (for example, CORE, NUC or CORE_N). These structures can include different elements and exhibit different arrangements of these elements (compare, for example, the RRG structures in Figures 1, 2 and 8). By contrast, constituents such as CORE_A or NUC_ADV tend to be non-problematic for the conversion since their structure is either highly predictable (CORE_A (A)) or is clearly indicated by the corresponding labels in the PTB (for example, ADVP

label	frequency	recall	precision	F1
(any)	100.00	91.18	90.21	90.69
NP	14.74	96.04	95.40	95.72
CORE_N	14.48	90.36	89.16	89.76
NUC_N	13.89	91.36	86.31	88.76
CORE	6.49	75.00	77.32	76.14
NUC	6.49	87.50	87.06	87.28
CLAUSE	5.19	78.75	86.90	82.62
NUC_P	5.16	100.00	98.15	99.07
PP	5.13	97.47	96.86	97.16
CORE_P	5.13	97.47	96.86	97.16
AP	3.80	90.60	92.17	91.38
CORE_A	3.73	93.91	93.10	93.51
NUC_A	3.73	97.39	96.55	96.97
ROOT	3.25	100.00	100.00	100.00
ADVP	2.30	81.69	96.67	88.55
NUC_ADV	2.21	100.00	95.77	97.84
CORE_ADV	2.21	92.65	88.73	90.60

Table 1: Preliminary results of evaluating the conversion algorithm on our 100-sentence validation corpus, overall and for the 15 most frequent constituent labels. The scores are labeled EVALB scores.

for adverbial phrases).

5 Conclusion

This paper reports on ongoing efforts towards creating a treebank for Role and Reference Grammar, a grammar theory that is widely used in typological research and that adopts a view on grammar as a complex system of syntax, semantics, morhpology, and information structure. We concentrate on the syntactic analyses assumed in RRG, and we first proposed a tree-based representation structure for them. We then started an iterative process of annotating PTB sentences with RRG structures, developing rules for an automatic transformation of PTB trees into RRG trees, and then feeding back information about errors on the gold data into the development of transformation rules. We plan to continue this cycle of annotation, rule development and testing for some time.

The work presented here will lead to RRGbank, an RRG annotation of the PTB. RRGbank will be the first large linguistic resource in the RRG community. It opens up new possibilities for using RRG in natural language processing (grammar implementation, grammar induction, data-driven parsing, semantic parsing when adding for instance the semantic information from PropBank etc.). Furthermore, the development of RRGbank will also lead to new insights about how to analyze certain constructions in English within RRG, and the treebank will be a valuable resource for empirical, corpus-based investigations of RRG structures.

We also plan to explore treebanks available in the framework of the Universal Dependencies project (Nivre et al., 2016) for conversion to RRG structures. An advantage of using Universal Dependencies is the coverage of many languages along with a uniform labeling while taking into consideration linguistic peculiarities of each language.

The transformation tool will be made available and, in addition, we plan to provide RRGbank via the Linguistic Data Consortium (LDC) as an alternative annotation layer to the PTB.

Acknowledgments

The work presented in this paper was partly funded by the European Research Council (ERC grant TreeGraSP) and partly by the German Science Foundation (CRC 991). We would also like to thank Robert D. Van Valin, Jr. for giving us valuable advice for our project. Furthermore, we are grateful to three anonymous reviewers whose comments helped to improve the paper.

References

Flickinger, D., Kordoni, V., and Zhang, Y. (2012). DeepBank: A Dynamically Annotated Treebank of the Wall Street Journal. In *Proceedings of the 11th International Workshop on Treebanks and Linguistic Theories*, Lisbon, Portugal.

Hockenmaier, J. and Steedman, M. (2007). CCGbank: A corpus of CCG derivations and dependency structures extracted from the Penn Treebank. *Computational Linguistics*, 33(3).

Hovy, E., Marcus, M., Palmer, M., Ramshaw, L., and Weischedel, R. (2006). Ontonotes: the 90% solution. In *Proceedings of the human language technology conference of the NAACL, Companion Volume: Short Papers*, pages 57–60. Association for Computational Linguistics.

Kallmeyer, L. (2016). On the mild context-sensitivity of k-Tree Wrapping Grammar. In Foret, A., Morrill, G., Muskens, R., Osswald, R., and Pogodalla, S., editors, *Formal Grammar: 20th and 21st International Conferences, FG 2015, Barcelona, Spain, August 2015, Revised Selected Papers. FG 2016, Bozen, Italy, August 2016, Proceedings*, number 9804 in Lecture Notes in Computer Science, pages 77–93, Berlin. Springer.

Kallmeyer, L. and Osswald, R. (2017). Combining Predicate-Argument Structure and Operator Projection: Clause Structure in Role and Reference Grammar. In *Proceedings of the 13th International Workshop on Tree Adjoining Grammars and Related Formalisms*, pages 61–70, Umeå, Sweden. Association for Computational Linguistics.

Kallmeyer, L., Osswald, R., and Van Valin, Jr., R. D. (2013). Tree Wrapping for Role and Reference Grammar. In Morrill, G. and Nederhof, M.-J., editors, *Formal Grammar 2012/2013*, volume 8036 of *LNCS*, pages 175–190. Springer.

Marcus, M., Santorini, B., and Marcinkiewicz, M. (1993). Building a Large Annotated Corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330.

Nivre, J., De Marneffe, M.-C., Ginter, F., Goldberg, Y., Hajic, J., Manning, C. D., McDonald, R. T., Petrov, S., Pyysalo, S., Silveira, N., et al. (2016). Universal dependencies v1: A multilingual treebank collection. In *LREC*.

Oepen, S., Flickinger, D., Toutanova, K., and Manning, C. D. (2004). Lingo redwoods. *Research on Language and Computation*, 2(4):575–596.

Sulger, S., Butt, M., King, T. H., Meurer, P., Laczkó, T., Rákosi, G., Dione, C. B., Dyvik, H., Rosén, V., De Smedt, K., et al. (2013). Pargrambank: The pargram parallel treebank. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 550–560.

Van Valin, Jr., R. D. (2005). *Exploring the Syntax-Semantics Interface*. Cambridge University Press.

Van Valin, Jr., R. D. (2010). Role and Reference Grammar as a framework for linguistic analysis. In Heine, B. and Narrog, H., editors, *The Oxford Handbook of Linguistic Analysis*, pages 703–738. Oxford University Press, Oxford.

Van Valin, Jr., R. D. and LaPolla, R. (1997). *Syntax: Structure, meaning and function*. Cambridge University Press.

Proceedings of the 13th Conference on Language Resources and Evaluation (LREC 2022), pages 4833–4841
Marseille, 20-25 June 2022

RRGparbank: A Parallel Role and Reference Grammar Treebank

Tatiana Bladier, Kilian Evang, Valeria Generalova, Zahra Ghane, Laura Kallmeyer, Robin Möllemann, Natalia Moors, Rainer Osswald, Simon Petitjean
Heinrich Heine University Düsseldorf
Universitätsstr. 1, 40225 Düsseldorf, Germany
first.last@hhu.de

Abstract

This paper describes the first release of RRGparbank, a multilingual parallel treebank for Role and Reference Grammar (RRG) that contains annotations of George Orwell's novel *1984* and its translations. The release comprises the entire novel for English and a constructionally diverse, parallel "seed" sample for German, French, Russian, and Farsi. The paper gives an overview of the annotation decisions taken and describes the adopted treebanking methodology. As a possible application, a multilingual parser is trained on the treebank data. RRGparbank is one of the first resources for which RRG has been applied to large amounts of real-world data. It enables comparative and typological corpus studies in RRG and creates new possibilities of data-driven NLP applications based on RRG.

Keywords: Syntax, Treebank, Parallel Corpus, Role and Reference Grammar, English, German, French, Russian, Farsi

1. Introduction

Role and Reference Grammar (RRG) (Van Valin and LaPolla, 1997; Van Valin, 2005; Van Valin, 2010) has been proposed as a theory of grammar with an emphasis on typological adequacy. More recently, RRG has also been studied from the perspective of formal and computational linguistics: A formalization of RRG has been proposed in (Kallmeyer et al., 2013; Osswald and Kallmeyer, 2018), based on which a symbolic parser for precision grammars has been developed (Arps et al., 2019). Moreover, there have been recent initiatives for creating treebanks for RRG (Bladier et al., 2018; Chiarcos and Fäth, 2019).

In this paper, we present the first release of RRGparbank, a multilingual parallel treebank for RRG, based on George Orwell's novel *1984* and translations thereof.[1] RRGparbank is the first effort to apply RRG to a parallel large-scale corpus, making RRG usable as a framework for data-driven NLP and corpus-linguistic research. We expect the parallel nature of the treebank to make it especially useful for comparative and typological studies, for which RRG has been designed. Applying RRG to large amounts of real data has raised (and already helped answer) a number of questions about the details of RRG analyses which were previously undefined. We have used several innovative techniques to make treebanking efficient, including rule-based conversion from Universal Dependencies to RRG (Evang et al., 2021) and statistical parsing as starting points for annotation, as well as incremental improvements of these starting points with human annotators in the loop. We make the resulting treebank available with various download and search options.

In this paper, we give a brief introduction to RRG (Section 2), describe our treebanking methodology and highlight important aspects of the annotation guidelines we developed (Section 3), describe the released resource and tools (Section 4), and demonstrate statistical parsing as one possible use of an RRG treebank (Section 5).

[1] RRGparbank is available at: https://rrgparbank.phil.hhu.de

2. Role and Reference Grammar

2.1. Background

Role and Reference Grammar (RRG) is a functional theory of grammar whose development has been strongly driven by the investigation of typologically varied languages. RRG aims at integrating syntactic, semantic and pragmatic levels of description which are related to each other by the "linking system", an elaborate system of linguistic rules and constraints (Van Valin and LaPolla, 1997; Van Valin, 2005; Van Valin, 2010). Since the focus of RRGparbank is primarily on syntactic annotation, RRG's syntax-semantics-pragmatics interface will not be discussed here in more detail.

A key syntactic concept of RRG is the "layered structure of the clause" comprising the layers *nucleus*, *core* and *clause*. The nucleus encodes the main predicate, the core consists of the nucleus and the syntactic realizations of the predicate's arguments, and the clause includes the core plus extracted arguments. Each layer can be accompanied by *peripheral* structures for attaching adjuncts. For instance, in a verbal constituent, aspectual modifiers attach to the nucleus, locative and temporal modifiers attach to the core, while modal adverbials attach to the clause. The layered structure is not restricted to verbal phrases but applies also to constituents headed by other elements such as nouns, prepositions, etc.

Closed-class morphosyntactic elements for encoding tense, modality, aspect, or definiteness, among others, are referred to as *operators* in RRG. They attach to the

layers over which they take scope, and it is a crucial assumption of RRG that the surface ordering of the operators is aligned with the height of their attachment site. Since the surface order of the operators relative to arguments and adjuncts would often require crossing branches in the syntactic representations, RRG considers the constituent structure and the operator structure as different syntactic *projections* of the clause.

Concerning the structure of complex sentences, RRG draws not only a distinction between embedded, dependent structures (*subordinations*) and non-embedded, independent ones (*coordinations*), but assumes in addition non-embedded dependent structures, so-called *cosubordinations*. Cosubordinate structures have the general form $[[\ ]_X\ [\ ]_X]_X$. In such constructions, operators that apply to category X are usually realized only once but have scope over both constituents. An example for a CORE cosubordination is en-2304[2], *Presumably [[she could be trusted]$_{CORE}$ [to find a safe place]$_{CORE}$]$_{CORE}$*, where we have a modal operator (*could*) that is part of the first CORE but scopes also over the second. The three different nexus types can occur at all layers. For instance, English resultative constructions such as *tore open* in *He tore open a corner of the packet* (en-2889) are analyzed as nuclear cosubordinations since they function as complex predicates. By comparison, raising constructions like *He seemed to know the place* (en-1788) are generally analyzed as core coordinations.

2.2. Formalization

The syntactic annotations in RRGparbank build on the formalized version of RRG proposed by Kallmeyer and Osswald (2017) and Osswald and Kallmeyer (2018) (see also Kallmeyer et al. (2013)). An important difference between the structures used in this formalization and the syntactic representations found in RRG textbooks is that operators are integrated into the constituent projection. They are attached where they take scope, e.g., tense attaches at the CLAUSE level and negation attaches for instance at the CORE level, see Figure 1. The attachment of operators and also of modifiers (periphery structures in RRG) can lead to crossing branches.

The proposed formalization treats RRG as a *Tree Wrapping Grammar (TWG)*, which is based on a tree-rewriting formalism in the spirit of Tree Adjoining Grammar (Joshi and Schabes, 1997). A TWG consists of a finite set of elementary trees that can be combined by the following three basic operations: *(simple) substitution* (replacing a leaf by a new tree); *sister adjunction* (adding a new tree as a subtree to an internal node); and *wrapping substitution* (splitting the new tree at a dominance-edge, filling a substitution node with the lower part and adding the upper part to the root of the target tree). For more details on this formalization,

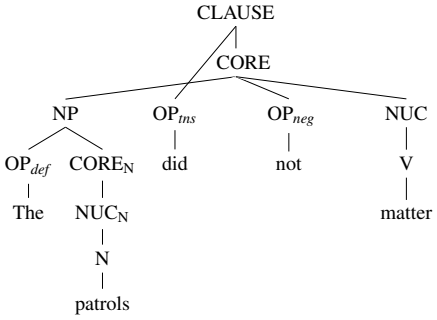


Figure 1: Examples of definiteness, tense and negation operators (en-28)

see Kallmeyer et al. (2013; Osswald and Kallmeyer (2018). While not directly relevant to the annotation task *per se*, viewing RRG as a TWG allows us to extract grammars from the annotated corpora, which in turn can be employed for parsing purposes (see Section 5), and which was also used for selecting the seed data (see Section 4.5).

3. Annotation

3.1. Annotation pipeline

We provide annotators with initial, automatically created trees for all sentences, which they then correct using a web-based annotation interface (see Figure. 2).[3] For creating the initial trees, we first parsed the sentences with an off-the-shelf Universal Dependencies parser and converted them to RRG using a rule-based algorithm (Evang et al., 2021). Later, as annotators produced enough corrected annotations to train a statistical parser (see Section 5), we started to use the results of this parser instead for selected languages because it provided more accurate syntactic structures [4].

In total, 11 annotators were involved in creating the data in the release described in this paper. They were presented with sentences pre-annotated using the automatically generated trees, corrected them using the drag-and-drop web interface, and finally marked their version as correct. Annotators do not see each other's annotations. A tree marked correct by at least one annotator has *silver* status (the release includes the latest silver tree in such cases).

Some of the annotators (who are RRG experts) are also allowed to sign in using a special *judge* account, where they can see all annotations, and a diff view highlighting the parts of trees where annotations differ (i.e.,

[2]Throughout the paper, *L-n* is used as id for sentence number *n* in language *L* in RRGparbank.

[3]The first prototype of the interface was implemented by Andreas van Cranenburgh using components of his disco-dop framework (van Cranenburgh et al., 2016).

[4]Evang et al. (2021) showed that a statistical parser starts to outperform the rule-based conversion algorithm from UD dependencies to RRG structures at about 2000 training sentences for English. Although a further fine-tuning of the rule-based approach is possible, it would not be practical due to a large number of additional required rules.

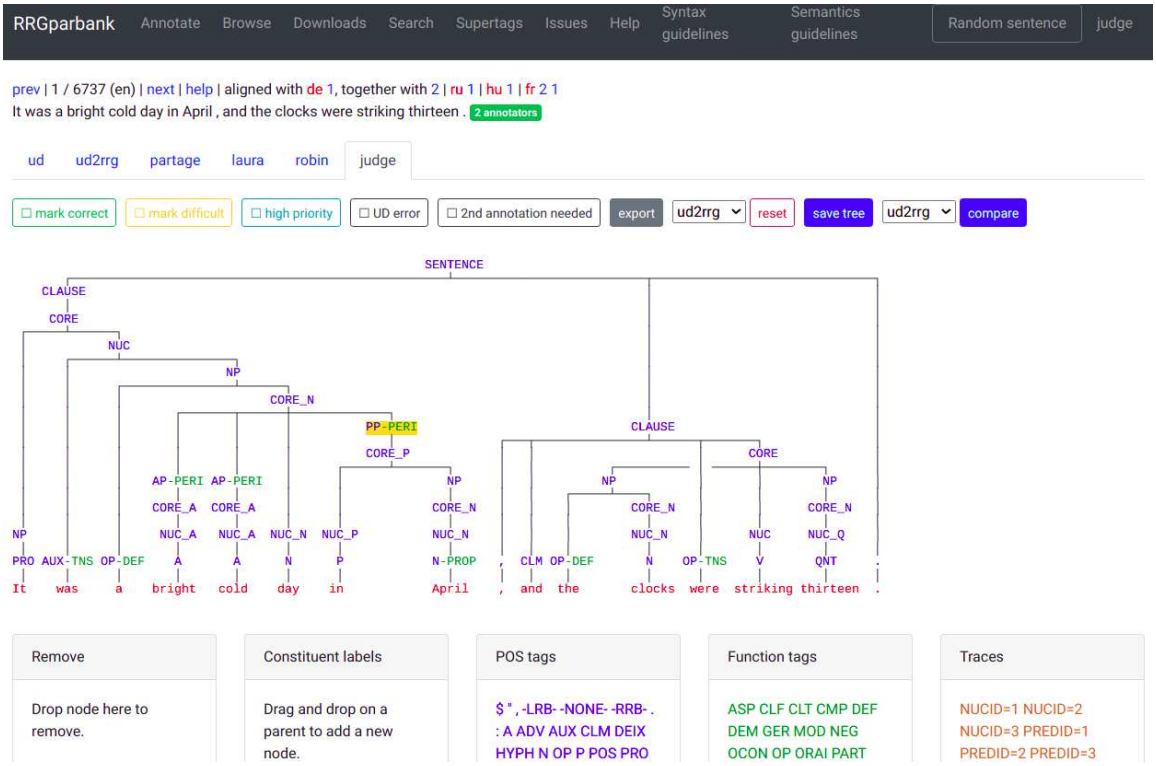


Figure 2: The drag-and-drop annotation interface of RRGparbank (view as judge)

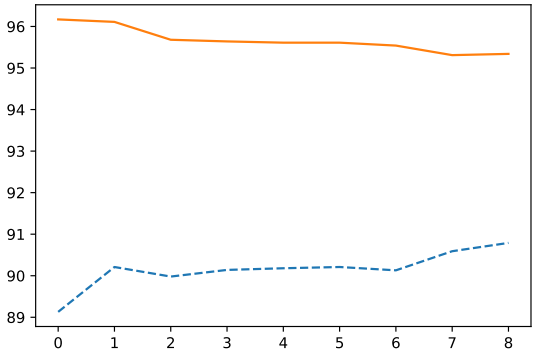


Figure 3: Cumulative quarterly inter-annotator f-score from January 2020 to March 2022, overall (solid curve) and for sentences with disagreements (dashed curve).

inter-annotator disagreements). The judge then has to decide which annotation decisions are the correct ones and create a final authoritative tree (based on the latest silver tree) using the normal tree editing operations. A tree marked correct by the judge has *gold status*. When it is not clear how to resolve a disagreement, the sentence is discussed between annotators at regular adjudication meetings before being marked as gold.

In the beginning, each sentence was annotated by at least two annotators before being judged. As annotators gained more experience and the guidelines were extended to cover more cases explicitly, we gradually moved to a more speedy annotation workflow where the expert annotators were allowed to use the judge account to directly mark a single annotation (that is not their own) as gold in easier cases, i.e., where they feel the existing annotation is clearly correct. They could also correct small, trivial annotation mistakes (for instance, deleting a second NP node below a first one with just a unary branch between the two). However, if, beyond that, they disagreed with something in the annotation, they were again instructed to create an alternative annotation using their regular annotator account, and leave the judging to another annotator. This workflow speeded up the annotation process considerably without sacrificing too many checks and balances compared to the complete workflow with at least two annotators and one judge. Between 30% (for Russian) and 60% (for English) of all released gold sentences have at least two annotations (see Section 4 for details). For these sentence pairs, we have an overall inter-annotator agreement of 91.04% measured as EVALB f-score (Collins, 1997). In Figure 3, we show cumulative agreement over time, binned by quarter. The solid curve represents overall agreement, counting sentences where the second annotator accepted the first annotation as agreeing perfectly. The dashed curve considers only sentences where a second annotation was provided. Overall agreement starts at 96.2% and goes down to 95.3% over time, as the "easy cases" tended to be annotated early. For sentences with two annotations, agreement starts at 89.1% and goes up to

almost 91% as annotation guidelines got more fleshed out and annotators gained experience. In order to find out whether the possibility for the second annotator to start from the first annotation unduly biases them, we also compared agreement in the month before and after this possibility was introduced, finding no dramatic difference (87.64% to 89.88%).

3.2. Selected phenomena

RRGparbank, along with RRGbank (Bladier et al., 2018), a previous RRG treebank of text from the Penn Treebank, also annotated by our group, is the first endeavour to annotate large amounts of corpus data with RRG structures. The only other electronic syntactic RRG resource is that of Chiarcos and Fäth (2019)[5], a corpus consisting of 351 examples from the textbook of Van Valin and LaPolla (1997). In contrast to them, we were faced with a variety of constructions that RRG had not considered so far, which means that, besides annotating, we also had to take numerous decisions concerning syntactic analyses in RRG.
For a detailed description of the annotation decisions, see the guidelines available on the treebank website. In the following, we discuss a few interesting questions that came up during the annotation process.

Copula constructions. Most copula constructions feature a verb (usually *'to be'*) annotated as AUX (auxiliary). It is placed under NUC and is thus one of the predicating parts. It can also bear some operator features, e.g., tense, aspect, or modality. The other part of the predicate (mostly AP, PP, NP, or participle) is also dependent on the NUC. There is no auxiliary in the present tense in Russian, so the only predicating part and the only descendant of the NUC is the non-verbal constituent.

Discontinuous structures. Discontinuities (i.e., crossing branches) can arise in the treebank trees due to elements belonging to a higher layer but being positioned between elements belonging to a lower layer. These can be not only operators or periphery elements (as mentioned above) but also arguments. In these cases as well, the annotation contains crossing branches. Examples are discontinuous NUC constituents as in Figure 4 and discontinuous CORE constituents as in 1 below (the relevant part of the tree is given in Figure 5). We found this type of discontinuous CORE mainly in German.

(1) *Merkwürdigerweise schien ihn das Schlagen*
Curiously seems him$_{acc}$ the chiming
der vollen Stunde mit neuem Mut
of.the full hour$_{nom}$ with new courage
erfüllt zu haben .
filled to have .

'Curiously, the chiming of the full hour seems to have filled him with new courage.'

[5]https://github.com/acoli-repo/RRG

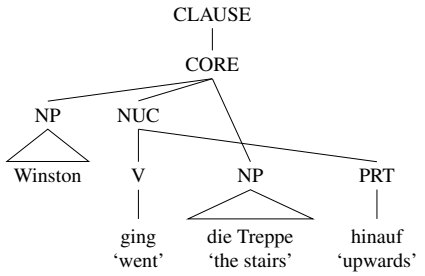


Figure 4: Discontinuous NUC for German particle verb (de-5)

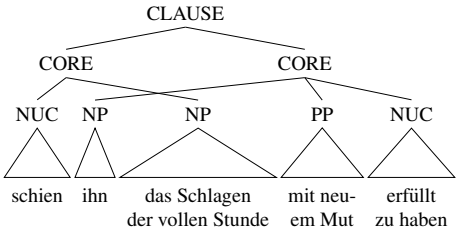


Figure 5: Discontinuous COREs in German (de-481)

Non-local dependencies. Two types of non-local dependencies are annotated in RRGparbank: one is long-distance dependencies arising from a fronted *wh*-phrase, or relative pronoun (in the pre-core slot (PrCS) in RRG) that does not belong to the CORE it precedes but to another CORE. In these cases, the feature NUCID identifies the predicate on which the PrCS depends. The PrCS, in turn, is provided with a PREDID feature pointing at the predicate. The coindexing of these two features expresses the predicate-argument relation in a long-distance dependency construction, see Figure 6.

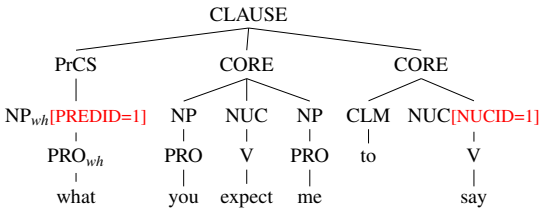


Figure 6: Subordinate interrogative clause with long-distance dependency (en-1761)

The second type of non-local dependency covered by the annotations are extraposed relative clauses (attached as a periphery element at the higher clause) that are linked to their antecedent NPs via coindexation in a feature REF. An example is given in Figure 7. Such constructions are particularly frequent in the German RRGparbank data (due to German's free word order); 4.8% of the German treebank sentences contain an extraposed relative clause.

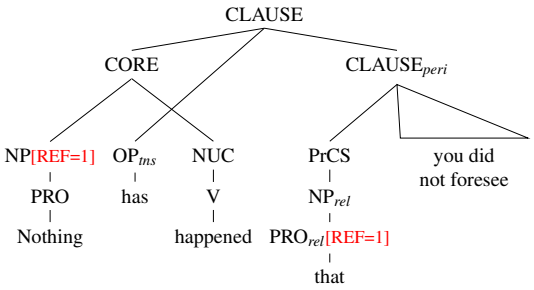


Figure 7: Extraposed relative clause (en-5922)

Multi-word expressions (MWEs). Fixed MWEs (e.g., *Big Brother*, *of course*) that cannot be modified and are syntactically inflexible are annotated in a flat way, i.e., with all POS tags below the same NUC (resp. NUC_X) node. This includes also inherently reflexive verbs (for instance German *sich erinnern* 'remember', or French *se trouver* 'be located', *se souvenir* 'remember'), where the reflexive pronoun and the verb are daughters of NUC, as well as fixed V N combinations such as English *give way*, *get hold*, and French *avoir lieu* 'take place'.

In contrast to this, light verb constructions (LVC), which are more flexible and productive, are annotated like full verbs, i.e., the non-verbal part (usually an NP or a PP) is placed under CORE, and the light verb is a V under NUC. An example is French *donner un bain* ('give a bath', fr-6400) and its English translation in en-5957.

Negation and modality. Expressions of negation are usually analyzed as operators (indicated by OP$_{neg}$ or OP-NEG). They can attach to any layer (CLAUSE, CORE or NUC) depending on their scope. Syntactic tests (for instance, addition of peripheral elements) show that English and German negation scopes over the CORE, and over the NUC in Russian. This difference is reflected in the annotation: negation elements are attached to NUC structures in Russian and to COREs in English and German (cf. en-106, de-105 vs. ru-104).

In French, the negation usually consists of two parts, i.e., we have negative concord. The particles *ne* and *pas* are annotated as operators exclusively as their unique function is to introduce the negation. In contrast, negative adverbs (like *jamais* 'never') and pronouns (like *rien* 'nothing') are heads of their respective phrases. In this case, the functional tag NEG is attached to the respective category labels.

The same applies to annotating modality: there are modality operators (e. g., the Russian particle *by* used for building the irrealis mood) as well as words that receive their own part-of-speech together with the functional MOD tag. For instance, Russian modal predicative adverbs, see ru-452 in 2, are annotated as ADVMOD and can take their own dependencies.

(2) *Proshloe* *umer-lo* *budushhee*
past die-3SG.PST future
nel'zja *voobrazi-t'* .
impossible.ADV.MOD imagine-INF .
'The past was dead, the future was unimaginable.'
(lit.: 'impossible to imagine')

Reported speech. The literary text contains many cases of direct and indirect reported speech. Direct speech includes a clause with a verb of saying and a quoted block with the contents of the utterance, e.g., *"And now let's see which of us can touch our toes!" she said enthusiastically* (en-611). In these cases, the quoted text is annotated as a separate SENTENCE subordinate to the main SENTENCE, while the reporting part appears under the usual spine, see Fig.8. Note, however, that not all cases of direct speech come with quotes; in French, we frequently have cases without quotes, for instance *Ils sont si bruyants! dit-elle.* ('"They are so noisy!", she says.', fr-434).

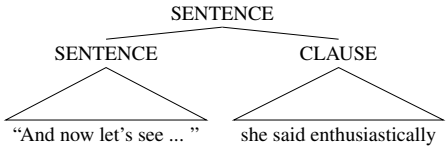


Figure 8: Direct speech (en-611)

Indirect reported speech often contains complementizers, anaphoric pronouns and relative tense marking. In these cases, the contents of the speech is treated as an argument of the saying predicate and appears as a subordinate CLAUSE, see Fig. 9 illustrating en-593: *The Party said that Oceania had never been in alliance with Eurasia.*

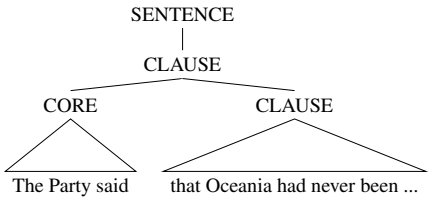


Figure 9: Indirect speech (en-593)

4. Resource

4.1. Source texts and sentence alignments

The annotated texts in RRGparbank are taken from George Orwell's novel *1984*. The English and Russian tokenized texts and sentence alignments are taken from the MULTEXT-East dataset (Erjavec, 2017). The corresponding French and German data was built manually using the published translations Orwell (1972) and Orwell (2003), respectively.

	EN	EN-SEED	DE-SEED	FR-SEED	RU-SEED	FA-SEED
Number of sentences	6 737	1 450	1 454	1 555	1 416	1 476
Number of tokens	122 843	23 750	23 444	24 670	17 697	22 456
Average sentence length	18.2	16.4	16.1	15.9	12.5	15.2
Not yet annotated	0	0	0	0	0	1 010
Silver	348	0	889	1 309	1 019	589
Gold with 1 annotation	2 691	575	286	112	183	0
Gold with ≥ 2 annotations	3 698	875	279	134	214	0

Table 1: Statistics beginning of May 2022 (preliminary—further annotations will be added for first release in June 2022). The release includes the entire novel for English and the seed sentences for German, French, Russian, and Farsi. Sentences that are not annotated yet (this concerns the Farsi seed data) will be released with so-called bronze trees, which means with automatically obtained parse trees.

4.2. Coverage

We make all sentences from the English text available. Currently, they are all at least silver, by the time of the conference they will be gold. This means that gold RRG trees for the entire English 1984 corpus will be provided in the planned release. Furthermore, we make a part of the German, French, Russian, and Farsi data publicly available as a "seed corpus".[6] We aim at representing a broad variety of linguistic phenomena across the languages in the seed data. We also aim at a high degree of parallelism in the seed, making cross-linguistic comparisons possible. We describe the selection of seed sentences in Section 4.5. Table 1 gives some statistics of the first release.

4.3. Download and search options

All released data can be downloaded in NEGra treebank export format (Brants, 1997), which is suitable to represent trees with crossing branches. We provide a suggested split into training, development, and test data for experiments: all sentences whose numbers end with [1-8] are used for training, sentences with numbers ending in 9 go into development and in 0 into the test set. The sentence alignments between the English text and each translation are availale for download as text files in a simple column-based format.

We make it easy for linguists to find certain constructions of interest in RRGparbank by providing the possibility to search the trees via RRGparbank's Web interface using the TGrep2 tree search tool (Rohde, 2005). Users can query the trees whose structure matches a specified pattern. For example, the search query 'NUC < (V $.. PRT)' returns all trees in which the node NUC directly dominates a verb V with a separate particle PRT, such that V precedes PRT, for example *stood up* in English or *fuhr fort* ('continued') in German. The query '/=SAID$/ . /=WINSTON$/' returns all trees in which the word WINSTON comes directly after SAID.

[6]For copyright reasons, we cannot provide all annotated sentences in the other languages.

4.4. Annotation guidelines

We document our annotation decisions in the form of an annotation manual, available on the RRGparbank website. These guidelines are work in progress since the annotation process still leads to discussions of previously unseen phenomena or, sometimes, to revisions of earlier decisions.

4.5. Selection of seed data

To select seed corpora with a high degree of parallelism and a broad coverage of constructions, we extracted a Tree Wrapping Grammar for each language (see Section 5 for more details), which included assigning syntactic supertags (unanchored elementary trees) to tokens. We then selected a set of sentences together with their translations in all four languages in a way that maximizes the number of distinct supertags per language. Doing this optimally is an NP-complete problem, so we opted for a greedy approximation. We selected seed training sentences for the language with the highest annotation coverage first (German) and then proceeded to add sentences for English, Russian, and French. For each language, we iterated until all supertags that occur in the silver and gold training split at least twice were included. At each iteration, we added the sentence that maximizes the ratio u/l, where u is the number of unseen supertags in the sentence (i.e., supertags that are not yet in the seed) and l is the length of the sentence. Before moving on to the next language, we added all sentences that occur in the same sentence-level translation unit according to our sentence alignments (regardless of whether they are training, development, or test sentences) in order to ensure parallelism. As a result, for English, Russian, and French, the seed was already initialized with parallel sentences, and the iterative algorithm only had to "fill the remaining gaps" in the supertag coverage.

For future languages added to RRGparbank, we will just add all sentences aligned to the English seed data. This is the case for Farsi, for instance, for which alignments were added only recently.

5. Applications

One of the motivating factors behind RRGparbank is to create a sufficiently large linguistic resource to be used in different NLP contexts. This is made possible by the formalization of RRG and the extraction of formal grammars for the languages in RRGparbank. These grammars consist of elementary tree templates (i.e., *supertags*). They can be used to formulate compositional analyses of sentence syntax and semantics, and to design both precision grammars and statistical parsers. We use such a (syntactical) statistical parser for generating trees as starting points for annotation (cf. Section 3). Beyond this, syntactic parsers can be useful for downstream NLP tasks such as semantic parsing. In this section, we describe our statistical syntactic parsing architecture and carry out parsing experiments that demonstrate the usefulness of RRGparbank as a resource for training syntactic parsers.

As mentioned in Section 2.2, the formalization of RRG that underlies RRGparbank is based on Tree Wrapping Grammars (TWG). TWGs can be extracted from treebanks using an automatic extraction process described in Bladier et al. (2020a). TWGs typically consist of several thousand unlexicalized elementary trees, about half of which appear only once in the corpus. As an example, Figure 10 shows the clause 'what you expect me to say' from Figure 6 annotated with elementary tree templates. TWGs can be used for statistical parsing, for example with the parser ParTAGe [7] (Waszczuk, 2017; Bladier et al., 2019; Bladier et al., 2020b). The pipeline of this parser consists of supertagging (i.e. assigning the *n*-best elementary tree templates to each word in a sentence) and a subsequent A* parsing step.

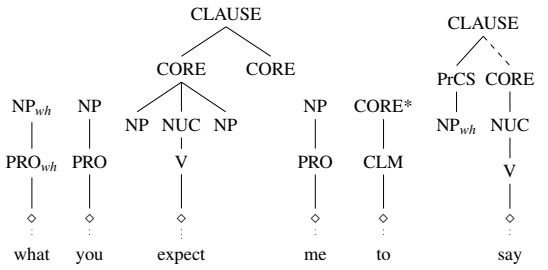


Figure 10: Extracted TWG supertags for the clause 'what you expect me to say' from Fig. 6. The sister-adjoining tree *to* is marked with an asterisk on the root node. The wrapping elementary tree *say* has a dominance link, notated as a dashed edge.

For our parsing experiments, we extract TWGs from the English, German, French, and Russian gold and silver subcorpora of a pre-release snapshot[8] of RRGparbank (including all training data, not just the one from the seed subcorpora). We train the statistical TWG parser ParTAGe using training and development sets and parse the corresponding test set to evaluate the language models. Table 2 gives an overview of the number of sentences and elementary trees for different languages. Many of the extracted supertags are common for all grammars. We found 426 supertags which appear in all four extracted TWGs.

We fine-tune the multilingual BERT model[9] and single-language BERT models for the supertagging component of the parser (similar to Schmidt (2021)) and compare the parsing accuracies. The experimental results are given in Table 3. We use the following single-language Transformer models: *bert-base-cased*[10] for English, *bert-based-german-cased*[11] for German, *camembert-base*[12] for French, *rubert-base-cased-sentence*[13] for Russian, and *bert-base-parsbert-peymaner-uncased* [14] for Farsi. We train all models for 20 epochs and use the same hyper-parameters across all models (see Table 4).

The results show that the TWG grammars extracted from RRGparbank have sufficient quality to be used for statistical multilingual parsing and that the parser trained on these grammars generalizes well. We also observe that the parser based on the multilingual model shows better performance compared to single-language models for all languages except English[15]. The single-language models on the other hand show a higher number of exactly matching parses. We assume that the better performance of the multilingual model can be explained by the cross-lingual transfer property of the multilingual BERT model (Wang et al., 2019; Ahmad et al., 2021) and some overfitting of the monolingual models. It would be interesting however to explore which role is played by the supertags common in all languages for the better performance of the multilingual parsing model. In our future work we will include further languages in the parsing experiments as the annotation of RRGparbank continues. We also plan to explore how to use extracted grammars and trained parsing models for cross-lingual parsing.

6. Conclusion

In this paper, we presented the first release of RRGparbank, a parallel treebank based on George Orwell's novel *1984* and its translations. The sentences in the corpus are annotated with RRG structures. For English, we include the entire novel, while for other languages (so far German, French and Russian), we provide a parallel seed corpus.

[7] https://rrgparser.phil.hhu.de/

[8] The data were downloaded on 2021-12-23.

[9] https://huggingface.co/bert-base-multilingual-cased

[10] https://huggingface.co/bert-base-cased

[11] https://huggingface.co/bert-base-german-cased

[12] https://huggingface.co/camembert-base

[13] https://huggingface.co/DeepPavlov/rubert-base-cased-sentence

[14] https://huggingface.co/HooshvareLab/bert-base-parsbert-peymaner-uncased

[15] All fine-tuned multilingual and single language models can be downloaded from the TWG parsing repository https://github.com/TaniaBladier/Transformer-based-TWG-parsing.

lang.	train	dev.	test	TWG size
en	4635 (4432)	574 (535)	566 (532)	3861 (2378)
de	4452 (1440)	566 (189)	561 (189)	4590 (2956)
fr	2324 (238)	273 (27)	289 (30)	2272 (1388)
ru	3877 (712)	480 (91)	486 (92)	3425 (2295)
fa	1169 (0)	146 (0)	128 (0)	1532 (992)
total	16457	2039	2030	–

Table 2: Number of sentences in data split for parsing experiments. The number in brackets indicate the gold sentences among the train, development and test data. The column TWG size shows the number of elementary trees in the extracted grammars, the numbers in brackets show how many supertags appear only once in each training set. Please note that the annotation of RRGparbank is not yet finished.

	multilingual model	single-language models	exact match (mult. model)	exact match (sing. model)	# sents	∅ len.
en	86.27	**86.56**	122	155	566	15.43
de	**85.19**	84.15	95	80	561	13.86
fr	**85.68**	85.21	66	71	289	11.66
ru	**86.16**	84.74	115	108	486	9.68
fa	**80.80**	74.37	37	17	127	8.66

Table 3: Parsing results (labeled F1 score) with the ParTAGe parser based on a fine-tuned multilingual BERT model and single-language BERT models. The results are shown for the test data without considering punctuation and function tags.

Hyper-parameters	**Value**
Max_seq_length	128
Train batch sizes	8
Learning rate	4e-05
Optimizer	AdamW
Lower case	False
Attention probability dropout rate	0.1
Hidden layer activation function	gelu
Hidden size	768
Warmup proportion	0.06
Warmup steps	1337
Number of hidden layers	12
Number of attend heads	12
Number of training epochs	20

Table 4: Hyper-parameters of the Transformer models.

RRGparbank is a valuable resource for several reasons. First, while building the treebank, we encountered numerous constructions that had not been taken into consideration in the RRG literature and for which we propose an analysis, documented in the guidelines. In this sense, RRGparbank contributes to the domain of syntactic analyses in RRG. Second, by building a treebank, in particular a parallel treebank, data-driven syntactic processing such as the parsing application presented in Section 5 become possible. Third, RRGparbank, together with options for download and for search, and also in combination with supertag extractions, enables corpus-based investigations of RRG structures, also across languages.

In the future, we plan to add further languages. Furthermore, besides syntactic annotations, we also started annotating semantic roles.

7. Acknowledgments

We thank all those that were involved in the creation and annotation of RRGparbank, aside from the authors of this paper (David Arps, Davy Baardink, Kata Balogh, Elif Benli, Andreas van Cranenburgh, Sarah Fabian, Naima Grebe). We also thank Behrang QasemiZadeh, Jule Pohlmann and Roland Eibers for preparing the German data. Furthermore, we are grateful to Robert D. Van Valin, Jr. for numerous invaluable in-depth discussions of annotation decisions. We also thank Jakub Waszczuk and Svetlana Schmidt who implemented parts of the ParTAGe parsing pipeline. This work was carried out as a part of the research project TreeGraSP[16] funded by a Consolidator Grant of the European Research Council (ERC).

8. Bibliographical References

Ahmad, W. U., Li, H., Chang, K.-W., and Mehdad, Y. (2021). Syntax-augmented multilingual BERT for cross-lingual transfer. *arXiv preprint arXiv:2106.02134*.

Arps, D., Bladier, T., and Kallmeyer, L. (2019). Chart-based RRG parsing for automatically extracted and hand-crafted RRG grammars. In *Role and Reference Grammar RRG Conference 2019*. University at Buffalo.

[16]https://treegrasp.phil.hhu.de

Bladier, T., van Cranenburgh, A., Evang, K., Kallmeyer, L., Möllemann, R., and Osswald, R. (2018). RRGbank: a Role and Reference Grammar Corpus of Syntactic Structures Extracted from the Penn Treebank. In *Proceedings of the 17th International Workshop on Treebanks and Linguistic Theories (TLT 2018), December 13–14, 2018, Oslo University, Norway*, pages 5–16. Linköping University Electronic Press.

Bladier, T., Waszczuk, J., Kallmeyer, L., and Janke, J. (2019). From partial neural graph-based LTAG parsing towards full parsing. *Computational Linguistics in the Netherlands Journal*, 9:3–26, Dec.

Bladier, T., Kallmeyer, L., Osswald, R., and Waszczuk, J. (2020a). Automatic extraction of tree-wrapping grammars for multiple languages. In *Proceedings of the 19th Workshop on Treebanks and Linguistic Theories*, pages 55–61.

Bladier, T., Waszczuk, J., and Kallmeyer, L. (2020b). Statistical parsing of tree wrapping grammars. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6759–6766.

Brants, T. (1997). The NEGRA export format. CLAUS Report 98, Universität des Saarlandes, Computerlinguistik, Saarbrücken, Germany.

Chiarcos, C. and Fäth, C. (2019). Graph-Based Annotation Engineering: Towards a Gold Corpus for Role and Reference Grammar. In Maria Eskevich, et al., editors, *2nd Conference on Language, Data and Knowledge (LDK 2019)*, volume 70 of *OpenAccess Series in Informatics (OASIcs)*, pages 9:1–9:11, Dagstuhl, Germany. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik.

Collins, M. (1997). Three generative, lexicalised models for statistical parsing. In *35th Annual Meeting of the Association for Computational Linguistics and 8th Conference of the European Chapter of the Association for Computational Linguistics*, pages 16–23, Madrid, Spain, July. Association for Computational Linguistics.

Erjavec, T. (2017). MULTEXT-East. In Nancy Ide et al., editors, *Handbook of Linguistic Annotation*, pages 441–462, Dordrecht. Springer Netherlands.

Evang, K., Bladier, T., Kallmeyer, L., and Petitjean, S. (2021). Bootstrapping Role and Reference Grammar treebanks via Universal Dependencies. In *Proceedings of Universal Dependencies Workshop 2021 (UDW 2021)*.

Joshi, A. K. and Schabes, Y. (1997). Tree-adjoining grammars. In *Handbook of formal languages*, pages 69–123. Springer.

Kallmeyer, L. and Osswald, R. (2017). Combining predicate-argument structure and operator projection: Clause structure in role and reference grammar. In *Proceedings of the 13th International Workshop on Tree Adjoining Grammars and Related Formalisms*, pages 61–70.

Kallmeyer, L., Osswald, R., and Van Valin, Jr., R. D. (2013). Tree Wrapping for Role and Reference Grammar. In G. Morrill et al., editors, *Formal Grammar 2012/2013*, volume 8036 of *LNCS*, pages 175–190. Springer.

Orwell, G. (1972). *1984*. Gallimard. French translation by Amélie Audiberti (first published 1950).

Orwell, G. (2003). *1984*. Ullstein, 37th edition. German translation by Kurt Wagenseil (first published 1950 by Alfons Bürger Verlag).

Osswald, R. and Kallmeyer, L. (2018). Towards a formalization of Role and Reference Grammar. In Rolf Kailuweit, et al., editors, *Applying and Expanding Role and Reference Grammar*, pages 355–378. Albert-Ludwigs-Universität, Universitätsbibliothek. [NIHIN studies], Freiburg.

Rohde, D. L. (2005). TGrep2 user manual version 1.15. Massachusetts Institute of Technology.

Schmidt, S. (2021). *Transformers-based Supertagging Model for Statistical Parsing of Tree Wrapping Grammars*. Bachelor's thesis, Heinrich-Heine-Universität Düsseldorf.

van Cranenburgh, A., Scha, R., and Bod, R. (2016). Data-oriented parsing with discontinuous constituents and function tags. *Journal of Language Modelling*, 4(1):57–111.

Van Valin, Jr., R. D. and LaPolla, R. J. (1997). *Syntax: Structure, meaning, and function*. Cambridge University Press.

Van Valin, Jr., R. D. (2005). *Exploring the syntax-semantics interface*. Cambridge University Press.

Van Valin, Jr., R. D. (2010). Role and Reference Grammar as a framework for linguistic analysis. In Bernd Heine et al., editors, *The Oxford Handbook of Linguistic Analysis*, pages 703–738. Oxford University Press, Oxford.

Wang, Z., Mayhew, S., Roth, D., et al. (2019). Cross-lingual ability of multilingual BERT: An empirical study. *arXiv preprint arXiv:1912.07840*.

Waszczuk, J. (2017). *Leveraging MWEs in practical TAG parsing: towards the best of the two worlds*. Ph.D. thesis, Université François Rabelais Tours.

Chapter 5

Syntax-enhanced Semantic Parsing

This chapter presents a publication as originally published, reprinted with permission from the corresponding publishers. The copyright of the original publication is held by the respective copyright holders, see the following copyright notices.

Improving DRS Parsing with Separately Predicted Semantic Roles

Tatiana Bladier[1], **Gosse Minnema**[2], **Rik van Noord**[2], **Kilian Evang**[1]
(1) University of Düsseldorf, Germany
(2) University of Groningen, the Netherlands
{tatiana.bladier, evang}@hhu.de
{g.f.minnema, r.i.k.van.noord}@rug.nl

Abstract

This paper addresses Semantic Role Labeling (SRL) within the context of English Discourse Representation Structure (DRS) parsing. In particular, we investigate whether semantic roles predicted by a near-state-of-the-art SRL model can be used to improve the outputs of modern end-to-end neural DRS parsers using a rule-based post-processing algorithm. We compare two methods of generating training data for the SRL model from the Parallel Meaning Bank, one DRS-based and one CCG-based. We also compare two different post-processing algorithms. Our results vary across different DRS parsers, but overall we find a small to moderate improvement of up to 0.5 F1 on the final DRSs. We find a small but consistent advantage of DRS-based over CCG-based training data generation, and of token-based over concept-based post-processing, where applicable.

1 Introduction

With the increasing availability of multi-layered semantically annotated corpora, semantic parsing today is typically approached as an end-to-end task of predicting a meaning representation in one go, including information on word senses, predicate-argument structure, scope, semantic roles, and more. Since each of these layers is complex in its own right, it might be beneficial to rely on multiple specialized components to separately predict individual semantic layers, and to combine their output. In this paper, we focus on separately predicting semantic roles in the context of Discourse Representation Structure (DRS) parsing.

DRSs are meaning representations grounded in Discourse Representation Theory (Kamp and Reyle, 1993). We use the English part of the Parallel Meaning Bank (PMB; Abzianidze et al., 2017), which contains sentences annotated with DRSs. Figure 1 shows an example. Events (e.g., `e1`) are related to their participants (e.g., `x1`, `x2`) via semantic roles (e.g., `Theme`, `Destination`) from the VerbNet/LIRICS inventory (Bonial et al., 2011). Semantic roles are a crucial aspect of meaning since they encode how each entity participates in an event (Fillmore, 1968).

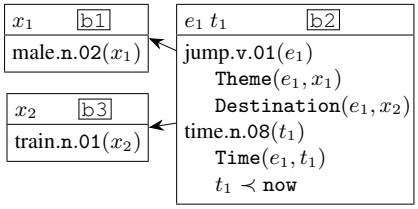


Figure 1: DRS for *He jumped into the train* (source: PMB, document `00/2759`)

Semantic role labelling (SRL) is typically approached as a task of labeling tokens or parse tree edges with predicate/role labels, independently of other aspects of meaning (e.g., Li et al., 2019, 2020b; Shi et al., 2020; Marcheggiani and Titov, 2020; Li et al., 2020a). Conversely, DRS parsers such as Evang (2019); Fancellu et al. (2020); van Noord et al. (2020); Liu et al. (2021) do not have dedicated SRL modules but predict a complete meaning representation of which roles are one part. In this paper, we explore the possibility of combining semantic parsers with a dedicated SRL system. The main research question we seek to answer is: can we in this way obtain DRSs with more accurate semantic roles?

Our approach is summarized in Figure 2: we first convert the PMB training data into a standard SRL annotation format (§2) in order to train a near-state-of-the-art SRL system on it (§3). At test time, we merge the output of DRS parsers with that of the SRL system using a rule-based post-processing algorithm (§4), aiming to produce

Proceedings of the ESSLLI 2021 Workshop on Computing Semantics with Types, Frames and Related Structures, pages 25–34
Utrecht, The Netherlands (online), 26–27 July, 2021.

a more accurate final DRS. We experiment with applying our procedure on top of several recent DRS parsing systems, and find that, albeit with some caveats, our procedure leads to overall better scores (§5). [1]

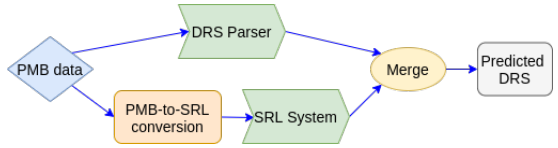


Figure 2: System overview

2 DRS-to-SRL Conversion

Before we can train an SRL system, we first need to convert semantic role annotations in the PMB to a more standard SRL format. Two characteristics of the PMB make this a non-trivial task. First, role annotations in the PMB are *predicate-based*, meaning that roles are carried by predicates instead of by arguments, as in standard SRL systems. Table 1 illustrates this: in standard SRL, the Theme role would be marked on *he*. Instead, in the PMB, the role is annotated on *jumped*, the predicate assigning the role; in a later step, the DRS parser makes sure that the role is associated to the discourse referent introduced by "He". Second, prepositional and adverbial roles (e.g. *into the train*, *slowly*) are treated differently from "core" semantic roles: they are carried by the preposition or adverb itself, instead of by the verbal predicate they are associated to.

Token	*He*	*jumped*	*into*	*the*	*train*
PMB		Theme	Destination		
SRL: head	Theme	PRED			Destination
SRL: span	Theme	PRED	{ ←	Destination	→ }

Table 1: PMB-style versus standard SRL annotations.

We experiment with two approaches for converting PMB role labels to a standard SRL format:

2.1 DRS-based conversion

Here, predicates and fillers for semantic roles are found via DRSs, which in the training data are *anchored*, i.e., most clauses are aligned to exactly one token. We extract predicate-role-filler triples such as ⟨jumped,Theme,he⟩ from the anchored DRSs by looking for role clauses such as `b2 Theme e1 x1` and then finding the clause introducing the filler (`b1 REF x1`, anchored to *He*), and the clause introducing the event (`b2 REF e1`, anchored to *jumped*). The process is illustrated in Figure 3.[2]

Disadvantages of this approach are 1) that it only yields the heads of the fillers, not full spans, and 2) that in some cases, the 'deep' semantic structure of the DRS does not directly match the surface realisations of the semantic roles we want to find. One example of the latter problem is found in sentences such as "She saw herself", where a DRS-based approach would return "She" as the Stimulus role, instead of "herself", which is the surface filler of this role but does not introduce a discourse referent of its own.

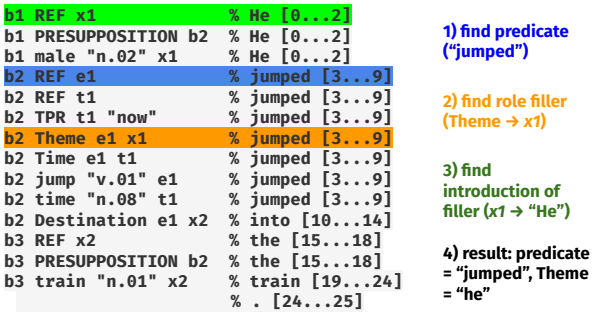


Figure 3: Example of DRS-based conversion.

2.2 CCG-based conversion

The second approach aims at overcoming both limitations of the DRS-based approach by making use of the CCG derivations in the PMB. Here, predicates and fillers for semantic roles are found via the CCG (Categorial Combinatorial Grammar, Steedman 2000) syntax trees and predicate-based role annotations in the PMB.

Main conversion process First, we transform the CCG trees using the `pmb_ccg_to_term` module in the `LangPro` package (Abzianidze, 2017), removing directionality of the combinatory rules and reducing the number of possible combinators, which simplifies tree traversing. In particular, long-distance dependencies (such as *wh*-movement) are handled using the λ-operator, which introduces a relationship between two variables at different points in the tree. An example of this kind of tree is given in Figure 4.

[1] Code and data at https://github.com/TaniaBladier/DRS_Parsing_with_SRL

[2] The DRS in *clause notation* in Figure 3 is equivalent to the one in *box notation* in Figure 1, but additionally shows the alignment with tokens in the sentence.

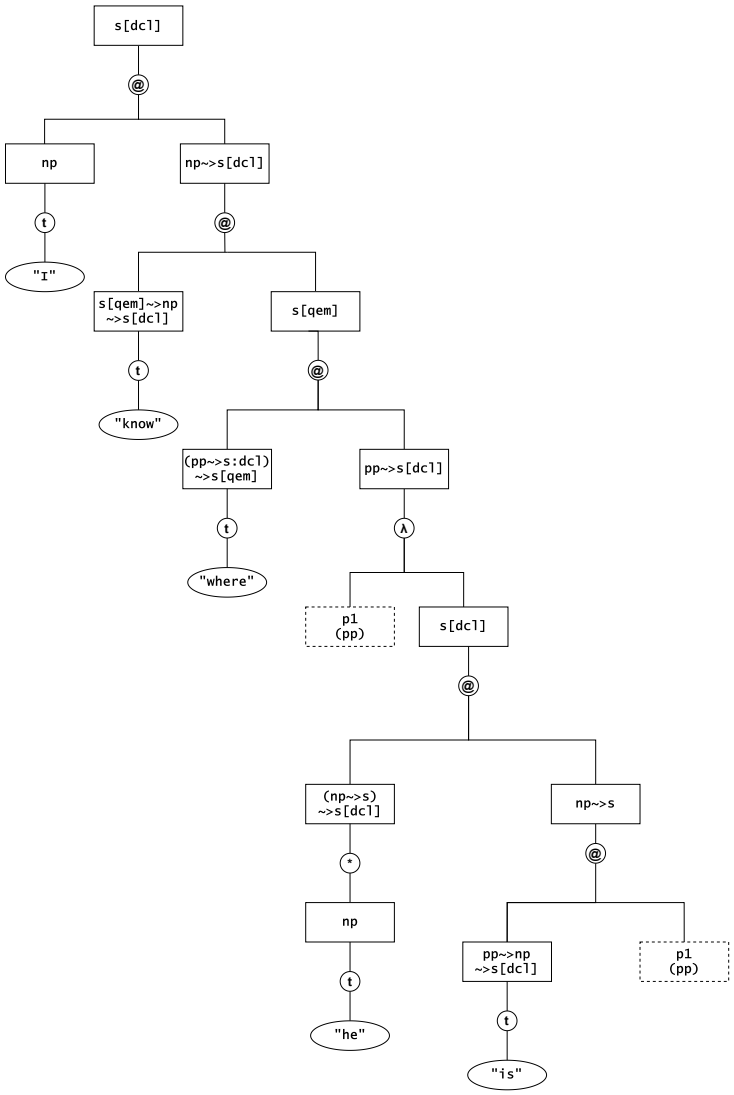


Figure 4: Simplified CCG tree with examples of all combinators (@: simple functional application; λ: variable introduction;, $*$: type-raising). Solid rectangles are types, circles are operators, dotted rectangles are lambda variables, and ovals are lexical nodes. `s[dcl]` means 'declarative sentence'; `s[qem]` means 'embedded question'.

Next, we deploy our role span extraction algorithm, which traverses the simplified tree and tries to match the semantic roles annotated on each predicate to the constituents filling these roles. Figure 5 displays a high-level overview of this process, showing how CCG arguments get mapped to constituents in the tree. This process is explained in more detail in Figure 6.

Given a simplified tree, we extract each predicate's syntactic roles from its CCG type signature and match them with the annotated semantic roles. For example, suppose *jump* has the type signature NP→S[3] and the role annotation `[Theme]`, then it has a single NP syntactic role, corresponding to

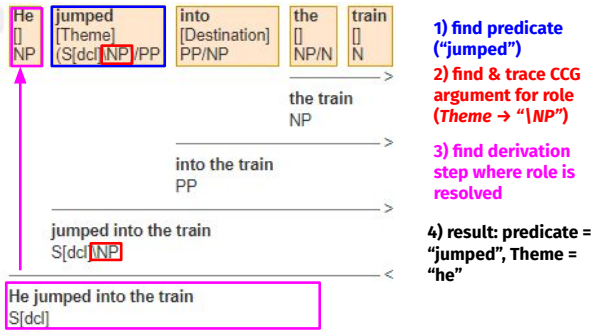


Figure 5: Example of CCG-based conversion.

a Theme semantic role. Then, we move upwards through the syntax tree, checking the type signature at every step; whenever we detect that a role has been filled, we process the constituent that was

[3]The original CCG category would be S\NP, which we simplify into the direction-agnostic NP→S.

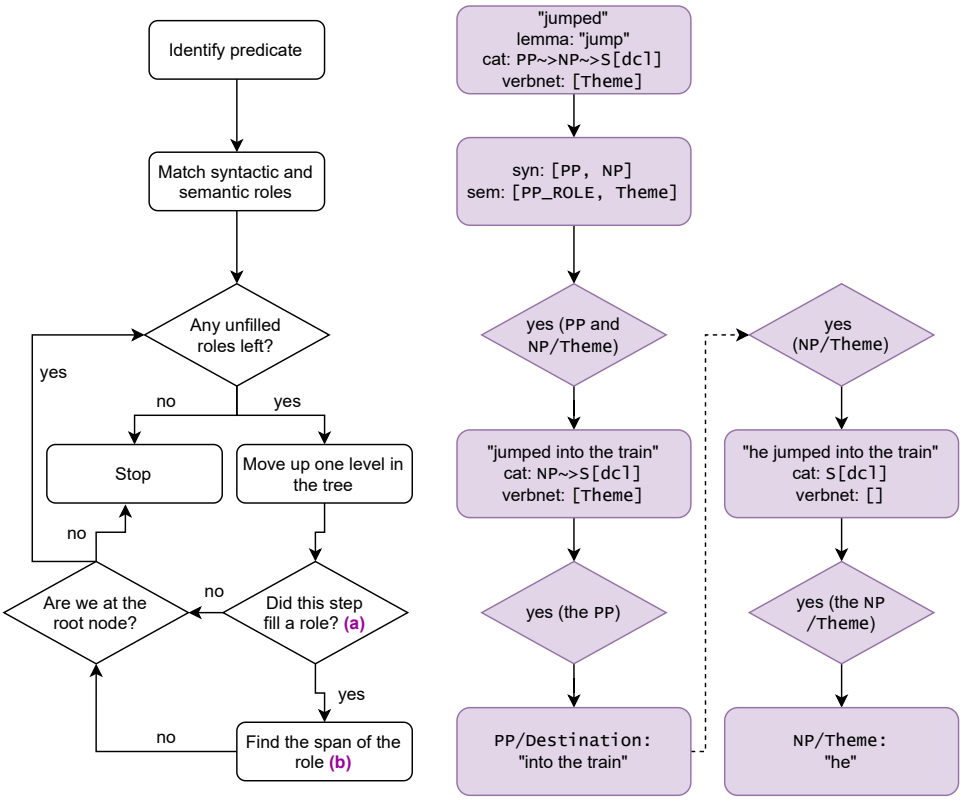


Figure 6: Flow chart of the main CCG-based conversion process. Algorithmic steps in white, example in purple.

merged at that point of the tree as the filler of the corresponding semantic role. This process is repeated until we have found a filler for every role, or until we reach the top of the tree.[4]

Detecting merged constituents A crucial step of our process (step *(a)* in Figure 6) is detecting, given a particular node in the tree, whether a role has been resolved at that node. In many cases, this is straightforward; for example, in the sentence in Figures 5 and 6, we can see that *he* fills the NP/Theme role of *jump* at the point where *he* is combined with *jumped into the train* through simple functional application, changing the type signature from NP→S to S. In other cases, more complicated rules are needed, for example when dealing with to-clauses (*She wants me to leave*), where, on combining *wants me* with *to leave*, the type signature of *to leave* changes from NP→S[to] to NP→S[dcl]. In such cases, at first glance, it appears as if not much has changed except a change of clause type (from a *to*-clause to a declarative sentence), whereas in fact, *me* has filled the subject NP of *leave*, and a new NP argument (the subject NP of *wants*) has been added. We have developed a set of heuristics that cover all such difficult cases occurring in the gold annotations in the PMB. While we believe that this amounts to a wide general coverage, it is likely that there exist other constructions that our algorithm does not (yet) cover.

Once it has been defined that a role is resolved at a given node in the tree, the next crucial step (step *b* in Figure 6) is to find the correct role span within the constituent that was combined. In many cases (like *he* in *he jumped*), the entire constituent is the role filler, but in other cases (like *wants me* in *She wants me to leave*), only a part of the constituent (*me*) is the role filler that we are looking for. To find this constituent, we designed a separate algorithm that moves down the tree starting from the merged constituent, until an argument with the correct type is found.

PP and adverbial roles Semantic roles carried by PP constituents (e.g. *into the train*) or by adverbial phrases (e.g. *quickly*) pose an additional chal-

[4] In some cases, e.g. *wh*-questions, it is possible that some roles remain unfilled.

lenge, since, in the PMB annotation framework, these roles are annotated on the syntactic head of the PP or adverbial phrase (e.g. *into* in *into the train*) rather than on the verb that they combine with. In cases where the PP is a syntactic argument of the verb (as in *jump into the train*), we solve this by first adding a placeholder role (see the `PP_role` at the top of Figure 6) corresponding to the verb's PP argument, and then replacing this by the semantic role carried by the PP at the point where it is combined with the predicate. In cases where a PP or adverb is an adjunct (e.g. with type signature `S→S` or `(NP→S)→(NP→S)`), we add the semantic roles introduced by the adjunct to the predicates in the constituent that is modified (e.g., *quickly* modifies *he ran* in *he ran quickly*. To ensure that adjuncts get the right scope, we added a constraint to our algorithm that forbids adding adjunct roles to predicates if doing so would cross a clause boundary; e.g., *loudly* in *he loudly said he was going to leave* can modify *said* but not *leave*.

Span-to-head conversion As a final step, to make the outputs of the CCG-based algorithm comparable to those of the DRS-based algorithm, we add a final step that converts the extracted role spans to their semantic heads. This algorithm consists of a set of (recursive) rules defining what the head of each type of phrase is. For example, $H(\text{the old woman}) = H(\text{old woman}) = H(\text{woman}) = \text{woman}$, where H is a function applying the appropriate rule for a given phrase type and returning the 'head part' of the phrase. There are many possible phrase types, but in general, the head of an NP is a noun, the head of a VP is a verb, the head of a PP is an NP, and the head of a sentence is the VP.

2.3 Comparing the approaches

Comparing the outputs of both conversion approaches, we find that 68% of documents match exactly, and 82% differ by at most one role. This shows that both approaches show significant differences worth further investigating. The differences mainly concern structural mismatches between syntax and semantics. For example, in sentences with co-referential NPs, CCG-based conversion gives more intuitive results than DRS-based conversion: in *she handed him*$_1$ *the money that she owed him*$_2$, DRS-based conversion treats the two *him*s as the same entity and assigns the Beneficiary role of *owe* to *him*$_1$, whereas CCG-based conversion correctly assigns it to *him*$_2$. Similarly, with reflexives, in *she saw herself*, DRS-based conversion is unable to assign any role to *herself*, since this word does not introduce a new discourse referent but refers back to *she*. The syntax-driven CCG-based conversion also allows for a better resolution of *hearer* and *speaker* discourse participants in such sentences as *I don't remember your name*.

On the other hand, CCG-based conversion has difficulties dealing with light verb constructions where the semantics of the main verb and the light verb interact. For instance, in *he had his wallet stolen*, the relationship between *he* and *stolen* is not detected. Finally, more heuristics will need to be added to CCG-based conversion to cover all adjunct semantic roles due to the way that these are annotated in the PMB, e.g. *by*-clauses in passive sentences. Also, the CCG-based conversion needs additional rules to distinguish between the semantic and syntactic head in such constructions as *all of the town* or *a kilo of plums*.

3 SRL Predictions

We predict semantic roles using the graph-based end-to-end coreference resolution system by He et al. (2018). This syntax-agnostic SRL model jointly predicts predicates, role fillers, and role labels. The SRL system builds contextualized representations for spans of arguments and predicate tokens based on BiLSTM outputs. The argument spans and predicates are predicted independently of each other and the aggressive beam pruning is used to discard the least probable combinations of predicate and argument spans. The output of the system is a graph, which lists predicted SRL roles as edges and the associated text spans as nodes. The SRL graph is predicted directly over text spans. Unlike He et al., we do not predict the full spans of semantic roles, but only syntactic heads of the semantic role spans, since the DRSs in the PMB do not contain information about full spans of arguments.[5] We experiment with GloVe (Pennington et al., 2014) and ELMo (Peters et al., 2018) embeddings to train the SRL system.[6]

We use the gold section of the English PMB data (release 3.0.0) to train and test the SRL system, which contains a train, dev, and test split of

[5]The full spans of semantic arguments can be reconstructed from head spans using syntactic information from dependency graphs (Gliosca and Amsili, 2019).

[6]The hyper-parameters are given in the appendix.

6 620, 885, and 898 documents, respectively. The SRL system is trained on the output of both DRS-to-SRL conversion tools separately. We include only verbal predicates and exclude the predicate *be* due to its inconsistent annotation in the PMB.

4 Merging DRS and SRL Predictions

As baseline DRS parsers without external SRL prediction, we use DRS parsers for which the output is publicly available: the transition-based compositional parser of Evang (2019) and three neural sequence-to-sequence models: the character-level model of van Noord et al. (2018b), an extension of this model that uses linguistic features (van Noord et al., 2019) and the best BERT-based model of van Noord et al. (2020). We refer to these models with E19, N18, N19, and N20.

We propose two methods for merging DRS and SRL output: a *token-based* method for parsers that are lexically anchored (each clause maps to one token), such as E19, and a *concept-based* method for parsers for which this is not the case (N18, N19, N20). Both methods only aim to *replace* roles in the DRS; no new full clauses are inserted.

Token-based merging When the SRL system predicts a predicate-role-filler tuple such as ⟨jumped,Theme,he⟩, we look for a corresponding role prediction in the parser output. A corresponding prediction is a role clause such as `b2 Agent e1 x1`, where the event discourse referent (`e1`) and the filler discourse referent (`x1`) are introduced by the corresponding tokens, i.e., *jumped*, and *he*, respectively. We say that a referent is introduced by a token if the token is anchored to a concept clause for that referent, such as `b2 jump "v.01" e1` or `b1 male "n.02" x1`. In this example, the DRS parser predicted a different role (Agent) than the SRL system (Theme), so we replace the former with the latter.

Concept-based merging Concept-based merging works similarly but does not rely on clauses being anchored to tokens. Instead, concept clauses are *matched* to tokens using corpus-level alignment and lemmatization. We say that a concept clause (e.g., `b1 male "n.02" x1`) *matches* a token (e.g., *he*) if it is observed anchored to the same word anywhere in the full PMB training data (bronze, silver, and gold). We also say that a concept clause (e.g., `b2 jump "v.01" e1`) matches a token (e.g., *jumped*) if there is a string match between the concept and the lemma[7] of the token (*jump*).

Restrictions In order to avoid some incorrect role replacements, we impose the following heuristics to restrict replacement: a role r is not replaced with r' if 1) r is one of the special roles `Time` and `Name`, 2) r' was predicted by the SRL system with $< 50\%$ precision, 3) r' already exists in the same box as r. For concept-based merging, the general concepts `person`, `be` and `entity` are never matched with any input tokens.

5 Experiments and Discussion

The main results of our experiments are shown in Table 2. Overall, we see small but consistent improvements for all parsers, except for N20, the most recent system. It seems that once the parser reaches a certain accuracy it is not straightforward to improve the scores by using an imperfect external system. This is also reflected by the number of replaced roles, which goes down as the parsers get better. Comparing the two conversion methods, we find that DRS-based conversion leads to higher scores. The difference with CCG-based conversion is small, though consistent between setups. In a sense, this is unsurprising given that DRS is also our target representation format. Furthermore, we found that using ELMo outperformed GloVe; while this is unsurprising, it supports the intuition that using a higher quality SRL system leads to more improvement. In other words, any development on the SRL parsing side is likely to lead to better performance on DRS parsing as well. Comparing token-based to concept-based merging on the output of the E19 parser (the only one where it is applicable), it makes more replacements and results in slightly higher accuracy, suggesting an advantage in terms of recall over concept-based merging.

Room for improvement As can be seen in Table 2, SRL performance seems to be a bottleneck; hence, using future, higher-quality SRL systems might also lead to better overall performance of our method. In particular, due to the merging step in our pipeline system, missing roles in SRL predictions are less costly than wrong predictions. Hence, we expect that SRL systems that are optimized for precision rather than for F-score will be more suited for use in our task. Furthermore, we

[7] We use spaCy (Honnibal et al., 2020) for this.

Experiments	SRL		E19-tok		E19		N18		N19		N20	
	dev	test	dev	test	dev	test	dev	test	dev	test	dev	test
Baseline	–	–	81.4 (0)	81.4 (0)	81.4 (0)	81.4 (0)	84.3 (0)	84.9 (0)	86.8 (0)	88.7 (0)	88.4 (0)	89.3 (0)
DRS conv.: upper	100	100	+1.5 (154)	+1.3 (144)	+1.3 (124)	1.2 (124)	+0.9 (92)	+1.2 (132)	+0.9 (88)	+1.1 (117)	+0.5 (51)	+0.7 (76)
CCG conv.: upper	100	100	+1.2 (145)	+1.2 (134)	+1.2 (115)	1.1 (118)	+0.9 (89)	+1.2 (129)	+0.8 (80)	+1.1 (114)	+0.5 (50)	+0.8 (78)
DRS conv. + GloVe	79.7	81.6	+0.3 (129)	+0.3 (113)	+0.4 (97)	+0.2 (102)	+0.2 (68)	+0.4 (92)	+0.1 (64)	+0.2 (90)	-0.2 (57)	-0.1 (70)
DRS conv. + ELMo	**85.8**	86.3	**+0.5** (128)	**+0.4** (120)	**+0.5** (104)	**+0.4** (110)	**+0.3** (73)	**+0.5** (107)	**+0.2** (74)	**+0.3** (104)	-0.1 (55)	0.0 (69)
CCG conv. + GloVe	80.7	83.0	+0.3 (129)	+0.3 (117)	+0.3 (107)	+0.2 (108)	+0.1 (96)	+0.4 (102)	0.0 (93)	+0.1 (103)	-0.2 (73)	-0.1 (74)
CCG conv. + ELMo	85.2	**87.0**	+0.4 (118)	**+0.4** (109)	+0.4 (99)	+0.3 (103)	+0.2 (81)	+0.4 (104)	+0.1 (73)	+0.2 (102)	-0.2 (63)	0.0 (66)

Table 2: Experiment results, including F-scores and number of replaced roles (in brackets). The F-scores are calculated using Counter (van Noord et al., 2018a). Scores for N19 and N20 are averaged over 5 runs. E19-tok uses token-based merging, E19 uses concept-based merging like the rest.

expect that further improvements in the conversion algorithms will lead to better overall performance.

Error analysis We identified four sources of errors in the SRL predictions. The data show an imbalanced role distribution towards the roles Theme and Agent, which take up 52% of all annotations out of 32 semantic roles. This leads to overprediction of these roles by the SRL-labeler. Indeed, for N20 we find that these roles have an insertion precision of $< 50\%$, or in other words, they were more often wrongly inserted than that they correctly replaced a non-matching role. Figure 7 shows the confusion matrix for the most frequent semantic roles.

pred./gold	Agent	Co-Theme	Dest.	Exper.	Loc.	Patient	Source	Stim.	Theme
Agent	337	0	0	5	0	1	0	0	5
Co-Theme	0	54	0	0	0	1	0	0	3
Destination	0	0	33	0	2	0	0	0	0
Experiencer	1	0		62	0	3	0	1	1
Location	0	0	0	0	62	0	0	0	0
Patient	2	0	1	1		77	0	1	7
Source	1	0	0	0	0	0	21	1	2
Stimulus	2	0	0	0	0	0	0	56	2
Theme	14	2	1	0	2	7	0	4	356

Figure 7: Confusion matrix for semantic labeling errors, showing the numbers of predicted labels for the most frequent labels.

The role Theme and Agent are also frequently predicted extra in cases where no semantic role should be predicted. For example, the pronoun *her* in the sentence *she ate her dinner* is erroneously assigned the role Agent. Semantic roles of prepositional phrases also lead to prediction errors. For example, the phrase *the field of biology* in the sentence *He is working in the field of biology* is wrongly recognized as Location instead of Theme. Another cause of prediction errors are possessive determiners which are wrongly predicted as role fillers. For example, both *her* and *dinner* are predicted as Patient in the following sentence: *She ate her dinner*. Also, no semantic roles are predicted by the SRL-labeler if the head word has no vector embedding due to a special character, for example like *post∼office*. Due to the merging step in our pipeline, the erroneously missing semantic roles in SRL predictions do not lead to a drop of parsing performance and also do not improve it.

6 Conclusions and Future Work

We have presented experiments on using externally predicted semantic roles to improve the output of four recent DRS parsers. We saw that there is considerable room for improvement and our method fills it – but not fully, especially as parsers get more accurate. We conclude that our approach is useful especially with parsers such as E19 which do not reach state-of-the-art accuracy but may have other advantages such as smaller models or lexical anchoring. An advantage of our approach is that it is very flexible: it can be applied on top of any DRS parsing model without having to alter or retrain the model itself. This means that our method, or an improved version of it, could also be applied to future DRS parsers, possibly with completely different architectures. In future work we intend to experiment with enhancing the SRL system using syntactic input from CCG-based supertags and also try out other SRL systems. We also plan to experiment with prediction of nominal and adjectival predicates along with their semantic roles. We also intend to reconstruct and predict full spans of semantic roles. Moreover, we plan to carry out parsing experiments with further languages in the PMB, including Dutch, German, and Italian, as our method should be universally applicable. Finally, it would be interesting to improve the SRL predictions by enforcing coherence of predicted predicates and corresponding semantic roles.

Acknowledgements

We would like to thank two anonymous reviewers for their valuable comments. The work presented in this paper has been partially funded by the European Research Council, within the ERC grant TreeGraSP[8].

References

Lasha Abzianidze. 2017. LangPro: Natural language theorem prover. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 115–120, Copenhagen, Denmark. Association for Computational Linguistics.

Lasha Abzianidze, Johannes Bjerva, Kilian Evang, Hessel Haagsma, Rik van Noord, Pierre Ludmann, Duc-Duy Nguyen, and Johan Bos. 2017. The Parallel Meaning Bank: Towards a multilingual corpus of translations annotated with compositional meaning representations. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 242–247, Valencia, Spain. Association for Computational Linguistics.

Claire Bonial, William Corvey, Martha Palmer, Volha V Petukhova, and Harry Bunt. 2011. A hierarchical unification of lirics and verbnet semantic roles. In *2011 IEEE Fifth International Conference on Semantic Computing*, pages 483–489. IEEE.

Kilian Evang. 2019. Transition-based DRS parsing using stack-LSTMs. In *Proceedings of the IWCS Shared Task on Semantic Parsing*, Gothenburg, Sweden. Association for Computational Linguistics.

Federico Fancellu, Ákos Kádár, Ran Zhang, and Afsaneh Fazly. 2020. Accurate polyglot semantic parsing with DAG grammars. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3567–3580, Online. Association for Computational Linguistics.

C.J. Fillmore. 1968. The case for case. In E. Bach and R. Harms, editors, *Universals in Linguistic Theory*. Holt, Rinehart and Winston, New York.

Quentin Gliosca and Pascal Amsili. 2019. Résolution de coréférence basée sur les têtes. In *Actes de la conférence TALN 2019 (articles courts)*, Toulouse.

Luheng He, Kenton Lee, Omer Levy, and Luke Zettlemoyer. 2018. Jointly predicting predicates and arguments in neural semantic role labeling. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 364–369, Melbourne, Australia. Association for Computational Linguistics.

[8] https://treegrasp.phil.hhu.de/

Matthew Honnibal, Ines Montani, Sofie Van Landeghem, and Adriane Boyd. 2020. spaCy: Industrial-strength Natural Language Processing in Python.

Hans Kamp and Uwe Reyle. 1993. *From Discourse to Logic*. Studies in Linguistics and Philosophy. Kluwer, Dordrecht, Boston, London.

Tao Li, Parth Anand Jawale, Martha Palmer, and Vivek Srikumar. 2020a. Structured tuning for semantic role labeling. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8402–8412, Online. Association for Computational Linguistics.

Zuchao Li, Shexia He, Hai Zhao, Yiqing Zhang, Zhuosheng Zhang, Xi Zhou, and Xiang Zhou. 2019. Dependency or span, end-to-end uniform semantic role labeling. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6730–6737.

Zuchao Li, Hai Zhao, Rui Wang, and Kevin Parnow. 2020b. High-order semantic role labeling. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1134–1151, Online. Association for Computational Linguistics.

Jiangming Liu, Shay B. Cohen, Mirella Lapata, and Johan Bos. 2021. Universal Discourse Representation Structure Parsing. *Computational Linguistics*, pages 1–33.

Diego Marcheggiani and Ivan Titov. 2020. Graph convolutions over constituent trees for syntax-aware semantic role labeling. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3915–3928, Online. Association for Computational Linguistics.

Rik van Noord, Lasha Abzianidze, Hessel Haagsma, and Johan Bos. 2018a. Evaluating scoped meaning representations. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Rik van Noord, Lasha Abzianidze, Antonio Toral, and Johan Bos. 2018b. Exploring neural methods for parsing discourse representation structures. *Transactions of the Association for Computational Linguistics*, 6:619–633.

Rik van Noord, Antonio Toral, and Johan Bos. 2019. Linguistic information in neural semantic parsing with multiple encoders. In *Proceedings of the 13th International Conference on Computational Semantics - Short Papers*, pages 24–31, Gothenburg, Sweden. Association for Computational Linguistics.

Rik van Noord, Antonio Toral, and Johan Bos. 2020. Character-level representations improve DRS-based semantic parsing Even in the age of BERT. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4587–4603, Online. Association for Computational Linguistics.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Tianze Shi, Igor Malioutov, and Ozan Irsoy. 2020. Semantic role labeling as syntactic dependency parsing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7551–7571, Online. Association for Computational Linguistics.

Mark Steedman. 2000. *The Syntactic Process*. MIT Press.

Appendix

Layer	**Hyper-parameters**	**Value**
Characters CNN	numb. of filters	50
Bi-LSTM	state size # layers	200 3
Words embedding	vector dim.	300
Char. embedding	dimension	8
	batch size	40
Dropout	dropout rate	0.5
	Max. gradient norm	5.0
	Optimizer	Adam
	Learning rate	0.001
	Decay rate	0.999
	Decay frequency	100

Hyper-parameters of the SRL system.

Data-Driven Frame-Semantic Parsing with Tree Wrapping Grammar

Tatiana Bladier **Laura Kallmeyer** **Kilian Evang**
Heinrich Heine University Düsseldorf, Germany
first.last@hhu.de

Abstract

We describe the first experimental results for data-driven semantic parsing with Tree Rewriting Grammars (TRGs) and semantic frames. While several theoretical papers previously discussed approaches for modeling frame semantics in the context of TRGs, this is the first data-driven implementation of such a parser.[1] We experiment with Tree Wrapping Grammar (TWG), a grammar formalism closely related to Tree Adjoining Grammar (TAG), developed for formalizing the typologically inspired linguistic theory of Role and Reference Grammar (RRG). We use a transformer-based multitask architecture to predict semantic supertags which are then decoded into RRG trees augmented with semantic feature structures. We present experiments for sentences in different genres for English data. We also discuss our compositional semantic analyses using TWG for several linguistic phenomena.

1 Introduction

While many user-facing applications of Natural Language Processing such as machine translation or sentiment analysis can these days be performed with state-of-the-art accuracy by syntax-agnostic machine learning models, grammar-based methods are still important. For one thing, they offer more transparency and insight into the decisions of a model, while in many cases having near-state-of-the-art performance (Xia et al., 2019; Kasai et al., 2019; Lindemann et al., 2019; Poelman et al., 2022). Secondly, they tend to be less data-hungry and therefore more readily adapted or transferred to low-resource languages. Symbolic methods for semantic parsing can also greatly contribute to grammar theory studies and to linguistic investigations of different languages.

[1]The code for our semantic parser can be found on https://github.com/TaniaBladier/Frame-Semantic-Parser-with-Lexicalized-Grammars

In this paper, we are interested in developing a methodology for deep semantic parsing (i.e., producing semantic representations for entire sentences) which would also allow easy transfer to different languages, including low-resource ones. We start from the typologically oriented linguistic theory of Role and Reference Grammar (RRG). This theory uses a common inventory of labels and structures to describe languages from different language families (Van Valin and Foley, 1980; Van Valin, 2005). The formalization of RRG using Tree Wrapping Grammar (TWG; Kallmeyer et al., 2013) has paved the way for using this theory in computational linguistics and for developing NLP applications such as syntactic parsers (Bladier et al., 2022; Evang et al., 2022).

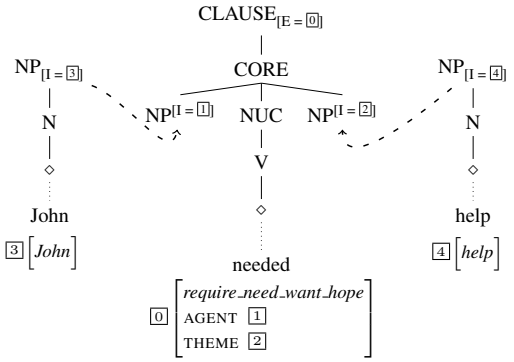


Figure 1: Frame-semantic derivation with TWG for *John needed help*

The TWG formalism is inspired by Tree-Adjoining Grammar (TAG; Joshi and Schabes, 1997) and allows for adequate modeling of long-distance dependencies. Since TWG is closely related to TAG, we can readily apply existing computational methods developed for TAG. In this work, we explore how well the methodology for compositional semantics with a tree-based syntax outlined in several theoretical papers on TAG (Kallmeyer and Osswald, 2012a,b; Zinova and

Proceedings of the 15th International Conference on Computational Semantics pages 223–232
June 21–23, 2023.

Kallmeyer, 2012) is suitable for TWG and can be used for a large scale implementation.

A small-scale frame-semantic parser based on the Tree Adjoining Grammar was already implemented by Arps and Petitjean (2018). Our approach differs from theirs in that it is data-driven and aims for a broad-coverage semantic parser. Our method is based on transformers and contextual embeddings and we do not use a metagrammar in our application, but go for an approach based on supertagging. Our work also differs from Semantic Role Labeling (i.e., shallow semantic parsing) with TAG (Liu and Sarkar, 2009; Kasai et al., 2019) since we are interested in deep semantic representations of the sentences. Figure 1 shows how the semantic representations for the sentence *John needed help* can be produced compositionally with elementary trees in TWG paired with frames, and Figure 3 shows the frame representation for this sentence.

The objective of this paper is to implement a broad-coverage semantic parser based on Tree Rewriting Grammars. Since this is the first broad-coverage implementation of a deep semantic parser for either TAG or TWG, we are particularly interested in modeling linguistic phenomena which we came across during this data-driven implementation. We describe this in §2. We also want to investigate if our syntax-aware methodology allows us to achieve state-of-the-art results on semantic parsing. We describe the theoretical background of our work and introduce our approach to frame-based semantics with TWG in §3 and present experimental results in §4. We discuss future work in §5.

2 Semantic Parsing with TWG

2.1 Tree Wrapping Grammar

TWGs consist of elementary trees which can be combined using the operations of a) *substitution* (replacing a leaf node with a tree), b) *sister adjunction* (adding a new daughter to an internal node), and c) *tree-wrapping substitution* (adding a tree with a d(ominance)-edge by substituting the lower part of the d-edge for a leaf node and merging the upper node of the d-edge with the root of the target tree, see Fig. 2). The latter is used to capture long distance dependencies (LDDs), see the wh-movement in Fig. 2. Here, the left tree with the d-edge (depicted as a dashed edge) gets split; the lower part fills a substitution slot while the upper part merges with the root of the target tree. TWG is more powerful than TAG (Kallmeyer, 2016). The reason is that a) TWG allows for more than one wrapping substitution stretching across specific nodes in the derived tree and b) the two target nodes of a wrapping substitution (the substitution node and the root node) do not have to come from the same elementary tree, which makes wrapping non-local compared to adjunction in TAG.

TWG emerged as a result of the formalization of Role and Reference Grammar (RRG; Van Valin and LaPolla, 1997; Van Valin, 2005). RRG is a linguistic theory strongly inspired by typological concerns. RRG was used to describe languages with diverse syntactic structures such as Lakhota, Tagalog, and Dyirbal. RRG's syntactic structures are rather flat in order to be applicable to all types of different languages. According to RRG, sentence structure is organized in layers: nucleus (containing the predicate), core (containing the nucleus and the arguments of the predicate) and clause (the core and extracted arguments). Each layer can have modifiers (called periphery elements), and operators attach to the layer over which they take semantic scope.

2.2 Frame Semantics and TWG

We adapt the syntax-semantics interface for LTAG proposed by Kallmeyer and Osswald (2013) to semantic parsing with TWG. Kallmeyer and Osswald represent semantic frames as base-labelled, typed feature structures. The frames can be understood as a straightforward representation of the semantic and conceptual knowledge about a situation, while having good computational properties as their composition relies on the unification of attribute-value structures. The frames represent genuine semantic representations, and not logical expressions, whose meaning has to be derived during semantic composition[2].

The elementary trees in a lexicalized TWG are paired with frames via interface feature structures, as shown in Figure 1. Here, the root of the elementary tree for 'needed' is augmented with an interface feature structure whose E (event) attribute value is a frame of type $require\_need\_want\_hope$, which has two attributes: an agent and a theme.

[2]The advantage of the unification is that the order of semantic argument filling is not specified by successive lambda abstraction or the like. Instead, semantic argument slots can be filled in any order (in particular, independently of surface word order) via unifications triggered by syntactic composition). For a more detailed discussion see Kallmeyer and Romero (2004) and Kallmeyer and Osswald (2014)

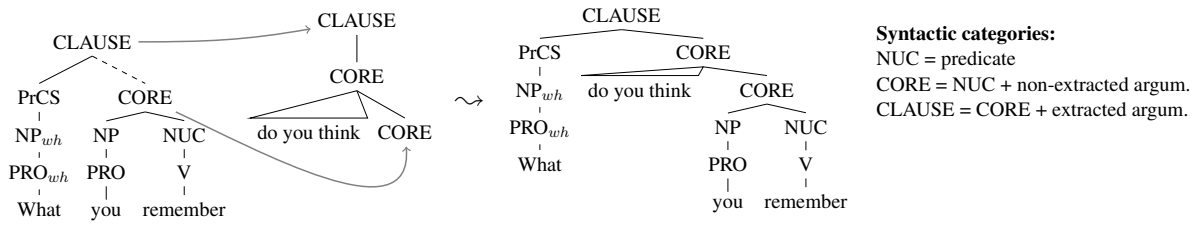


Figure 2: Tree-wrapping substitution for the sentence *"What do you think you remember"* with long-distance wh-movement.

$$\boxed{0}\begin{bmatrix} \textit{require\_need\_want\_hope} \\ \text{AGENT} \;\; \textit{John} \\ \text{THEME} \;\; \textit{help} \end{bmatrix}$$

Figure 3: Frame-semantic representation for *John needed help*.

The values of these attributes are shared with the feature structures paired with the NP substitution nodes for the subject and the object, where they are the values of the I (individual) attribute[3]. The roots of the elementary trees for 'John' and 'help' are augmented with feature structures for whose I attribute values are feature structures for whose types we use the respective lemmas (more detailed semantic representations of NPs are beyond the scope of this paper).

During parsing, as syntactic trees are combined (by adjunction, substitution or wrapping substitution), the semantic representations are also combined. The unification of interface feature structures triggers unification of feature values in the frames. In our example, as the substitution of the subject NP takes place (combining the elementary trees of 'needed' and 'John'), the respective values associated to the attribute I in the interface feature structures are unified. This results in the unification of the feature structures 3 and 1, which makes the frame for John become the agent of the event 'needed'. The same happens when the tree for 'help' is substituted at the object NP node of the 'needed' tree: 4 and 2 unify to let the frame for 'help' become the value of the theme attribute in the frame 0.

To build our frame lexicon, we use the inventory of the lexical-semantic resource VerbAtlas (Di Fabio et al., 2019). VerbAtlas covers over 13 700 verbal WordNet (Fellbaum, 2000) senses, but organizes them into a relatively small number of frames (466) with only 25 cross-frame semantic roles, which makes it well suited for training neural language models. The frames in VerbAtlas are mapped to PropBank (Palmer et al., 2005) framesets and multilingual BabelNet (Navigli and Ponzetto, 2010) frames, and can potentially be linked to FrameNet (Baker et al., 1998; Baker, 2014) frames.

2.3 Complex linguistic cases

In the process of developing our data-driven semantic parser, we came across several complex linguistic constructions which were not previously described in papers dealing with the combination of Tree Rewriting formalisms and semantics. Depending on the syntactic complexity of the sentences, such constructions occur in about 20% of all sentences in our data, distributed unevenly among the subcorpora we used for the experiments. We describe some of our semantic modeling choices in this section[4].

Control constructions We introduce the variable *pivot* for cases in which an elementary tree does not have an explicit syntactic argument, but shares the argument with an elementary tree it combines with. Figs. 4 and 5 show an example. The *pivot* variable is only assigned to CORE nodes and is used to propagate the semantic representation of the controlled argument.

Constructions with a peripheral subordinate clause The representation of discourse relations is beyond the scope of this work, so for now we generate semantic representations for such clauses separately. Fig. 6 shows the elementary tree-frame pairs and Fig. 7 shows a representation for the sentence *The sheep follow him because they know his voice*.

Constructions with a non-peripheral subordinate clause If a subordinate clause is not a modi-

[3]The feature I is used as a variable in untyped frames referring to an argument (possibly syntactically complex) which fills the substitution slot.

[4]For the sake of space we only represent the relevant elementary trees in the figures of this section and skip some initial elementary trees that are substituted or adjoined into the larger trees.

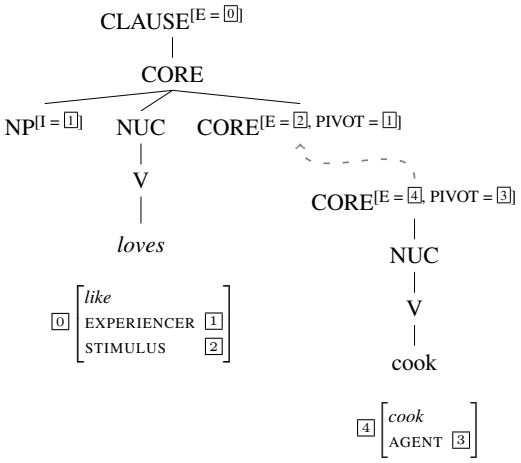


Figure 4: The pivot variable in semantic representation of the sentence *She loves to cook*.

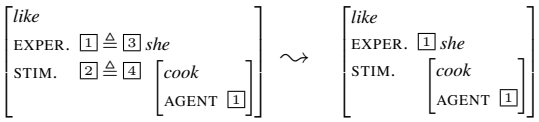


Figure 5: Label unifications and resulting frame for *she loves to cook.*

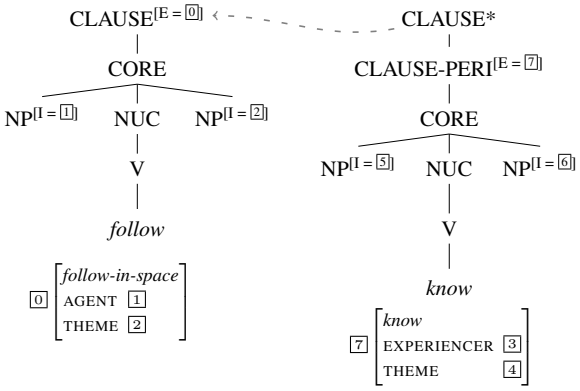


Figure 6: Tree-frame pairs for the sentence *The sheep follow him because they know his voice*

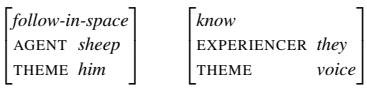


Figure 7: Semantic representations of a main clause and a peripheral subordinate clause in sentence *The sheep follow him, because they know his voice*

fier, but an argument of a main clause, the frame of the subordinate clause fills the corresponding argument slot of the parent frame (see the elementary trees and frame representation in Fig. 8, 9 for the sentence *What people say about themselves means nothing*).

Treatment of prepositional phrases The treatment of prepositional phrases depends on whether

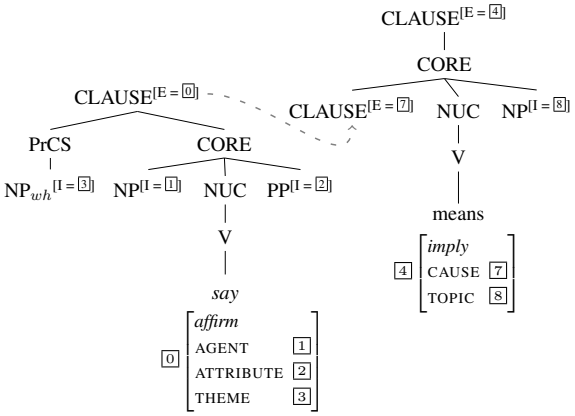


Figure 8: Tree-frame pairs for constructions with subordinate clauses

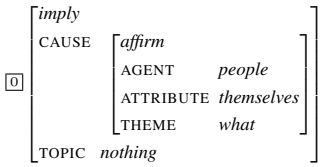


Figure 9: Constructions with subordinate clauses, here *What people say about themselves means nothing*

the PP is an argument or an adjunct of the predicate. In (1-a) below, the PP fills a core role of the predicate *lowered*. However, the role filler *well* for this argument slot should itself be substituted first into the elementary tree of the preposition *into*. Thus, to propagate the filler of the *destination* role to the designated argument slot of *lowered*, we check during the substitution of the PP subtree and the subsequent frame unification that the argument role of the PP corresponds to the required argument role of the sentential predicate (see Fig. 10). If the prepositional phrase is an adjunct of the predicate (as, for example, in (1-b), where *with a check* modifies the predicate *pay*), the subframe of the prepositional phrase is added as an additional semantic role of the predicate after adjoining the PP subtree.

Since we focus on verbal predicates in this work, we do not explore an explicit frame representation of different prepositions, as outlined in Kallmeyer and Osswald (2013). Instead, we leave the representation of prepositions and other non-verbal predicates for future work.

(1) a. Tom lowered the bucket into the well.
b. I want to pay with a check.

Constructions with non-local dependencies Constructions with non-local dependencies (e.g.

long-distance wh-movement or extraposed relative clauses) can be handled via unification during wrapping substitution (see tree-frame pairs in Fig. 11 and the resulting representation in Fig. 12).

	Supertag	**Frame**	**Arg. Link.**
she	(NP (PRO ◇))	(entity)	(–)
loves	(CL (CO (NP) (NUC (V ◇)) (CORE)))	(like)	((1, 'Exp.'), (2, 'Stim.'))
to	(CO* (CLM ◇))		(–)
cook	(CO (NUC (V ◇)))	(cook)	((0, 'Agent'))

Table 1: Example of the training data, CL stands for Clause, CO means Core.

3 Method

3.1 Argument linking

As outlined in the previous section, our approach to semantic parsing requires two components which are used to compositionally produce a deep semantic representation of the sentences: TWG elementary trees and the corresponding semantic frames. We divide prediction of semantic frames into two subtasks: prediction of the correct frame and learning the argument linking within those frames.

The argument linking mechanism relies on the elementary tree of the predicate and predicts which substitution slot of the supertag carries which semantic role. For example, in Table 1 the argument linking for the predicate *likes* means that the first substitution slot of the corresponding supertag should get the role label "Experiencer" and the second slot gets the label "Stimulus", hence the numbers 1 and 2. In case an elementary tree has a semantic role with no local filler, as in control or raising constructions (see Figure 4) or in sentences with conjoining predicates, we mark the semantic role with the index 0, indicating that there is

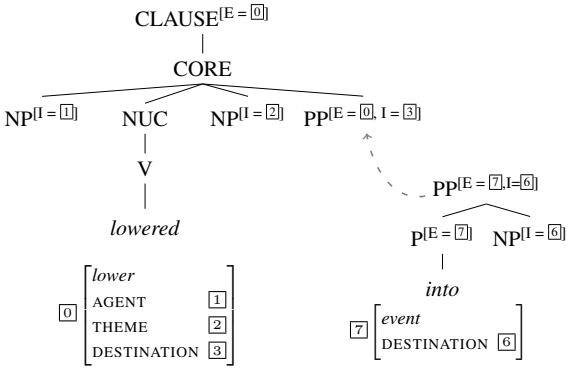


Figure 10: Propagating the role of the argument PP *into* to the main frame *lower* for the example (1-a)

no substitution slot for this role (see, for example the frame *cook* in Table 1). For non-predicative frames we learn the frame with the dummy type ENTITY and resolve the type of the frame to the corresponding lemma after parsing.

3.2 Reducing the size of TWG grammars

Since the TWG grammars are usually large and contain several thousands distinct elementary trees, which is potentially hard for a neural model to learn, we reduce the size of the grammar by flattening the elementary trees and thus simplifying the syntactic structure of the trees from which we induce the TWG grammar. We collapse the internal structure of the trees, so that it preserves the relevant syntactic information about the lexical anchor and its argument structure. In particular, we delete the internal nodes of the tree which are not relevant for syntactic composition (i.e. the nodes are not involved in any tree combination operations) while leaving the root node and unlexicalized leaves untouched. We delete all SENTENCE nodes while keeping however the spine of CLAUSE, CORE and NUC since these are important targets for modifier and operator adjunctions. Figure 13 shows an example. After flattening the trees, we extract a TWG elementary trees using the automated grammar extraction approach of Bladier et al. (2020a). Since the syntactic trees in TWG grammars can have crossing branches, but the algorithm for TWG parsing (Bladier et al., 2020b), which we use to obtain syntactic representations for our data, does not support crossing branches, some nodes in trees have to be reattached before grammar extraction and re-attached to the correct nodes after parsing.

3.3 Multi-task transformer-based learning

We use the MaChAmp toolkit (van der Goot et al., 2021) to build a multi-task neural model for simultaneous learning of the elementary tree templates (i.e. supertags), frame selection, and argument linking, all cast as sequence labeling tasks. The MaChAmp multi-task models share a BERT-based encoder, but use task-specific decoders for the subtasks. Table 1 shows an example of the input for the multi-task neural model. We initially experimented with training a single-task model for each subtask and tried out different combinations of multi-task models. Since the results of a multi-task model turned out to be comparable with the single-task models (showing only around 0.1 percent of difference), we therefore carry out our ex-

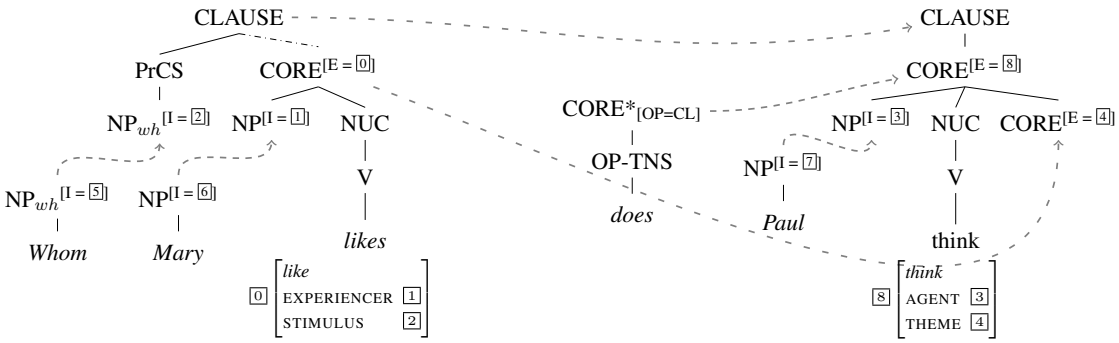


Figure 11: Wrapping substitution for wh-LDD in sentence *Whom does Paul think Mary likes?* The OP=CL notion means that the node will be attached to the CLAUSE node of the parent tree after the parsing step.

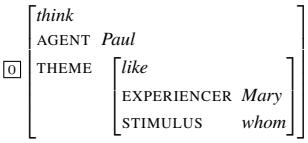


Figure 12: Semantic representation for an LDD construction in *Whom does Paul think Mary likes?*

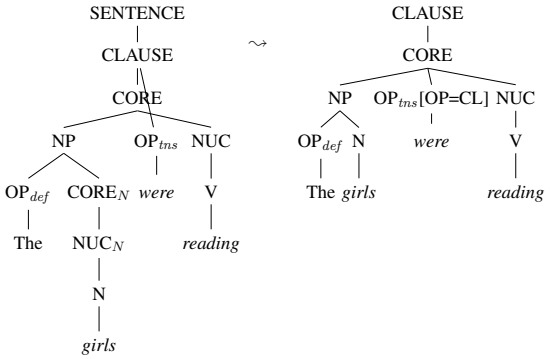


Figure 13: Example of a transformed tree before grammar extraction: the crossing branch from the original tree (on the left) is reattached and some of the internal nodes are removed. OP=CL indicates that the OP$_{tns}$ node was originally immediately below CLAUSE.

periments with the multi-task model. This model has the advantage of predicting all the components of our semantic parsing approach at once, resulting in lower training and prediction times. We tried to apply different weights on the loss function of each subtask to see if it affects the performance of the multi-task model, however the results did not change significantly. Apart from experimenting with different loss functions, we used the default values of the MaChAmp Bert model for training. The model is trained for 10 epochs, and we select the model with the highest F1-checkpoint for the evaluation.

4 Experiments and Discussion

4.1 Data

Since there is currently no manually annotated gold dataset for semantic parsing with TWG, we use alternative resources to train our model. We use the statistical neural TWG parser ParTAGe (Bladier et al., 2020b) developed for syntactic parsing with TWGs and train it on multilingual data from RRGparbank, the first large resource for TWG and Role and Reference Grammar (Bladier et al., 2022). The ParTAGe parser predicts the syntactic trees based on predicted n-best supertags for each sentence and also predicts the dependency heads based on the produced syntactic tree. The performance of this parser is different for sentences with different sentence length, but is sufficiently high for shorter sentences. We measured the ParTAGe performance on English sentences from the RRGparbank corpus (since the parser was originally trained on this data). We found that the performance of the parser on sentences with less then 7 tokens had the labeled F1 score of 93.52 for the produced syntactic trees, and the labeled F1 score of longer sentences was around 85.26.

We use the Parallel Meaning Bank v3.0.0 (PMB; Abzianidze et al., 2017) and the CoNLL-2012 English dataset based on OntoNotes 5.0 (Pradhan et al., 2012) for the frame-semantic parsing experiments. The PMB provides deep semantic representations of sentences following Discourse Representation Theory. It has rather short sentences (around 6.7 tokens on average) consisting of Web texts, newspaper articles and the Bible. The English part of the CoNLL-2012 corpus is a large resource which includes over 94 000 sentences from different genres, including journal articles, web data, broadcast news and phone conversations. We

use the pre-defined train, development and test sets for both resources (see Table 2).

	PMB	**OntoNotes**
# sents (train, dev, test)	6654, 886, 902	75187, 9480, 9260
avg. sent length	6.94	16.71
# tokens	54205	201300
# lemmas	5463	10975
# dist. frames	350	436
# dist. frame/lemma pairs	949	2965
# frame occurrences	4783	34930
# role occurrences	13495	45496
# supertags	782	4158
# supertags occ. once	354	2204

Table 2: Statistics on the used data.

PMB and OntoNotes are not explicitly annotated with VerbAtlas frames, but PMB provides WordNet senses and VerbNet semantic roles, and OntoNotes is annotated with PropBank framesets and semantic roles. Since VerbAtlas provides manually created mappings to these resources, we used these mappings to create a sufficient amount of semantically annotated data. In order to obtain syntactic representations needed for our frame-semantic parser, we parse all sentences with the pretrained ParTAGe models available from Bladier et al. (2022).

4.2 Frame-semantic parsing experiments

Our frame-semantic parser predicts supertags needed to produce syntactic trees in parallel with the frame labels and corresponding semantic roles. We predict only heads of the semantic roles, since the full spans can be reconstructed deterministically from the predicted syntactic trees. We use the constituent trees produced by our parser to reconstruct the full spans of semantic roles[5].

VerbAtlas has 466 frames, 350 of which we observe in PMB and 436 in the OntoNotes data. The distribution of the frames is relatively even, without any frames occurring particularly more frequent then other frames. We do not consider frames associated with modal verbs. Since some of the frames occur only in test or development set and thus cannot be learned, we calculate the upper bound for the data to determine what would be the highest possible achievable score. The evaluations show a long tail of prediction errors without particular errors occurring more often then the others. Table 4 shows some of the most frequent mistakes.

[5]We reconstructed full spans of semantic roles only for OntoNotes, since the data from PMB are not annotated with full-span semantic roles.

The distribution of the supertags is uneven with a couple of most frequent ones occurring in the majority of the cases. We found 225 distinct predicative supertags in the PMB data, and 1358 in OntoNotes. Table 5 shows that the first three most common predicative supertags make up around two thirds of all predicates in PMB. A similar distribution is also present in the larger OntoNotes corpus, although the frequency of the most common supertags is less prominent.

The results of the frame-semantic parsing show that we achieve results comparable with the baseline Semantic Role Labeling (SRL) results on the OntoNotes and show a slight improvement on the PMB data (see Table 3[6]). The results on different genres in OntoNotes show a significant increase in performance on the Bible data and the worst results for the web texts. This result is due to the greater sentence length for the web data and a high amount of internet slang and deviations from standard English orthography and syntax.

4.3 Error analysis

Although VerbAtlas has rather coarse-grained frame lexicon, the number of frames (466) is still large and some frame pairs have only a subtle difference in its definition (e.g. the frame pairs GO-FORWARD and LEAVE_DEPART_RUN-AWAY or AFFIRM and SPEAK). Also there are some verbs, like for example *go*, which are polysemous and can be assigned different frames which appear more or less frequent in the annotated data. Since the majority of the frames appear only a couple of times in the training data, the model sometimes predicts the wrong frame which appears more frequently, as for example the frame LEAVE_DEPART_RUN-AWAY is wrongly predicted instead of CONTINUE in example (2).

(2) [...] but they're determined to keep going[leave_depart_run-away]

Each frame in VerbAtlas comes with its own set of semantic roles. Although the number of the roles is small (26), the model has to learn the correct labels for each of the 466 frames. Since for most frames in VerbAtlas, the agentive and patientive role have the labels AGENT and THEME, the

[6]We use the following terms while describing our semantic parsing experiments: the term *trigger* stands for a lexical unit that can evoke a frame, the term *role* for frame element, and *role candidate* for the sequence of words that instantiates a role.

	PMB	OntoNotes					
		avg.	bn+bc	nw+mz	pt	tc	wb
frame trigger detection	93.75	92.92	92.35	92.14	96.41	94.79	91.56
frame label selection (w. *entity* and *event* labels)	89.75	89.57	88.56	88.65	95.81	92.5	86.15
frame label selection (only VA-labels)	83.9	89.06	87.93	87.78	97.11	92.06	85.48
*upper bound	99.81	99.46	99.59	99.38	99.71	99.65	98.88
role candidate detection	91.1	87.47**	86.54**	87.91**	91.45**	86.45**	86.25**
role label selection (head)	**86.15**	89.67**	88.36**	90.08**	93.16**	89.56**	88.15**
role label selection (full span)	–	**88.34****	87.61**	88.63**	92.11**	88.82**	86.43**
role label selection (baseline, head)	85.8 Bladier et al. (2021)	**92.1** InVeRo-XL (Conia et al., 2021)					
role label selection (baseline, full span)	–	86.8 InVeRo-XL (Conia et al., 2021)					
avg. sent. length	5.99	14.73	14.36	20.09	11.02	8.04	16.71
# sents	902	9260	2968	2568	1051	1618	1055

Table 3: Frame-semantic parsing results. We use the frame inventory from VerbAtlas (VA; Di Fabio et al., 2019) in our semantic representations. The role label selection for full spans is not evaluated for the PMB experiment, since only semantic heads of role spans are annotated in gold PMB data. *Since some labels from the test set are not present in the training data, we measure the highest possible upper bound for the VA-label selection. **We measure the scores for OntoNotes only for pre-identified predicates to make the evaluations comparable with the reported baseline. bn+bc = broadcast, nw+mz = newswire, pt = bible, tc = telephone conversations, wb = web.

Gold frame	Predicted frame	%
GO-FORWARD	LEAVE_DEPART_RUN-AWAY	0.7
CONTINUE	LEAVE_DEPART_RUN-AWAY	0.48
INCITE_INDUCE	EXIST-WITH-FEATURE	0.42
KNOW	MEET	0.42
RESULT_CONSEQUENCE	ARRIVE	0.42

Table 4: Most frequent frame label prediction mistakes with the percentage from the overall frame label prediction errors, measured on OntoNotes data.

Supertag	% (PMB)	% (ON)
(CL (CO (NP) (NUC (V ◇)) (NP)))	38.82	8.5
(CL (CO (NP) (NUC (V ◇))))	14.37	6.64
(CL (CO (NP) (NUC (V ◇)) (PP)))	10.62	3.3
(CL (CO (NP) (NUC (V ◇)) (NP) (NP)))	7.6	0.1
(CL (CO (NP) (NUC (V ◇)) (P) (NP)))	5.28	0.01

Table 5: Most common predicative supertags for PMB and OntoNotes (ON) data.

model frequently picks these two labels instead of some less frequent frame-specific role labels. For example in (3), the correct role set for the COME-AFTER_FOLLOW-IN-TIME frame is THEME and CO-THEME, but the model predicts the more common AGENT and THEME role labels.

(3) That$_{[agent]}$ follows$_{[come\text{-}after_follow\text{-}in\text{-}time]}$ a decline$_{[theme]}$ in the prior six months [. . .]

As for the errors in prediction of argument linking, the most errors emerge when an infinitive modifies a noun or an adjective (see an example in (4)). The supertag for the verb in such constructions has the type of an auxiliary tree and thus lacks the agentive argument slot. In these cases, the semantic role corresponding to the PIVOT variable sometimes is not predicted (we described the PIVOT in greater detail in Section 2.3). For example, in (4) for the MANAGE frame, only the role THEME is predicted, but not the AGENT role for *strategy*.

(4) A time-honored strategy to control$_{[manage]}$ the masses$_{[theme]}$.

5 Conclusion and Future Work

In this paper, we presented the first broad-coverage frame-semantic parser with Tree Wrapping Grammar, a grammar formalism closely related to Tree Adjoining Grammar. To develop our parser, we adapted the theoretical approach of Kallmeyer and Osswald (2013) to semantic parsing with TAG and transferred it to TWG. We explored parsing strategies for several complex linguistic constructions. We developed our transformer-based language model based on the VerbAtlas frame lexicon, and experimented with English data in several genres. We could see that our semantic parser shows results close to the state-of-the-art semantic parsers.

In future work we want to explore the transferability of our approach to different languages, in-

cluding low-resource ones. Our approach to semantic parsing starts from statistical syntactic parsing for TWG proposed by Waszczuk (2017); Bladier et al. (2020b). A recent work by Evang et al. (2022) presents a modification of this method for cross-lingual syntactic parsing based on word embeddings and English glosses. The underlying idea is to transfer supertag information from an English translation to the target sentence via word alignments. We plan to extend this method to semantics.

The frame lexicon VerbAtlas, which we use as a frame inventory for the semantic representations, lacks relations between frames. In order to enable semantic inference and logical reasoning with our parser, we currently investigate possibilities to develop a rule-based mapping from VerbAtlas frames to FrameNet frames, which would then yield also hierarchical relations between frames.

Acknowledgments

We thank three anonymous reviewers for their advice and useful comments. This work was carried out as a part of the research project TreeGraSP[7] funded by a Consolidator Grant of the European Research Council (ERC).

References

Lasha Abzianidze, Johannes Bjerva, Kilian Evang, Hessel Haagsma, Rik van Noord, Pierre Ludmann, Duc-Duy Nguyen, and Johan Bos. 2017. The Parallel Meaning Bank: Towards a multilingual corpus of translations annotated with compositional meaning representations. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 242–247, Valencia, Spain. Association for Computational Linguistics.

David Arps and Simon Petitjean. 2018. A Parser for LTAG and Frame Semantics. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Collin F. Baker. 2014. FrameNet: A knowledge base for natural language processing. In *Proceedings of Frame Semantics in NLP: A Workshop in Honor of Chuck Fillmore (1929-2014)*, pages 1–5, Baltimore, MD, USA. Association for Computational Linguistics.

Collin F. Baker, Charles J. Fillmore, and John B. Lowe. 1998. The Berkeley FrameNet project. In *36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics, Volume 1*, pages 86–90, Montreal, Quebec, Canada. Association for Computational Linguistics.

Tatiana Bladier, Kilian Evang, Valeria Generalova, Zahra Ghane, Laura Kallmeyer, Robin Möllemann, Natalia Moors, Rainer Osswald, and Simon Petitjean. 2022. RRGparbank: A parallel role and reference grammar treebank. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 4833–4841, Marseille, France. European Language Resources Association.

Tatiana Bladier, Laura Kallmeyer, Rainer Osswald, and Jakub Waszczuk. 2020a. Automatic extraction of tree-wrapping grammars for multiple languages. In *Proceedings of the 19th Workshop on Treebanks and Linguistic Theories*, pages 55–61.

Tatiana Bladier, Gosse Minnema, Rik van Noord, and Kilian Evang. 2021. Improving DRS Parsing with Separately Predicted Semantic Roles. In *Workshop on Computing Semantics with Types, Frames and Related Structures: Workshop at ESSLLI 2021*. ESSLLI.

Tatiana Bladier, Jakub Waszczuk, and Laura Kallmeyer. 2020b. Statistical parsing of tree wrapping grammars. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6759–6766.

Simone Conia, Riccardo Orlando, Fabrizio Brignone, Francesco Cecconi, and Roberto Navigli. 2021. Invero-xl: Making cross-lingual semantic role labeling accessible with intelligible verbs and roles. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 319–328.

Andrea Di Fabio, Simone Conia, and Roberto Navigli. 2019. Verbatlas: a novel large-scale verbal semantic resource and its application to semantic role labeling. 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)*, pages 627–637.

Kilian Evang, Laura Kallmeyer, Jakub Waszczuk, Kilu von Prince, Tatiana Bladier, and Simon Petitjean. 2022. Improving low-resource RRG parsing with cross-lingual self-training. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 4360–4371, Gyeongju, Republic of Korea. International Committee on Computational Linguistics.

Christiane D. Fellbaum. 2000. WordNet: an electronic lexical database. *Language*, 76:706.

Rob van der Goot, Ahmet Üstün, Alan Ramponi, Ibrahim Sharaf, and Barbara Plank. 2021. Massive choice, ample tasks (MaChAmp): A toolkit for multi-task learning in NLP. In *Proceedings of the 16th*

[7] https://treegrasp.phil.hhu.de

Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations, pages 176–197, Online. Association for Computational Linguistics.

Aravind K Joshi and Yves Schabes. 1997. Tree-adjoining grammars. In *Handbook of formal languages*, pages 69–123. Springer.

Laura Kallmeyer. 2016. On the mild context-sensitivity of k-Tree Wrapping Grammar. In *Formal Grammar: 20th and 21st International Conferences, FG 2015, Barcelona, Spain, August 2015, Revised Selected Papers. FG 2016, Italy, August 2016, Proceedings*, number 9804 in Lecture Notes in Computer Science, pages 77–93, Berlin. Springer.

Laura Kallmeyer and Rainer Osswald. 2012a. An analysis of directed motion expressions with lexicalized tree adjoining grammars and frame semantics. In *Logic, Language, Information and Computation. 19th International Workshop, WoLLIC 2012*, number LNCS 7456 in Lecture Notes in Computer Science, pages 34–55, Buenos Aires, Argentina. Springer. Proceedings.

Laura Kallmeyer and Rainer Osswald. 2012b. A frame-based semantics of the dative alternation in lexicalized tree adjoining grammars. *Empirical Issues in Syntax and Semantics*, 9:167–184.

Laura Kallmeyer and Rainer Osswald. 2013. Syntax-driven semantic frame composition in lexicalized tree adjoining grammars. *Journal of Language Modelling*, 1:267–330.

Laura Kallmeyer and Rainer Osswald. 2014. Syntax-driven semantic frame composition in lexicalized tree adjoining grammars. *Journal of Language Modelling*, 1(2):267–330.

Laura Kallmeyer, Rainer Osswald, and Robert D. Van Valin, Jr. 2013. Tree Wrapping for Role and Reference Grammar. In *Formal Grammar 2012/2013*, volume 8036 of *LNCS*, pages 175–190. Springer.

Laura Kallmeyer and Maribel Romero. 2004. LTAG semantics with semantic unification. In *Proceedings of TAG+7*, pages 155–162, Vancouver.

Jungo Kasai, Dan Friedman, Robert Frank, Dragomir Radev, and Owen Rambow. 2019. Syntax-aware neural semantic role labeling with supertags. *arXiv preprint arXiv:1903.05260*.

Matthias Lindemann, Jonas Groschwitz, and Alexander Koller. 2019. Compositional semantic parsing across graphbanks.

Yudong Liu and Anoop Sarkar. 2009. Exploration of the ltag-spinal formalism and treebank for semantic role labeling. In *Proceedings of the 2009 Workshop on Grammar Engineering Across Frameworks*, pages 1–9. Association for Computational Linguistics.

Roberto Navigli and Simone Paolo Ponzetto. 2010. BabelNet: Building a very large multilingual semantic network. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 216–225, Uppsala, Sweden. Association for Computational Linguistics.

Martha Palmer, Daniel Gildea, and Paul Kingsbury. 2005. The proposition bank: An annotated corpus of semantic roles. *Computational linguistics*, 31(1):71–106.

Wessel Poelman, Rik van Noord, and Johan Bos. 2022. Transparent semantic parsing with universal dependencies using graph transformations. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 4186–4192.

Sameer Pradhan, Alessandro Moschitti, Nianwen Xue, Olga Uryupina, and Yuchen Zhang. 2012. Conll-2012 shared task: Modeling multilingual unrestricted coreference in ontonotes. In *Joint conference on EMNLP and CoNLL-shared task*, pages 1–40.

Robert D. Van Valin, Jr. 2005. *Exploring the Syntax-Semantics Interface*. Cambridge University Press.

Robert D. Van Valin, Jr. and William A. Foley. 1980. Role and reference grammar. In E. A. Moravcsik and J. R. Wirth, editors, *Current approaches to syntax*, volume 13 of *Syntax and semantics*, pages 329–352. Academic Press, New York.

Robert D. Van Valin, Jr. and Randy LaPolla. 1997. *Syntax: Structure, meaning and function*. Cambridge University Press.

Jakub Waszczuk. 2017. *Leveraging MWEs in practical TAG parsing: towards the best of the two worlds*. Ph.D. thesis, Université François Rabelais Tours.

Qingrong Xia, Zhenghua Li, Min Zhang, Meishan Zhang, Guohong Fu, Rui Wang, and Luo Si. 2019. Syntax-aware neural semantic role labeling. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 7305–7313.

Yulia Zinova and Laura Kallmeyer. 2012. A frame-based semantics of locative alternation in LTAG. In *Proceedings of the 11th International Workshop on Tree Adjoining Grammar and Related Formalisms (TAG+11)*, pages 28–36, Paris.

Chapter 6

Discussion and Conclusion

Key findings In this dissertation, we worked on the implementation of the syntax-semantics interface for the typologically grounded structural-functionalist theory of language Role and Reference Grammar (RRG). Since its starting point in the 1980s, Role and Reference Grammar was used to describe several dozens of languages from around the globe, and proved to be useful both for language description as well as for exploring central theoretical issues in linguistics. Kallmeyer et al. (2013) developed a formalization of RRG - Tree Wrapping Grammar - in order to use RRG in computational linguistics and to potentially develop NLP tools for a wide variety of languages, including rare, endangered, and low-resource ones. Starting from this formalization, we implemented a full system for probabilistic RRG grammar induction together with a syntactic and semantic parsing tool based on TWG/RRG. Our system can also be used for other Tree Rewriting Formalisms, such as TAG and its different variations, since it allows to extract different types of syntactic trees used in both formalisms (traditional initial and auxiliary trees used for TAG, but also sister-adjoining and d-edge trees used for TWG).

Since TWG is closely related to TAG, we have shown that a variety of algorithms and NLP tools existing for TAG can be re-used for RRG. In particular, we adapted an existing TAG induction algorithm to develop a methodology for extraction of linguistically motivated probabilistic RRG grammars from constituency treebanks. We also adapted and modified an existing syntactic parser for TAG ParTAGe (Waszczuk, 2017) and enhanced it with the supertagging component to boost the parsers accuracy and speed.

Since one of the main strengths of TWG is treatment of non-local dependencies (NLDs), we compare our parsing system with two LCFRS-based syntactic parsers DiscoDOP (van Cranenburgh et al., 2016) and Discoparset (Coavoux and Cohen, 2019) and show that our parser outperforms them in recognizing NLDs, while using

linguistically motivated syntactic constructions in the underlying grammar. Extraction of the elementary trees for NLDs from treebanks however remains challenging due to the heterogeneous nature of NLDs and their language specificity. In our work we present several types of NLDs we encountered in English and German data and outline a strategy to induce elementary trees for them from the treebanks.

According to RRG, the syntax of a natural language can only be understood with reference to its semantic and communicative functions (cf. Van Valin (2005)). In our dissertation we investigated the ways to computationally implement the outlined syntax-semantics interface and to use it for RRG-based semantic parsing. The implementation revealed several challenges, since the methodology for such a parser has been so far only theoretically described in the literature (Van Valin, 2005; Kallmeyer and Osswald, 2014) and only prototypically implemented for LTAG in Arps and Petitjean (2018) for a small number of verbal predicates. Arps and Petitjean (2018) also did not pursue a large scale implementation and their parser is based on the metagrammar and not on a grammar directly extracted from a treebank. A data-driven implementation of the syntax-semantics interface for RRG revealed challenges with certain linguistic phenomena which we investigated in this dissertation.

As a part of our dissertation project, we also co-worked on the creation of two first large treebanks for RRG - a monolingual corpus RRGbank and a parallel multilingual resource RRGparbank. These resources provided insights on not yet described linguistic phenomena in RRG and enabled first investigations of the "core" RRG grammar, a set of syntactic constructions common to multiple languages. We could also leverage these resources for the implementation of both syntactic and semantic parsers. Our practical implementation proved these treebanks suitable for integration within downstream NLP applications. Moreover, our parsing experiments establish a benchmark for subsequent RRG-based applications.

Detailed discussion We included eight studies in this cumulative dissertation to discuss various aspects of syntactic and semantic parsing based on lexicalized Tree Adjoining Grammar (TAG) and Tree Wrapping Grammar (TWG). We started our dissertation project with four main research questions in mind, which we can now answer. Let us begin with the first one:

(i) Can we propose an algorithm for inducing Tree Rewriting Grammars from an RRG-annotated treebank? Can this grammar extraction algorithm be applied across multiple languages?

The elementary trees which constitute Tree Rewriting Grammars capture the syntactic arguments of their lexical anchor, thus making the predicate-argument relations explicit - a property which was previously discussed as particularly promising for

semantic parsing (Liu and Sarkar, 2009; Kallmeyer and Osswald, 2014; Arps and Petitjean, 2018). We pursued the idea proposed by Kallmeyer and Osswald (2014) to map the elementary trees to meaning representations and develop a semantic parser which would derive the meaning of utterances compositionally during syntactic parsing. The development of the syntax-mediated semantic parser from scratch includes several key components, which we addressed in this dissertation. We described the process of obtaining probabilistic grammars to be used for parsing and outlined our methodologies for syntactic and semantic parsing. Finally, we showed the process of creating sufficiently large treebanks to train our language models on.

Our created semantic parser is of interest to several fields of linguistics for various reasons: to the best of our knowledge, this is the first large-scale implemented semantic parser based on tree-rewriting grammars which uses the theoretical basis of the typologically oriented Role and Reference Grammar. We showed that our parser reaches state-of-the-art semantic parsing performance while bringing some important advantages for natural language modeling and processing. The grammar-aware approach to semantic parsing allows to develop more transparent language models, compared to "black boxes" of neural syntax-agnostic tools. Grammar-based methods which we used in the dissertation also require less data and can be potentially extended to a broad variety of languages, even to low-resource ones, as has been done in Evang et al. (2022). The methodology we developed in this dissertation is particularly relevant in the context of typological linguistic theories such as RRG, Functional Discourse Grammar (Dik, 1987), or Systemic Functional Linguistics (Halliday, 1961), for which a lot of typologically oriented research is done, which means research mostly on low resource languages.

TWG was developed as a part of the overall goal of formalizing RRG. One of the most important linguistic properties of TWG is that it allows for a linguistically motivated modeling of non-local dependent elements in utterances, such as for example an extraposed wh-clause in the sentence *'What do you think you remember?'* in Fig. 6.1. The wh-phrase *'what'* in such sentences does not appear in the canonical position of a direct object in the CORE of the predicate (here after the verb *'remember'*), but in the front of the phrase. In TWG, such constructions are handled by the tree combination operation *wrapping substitution*. During wrapping substitution, the elementary tree with a dominance edge (on the left in Fig. 6.1) is split in two so that the lower part fills the substitution node of the target tree and the upper part is added to the root of the target tree). In our first study **(1)** "Automatic Extraction of Tree-Wrapping Grammars for Multiple Languages" we developed an algorithm to extract TWG grammars from treebanks. We adapted and extended the top-down algorithm developed by Xia et al. (2000) for TAG extraction. We introduced special extraction operations to account for sister-adjunction

and wrapping substitution which are used in TWG. Since the frequency and the types of non-local dependencies (NLDs) vary broadly across the languages, only a few types of constructions with non-local dependencies (NLDs) can be recognized automatically based on predefined rules (for example, relative clauses introduced by the pronoun *que* in French with a raising construction between *que* and the governour predicate) (Candito and Seddah, 2012). Thus, we found it necessary to manually mark all parts of NLDs in the corpus prior to grammar extraction. It should be noted that the frequency of such constructions in corpus data depends on the language (accounting typically around 1% or 2% of sentences in English newspaper or literary texts, but occurring more frequently in languages with free constituent order like German or Russian). We identified two types of NLDs: (1) the long-distance dependencies (LDD) include the non-local dependencies in which an obligatory syntactic element is moved to the front position within the scope of syntactic element. This includes such constructions as wh-extraction, relativization or topicalization. We called the second type (2) of NLDs extraposed relative clauses (ERCs). In ERCs, the relative clause is extracted from its canonical position (i.e. directly following the introducing relative pronoun) typically to the position right of the scope, which leads to a non-local coreference link between the relative clause and its antecedent nominal phrase (like for example in *Nothing has happened that you did not foresee.*). In the first study **(1)** we showed that these two types of NLDs require slightly different procedures to extract elementary trees, since LDDs concern obligatory syntactic arguments and ERCs contain syntactic modifiers. For example, the sentence in example 6 has a non-local dependency (NLD) of the type of a long-distance wh-extraction (wh-LDD):

(1) What do you think you remember

The NLD in this sentence is handled with *wrapping substitution* in TWG, since it is an extraction out of an argument, i.e., the wrapping substitution fills a CLAUSE argument slot while adding the extracted part to the left of the entire CLAUSE. Figure 6.1 shows an example of wrapping substitution between the elementary trees of *'remember'* and *'think'*. The wrapping substitution allows to simultaneously put both parts of a discontinuous constituent to the slots of the target elementary tree.

The ERC type of non-local dependencies is slightly different from the LDD type since it involves the extraposition of a modifier. It therefore requires a slightly different modeling with putting the antecedent NP and the relative clause into the same elementary d-edge tree, as exemplified in Fig. 6.2 (cf. Kallmeyer (2021)).

In the study **(1)** "Automatic extraction of tree-wrapping grammars for multiple

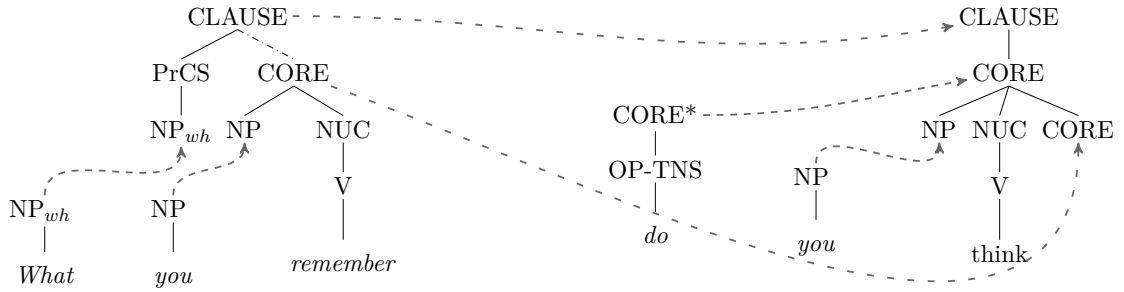


Figure 6.1: Wrapping substitution for the wh-long distance dependency (wh-LDD) in sentence *What do you think you remember?*

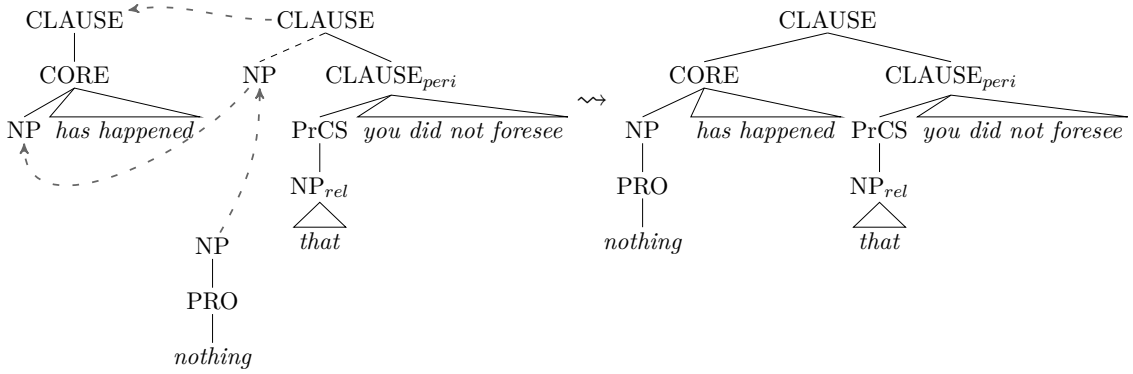


Figure 6.2: Wrapping substitution for the extraposed relative clause (ERC) in the sentence *nothing has happened that you did not foresee.*

languages" we extracted TWG grammars for English, German, French, and Russian. We thus showed that our developed algorithm for TWG extraction can be used to extract grammars from different languages and also adapted to grammar extraction for other languages in future research. Potentially, the algorithm can be applied to every constituency treebank and also dependency treebanks (after converting dependency trees to constituency trees).

Our answer to the first research question is thus that we could indeed adapt the existing induction algorithm for TAG induction and use it for Tree Wrapping Grammar induction. However, the TWG-specific operation of *wrapping substitution* requires special manually added node markers in order to identify the parts of non-local dependent elements in sentences.

Let us now look at the second research question:

(ii) Is it feasible to develop a syntactic parser for RRG based on its formalization as a Tree Wrapping Grammar?

The three studies **(2)**-**(4)** comprising Chapter 3 *Supertagging and Parsing with Tree Rewriting Grammars* discuss parsing strategies with the extracted grammars. We

adapted the classical parsing method of the pipeline supertagging and a subsequent actual parsing step. Supertagging is traditionally referred to as being 'almost parsing' because it reduces the number of choices which an actual parser has to make, since the actual parsing step is in general computationally costly. For example, a TAG for different languages usually consists of around two to three thousands trees, but the supertagger reduces this number to several most probable supertags per token in a sentence. The supertagging step also helps to increase the parsing speed. In **(2)** "German and French Neural Supertagging Experiments for LTAG Parsing", we investigated the supertagging strategies for German and French and developed a neural architecture of the supertagger which shows results for both languages comparable with the supertagging results for English. We also showed that accuracy prediction is improving up to 5-best predicted supertags per token, while for ranks n ≥ 6 the improvement of accuracy is not as big. We find that the supertagging accuracy depends on the sentence length (thus, we had good results for German despite the free word order) and the amount of multiword constructions in the treebank, as such constructions are usually annotated differently from the rest of the corpus, which can lead to a confusion by the training of the neural supertagger.

Supertagging, however, is not yet parsing. In the study **(3)** "From partial neural graph-based LTAG parsing towards full parsing" we discussed the results presented by Kasai et al. (2017, 2018) and showed that a high accuracy in 1-best supertagging does not lead to a full parse tree in many cases. We demonstrated that for the English TAG even if just one supertag from the sequence of most probable supertags for the sentence is not correct, in almost half of the sentences these supertags cannot be combined to form a syntactic tree for the sentence. We showed that prediction of n-best supertags is necessary to produce syntactic trees for the majority of the sentences. We investigated how many predicted supertags per token are sufficient for parsing and show that the rank of n depends on the average sentence length, but that in general $n = 10$ is sufficient to parse sentences from the French Treebank (which has an average sentence length of 30 tokens). We developed a parsing pipeline consisting of a supertagger based on the BiLSTM Deep Learning algorithm and adapt the A* star parser for TAG developed by Waszczuk (2017) to be able to deal with supertags. The new version of this parser considers each set of n-best supertags predicted sentence-wise as complete grammars, from which the parser chooses a sequence of one supertag per token to produce a constituency tree for the sentence. This accelerates the parsing speed as it reduces the grammar and thus also the number of choices the A* star parser has to make. We also trained the parser to predict the bilexical dependencies between the supertags, i.e. the information which supertags should be combined with each other. We showed that prediction of such dependencies along with the supertags further speeds up the A* parsing architecture,

as it reduces the size of the hypergraph created by the parser. It should be noted however, that the bilexical dependencies do not resolve the attachment ambiguity - thus, if a target supertag has two internal nodes with the same label, a bilexical dependency does not tell explicitly the exact node of the tree combination operation. Supertagging is a sequence labeling problem, thus machine learning algorithms, such as neural networks of different architectures are well suitable for this task. In this paper we used a BiLSTM-based model, but as we showed in later papers, other more advanced deep learning architectures can be employed as well, for example transformer-based models, which are in the last years increasingly used for NLP tasks (Vaswani et al., 2017; Wolf et al., 2020).

In the study **(4)** "Statistical Parsing of Tree Wrapping Grammars" we elaborated on our modification of an A* parser, which was initially developed for TAG, to handle tree combination operations unique to TWGs, such as sister adjunction and wrapping substitution. In this study we investigated if statistical parsing with TWGs is better suitable for handling the non-local dependencies than other syntactic parsers, since representation of NLDs is a particular strength of TWG. We showed that such constructions appear differently frequent in different languages. For instance, German tends to have more extraposed relative clauses than English or French. This is because German allows for more flexibility in how sentences are structured, and there is a tendency to avoid heavy NPs, particularly those with relative clauses, which often occur before the sentence's final position because of the verb-final order. We compared our parsing approach with the discontinuous data-oriented parsing model DiscoDOP developed by van Cranenburgh and Bod (2013). We also compared our results with the state-of-the-art transition-based parser Discoparset (Coavoux and Crabbé, 2017) and show that our parser shows a better performance with regard to NLD recognition. We investigated the errors of the parser and find that the cases where a relative clause or *wh*-phrase of the NLD is an adjunct are harder to process for the parser. Also the cases in which a relative clause is introduced with a wh-word and contains a verb that usually takes a wh-element as argument, as for example with verbs of communication (i.e. *'say'*) were not processed correctly, i.e. in the phrase *slip of paper which they said was the bill* the pronoun *which* is wrongly analyzed as an argument of the verb *said*).

Our answer to the research question (ii) is thus that it is possible to develop a syntactic parser for RRG based on TWGs and that it works well for parsing non-local dependencies, modeling of which is one of the strengths of TWG. Our parsing strategy exploits the idea of using TAG-like syntactic templates described in Van Valin (2005) and Van Valin (2023) for parsing. Some other approaches for RRG parsing has been discussed previously (see Guest (2009); Diedrichsen (2014); Cortes-Rodriguez (2016); Mairal-Usón and Cortés-Rodríguez (2017)). These approaches

are, however, not able to deal with long-distance dependencies and two of them are either language- or application-specific Diedrichsen (2014); Cortes-Rodriguez (2016) and none of the existing RRG parsing approaches has a large-scale implementation.

Let us now turn to the third research question:

(iii) What strategies are effective in building a large linguistic resource for RRG to support NLP development?

We addressed the research question (iii) in papers **(5)** "RRGbank: a Role and Reference Grammar Corpus of Syntactic Structures Extracted from the Penn Treebank" and **(6)** "RRGparbank: A Parallel Role and Reference Grammar Treebank". Although it is possible to extract a TWG from any kind of constituency treebank, TWGs are tightly linked to Role and Reference Grammar, as the TWG methodology was originally developed for its formalization in order to use it in computational applications. Since there were no suitably large linguistic resources for RRG before we started our dissertation project, we built two resources to have enough data for statistical parsing experiments and grammar induction: RRGbank **(5)** "RRGbank: a Role and Reference Grammar Corpus of Syntactic Structures Extracted from the Penn Treebank" and RRGparbank **(6)** "RRGparbank: A Parallel Role and Reference Grammar Treebank". We were interested to build a resource which would cover as many linguistic phenomena as possible for different languages. We were also interested in finding a treebanking methodology which would allow a relatively speedy annotation while also producing gold standard RRG trees. We started out our annotations converting the constituency trees from the Penn Treebank to RRG structures. For RRGparbank we wanted to build a parallel multilingual corpus in order to be able to better compare the grammar phenomena in different languages and to build NLP tools based on gold data for several languages, thus we decided to annotate the novel '1984' by George Orwell which has translations into many languages, some of which were annotated with Universal Dependencies. We decided to manually validate the resulting annotations in order to achieve a reliable RRG-based treebank. We found that the majority of the trees from Penn Treebank could be converted with a rule-based script to the RRG structures. However, we identified several reasons for systematic conversion errors, among which were the annotation errors or inconsistencies in PTB and some underspecifications in PTB with regard to the RRG theory (for example, the attachment site of the negation operator in RRG depends on its scope, which is not reflected in PTB). In order to bootstrap our rule-based conversions and to potentially create RRG-based corpora in several languages, we decided to implement an additional conversion script from Universal Dependencies to RRG annotations. We used both conversion algorithms for RRG-

parbank. As we had manually validated enough annotated data, we trained our developed RRG/TWG parser on them and used the parser for annotations, since it produced more accurate results. We showed that approximately 1000 annotated sentences in every language were enough to train a sufficiently accurate parser.

Since the RRG annotations differ from traditional constituency trees in several ways (for example, the RRG annotations have disconnected nodes and a constituent along with an operator projection for each sentence), we developed a notational variant for RRG syntactic structures in order to make annotations conform with the usual constituency treebank notations. While Figure 6.3 shows the sentence with traditional RRG annotations, the Figure 6.4 shows the tree as it would be represented with our notation in the RRGparbank. This notational variant can easily be converted to a traditional RRG annotation To reconstruct the operator projection in Figure 6.4 one should mirror the spine of our tree notation. The node label for *"do"* indicates that this is a tense operator (OP-TNS) and its parent node, CORE, is the site of attachment on the operator projection.

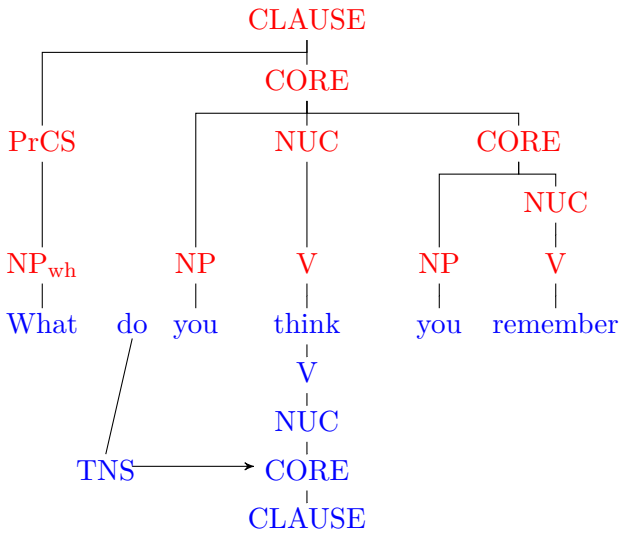


Figure 6.3: Original RRG notation for the sentence 'What do you think you remember'.

Generally, the majority of studies based on RRG are focused around certain linguistic phenomena and are not based on English or other European languages. Since we opted for a data-driven approach, we were able to discover language phenomena not yet covered by the RRG theory. Some of those phenomena were rather language-specific (for example, the clitics in French, stranding of CLM '*to*' in English or constructions with modal adverbials in Russian, such as lemmas *'mozhno'* (*'to be allowed'*, *'to can'*) or *'nuzhno'* (*'to need'*). Another example are Russian particles, which can scope over different token spans and do not have a fixed sentential position due to the free constituent order in Russian. Other open questions

concerned phenomena from a potentially large variety of languages (for example, quantifier phrases, multiword expressions or reported speech require a detailed analysis within the RRG theory). Since no systematic analysis of the languages we used for RRGbank and RRGparbank was made in RRG, our annotations revealed several open questions for these languages, which required personal consultations with the founder of the RRG theory Robert D. Van Valin Jr.

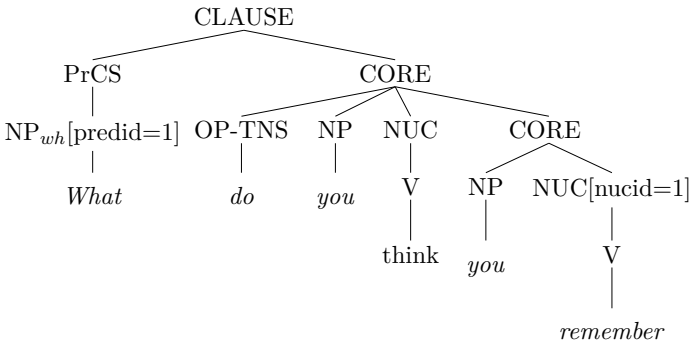


Figure 6.4: Constituency tree for 'What do you think you remember' from previous Figure in our RRGparbank notation. PREDID (predicate identification number) and NUCID (nucleus identification number) with the same ID number indicate the parts of the non-local dependency. We use these indications for automatic extraction of d-edge trees during the TWG induction procedure.

In order to annotate semantics in RRGparbank, we chose the VerbAtlas frame lexicon (Di Fabio et al., 2019). We chose VerbAtlas because it uses a relatively small set of roles, which is beneficial for training NLP models. However, it also offers a sizable range of roles that adequately capture various aspects of meaning. The VerbAtlas frames are directly associated with the English senses of verbal words, covering all the verbal synsets present in WordNet. These synsets are further connected to verb senses across other languages. Additionally, the VerbAtlas roleset aligns closely with the roleset of VerbNet, which is structured in a hierarchy ranging from coarse-grained to fine-grained roles. This hierarchical organization enables the training of models with varying levels of granularity, depending on specific applications. We annotated only heads of semantic role spans and not full spans—for example, if the whole NP *the swift answer* was supposedd to fill a Theme role, only the word *answer* was annotated as Theme. The full spans of semantic roles can be reconstructed deterministically from the corresponding syntactic trees (cf. Bladier et al. (2023)).

In the last chapter of the dissertation we addressed the question of the role of syntactic input for semantic parsers and whether multi-task models can be beneficial for the parsing performance. This led to our fourth and final research question:

(iv) How can tree rewriting formalisms be used for data-driven RRG-based se-

mantic parsing? Can state-of-the-art semantic parsing results be achieved by combining Extended Domain of Locality (EDL)-based syntax and semantic frames?

In paper **(7)** "Improving DRS Parsing with Separately Predicted Semantic Roles" we investigated another type of semantic representations - the Discourse Representation Structures as introduced by Discourse Representation Theory (Kamp and Reyle, 1993). We developed an approach to convert existing DRS representations to semantic role labels (SRLs) in order to be able to predict SRLs separately and afterwards use to improve the parsers. We showed a syntax-aware and a syntax-agnostic way to convert DRSs to SRLs. We experimented with several DRS-based full semantic parsers ans show that the use of separately predicted SRLs to improve the performance of the existing DRS parsers, even of the neural based ones, depending on the accuracy of the SRL predictions. We developed two conversion scripts from DRSs to semantic role labeled spans. The first script uses only the DRS notations for conversion, while the second one makes use of the CCG annotations which are associated with the DRSs in Parallel Meaning Bank (PMB; (Abzianidze et al., 2017)) to reconstruct the correct SRL-spans. The conversion from DRSs to SRLs revealed several challenges resulting from the structural mismatches be- tween syntax and semantics. For example, the syntax-aware CCG-based conversion is better at handling co-referential NPs (like for example assigning correct roles and predicates to both *'him'* tokens in sentence 'she handed him_1 the money that she owed him_2') or the reflexives, as for example in 'she saw herself'. In the latter case, the syntax-agnostic DRS-based conversion is unable to assign a semantic role to 'herself', since this token does not introduce a new discourse referent. On the other hand, the syntax-agnostic conversion approach is better at dealing with light verb constructions in which the semantics of the main verb interacts with the semantics of the light verb. For example, the CCG-based approach does not detect the relationship between '*he*' and '*stolen*' in '*he had his wallet stolen*'. After extracting the gold SRLs from the PMB-data, we train a neural semantic role labeling system and add the results on top of the existing DRS parsers, merging the SRL predictions with the output of the parsers. We showed that our approach is especially useful for parsers which might not reach state-of-the-art accuracy, but may provide other advantages such as smaller models or lexical anchoring. We also showed that our approach is very flexible and can be applied on top of any DRS parsing model without having to alter or retrain the model itself.

Finally, in the last paper **(8)** "Data-Driven Frame-Semantic Parsing with Tree Wrapping Grammar" we combined the developments presented in previous papers comprising this dissertation and discuss the syntax-aware semantic parser based on TWGs extracted from the RRGparbank. This is the first large-scale implementa-

tion of the syntax-semantics interface in RRG presented for TAG Kallmeyer and Osswald (2014) and prototypically implemented in Arps and Petitjean (2018). The semantic representations, i.e. frames, are assigned to elementary trees, as represented in Figure 6.5. We followed Kallmeyer and Osswald (2014) to represent frames as base-labeled feature structures. During parsing, the combination operations for elementary trees triggers also the semantic composition of the frames mapped to them. The semantic representation of the complete utterance after the combinations is shown in Figure 6.6.

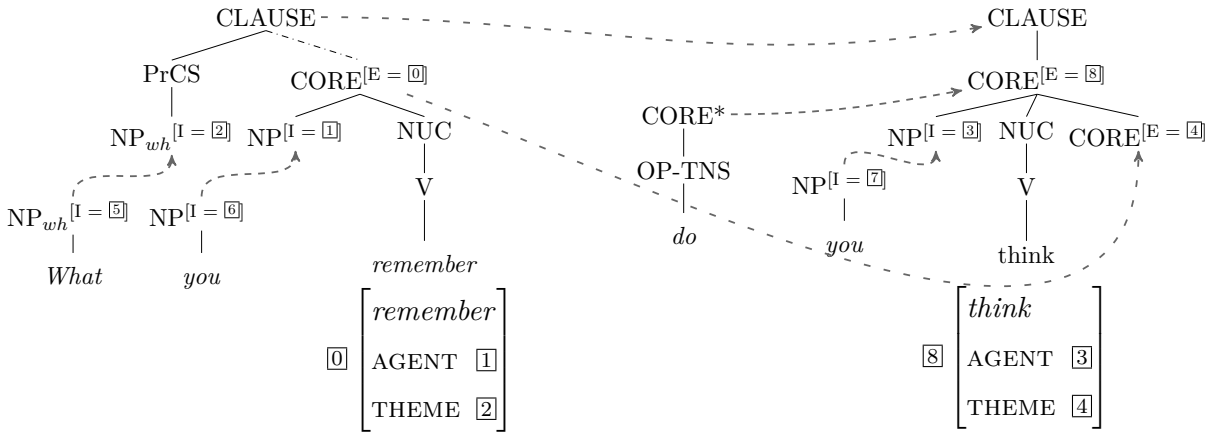


Figure 6.5: Wrapping substitution for wh-LDD in the sentence *What do you think you remember?*

Table 6.1 shows an example of the input data to train our semantic parser.

	Supertag	**Frame**	**Arg. Link.**
What	(NP$_{wh}$ (PRO ⋄))	(entity)	(–)
do	(CORE* (OP-TNS ⋄))	(–)	(–)
you	(NP (PRO ⋄))	(–)	(–)
think	(CL (CORE (NP) (NUC (V ⋄)) (CORE)))	(think)	((1, 'Agent'), (2, 'Theme'))
you	(NP (PRO ⋄))	(–)	(–)
remember	(CL (PrCS (NP$_{wh}$))(CORE (NP) (NUC (V ⋄))))	(remember)	((1, 'Agent'), (2, 'Theme'))

Table 6.1: Example of the training data, CL stands for Clause, CO means Core.

$$\boxed{0}\begin{bmatrix} \textit{think} & \\ \text{AGENT} & \textit{you} \\ \text{THEME} & \begin{bmatrix} \textit{remember} & \\ \text{AGENT} & \textit{you} \\ \text{THEME} & \textit{what} \end{bmatrix} \end{bmatrix}$$

Figure 6.6: Semantic representation for an LDD construction in *What do you think you remember*

We showed in this last paper **(8)** "Data-Driven Frame-Semantic Parsing with Tree Wrapping Grammar" that our syntax-aware approach leads to an almost state-of-the-art result in dependency semantic parsing (i.e. semantic parsing approach which predicts only heads of semantic roles) and outperforms the syntax-agnostic parser when it comes to predict the full spans of semantic roles in a frame-based semantic parser. The results for the full spans are better with our parser (compared to the baseline systems) since it appears to be easier to reconstruct the full spans from the constituent heads using predicted syntactic tree than to predict the full spans. Due to its being the first data-driven implementation of the syntax-semantics interface in RRG, we were confronted with several linguistic phenomena in English data not yet covered in the research on TRG-based semantics, such as control and raising constructions, peripheral and non-peripheral subordinate clauses, prepositional phrases and non-local dependencies. In this paper we described our implementation decisions for these phenomena.

Our answer to the last research question (iv) is thus that we could implement a large-scale semantic parser based on RRG and its formalization TWG, which reaches a state-of-the-art performance in predicting full spans of semantic roles in frames and performs also sufficiently well in predicting heads of semantic roles.

Future work In our future work, we plan to pursue several directions:

- We explored PropBank, VerbNet, FrameNet and VerbAtlas as possible sources for meaning representation inventory in our work. We decided to choose VerbAtlas since it has the largest coverage of verb senses among the four resources and also a relatively small (but also not too general) set of frames and semantic roles, so this inventory can be better learned with machine learning algorithms. FrameNet for example, has a large number of semantic roles, which are frequently frame-specific and thus hard to learn, whilst the semantic roles in PropBank are too general and hard to interpret outside of the context. The only disadvantage of VerbAtlas is the lack of the FrameNet-like relations and the hierarchy of frames, which would enable reasoning and logical inference. However, since VerbAtlas provides hand-crafted mappings to the mentioned verbal and frame resources, we would like to explore the possibility of creating a relation map for the frame-sense pairs in VerbAtlas through rule-based mappings to FrameNet frames.

- In this work we only explored verbal predicates as frame-evoking elements for our semantic parser. In future work we plan to extend our approach to nominal, adjectival, and prepositional predicates.

- We would like to investigate the challenges arising from our implementation

for other languages, starting from the multilingual subcorpora in RRGparbank (i.e. German, French, and Russian subcorpora).

- We started our dissertation project with the aim of facilitating creation of NLP tools for languages from different language families, including low-resource ones. Evang et al. (2022) described a methodology for cross-lingual syntactic parsing by using the English glosses and aligning them with the words in original language. In our future work we want to extend our semantic parser according to their methodology and to investigate how well our semantic parsing system can be adapted for use with low-resource languages. In particular, we plan to to start from extending our semantic parser for the Daakaka language, an RRG-based treebank for which is currently being developed.

- The Fillmore-style semantic frames we use in our parsing implementation have shown to adequately capture lexical meaning (Fillmore, 1982; Löbner, 2014), but they cannot be easily extended to integrate logical operators to enable computational reasoning. Based on previous work (Kallmeyer and Richter, 2014; Kallmeyer et al., 2015), we plan to explore incorporating logical operators, e.g. quantifiers and negation, into our frame-semantic parsing tool.

- The syntax-semantics interface in the RRG theory includes the discourse-pragmatics component alongside syntax and semantics. We left out the discourse pragmatics in our implementation of the syntax-semantics interface in RRG, as this goes beyond the scope of this dissertation, but we want to explore it in future work.

References

Abeillé and Rambow, O. (2000). Tree adjoining grammars : formalisms, linguistic analysis, and processing.

Abeillé, A. (1991). *Une grammaire lexicalisée d'arbres adjoints pour le français: application à l'analyse automatique*. PhD thesis, Université Paris 7.

Abeillé, A. (2002). *Une Grammaire Électronique du Français*. CNRS Editions, Paris.

Abeillé, A., Barrier, N., Barrier, S., and Gerdes, K. (2002). La grammaire LTAG du Français (FTAG). Technical report, Technical Report, Université Paris 7/Denis Diderot.

Abeillé, A., Candito, M.-H., and Kinyon, A. (2000). The current status of FTAG. In *Proceedings of TAG+5*, pages 11–18, Paris.

Abzianidze, L., Bjerva, J., Evang, K., Haagsma, H., van Noord, R., Ludmann, P., Nguyen, D.-D., and Bos, J. (2017). The Parallel Meaning Bank: Towards a Multilingual Corpus of Translations Annotated with Compositional Meaning Representations. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 242–247, Valencia, Spain. Association for Computational Linguistics.

Arps, D. and Petitjean, S. (2018). A Parser for LTAG and Frame Semantics. In chair), N. C. C., Choukri, K., Cieri, C., Declerck, T., Goggi, S., Hasida, K., Isahara, H., Maegaard, B., Mariani, J., Mazo, H., Moreno, A., Odijk, J., Piperidis, S., and Tokunaga, T., editors, *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Babko-Malaya, O. (2004). LTAG Semantics of Focus. In *Proceedings of TAG+7*, pages 1–8, Vancouver.

Bangalore, S. and Joshi, A. K. (1999). Supertagging: An approach to almost parsing. *Computational linguistics*, 25(2):237–265.

Banik, E. (2004). Semantics of VP Coordination in LTAG. In *Proceedings of TAG+7*, pages 118–125, Vancouver.

Barsalou, L. W. (1992). Frames, Concepts, and Conceptual Fields. In Lehrer, A. and Kittay, E. F., editors, *Frames, Fields, and Contrasts*, New Essays in Semantic and Lexical Organization, chapter 1, pages 21–74. Lawrence Erlbaum Associates, Hillsdale, New Jersey.

Berg-Kirkpatrick, T. and Klein, D. (2010). Phylogenetic grammar induction. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 1288–1297.

Bladier, T., Kallmeyer, L., and Evang, K. (2023). Annotating Semantic Frames in RRGparbank. Heinrich Heine University Düsseldorf, Role and Reference Grammar RRG Conference 2023.

Boonkwan, P. (2014). Scalable semi-supervised grammar induction using cross-linguistically parameterized syntactic prototypes.

Bos, J. (2008). Wide-coverage semantic analysis with Boxer. In *Semantics in text processing. step 2008 conference proceedings*, pages 277–286.

Bos, J. (2015). Open-domain semantic parsing with boxer. In *Proceedings of the 20th nordic conference of computational linguistics (NODALIDA 2015)*, pages 301–304.

Brainerd, W. S. (1967). *Tree generating systems and tree automata*. Purdue University.

Candito, M., Crabbé, B., and Denis, P. (2010). Statistical French dependency parsing: Treebank conversion and first results. In *Seventh International Conference on Language Resources and Evaluation-LREC 2010*, pages 1840–1847. European Language Resources Association (ELRA).

Candito, M. and Seddah, D. (2012). Effectively long-distance dependencies in French: annotation and parsing evaluation.

Chen, J., Bangalore, S., and Vijay-Shanker, K. (2006). Automated extraction of Tree-Adjoining Grammars from treebanks. *Natural Language Engineering*, 12(3):251–299.

Chen, J. and Rambow, O. (2003). Use of deep linguistic features for the recognition and labeling of semantic arguments. In *Proceedings of the 2003 conference on Empirical methods in natural language processing*, pages 41–48.

Chiarcos, C. and Fäth, C. (2019). Graph-based annotation engineering: towards a gold corpus for Role and Reference Grammar. In *2nd Conference on Language, Data and Knowledge (LDK 2019)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik.

Clark, S. (2002). Supertagging for combinatory categorial grammar. In *Proceedings of the Sixth International Workshop on Tree Adjoining Grammar and Related Frameworks (TAG+ 6)*, pages 19–24.

Coavoux, M. and Cohen, S. B. (2019). Discontinuous constituency parsing with a stack-free transition system and a dynamic oracle. *arXiv preprint arXiv:1904.00615*.

Coavoux, M. and Crabbé, B. (2017). Multilingual Lexicalized Constituency Parsing with Word-Level Auxiliary Tasks. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 331–336, Valencia, Spain. Association for Computational Linguistics.

Cohn, T., Blunsom, P., and Goldwater, S. (2010). Inducing tree-substitution grammars. *The Journal of Machine Learning Research*, 11:3053–3096.

Collins, M. and Duffy, N. (2002). New Ranking Algorithms for Parsing and Tagging: Kernels over Discrete Structures, and the Voted Perceptron. pages 263–270, Philadelphia, PA.

Conia, S., Orlando, R., Brignone, F., Cecconi, F., Navigli, R., et al. (2021). InVeRo-XL: Making cross-lingual Semantic Role Labeling accessible with intelligible verbs and roles. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 319–328. Association for Computational Linguistics.

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

Cortes-Rodriguez, F. (2016). Towards the computational implementation of Role and Reference Grammar: Rules for the syntactic parsing of RRG phrasal constituents. *Círculo de lingüística aplicada a la comunicación*, 65:75–108.

Curran, J. R., Clark, S., and Bos, J. (2007). Linguistically motivated large-scale NLP with C&C and Boxer. In *Proceedings of the 45th annual meeting of the Association for Computational Linguistics Companion volume proceedings of the demo and poster sessions*, pages 33–36.

Das, D., Chen, D., Martins, A. F. T., Schneider, N., and Smith, N. A. (2014). Frame-Semantic Parsing. *Computational Linguistics*, 40(1):9–56.

Di Fabio, A., Conia, S., and Navigli, R. (2019). VerbAtlas: a novel large-scale verbal semantic resource and its application to semantic role labeling. 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)*, pages 627–637.

Diedrichsen, E. (2014). A Role and Reference Grammar Parser for German. *B. Nolan and C. Periñán-Pascual (eds)*, pages 105–142.

Dik, S. C. (1987). Some principles of functional grammar. *Functionalism in linguistics*, 20:81.

Dowty, D. R., Wall, R. E., and Peters, S. (1981). *Introduction to Montague Semantics*. Studies in Linguistics and Philosophy. Kluwer, Dordrecht.

Evang, K. (2019a). Cross-lingual CCG Induction. In Burstein, J., Doran, C., and Solorio, T., editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1577–1587, Minneapolis, Minnesota. Association for Computational Linguistics.

Evang, K. (2019b). Transition-based DRS parsing using stack-LSTMs. In *Proceedings of the IWCS shared task on semantic parsing*.

Evang, K., Kallmeyer, L., Waszczuk, J., von Prince, K., Bladier, T., and Petitjean, S. (2022). Improving Low-resource RRG Parsing with Cross-lingual Self-training. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 4360–4371, Gyeongju, Republic of Korea. International Committee on Computational Linguistics.

Fellbaum, C. D. (2000). WordNet : an electronic lexical database. *Language*, 76:706.

Fillmore, C. J. (1982). Frame Semantics. In of Korea, T. L. S., editor, *Linguistics in the Morning Calm*, pages 111–137. Hanshin Publishing Co., Seoul.

Fillmore, C. J. et al. (1976). Frame semantics and the nature of language. In *Annals of the New York Academy of Sciences: Conference on the origin and development of language and speech*, volume 280, pages 20–32.

Fillmore, C. J., Johnson, C. R., and Pertuck, M. R. L. (2003). Background to FrameNet. *International Journal of Lexicography*, 3(16):235–250.

Flickinger, D., Kordoni, V., and Zhang, Y. (2012). DeepBank: A Dynamically Annotated Treebank of the Wall Street Journal. In *Proceedings of the 11th International Workshop on Treebanks and Linguistic Theories*, Lisbon, Portugal.

Frank, R. (1992). *Syntactic Locality and Tree Adjoining Grammar: Grammatical, Acquisition and Processing Perspectives*. PhD thesis, University of Pennsylvania.

Frank, R. (2002). *Phrase Structure Composition and Syntactic Dependencies*. MIT Press, Cambridge, Mass.

Gardent, C. (2008). Integrating a unification-based semantics in a large scale Lexicalised Tree Adjoining Grammar for French. In *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008)*, pages 249–256.

Gorn, S. (1965). Explicit definitions and linguistic dominoes. In *Systems and Computer Science, Proceedings of the Conference held at Univ. of Western Ontario*, pages 77–115.

Guest, E. (2009). Parsing Using the Role and Reference Grammar Paradigm.

Habash, N. and Rambow, O. (2004). Extracting a tree adjoining grammar from the Penn Arabic Treebank. In *Proceedings of Traitement Automatique du Langage Naturel (TALN-04)*, pages 277–284.

Hajic, J., Ciaramita, M., Johansson, R., Kawahara, D., Martí, M. A., Màrquez, L., Meyers, A., Nivre, J., Padó, S., Štěpánek, J., et al. (2009). The CoNLL-2009 shared task: Syntactic and semantic dependencies in multiple languages. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL 2009): Shared Task*, pages 1–18.

Halliday, M. A. K. (1961). Categories of the theory of grammar. *Word*, 17(2):241–292.

Headden III, W. P., Johnson, M., and McClosky, D. (2009). Improving unsupervised dependency parsing with richer contexts and smoothing. In *Proceedings of human language technologies: the 2009 annual conference of the North American chapter of the association for computational linguistics*, pages 101–109.

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

Hockenmaier, J. and Steedman, M. (2007). CCGbank: A Corpus of CCG Derivations and Dependency Structures Extracted from the Penn Treebank. *Computational Linguistics*, 33(3).

Howell, K. and Bender, E. M. (2022). Building analyses from syntactic inference in local languages: An HPSG grammar inference system. In Derczynski, L., editor, *Northern European Journal of Language Technology, Volume 8*, Copenhagen, Denmark. Northern European Association of Language Technology.

Jin, L., Doshi-Velez, F., Miller, T., Schuler, W., and Schwartz, L. (2018). Unsupervised grammar induction with depth-bounded PCFG. *Transactions of the Association for Computational Linguistics*, 6:211–224.

Johnson, M. (1987). *A New Approach to Clause Structure in Role and Reference Grammar*, pages 55–59. 2.

Joshi, A. K. (1985). Tree adjoining grammars: How much contextsensitivity is required to provide reasonable structural descriptions? In Dowty, D., Karttunen, L., and Zwicky, A., editors, *Natural Language Parsing*, pages 206–250. Cambridge University Press.

Joshi, A. K. (1987). An introduction to Tree Adjoining Grammars. In Manaster-Ramer, A., editor, *Mathematics of Language*, pages 87–114. John Benjamins, Amsterdam.

Joshi, A. K., Kallmeyer, L., and Romero, M. (2007). Flexible Composition in LTAG: Quantifier Scope and Inverse Linking. In Muskens, R. and Bunt, H., editors, *Computing Meaning Volume 3*, volume 83 of *Studies in Linguistics and Philosophy*, pages 233–256. Springer.

Joshi, A. K., Levy, L. S., and Takahashi, M. (1975). Tree Adjunct Grammars. *Journal of Computer and System Science*, 10:136–163.

Joshi, A. K. and Schabes, Y. (1997). Tree-adjoining grammars. In *Handbook of formal languages*, pages 69–123. Springer.

Joshi, A. K. and Srinivas, B. (1994). Disambiguation of super parts of speech (or supertags): Almost parsing. In *Proceedings of the 15th conference on Computational linguistics-Volume 1*, pages 154–160. Association for Computational Linguistics.

Kallmeyer, L. (2003). LTAG Semantics for Relative Clauses. In Bunt, H., van der Sluis, I., and Morante, R., editors, *Proceedings of the Fifth International Workshop on Computational Semantics IWCS-5*, pages 195–210, Tilburg.

Kallmeyer, L. (2005). Tree-local Multicomponent Tree Adjoining Grammars with Shared Nodes. *Computational Linguistics*, 31(2):187–225.

Kallmeyer, L. (2016). On the Mild Context-Sensitivity of k-Tree Wrapping Grammar. In Foret, A., Morrill, G., Muskens, R., Osswald, R., and Pogodalla, S.,

editors, *Formal Grammar: 20th and 21st International Conferences, FG 2015, Barcelona, Spain, August 2015, Revised Selected Papers. FG 2016, Bozen, Italy, August 2016, Proceedings*, number 9804 in Lecture Notes in Computer Science, pages 77–93, Berlin. Springer.

Kallmeyer, L. (2021). Extraposed relative clauses in Role and Reference Grammar. An analysis using Tree Wrapping Grammars. *Journal of Language Modelling*, 9(2):225–290.

Kallmeyer, L. and Osswald, R. (2012). A Frame-Based Semantics of the Dative Alternation in Lexicalized Tree Adjoining Grammars. Submitted to Empirical Issues in Syntax and Semantics 9.

Kallmeyer, L. and Osswald, R. (2013). Syntax-driven semantic frame composition in Lexicalized Tree Adjoining Grammars. *Journal of Language Modelling*, 1(2):267–330.

Kallmeyer, L. and Osswald, R. (2014). Syntax-driven semantic frame composition in Lexicalized Tree Adjoining Grammars. *Journal of Language Modelling*, 1(2):267–330.

Kallmeyer, L. and Osswald, R. (2018). Towards a formalization of Role and Reference Grammar. In Kailuweit, R., Künkel, L., and Staudinger, E., editors, *Proceedings of the 2013 Conference on Role and Reference Grammar*.

Kallmeyer, L. and Osswald, R. (2023). *Formalization of RRG Syntax*, page 737–784. Cambridge Handbooks in Language and Linguistics. Cambridge University Press.

Kallmeyer, L., Osswald, R., and Pogodalla, S. (2015). Progression and Iteration in Event Semantics – An LTAG Analysis Using Hybrid Logic and Frame Semantics. In *Colloque de Syntaxe et Sémantique à Paris (CSSP 2015)*.

Kallmeyer, L., Osswald, R., and Van Valin, Jr., R. D. (2013). Tree Wrapping for Role and Reference Grammar. In Morrill, G. and Nederhof, M.-J., editors, *Formal Grammar 2012/2013*, volume 8036 of *LNCS*, pages 175–190. Springer.

Kallmeyer, L. and Richter, F. (2014). Quantifiers in frame semantics. In *Formal Grammar: 19th International Conference, FG 2014, Tübingen, Germany, August 16-17, 2014. Proceedings 19*, pages 69–85. Springer.

Kallmeyer, L. and Romero, M. (2004). LTAG Semantics with Semantic Unification. In *Proceedings of TAG+7*, pages 155–162, Vancouver.

Kalyanpur, A., Biran, O., Breloff, T., Chu-Carroll, J., Diertani, A., Rambow, O., and Sammons, M. (2020). Open-domain frame semantic parsing using transformers. *arXiv preprint arXiv:2010.10998*.

Kamp, H. and Reyle, U. (1993). *From Discourse to Logic*. Studies in Linguistics and Philosophy. Kluwer, Dordrecht, Boston, London.

Kaplan, R. M. and Bresnan, J. (1982). Lexical-Functional Grammar: A Formal System for Grammatical Representations. In *The Mental Representation of Grammatical Relations*, pages 173–281. MIT Press.

Kasai, J., Frank, R., McCoy, T., Rambow, O., and Nasr, A. (2017). Tag parsing with neural networks and vector representations of supertags. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1713–1723.

Kasai, J., Frank, R., Xu, P., Merrill, W., and Rambow, O. (2018). End-to-end Graph-based TAG Parsing with Neural Networks. *CoRR*, abs/1804.06610.

Kasai, J., Friedman, D., Frank, R., Radev, D., and Rambow, O. (2019). Syntax-aware Neural Semantic Role Labeling with Supertags. *arXiv preprint arXiv:1903.05260*.

Kipper, K., Dang, H. T., Schuler, W., and Palmer, M. (2000). Building a class-based verb lexicon using TAGs. In *Proceedings of the Fifth International Workshop on Tree Adjoining Grammar and Related Frameworks (TAG+ 5)*, pages 147–154.

Kroch, A. (1989). Asymmetries in long-distance extraction in a Tree Adjoining Grammar. In Baltin and Kroch, editors, *Alternative Conceptions of Phrase Structure*. University of Chicago.

Lafferty, J., McCallum, A., and Pereira, F. C. (2001). Conditional random fields: Probabilistic models for segmenting and labeling sequence data.

Lafont, Y. (1989). Interaction nets. In *Proceedings of the 17th ACM SIGPLAN-SIGACT symposium on Principles of programming languages*, pages 95–108.

Lewis, M. and Steedman, M. (2013). Combined distributional and logical semantics. *Transactions of the Association for Computational Linguistics*, 1:179–192.

Lewis, M. and Steedman, M. (2014). A* CCG Parsing with a Supertag-factored Model. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 990–1000. Association for Computational Linguistics.

Lindemann, M., Groschwitz, J., and Koller, A. (2019). Compositional semantic parsing across graphbanks. *arXiv preprint arXiv:1906.11746*.

Liu, Y. (2009). *Semantic role labeling using lexicalized tree adjoining grammars*. PhD thesis, School of Computing Science-Simon Fraser University.

Liu, Y. and Sarkar, A. (2007). Experimental evaluation of LTAG-based features for semantic role labeling. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 590–599.

Liu, Y. and Sarkar, A. (2009). Exploration of the LTAG-spinal formalism and treebank for semantic role labeling. In *Proceedings of the 2009 Workshop on Grammar Engineering Across Frameworks*, pages 1–9. Association for Computational Linguistics.

Löbner, S. (2014). Evidence for Frames from Human Language. In Gamerschlag, T., Gerland, D., Osswald, R., and Petersen, W., editors, *Frames and Concept Types*, number 94 in Studies in Linguistics and Philosophy, pages 23–67. Springer, Dordrecht.

Long, C., Kallmeyer, L., and Osswald, R. (2022). A Frame-Based Model of Inherent Polysemy, Copredication and Argument Coercion. In Zock, M., Chersoni, E., Hsu, Y.-Y., and Santus, E., editors, *Proceedings of the Workshop on Cognitive Aspects of the Lexicon*, pages 58–67, Taipei, Taiwan. Association for Computational Linguistics.

Mairal-Usón, R. and Cortés-Rodríguez, F. (2017). Automatically representing TExt Meaning via an Interlingua-based System (ARTEMIS). A further step towards the computational representation of RRG. *Journal of Computer-Assisted Linguistic Research*, 1(1):61–87.

Marcus, M., Santorini, B., and Marcinkiewicz, M. (1993). Building a Large Annotated Corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330.

Minnema, G. and Nissim, M. (2021). Breeding Fillmore's chickens and hatching the eggs: Recombining frames and roles in frame-semantic parsing. In *Proceedings of the 14th International Conference on Computational Semantics (IWCS)*, pages 155–165.

Mor, B., Garhwal, S., and Kumar, A. (2021). Mathematical modeling of Indian Tala's Kaidas and Paltas using formal grammar. *Journal of Ambient Intelligence and Humanized Computing*, 12:7891–7902.

Muralidaran, V., Spasić, I., and Knight, D. (2021). A systematic review of unsupervised approaches to grammar induction. *Natural Language Engineering*, 27(6):647–689.

Nesson, R., Rush, A. M., and Shieber, S. M. (2006). Induction of Probabilistic Synchronous Tree-Insertion Grammars for Machine Translation. In *Conference of the Association for Machine Translation in the Americas*.

Nguyen, T. M. H., Romary, L., Roussanaly, A., et al. (2006). A Lexicalized Tree-Adjoining Grammar for Vietnamese. In *International Conference on Language Resources and Evaluation-LREC 2006*.

Nivre, J., De Marneffe, M.-C., Ginter, F., Goldberg, Y., Hajic, J., Manning, C. D., McDonald, R. T., Petrov, S., Pyysalo, S., Silveira, N., et al. (2016). Universal Dependencies v1: A Multilingual Treebank Collection. In *LREC*.

Nolan, B. (2004). First steps toward a computational RRG. *Linguistic theory and practice: description, implementation and processing*, page 196.

Oepen, S., Flickinger, D., Toutanova, K., and Manning, C. D. (2004). Lingo redwoods. *Research on Language and Computation*, 2(4):575–596.

Osswald, R. (2021). Activities, accomplishments and causation. *Challenges in the analysis of the syntax-semantics-pragmatics interface: A Role and Reference Grammar perspective*, pages 3–30.

Palmer, M., Gildea, D., and Kingsbury, P. (2005). The Proposition Bank: An Annotated Corpus of Semantic Roles. *Comput. Linguist.*, 31(1):71–106.

Petersen, W. (2007). Representation of Concepts as Frames. In *The Baltic International Yearbook of Cognition, Logic and Communication*, volume 2, pages 151–170.

Poelman, W., van Noord, R., and Bos, J. (2022). Transparent Semantic Parsing with Universal Dependencies using Graph Transformations. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 4186–4192.

Pollard, C. and Sag, I. A. (1994). *Head-Driven Phrase Structure Grammar*. Studies in Contemporary Linguistics. The University of Chicago Press, Chicago, London.

Rambow, O., Vijay-Shanker, K., and Weir, D. (2001). D-tree substitution grammars. *Computational Linguistics*, 27(1):87–121.

Ratnaparkhi, A. (1996). A maximum entropy model for part-of-speech tagging. In *Conference on empirical methods in natural language processing*.

Regnault, M. (2019). Adaptation d'une métagrammaire du français contemporain au français médiéval. In *TALN-RECITAL 2019-26e édition de la conférence TALN (Traitement Automatique des Langues Naturelles) et 21e édition de la conférence jeunes chercheur · euse · s RECITAL*.

Ringgaard, M., Gupta, R., and Pereira, F. C. (2017). SLING: A framework for frame semantic parsing. *arXiv preprint arXiv:1710.07032*.

Rogers, J. (1999). Generalized Tree-Adjoining Grammars. In *Sixth Meeting on Mathematics of Language*, pages 189–202, Orlando, Florida.

Romero, M. (2002). Quantification over situations variables in LTAG: some constraints. In *Proceedings of TAG+6*, Venice, Italy.

Romero, M. and Kallmeyer, L. (2005). Scope and Situation Binding in LTAG using Semantic Unification. In *Proceedings of IWCS-6*, Tilburg.

Romero, M., Kallmeyer, L., and Babko-Malaya, O. (2004). LTAG Semantics for Questions. In *Proceedings of TAG+7*, pages 186–193, Vancouver.

Sarkar, A. (1998). Separating dependency from constituency in a tree rewriting system. *arXiv preprint cs/9809028*.

Schabes, Y. and Joshi, A. K. (1988). An Earley-type Parsing Algorithm for Tree Adjoining Grammars. In *Proceedings of the 26th Annual Meeting of the Association for Computational Linguistics*, pages 258–269.

Schabes, Y. and Waters, R. C. (1995). Tree insertion grammar: a cubic-time, parsable formalism that lexicalizes context-free grammar without changing the trees produced. *Computational Linguistics*, 21(4).

Schuler, K. K. (2005). VerbNet: A broad-coverage, comprehensive verb lexicon.

Schuster, M. and Paliwal, K. K. (1997). Bidirectional recurrent neural networks. *IEEE transactions on Signal Processing*, 45(11):2673–2681.

Shieber, S. M. and Schabes, Y. (1990). Synchronous Tree-Adjoining Grammars. In *Proceedings of COLING*, pages 253–258.

Steedman, M. (2000). *The Syntactic Process*. MIT Press.

Sulger, S., Butt, M., King, T. H., Meurer, P., Laczkó, T., Rákosi, G., Dione, C. B., Dyvik, H., Rosén, V., De Smedt, K., et al. (2013). Pargrambank: The pargram parallel treebank. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 550–560.

Swanson, B., Yamangil, E., Charniak, E., and Shieber, S. (2013). A context free TAG variant. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 302–310.

Swayamdipta, S. (2017). *Learning Algorithms for Broad-Coverage Semantic Parsing*. PhD thesis, Ph. D. thesis, Carnegie Mellon University Pittsburgh, PA.

Uemura, Y., Hasegawa, A., Kobayashi, S., and Yokomori, T. (1999). Tree adjoining grammars for RNA structure prediction. *Theoretical computer science*, 210(2):277–303.

van Cranenburgh, A. and Bod, R. (2013). Discontinuous parsing with an efficient and accurate DOP model. In *Proceedings of the International Conference on Parsing Technologies (IWPT 2013)*. Citeseer.

van Cranenburgh, A. et al. (2016). Rich statistical parsing and literary language.

van Noord, R., Abzianidze, L., Toral, A., and Bos, J. (2018). Exploring neural methods for parsing discourse representation structures. *Transactions of the Association for Computational Linguistics*, 6:619–633.

van Noord, R., Toral, A., and Bos, J. (2019). Linguistic information in neural semantic parsing with multiple encoders. In *Proceedings of the 13th International Conference on Computational Semantics-Short Papers*, pages 24–31.

van Noord, R., Toral, A., and Bos, J. (2020). Character-level representations improve DRS-based semantic parsing even in the age of BERT. In Webber, B., Cohn, T., He, Y., and Liu, Y., editors, *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4587–4603, Online. Association for Computational Linguistics.

Van Valin, R. D. (2023). *Principles of Role and Reference Grammar*, page 17–178. Cambridge Handbooks in Language and Linguistics. Cambridge University Press.

Van Valin, Jr., R. D. (1993). Role and reference grammar. *Work Papers of the Summer Institute of Linguistics, University of North Dakota Session*, 37(1):5.

Van Valin, Jr., R. D. (2005). *Exploring the Syntax-Semantics Interface*. Cambridge University Press.

Van Valin, Jr., R. D. and Foley, W. A. (1980). Role and Reference Grammar. In Moravcsik, E. A. and Wirth, J. R., editors, *Current approaches to syntax*, volume 13 of *Syntax and semantics*, pages 329–352. Academic Press, New York.

Van Valin, Jr., R. D. and LaPolla, R. (1997). *Syntax: Structure, meaning and function*. Cambridge University Press.

Vaswani, A., Bisk, Y., Sagae, K., and Musa, R. (2016). Supertagging with LSTMs. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 232–237.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. (2017). Attention is all you need. *Advances in neural information processing systems*, 30.

Vendler, Z. (1967). *Linguistics and Philosophy*. Cornell University Press, Ithaca.

Vijay-Shanker, K. and Joshi, A. K. (1985). Some computational properties of Tree Adjoining Grammars. In *Proceedings of the 23rd Annual Meeting of the Association for Computational Linguistics*, pages 82–93.

Vijay-Shanker, K. and Joshi, A. K. (1988). Feature Structures Based Tree Adjoining Grammar. In *Proceedings of COLING*, pages 714–719, Budapest.

Vijay-Shanker, K. and Schabes, Y. (1992). Structure Sharing in Lexicalized Tree Adjoining Grammars. In *Proceedings of COLING*, Nantes.

Waszczuk, J. (2017). *Leveraging MWEs in practical TAG parsing: towards the best of the two world*. PhD thesis.

Waszczuk, J., Savary, A., and Parmentier, Y. (2016). Promoting multiword expressions in A* TAG parsing. In *26th International Conference on Computational Linguistics (COLING 2016)*.

Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M., et al. (2020). Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 conference on empirical methods in natural language processing: system demonstrations*, pages 38–45.

Xia, F. (1999). Extracting tree adjoining grammars from bracketed corpora. In *Proceedings of the 5th Natural Language Processing Pacific Rim Symposium (NLPRS-99)*, pages 398–403.

Xia, F. (2001a). *Automatic grammar generation from two different perspectives*. University of pennsylvania.

Xia, F. (2001b). *Investigating the Relationship between Grammars and Treebanks for natural languages*. Doctoral dissertation, University of Pennsylvania, Philadelphia, PA.

Xia, F., Palmer, M., and Joshi, A. (2000). A uniform method of grammar extraction and its applications. In *Proceedings of the 2000 Joint SIGDAT conference on Empirical methods in natural language processing and very large corpora: held in conjunction with the 38th Annual Meeting of the Association for Computational Linguistics-Volume 13*, pages 53–62. Association for Computational Linguistics.

Xia, Q., Li, Z., Zhang, M., Zhang, M., Fu, G., Wang, R., and Si, L. (2019). Syntax-aware neural semantic role labeling. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 7305–7313.

XTAG Research Group (2001). A Lexicalized Tree Adjoining Grammar for English. Technical report, Institute for Research in Cognitive Science, Philadelphia. Available from `ftp://ftp.cis.upenn.edu/pub/xtag/release-2.24.2001/tech-report.pdf`.

Xu, W., Auli, M., and Clark, S. (2015). CCG supertagging with a recurrent neural network. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 250–255.

Yoshikawa, M., Noji, H., and Matsumoto, Y. (2017). A* CCG Parsing with a Supertag and Dependency Factored Model. *CoRR*, abs/1704.06936.

Zettlemoyer, L. and Collins, M. (2007). Online learning of relaxed ccg grammars for parsing to logical form. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 678–687.

Zhang, Y.-z., Matsuzaki, T., and Tsujii, J. (2009). Hpsg supertagging: A sequence labeling view. In *Proceedings of the 11th International Conference on Parsing Technologies (IWPT'09)*, pages 210–213.

Statement of Own Contribution

This chapter declares my contributions to each of the presented papers.

(1) **Bladier, T.**, Kallmeyer, L., Osswald, R., & Waszczuk, J. (2020). Automatic extraction of tree-wrapping grammars for multiple languages. In *Proceedings of the 19th International Workshop on Treebanks and Linguistic Theories* (pp. 55-61).

I was responsible for the experimental design and the research idea in **(1)**. I proposed and implemented the grammar extraction algorithm, collected and prepared experimental data. Jakub Waszczuk and I conducted the parsing experiments together, while I was responsible for the experiment evaluation and interpretation of the results. During the process of grammar extraction and adjustment of the extraction algorithm, I was supervised by Laura Kallmeyer and Rainer Osswald. They also conducted manual validation of the induced grammars.

(2) **Bladier, T.**, van Cranenburgh, A., Samih, Y., & Kallmeyer, L. (2018). German and French neural supertagging experiments for LTAG parsing. In *Proceedings of ACL 2018, Student Research Workshop* (pp. 59-66).

I was primarily responsible for the experimental design and the research idea in **(2)**. The concept and design of the RNN-based supertagger were proposed by Younes Samih. Younes Samih supervised me during the implementation of the supertagger. I collected and prepared experimental data, conducted parsing experiments, and evaluated and interpreted the results. The other authors contributed to this paper primarily in an advisory role.

(3) **Bladier, T.**, Waszczuk, J., Kallmeyer, L., & Janke, J. H. (2019). From partial neural graph-based LTAG parsing towards full parsing. *Computational Linguistics in the Netherlands Journal*, 9, 3-26.

The research idea in **(3)** was a joint effort between me, Jakub Waszczuk, and Laura Kallmeyer. The idea to adapt an existing probabilistic A* parser for LTAGs and to use it with n-best extracted supertags in order to enhance parsing performance, was proposed by me and Jakub Waszczuk. The adaptation of the A* parser was

carried out by Jakub Waszczuk, while I was responsible for the supertagging component of the proposed parsing pipeline. I conducted collection and pre-processing of data and was responsible for the experimental design. Jörg Hendrik Janke, who pursued his Bachelor thesis project in this context, conducted the experiments with my assistance, while I was responsible for the evaluation and interpretation of the results.

(4) **Bladier, T.**, Waszczuk, J., & Kallmeyer, L. (2020). Statistical parsing of Tree Wrapping Grammars. In *Proceedings of the 28th International Conference on Computational Linguistics* (pp. 6759-6766).

The design of the study in **(4)** were the joint task of all contributors. The adaptation of the symbolic A*-based LTAG parser by adding new rules for processing Tree Wrapping Grammars was conducted by Jakub Waszczuk and Laura Kallmeyer. I was responsible for collection and preparation of the data, conduction of experiments and interpretation of the results with other contributors having an advisory role.

(5) **Bladier, T.**, van Cranenburgh, A., Evang, K., Kallmeyer, L., Möllemann, R., & Osswald, R. (2018). RRGbank: a role and reference grammar corpus of syntactic structures extracted from the penn treebank. In *17th International Workshop on Treebanks and Linguistic Theories (TLT)*.

The design of the study in **(5)** were the joint task of all contributors. I, Kilian Evang and Andreas van Cranenburgh were responsible for the development, implementation, automatic evaluation, and maintenance of the conversion algorithm from constituency trees in the Penn Treebank to the trees in RRGbank, while other contributors carried out the manual validation of the algorithm output. I was also responsible for the development of the first RRG/TWG-based syntactic parser as soon as there was enough RRG-annotated data available, so that the research team had a better starting point for further annotations. The author was supervised by Laura Kallmeyer, Andreas van Cranenburgh, and Kilian Evang during the implementation of the parser.

(6) **Bladier, T.**, Evang, K., Generalova, V., Ghane, Z., Kallmeyer, L., Möllemann, R., Moors, N., Osswald, R., & Petitjean, S. (2022). RRGparbank: A parallel role and reference grammar treebank. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference* (pp. 4833-4841).

The design of the study in **(6)** was the joint task of all contributors. I was primarily responsible for the implementation and maintenance of certain functionalities of the RRGparbank website, such as data search and download options. I was also responsible for Section 5 *Applications* of the paper, i.e. I developed a multilingual

syntactic parser based on the data from RRGparbank. For this, I adapted the syntactic parser described in the above-mentioned papers **(4)** and **(5)** to parse multiple languages, thereby providing the baseline annotations for the multilingual corpus. Furthermore, I was responsible for collecting and preparing data, conducting parsing experiments, and interpreting the results. Additionally, I assisted Kilian Evang and Simon Petitjean in the development of the conversion algorithm from UDs to RRG annotations.

(7) **Bladier, T.**, Minnema, G., van Noord, R., & Evang, K. (2021). Improving DRS Parsing with Separately Predicted Semantic Roles. In *Proceedings of the ESSLLI 2021 Workshop on Computing Semantics with Types, Frames and Related Structures* (pp. 25-34).

I was responsible for the idea of the study and the conception of the paper **(7)**. I was also responsible for the development of the syntax-agnostic algorithm of the conversion of the DRSs to SRL. I was also responsible for the preparation of experimental data, extraction of supertags, and interpretation of the experimental results.

(8) **Bladier, T.**, Kallmeyer, L., & Evang, K. (2023). Data-Driven Frame-Semantic Parsing with Tree Wrapping Grammar. In *Proceedings of the 15th International Conference on Computational Semantics (IWCS)*.

I was responsible for the research design and the conception of the paper **(8)**. I was also responsible for planning, preparation and conduction of the semantic parsing experiments and interpretation of the results. The other authors contributed to this paper primarily in an advisory role.

Eidesstattliche Versicherung

Ich versichere an Eides statt, dass die Dissertation von mir selbständig und ohne unzulässige fremde Hilfe unter Beachtung der 'Ordnung über die Grundsätze zur Sicherung guter wissenschaftlicher Praxis an der Heinrich-Heine-Universität Düsseldorf' erstellt worden ist.

Tatiana Bladier

Rognac, France, 26.04.2024