DESIGNING SYNTACTIC REPRESENTATIONS FOR NLP: AN EMPIRICAL INVESTIGATION

A DISSERTATION
SUBMITTED TO THE DEPARTMENT OF LINGUISTICS
AND THE COMMITTEE ON GRADUATE STUDIES
OF STANFORD UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

Natalia G. Silveira
August 2016

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

Christopher Manning, Primary Adviser

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

Dan Jurafsky

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

Christopher Potts

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

Marie-Catherine de Marneffe

Approved for the Stanford University Committee on Graduate Studies.

Patricia J. Gumport, Vice Provost for Graduate Education

This signature page was generated electronically upon submission of this dissertation in electronic format. An original signed hard copy of the signature page is on file in University Archives.

Abstract

This dissertation is a study on the use of linguistic structure in Natural Language Processing (NLP) applications. Specifically, it investigates how different ways of packaging syntactic information have consequences for goals such as representing linguistic properties, training statistical parsers, and sourcing features for information extraction. The focus of these investigations is the design of Universal Dependencies (UD), a multilingual syntactic representation for NLP.

Chapter 2 discusses the theoretical foundations of UD and its relations to other frameworks for the study of syntax. This discussion shows specific design decisions that characterize UD, and the principles motivating those decisions. The rationale for headedness criteria and type distinctions in UD is introduced there.

Chapter 3 studies how choices of headedness in dependency representations have consequences for parsing and crosslinguistic parallelism. UD strongly prefers lexical heads in dependency trees, and this chapter presents quantitative results supporting this preference for its impact on parallelism. However, that design can be suboptimal for parsing, and in some languages parsing accuracy can be improved by using a parser-internal representation that favors function words as heads.

Chapter 4 presents the first detailed linguistic analysis of UD-represented data, taking four Romance languages for a case study. UD's conciseness and orientation to surface syntax allows for a simple and straightforward analysis of Romance SE constructions, which are very difficult to unify in generative syntax. On the other hand, complex predicates require us to choose between representing syntactic or semantic properties. The Romance case also shows why maximizing the crosslinguistic uniformity of the distinction between function and content words requires a small amount

of semantic information, in addition to syntactic cues.

Chapter 5 investigates the actual usage of UD in a pipeline, with an extrinsic evaluation that compares UD to minimally transformed versions of it. The main takeaway is methodological: it is very difficult to obtain consistent improvements across data sets by manipulating the dependency representation. The most consistent result obtained was an improvement in performance when using a version of UD that is restructured and relabeled to have shorter predicate-argument paths.

The results and analyses presented in this work show that the main (and perhaps only) reason to use a lexical-head design is to support crosslinguistic parallelism. However, that is only possible if function words are defined uniformly across languages, and doing so satisfactorily requires the use of criteria outside syntax.

Moreover, the complexity of the results shows that a single design cannot necessarily serve every purpose equally well. Knowing this, one of the most useful things that designers can do is provide a discussion of the properties of their representation for users, empowering them to make transformations such as the many examples illustrated in this dissertation. A deep understanding of syntactic representations creates flexibility for users exploit their properties in the way that is most suitable for a particular task and data set. This dissertation creates such a deep understanding about UD, thereby, hopefully, enabling users to utilize it in the way that is most suitable for them.

Acknowledgements

Studying and working at Stanford was never easy, but it was always joyful. It's been amazing to be part of this university and this community, and I've been so unbelievably lucky to have the chance to come here and learn as much as I did. I hope the reader will forgive my overuse of superlatives—they are my attempt to do justice to an experience that was itself superlative.

I came to Stanford to develop research in NLP, and had the opportunity to do that under the supervision of Chris Manning. Working with Chris has been a privilege. It goes without saying that he is incredibly knowledgeable and intimidatingly smart; less obvious to the outside world is that he is also a truly kind and unfailingly patient advisor, who has always impressed me with his willingness to listen and to change his mind. I want to thank him deeply for everything he has taught me, and for always looking out for me.

It was thanks to Chris that I got involved in the project that led to the present work. This dissertation would not have been possible without the ongoing collaboration that led to the inception and development of Universal Dependencies. I'm very grateful to be able to exchange ideas with the entire UD core team—Marie-Catherine de Marneffe, Filip Ginter, Yoav Goldberg, Jan Hajic, Chris Manning, Ryan McDonald, Slav Petrov, Sampo Pyysalo, Reut Tsarfaty, Dan Zeman—and I'm especially thankful to our "chief cat herder" Joakim Nivre, without whose enthusiasm UD would never have reached the scale that it has, in so little time. I've also been lucky to collaborate with some very smart linguists on the construction of the EWT—Miriam Connor, Samuel Bowman, Hanzhi Zhu, Daniel Galbraith, Timothy Dozat and once again Chris Manning and Marie-Catherine de Marneffe—as well as John Bauer and

Sebastian Schuster, who worked alongside us improving the Stanford converter as we migrated from Stanford Dependencies to Universal Dependencies. I'm especially indebted to Tim and Sebastian, who helped me in so many ways as we put annotations and papers together, and later as I worked on this dissertation.

In the later stages of this project, as I put the dissertation itself together, I had the feedback of the all-star team that was my reading committee—Chris Potts, Dan Jurafsky and Marie-Catherine de Marneffe—and I'm very grateful for their careful attention and thoughtful feedback. All three were role models long before they were on my committee, and having them together at my defense was a delight.

I've always told my non-computationally inclined friends at Stanford Linguistics that it was a shame they didn't do NLP, because they didn't get to work with the amazing people I worked with. But the truth is that these friends could say the same of me, because everyone I have worked with at Stanford Linguistics has impressed me deeply. A number of wonderful people at Stanford who have contributed to my work in less direct but no less important ways. I am indebted to Beth Levin, who is in many ways the heart of our department. Her resourcefulness and attention to detail—not to mention, sincere and compassionate concern for all her students—was essential in getting me through my first years at Stanford. I also owe a heartfelt thanks to John Rickford for pulling me into his office one evening, during my hardest quarter of Stanford, to tell me that the faculty was rooting for me and wanted me to succeed. I needed to hear that, and I'm glad John made sure I did.

I'm grateful for the many dear friends I made at Stanford, who I hope will always stay in my life. I'd be remiss if I didn't record a special 'thank you' for those friends who literally showed up in the middle of the night to help with various crises—Naira Khan, Mason Chua, Masoud Jasbi. Jody Greenberg has made my life so much better in so many ways that it's hard to imagine doing this without him. (And he made me a chart!) I can't thank my sister Marina enough for all the ways in which she helped me. My debt to my parents, Inês and Pedro, who raised me in a home filled with books, is one I can never repay.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

This dissertation is a study on the use of linguistic structure in Natural Language Processing (NLP) applications. Specifically, it investigates how different properties of that structure can be emphasized or represented differently to be more suitable for particular applications. It shows how different ways of packaging syntactic information have consequences for goals such as representing linguistic properties, training statistical parsers, and sourcing features for information extraction.

NLP systems are ultimately built to serve the complex purposes of human users, performing such tasks as summarization, translation, or search. In order to perform these end-goal tasks, systems leverage different layers of linguistic information. The path from raw language data to the final output of the system is often divided into sequential steps, which begin with layers of structured annotation being added to the raw input in order to provide scaffolding for processing that input. These annotation layers are usually called the **pre-processing pipeline**, and the structured annotation that they produce can take multiple forms; some of the most common are adding **part-of-speech (POS)** tags to words and creating a representation of the **syntactic structure** of sentences.

From the point of view of most NLP research, the output produced by a pre-processing pipeline is largely commodified. Research focused on the end-goal or **downstream** applications takes these steps for granted, reflecting a tacit understanding of linguistic structure as mostly uncontroversial, and of widely used pre-processing

tools as sufficient for capturing it. Research focused on the pipelines themselves, on the other hand, is sometimes concerned with the nature of the annotations, but more often zooms in on the algorithms used to produce those annotations from examples. Overall, comparatively less attention is given to questions about the desirable properties of an annotation standard, and about specific applications of that standard. Outside the small community that is directly concerned with developing linguistic annotations, there seems to be a general belief that they consist of "a solved problem," ready to be consumed.

This belief notwithstanding, a close look at the history and use of linguistic annotations clearly shows that they were never a pure commodity: some can be better than others, given particular purposes. Take the case of syntactic representations, which are the focus of this dissertation. Historically, one clear sign of this has been a gradual (and certainly not complete) shift in NLP from constituency representations to **dependency representations**. While constituency representations of syntax have a long tradition in linguistics, and benefited from the Penn Treebank (PTB, Marcus et al. 1993) boom in the early 1990s, dependency representations have gained a lot of ground in the last few years; they are taught alongside constituency representations in many introductory classes and textbooks, and are used in pre-processing pipelines for all manner of NLP tasks. The enthusiasm for dependencies is attributed to various reasons: the possibility of faster parsing, the usability of dependency annotations compared to constituency trees, and to the close parallelism between dependency relations and the predicate-argument relations that are often the ultimate target of many NLP systems.

More recently, another high-level shift has started to take place within dependency syntax. It has almost always been the case in NLP that syntactic representations (either constituency- or dependency-based) are specified and then used for a single language; the typical use case for these representations is a monolingual NLP task applied to one language only. However, there has been growing interest in supporting multilingual and crosslingual work (Petrov et al., 2012; Zeman et al., 2012; McDonald et al., 2013; Tsarfaty, 2013; Rosa et al., 2014; de Marneffe et al., 2014), for multiple reasons. One is a practical engineering need for NLP pipelines that process many

languages: it is cost-efficient in multiple ways to enable easy adaptation and as much reuse as possible across languages. Another related motivation comes from the parsing community, where research towards a universal parser and towards crosslinguistic parser reuse depends crucially on multilingual representation. Moreover, in intrinsically crosslingual applications such as machine translation, there may be gains from bringing out parallel structure. Finally, there is a scientific motivation in the pursuit of universal grammar, which has deep theoretical implications beyond NLP.

In any situation where one of these motivations is relevant, language-specific annotation standards are not ideal. Standards developed for English, such as Stanford Dependencies (de Marneffe et al., 2006) for syntax, or the PTB POS tagset, are imprinted with language-specific quirks. For example, Stanford Dependencies (SD, de Marneffe et al. 2006; de Marneffe and Manning 2008) has a relation called `prt` for English particle verbs, illustrated in (1); but this relation is completely irrelevant for, say, Spanish.

(1) I picked up the book. (prt: picked → up)

Language-specific development also leads representations to deemphasize crosslinguistic similarities. Take the English example in (2a). In the SD analysis depicted, the words *searched* and *clue* are not connected by any one dependency edge. However, when we consider the Finnish translation in (2b), it becomes clear that *johtolangatta* and *without a clue* have the same function with respect to their governing predicates, and the fact that this function is encoded with different morphosyntactic strategies in each language can be factored out to reveal a deeper commonality. In order to bring out this parallel, the English phrase can be represented with an edge between *search* and *clue*; nevertheless, that is not the most obvious or most natural way to annotate this when English is considered by itself, as reflected in the SD annotation of this example, depicted below.

(2) a. to search without a clue (prep: search → without; pobj: without → clue)

b. etsiä johtolangatta
search.INF clue.ABSESS
to search without a clue

Language-specific annotations can be harmonized to a common denominator, but that process is not always straight-forward. As would be the case in the example above, knowledge of the languages may be required to understand how the harmonization can work. Furthermore, two representations may have inherently different expressiveness,[1] and in some cases it is impossible to convert from one to the other without some degree of human interpretation.

For multilingual work, it is better to use a standard specifically designed to capture phenomena across languages. Ideally, that should be done at a level of abstraction that highlights their commonalities, while still characterizing dependency types with nontrivial syntactic properties that make them informative for applications. Such a standard would factor out language-specific quirks such as the `prt` relation by identifying them as instances of more general phenomena, and highlight the parallelism between *clue* and *johtolangatta* by designating both as nominal modifiers, despite the difference between coding the modifier's role with morphosyntactic case or with a prepositional head.

That is what Universal Dependencies (UD, Nivre et al. 2016) proposes to do. UD is a new multilingual annotation standard for morphosyntax in NLP; it merges and subsumes a number of previous efforts in the areas of POS and morphological tagsets and dependency syntax (Zeman, 2008; Petrov et al., 2012; de Marneffe et al., 2014).

UD can be understood as a standard comprising three concrete products: a set of tags for three levels of morphosyntactic representation (one of which are syntactic dependencies); a set of guidelines for using those tags; and a multilingual set of treebanks that implement those guidelines (with some variance in the degree of compliance).[2] The standard is supposed to be applicable to any language in the world;

[1] An example of this, involving two different representations of copulas, is discussed in Chapter 3; see Section 3.3.3.

[2] The UD tags and guidelines have been created and maintained mostly by a small set of researchers, myself included, while the treebanks have been contributed by dozens of researchers who produced resources following the published guidelines.

spelled out, this amounts to the claim that for every word in every sentence of every language, there should be a very good candidate, by UD guidelines, to the position of that word's governor, forming a dependency relation;[3] and there should be a very good candidate in the UD type set for labeling that dependency relation.

This is not a modest goal, but the accelerated growth of the UD treebanks over the last two years, with expansion to 40 languages in v.1.2, suggests that it is within reach. UD presents a dependency type set representing broad categories of syntactic relations that reflect typologically widespread paradigmatic distinctions, such as the difference between modification, complementation and specification of grammatical categories, or between argumental and predicative heads. The dependency relations are underspecified with respect to many syntactic properties that are explicitly represented in other syntactic frameworks; but this underspecification allows for broad coverage both within and across languages, by providing a degree of abstraction over differences between languages and flexibility for the annotation of innovative or marginal language uses in real-world data.

This toolkit allows linguists working on different languages to annotate a very wide range of data with a very limited set of building blocks, and to do so in a way that focuses on broad-stroke syntactic properties that allow important parallels to be drawn from language to language under translation. In the context of pre-processing pipelines, the main advantage of using UD is that the annotations for one language will be comparable to annotations of other languages, which greatly simplifies the work of defining or learning syntactic patterns on multilingual data; this is important because syntactic patterns are used in one form or another in tasks such as parsing, translation, typological studies, or various forms of information extraction.

As it stands now, UD has emerged from two gradual shifts in the needs and preferences of the NLP community with respect to syntactic representations: one towards dependency representations in general, another towards multilingual dependency representations. On both counts, the shift happens from one linguistically plausible standard to another, and the motivation lies in a higher-order requirement

[3] A technicality must be noted here: the claim that every word has a governor requires the assumption that the governor may be an artificial root token.

that goes beyond linguistic adequacy: in one case, a preference for dependencies over constituents, and in another, for language-neutral representations to language-specific ones. Once these higher-order requirements are clear, a linguistically suitable representation can be developed. These shifts illustrate two premises that I assume going forward: one, that syntactic representations for NLP should not be entirely commodified, since they clearly adapt to changing needs; and second, that linguistic arguments alone cannot be the only resource for designing such representations. Notably, however, both shifts have happened organically, without specific experimental evidence to push change in a particular direction.

This dissertation proposes an approach to the problem of adapting syntactic representations to higher-order requirements imposed by their usage: rather than happening by slow, intuition-based shifts, changes can be driven by careful empirical analysis of how a representation addresses specific needs. The recent inception of UD creates a good opportunity for this new approach, since there is already a significant amount of annotated data to analyze, and at the same time, revisions of the UD guidelines are expected soon, and there is still room for change.

Because UD is meant to be used in NLP pipelines, its design depends inevitably on multiple trade-offs that go beyond linguistic argumentation. As an NLP resource, UD is useful only to the extent that it can be parsed automatically with high accuracy, used consistently to annotate varied linguistic phenomena in a way that is common across languages, and exploited successfully as scaffolding for semantic applications. At the same time, some of the high-level choices regarding the design of UD are clearly the type of decisions that need to be made prior to structure-based argumentation, such as choices about headedness criteria and granularity of the type set.

Taking v.1 of UD as a starting point, this work investigates the consequences of UD's design by two methods, one qualitative, and another quantitative.

The qualitative method is to analyze the suitability of UD annotations, and the extent to which particular dependency type definitions are appropriate or sustainable in the face of varied linguistic phenomena. This is done by laying out the linguistic and metalinguistic principles that UD is committed to, and then dropping down to naturally occurring data to understand how those principles interact in practice

and what kinds of compromise need to be achieved. Providing such an analysis is a way of studying to what extent the specification of UD meets two important desiderata: that the proposed dependency types be suitable for their purpose of universal adequacy without becoming trivialized; and that the underlying principles proposed for annotation be tenable in practice, and do not conflict in fundamental ways.

The second, quantitative, method is to embed UD-annotated data in actual NLP applications, and compare the results to plausible alternative annotations that reflect roads not taken. Doing this in a generalizable way is very difficult because of the variety of contexts in which UD can be used; my attempt at completeness here includes more thorough experiments that I have found in the existing literature, but it should still be understood as a set of case studies. These case studies do, however, help us understand the complex picture of how simple modifications to UD annotation can impact the components of an NLP pipeline, and provide an opportunity to discuss how these modifications can be implemented and used in practice.

Chapter 2 of this dissertation introduces UD and contextualizes it in both linguistic theory and NLP. Historically, I briefly survey key developments in the Dependency Grammar tradition, and show how the fundamental premises of major theories of Dependency Grammar are echoed in UD. I also discuss the more recent history of dependencies in the NLP community and how it led to the development of UD. Theoretically, I lay down tenets that situate UD as a theory of syntax, drawing attention to the differences between its prototype-based, performance-oriented approach and the generative, competence-oriented approach of many other syntactic theories. From these high-level principles, I move on to the essential commitments of the UD type set, presenting a short introduction to the different dependency types and their defining properties.

Chapter 3 takes on one of the abstract design choices that fundamentally characterize UD: the primacy of lexical words (over function words) as heads of phrases. The focus of the comparison is parsing, a domain in which existing work suggests the functional-head design can yield better predictive accuracy. However, methodological objections to previous work, concerns about its applicability to UD, and the need for

a deeper discussion of the differences between choosing lexical or functional heads justify a more detailed discussion of this issue here, with UD in mind specifically.

To that end, I design possible parsing representations for UD that can be obtained automatically from UD-compliant annotation; I show that these representations can be useful for different parser architectures and data sets, with significant performance differences coming out in some conditions, and a clear winner as to the best parsing representation to use. The most striking result, however, comes from a comparison of different languages, where it becomes clear that the decision to use a parsing representation or not depends crucially on the language in question—an essential insight for a multilingual standard. Some of these results have previously appeared in Silveira and Manning (2015).

Still with respect to the question of headedness in UD, Chapter 3 tackles another related issue: the problem of crosslinguistic parallelism, which is the main motivation for the lexical-head approach taken in UD. The choice for lexical heads has often been justified on the basis of highlighting syntactic parallelism between languages that have different morphosyntax. Does this work in practice? A small-scale study with a parallel Spanish-English corpus reveals that there is in fact a large difference in the extent to which parallelism can be achieved with lexical heads and functional heads, presenting an empirical justification for the preference for lexical heads. Chapter 3 also discusses the differences in expressiveness of each design, which is crucial for a better understanding of how their respective advantages can be explored.

Chapter 4 focuses on the suitability of UD for representing linguistic phenomena in challenging constructions across multiple languages (other than English). The chapter presents a study of two complex syntactic phenomena in four Romance treebanks from UD. By extending its scope beyond English, this study diagnoses the actual applicability of the historically English-centric UD type set and guidelines to other languages; by examining multiple languages together, it reveals the particular difficulties of finding the right level of abstraction to allow the representation to highlight parallels between languages without drowning out important properties in any single one of them. Additionally, the chapter includes a discussion of the uses of two functional labels from the type set in the four treebanks, and of how usage decisions

interact with representational commitments for some types of syntactic phenomena.

The phenomena examined in Chapter 4 offer a chance to think about two problems at the syntax-semantics interface: how dependency types are related to semantic roles; and to what extent dependencies can represent mismatches in the domain of complex predicates. Both of these challenges force us to think carefully about what guarantees UD can make about the properties of different dependency types in crosslinguistic analysis, with a focus on the dependency types that represent arguments of predicates and functional verbal categories. The analysis shows that, even in a limited set of languages, there are important conflicts between the goals of representing surface structure and preserving crosslinguistic parallelism, and that a strict interpretation of what UD should represent leads to unsatisfying results.

Finally, Chapter 5 moves downstream and on to the perspective of a client application. One of the important uses of UD is as a source of features for various information extraction tasks, many of which will have a focus on predicate-argument relations. The important question with respect to these downstream tasks is whether different designs for UD can provide better features. The aspects of design investigated concern mostly label granularity, but also headedness and enrichment strategies that have been used in the past (for SD) and the effectiveness of which has been questioned previously. The study makes rigorous use of experiments to understand the extent to which the effects observed are consistent. It shows that small changes to the dependency representation can be a source of large performance gains for downstream understanding tasks, but the results are highly variable and difficult to explain. Importantly, enrichment strategies that involve directly encoding implicit semantic relations in the dependency structure, by inferring them from the syntax, give consistently positive results, showing the value of this type of strategy.

Chapter 2

Universal Dependencies

2.1 Introduction

This chapter introduces the **Universal Dependencies** representation (UD, Nivre et al. 2016), a system of typed word-to-word syntactic relations that is meant to be universally applicable for syntactic annotation across languages. It lays the foundation for a discussion of the empirical aspects of using UD in Natural Language Processing (NLP) applications, which I develop in the next three chapters. Much of that discussion will unfold around questions of which design principles of UD can be adapted to better suit applications, and which cannot; in the current chapter, I start delineating some possible (and impossible) directions for those explorations.

As such, the chapter has three high-level goals. The first goal is to situate UD in the context of its theoretical background in linguistics and the NLP issues that its development addresses; this historical and comparative perspective will help constitute a space of possible representations, which the remaining chapters will explore. My second goal is to lay down high-level principles, in a first attempt to explictly articulate which aspects of linguistic structure UD proposes to represent; these principles will be taken as essentially nonnegotiable as I discuss the design of UD. The third goal is to provide a brief introduction to the UD type system, in order to enable the reader to understand the analyses and arguments that are used in the rest of the dissertation.

2.2 Dependency Syntax

Dependency Grammar is a syntax framework with a rich history that extends long before its use in NLP, dating back at least to the Arabic tradition of the 8th century, and in a broader sense to Panini (Kruijff, 2006). It encompasses a number of syntactic representations that have at their core asymmetric relations between words.

(3) The small dog chased the cat.

nominal subject, direct object, adjectival modifier, determiner

Example (3) illustrates some basic relations of the UD standard (using expanded edge labels for readability). This standard is a modern incarnation of Dependency Grammar that has been finding widespread use in NLP. It comprises a set of typed dependencies for syntactic annotation, along with a part-of-speech (POS) tagset and morphological feature set, and universal guidelines for application to any language. The standard has currently been applied to over 50 treebanks in 40 languages, in a project known as the **UD Treebank**.[1]

The representation of the structure in (3) is a **tree**, whose nodes are the words in the sentence. The edges in this tree are labeled and directed, and each edge represents an asymmetric syntactic relation known as a **dependency**. Each word has exactly one incoming edge, specified by a **type** or **label**: the edge `nominal subject` from *chased* to *dog* (which can be represented as `nominal subject`(*dog*, *chased*)), for example, tells us that *dog* is the subject of *chased*, while `adjectival modifier`(*dog*, *small*) indicates that *small* is an adjectival phrase modifying the noun *dog*. A single word has no incoming edges (*chased* in this example); that word is considered the **root** of the sentence.[2] It is important to note that the functions indicated by edge labels are performed not by single words, but by linguistic units headed by those words. The subject of this clause, for example, is not simply *dog*, but *the small dog*, which is the concatenation of all the words contained in the subtree that is rooted by *dog*.

[1] universaldependencies.org

[2] This word is sometimes said to be a child of an artificial root node, in order to simplify formal statements about dependency trees.

UD is only one of many current-day interpretations of dependency syntax. This section focuses on summarizing the common threads underlying some of these interpretations and the major distinctions that characterize them individually, as well as surveying their historical roots. The discussion of these differences will shed light on the aspects of dependency representation design that are investigated empirically in the remainder of this dissertation.

2.2.1 Defining dependencies

Dependency representations, while very diverse, have a common core that characterizes them in opposition to other ways of encoding syntactic information, such as constituency representations. As stated in (Kübler et al., 2009, p. 2), "The basic assumption underlying all varieties of dependency grammar is the idea that syntactic structure essentially consists of words linked by binary, asymmetrical relations called dependency relations (or dependencies for short)". These relations hold between a **head** and a **dependent**, both of which are lexical units.

The fundamental concern of dependency grammar is establishing criteria for determining which pairs of lexical units, in a sentence, stand in such a relation. Kübler et al. (2009) present a list of criteria that have been proposed in different frameworks for identifying the head within a construction:

1. The head determines the syntactic category of the construction, and can often replace it. When this criterion holds, the construction is called **endocentric**, as opposed to **exocentric** when it is violated.

2. The head determines the semantic category of the construction; the dependent gives semantic specification.

3. The head is obligatory; the dependent may be optional.

4. The head selects the dependent and determines whether it is obligatory or optional.

5. The form of the dependent depends on the head (a phenomenon known as agreement or government).

6. The linear position of the dependent is specified with reference to the head.

While all these criteria characterize heads under some view of syntactic dependencies, it becomes clear even in small amounts of data that not all heads can satisfy all these criteria in all constructions. The notion of head is better understood as a prototype, applicable when most of the typical characteristics are present. The heterogeneity and sometimes incompatibility of reasonable criteria forces any representation to eliminate or preempt items from the list above. Selecting or prioritizing headedness criteria differently leads to different dependency structures, depending mostly on whether semantic or syntactic criteria for headedness are prioritized.

For this reason, many constructions involving function words, such as prepositions and determiners, have different representations across frameworks. Another important source of differences is the analysis of coordination. In most languages, coordination is characterized by the equal status of conjuncts, which makes it difficult to represent in terms of asymmetric relations.

Dependency relations are often typed, as in (3); relation types are used to encode important distributional properties. Some of the conflict between criteria for establishing dependencies can be resolved in the type system, by assuming that specific dependency types are established according to specific criteria. This possibility is explored by Mel'čuk in his grammar, as shown below in Section 2.2.2, and I discuss its use in UD in Section 2.4.1.

Note on the term 'head' It is useful to make a clarification here on terminology. Other terms for head and dependent appear in the literature: 'regent', 'ruler', 'governor' are used for heads; 'subordinate' or 'modifier', for dependents. However, as noted in Mel'čuk (2009), the term 'head' is also popular in constituency grammar, to mean the head of a constituent, which introduces an ambiguity. Take, for example, sentence (3). The "head" of *the cat* could be taken to mean *cat*, which is the head of that constituent (and which Mel'čuk would call the internal head); or *chased*, which

is the node in the dependency tree that that constituent is attached to, by means of a dependency from *chased* to *cat* (or the external head).

Because of the widespread use of the term 'head' in NLP, I will adopt it here; however, when the term comes to introduce an ambiguity between external and internal heads, I will adopt the term 'governor' to refer to external ones.

2.2.2 Recent history of dependencies in linguistics

This core understanding of dependency relations (as laid out in Section 2.2.1) and the appreciation for their usefulness for representing linguistic structures has existed for a long time, as mentioned above. But much like what happened in constituency syntax, theories of dependency syntax flourished in the 20th century, with more formal approaches taking center stage. Our modern use of dependency grammar owes much to two modern linguists: Tesnière and Mel'čuk.

Tesnière

Our modern notion of dependency grammar is largely due to Tesnière's theory of syntax (Tesnière, 2015), named **dependency grammar** by his students. The author describes syntax as an autonomous level of linguistic description, governed by its own rules. Much as Chomsky, Tesnière acknowledges the possibility of absurd but grammatical sentences, and uses it to argue for a theory of syntax that does not make reference to semantics. In his view, nonetheless, syntax interfaces with semantics: there is never a syntactic relation where there is no semantic relation, which means that dependencies in his approach have a semantic flavor. In Tesnière's representation, the constituency structure of the sentence can be derived from the dependency structure: each subtree in the dependency tree forms a constituent. Word order, however, is not represented, and requires the specification of additional linearization rules. Tesnière describes producing a language as mapping the structural connections to a linear order, and understanding it as the reverse.

For Tesnière, the organization of words in a sentence—which transforms them from isolated ideas into articulated propositions—can be described in terms of dependency

relations between linguistic units. These units come in four categories: verbs, nouns, adjectives and adverbs. The categories inform a theory of constraints on dependency relations: verbs are ungoverned; nouns depend on verbs; adjectives depend on nouns; adverbs can depend on verbs, adjectives, or other adverbs.

Crucial for the expressivity of this model, and a distinctive feature of Tesnière's syntax, is the fact that the nodes in the dependency trees are not required to be single words. They can also be complex, or dissociated, ***nuclei***, formed from multiple words engaged in two other types of structure-building relations: **junctions** and **transfers**.

Junctions are nonhierarchical relations, characterized by their symmetry. They allow coordinated elements to stand as one *nucleus* in a dependency relation; for example, coordinated nouns as the subject of a verb, or a noun and its appositive. Elements standing in a junction relation always share a head; if they also share all their dependents, the junction is called total; otherwise, it is called partial.

Transfer is a relation that forms dissociated *nuclei* from a conveyor of lexical information and one or more conveyors of grammatical information. A dissociated *nucleus* formed via transfer has its category determined by the functional or grammatical elements in the transfer; that is how a prepositional phrase, for example, can form a *nucleus* of type adverb, or how a genitive phrase receives the type adjective, despite both having nominal heads: the distributional properties of the resulting *nucleus* emerge from the transfer. Tesnière acknowledges transfer with an analytical marker (a function word) or a synthetic marker (a bound grammatical morpheme). A verb group, for example, is generally a dissociated *nucleus* formed by association of a lexical verb with one or more auxiliaries. In the dependency tree, it acts as a single verbal *nucleus*. This device allows Tesnière to remain agnostic as to the heads of these constructions.

Tesnière distinguishes a particular type of dependency relation: the **valency** relation. This is essentially a distinction between complements and adjuncts—or, in Tesnière's parlance, **actants** and **circumstants**. The dividing line between actants and circumstants, the author admits, is difficult to establish precisely. The criteria are of form and meaning: actants are nouns (although he does admit that actants can occasionally take prepositions) whereas circumstants are adverbs; actants are

indispensable for understanding the verb, whereas circumstants are optional.

In more current dependency representations (UD included), these distinctions between dependencies are represented by means of rich type systems; however, Tesnière did not formalize such a system, and his style of dependency grammar is usually considered unlabeled.[3]

This approach to syntax is informed by a typological outlook on language. In addition to his theory of dependencies, Tesnière's conception of syntax includes the concept of **metataxis**, which consists of fundamental structural changes between translations of an expression in different languages. There are several classifications of metataxis offered by the author. For the most part, they concern argument realization, such as the contrast between English *I miss you* and French *Vous me manquez* 'You are missing to me', but also comprise head switching, such as between *I like to read* and German *Ich lese gern* 'I read with pleasure': in English, *read* depends on *like*, but in German, *gern* 'with pleasure' depends on *lese* 'read'.

Mel'čuk

Another influential school of dependency grammar, Meaning to Text Theory, has flourished from the work of Mel'čuk 1988. Mel'čuk first started developing his theory of dependencies while working on a machine translation system; in this light, his concerns are very similar to the concerns that led to the design of UD: a desire to represent syntax in a way that is crosslinguistically uniform, and useful for practical NLP applications.

One of the essential aspects of Mel'čuk's linguistic theory is the separation of language in **strata**. There are several layers of representation, which can be interpreted from a production perspective as going from meaning to text (hence the name Meaning to Text Theory). Mel'čuk proposes a semantic representation, a deep and a surface syntactic level, a deep and a surface morphological level, and a deep and a surface phonological level. There are also multiple devices for representation: for

[3]It is tempting to see 'transfer' and 'junction' as dependency types. The objection to that is that such relations are defined to be symmetric, while dependencies are defined as asymmetric. Note, however, that UD uses the type system to define some relations as essentially symmetric, as will be shown in Section 2.4.2.

example, the semantic level is represented by a network, but both syntactic levels are represented with dependency trees. I focus here on the two syntactic levels.

Mel'čuk observes that, while meaning is understandable and morphology is perceivable, syntax is neither. This explains why syntactic dependencies are, according to him, more subtle, harder to identify and even to justify theoretically. For this reason, Mel'čuk assigns such dependencies based on a series of nondefinitional diagnostic criteria, divided into three sets. Firstly, there are criteria for establishing when a syntactic dependency is present; next, there are criteria for identifying the head in the dependency; finally, there are criteria for labeling the specific type of dependency. This illustrates two fundamental differences between Tesnière's and Mel'čuk's approaches to syntactic dependencies: Mel'čuk's dependencies are typed; and he established a formal mechanism for characterizing dependencies.

In addition to the type system, he defined three major classes of syntactic dependencies: complementation, which is exocentric (because the head cannot stand alone in place of the phrase), and modification and coordination, which are both endocentric (since the head, in the case of modification, or either conjunct, in the case of conjunction, can syntactically stand in for the construction). There are parallels with Tesnière's distinctions: coordination is clearly related to junction; many types of complementation (as understood in modern views of syntax) are related to transfer.

In terms of establishing whether there is a dependency, Mel'čuk offers two criteria, which must be met simultaneously. The first one is that rules about the linear position of one word have to make reference to the other word in some way. An example of this would be how the position of an adverb, for example, can be described as grammatical or not according to whether it occurs before or after a verb, rather than before or after that verb's subject. (This echoes the list from Kübler et al. 2009.) The second criterion is that the two words must form a **phrase**. This criterion is better supported by well-established constituency diagnostics; we can say confidently that *a boy* is a phrase and *good boy* is a phrase, but *a good* is not a phrase.

Once it is established that a dependency exists between two words, the question arises of which word is the head. For determining that, Mel'čuk observes that the head determines the distributional properties of the entire subtree to a greater extent than

the dependent (Mel'čuk, 2009). This does not require exact distributional equivalence between the head and the head-dependent combination, since that would not apply to exocentric constructions. The next criterion, to be applied if the first fails, is morphological: the head is the morphological contact point with the rest of the phrase; that is, it controls the form of other words outside the construction, or has its form controlled by them. He takes an asymmetric view of agreement: for example, if *the boys* is in subject position in an English sentence, *boys* is seen to enter agreement with the predicate. Finally, if neither of the first criteria are applicable, a semantic criterion can be used: the denotation of the entire construction is a subtype of the denotation of the head (e.g., *jam sandwich* is a sandwich). All of these criteria have correspondents in the list given in Section 2.2.1.

After the dependency and its direction are settled, a dependency type must be chosen. Mel'čuk describes a set of criteria for determining whether two relations have the same label. The first criterion is the minimal pair test: if there are two ways of building a dependency with two lexical items and they have different interpretations, those two ways should have different labels. Mel'čuk gives the example of the contrast, in English, between *stars visible* and *visible stars*. In both, *visible* depends on *stars*, but the two phrases have different meanings, which indicates that different dependency types should be used. The next criterion is unidirectional substitutability: two dependents have the same label if it is true that at least one of them can be substituted by the other in any syntactic context, without affecting well-formedness; so adjectives with the same distributional properties, for example, should be typed identically. The third and final criterion says that a dependent of a certain type must be either unique (that is, the head can only have one dependent of that type) or arbitrarily repeatable, but nothing in between; this guarantees that each argument type receives a distinct label, and that adjuncts receive labels of their own.

While Mel'čuk's syntax is split into two levels, **deep** and **surface**. Some of the differences between the levels are lexical in nature: for example, idioms are considered to be expanded in surface syntax. Importantly, function words are present in surface syntax but not deep syntax. The type systems are also different: there is a

very small set of deep syntactic relations, comprising argument relations, coordinative relations, attributive relations, and a parenthetical relation. These relations are crosslinguistically stable, comprising the universal module of syntax.

Like Tesnière, Mel'čuk does not attempt to encode linearization information in the dependency representation. In fact, the author notes that linear position is a means of expressing syntactic relations, and therefore not a syntactic phenomenon in itself.

Other theories

There are many other dependency-based accounts of syntax, from multiple linguistic theories. (Many of these have much to say about other aspects of language, such as semantics, but I will focus on the syntactic layer here.) Hudson's (1984) Word Grammar is a theory of language with a monostratal representation of syntax that is realized entirely as labeled dependencies between words. The sparse label set focuses on differentiating between arguments and adjuncts, and encodes some linear order. Coordination is treated in a small constituency-like module of the grammar. In special cases of structure sharing (in the sense of Pollard and Sag 1994), words are allowed to have more than one head.

Lexicase Grammar (Starosta, 1988) is another theory of language that relies on a monostratal dependency-based representation of syntax. The basic units of the syntax are words, with no empty nodes or sub-word units. It is a strongly lexicalist theory that posits syntactic features to words, in addition to semantic and phonological ones. In Lexicase syntax, words enter untyped dependencies that always have a single head.

Other dependency-based approaches to syntax include Functional Generative Description (Sgall et al., 1986) and Dependency Unification Grammar (Hellwig, 2003). Even linguistic theories that are not explicitly based on dependencies can be interpreted in terms of dependency relations; for a dependency-based perspective on Head-Driven Phrase Structure Grammar and Tree Adjoining Grammar, see Oliva (2003) and Bangalore et al. (2003), respectively.

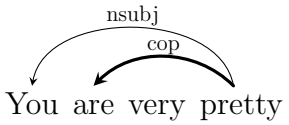
2.2.3 Dependencies in NLP

UD draws on the linguistic roots of Dependency Grammar, but it is more closely related to the recent developments that have brought dependency syntax into NLP pipelines. Its development and design are better understood in light of its alternatives and predecessors in that domain, a few of which I briefly introduce here.

Stanford Dependencies

The roots of UD are in the original Stanford Dependencies (SD, de Marneffe et al. 2006; de Marneffe and Manning 2008) representation. The authors originally proposed that standard as a mapping from English phrase structure configurations to typed dependency relations.

The SD representation is geared for practical use, as is clear from de Marneffe and Manning (2008). This is evident in multiple aspects of the design. Heads in the SD representation are semantically motivated, and therefore tend to be content words. In copular clauses, for example (4), the nominal or adjectival predicate is treated as the head, and the copular verb is one of its dependents (along with the subject of the predicate).

(4) 

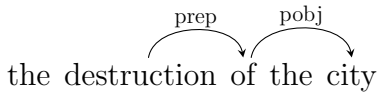
This results in many relations between content words, which in the designers' view approximates SD's binary relations to representations such as RDF (Candan et al., 2001), which are subject-predicate-object triples commonly used in knowledge representation for web applications (Jurafsky and Martin, 2009). Additionally, the set of relation labels itself is driven by practical concerns; it draws on the work of Carroll and Minnen (1999), which is described as grammatical relation annotation for parser evaluation, but makes adjustments in the granularity of labels to maximize potential usefulness in downstream applications. (This claim is examined in practice in Chapter 5; see Section 5.5.5.)

One such adjustment is in the fine-grained set of relations that SD introduces within the nominal domain. While SD distinguishes between adjectival modifiers, appositives, abbreviations and numeric modifiers (among others), CoNLL (Johansson and Nugues, 2007), for example, labels all such dependents uniformly as NMOD. SD also drops some of the distinctions made in more linguistically oriented proposals such as Link Grammar (Sleator and Temperley, 1991), which assigns different labels, e.g., to dependencies in questions and dependencies in affirmatives. The label inventory in SD is also heavily influenced by the set of grammatical functions in Lexical-Functional Grammar (LFG, Bresnan 2015), a widely-known theory of syntax that has been used to describe many languages in functional computational implementations. This influence persists in UD, and is discussed further in Section 2.4.1.

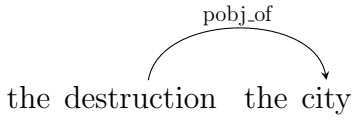
Also largely for practical reasons, SD does not allow empty nodes: all dependency relations are between words. This makes some syntactic phenomena involving unpronounced units difficult to represent clearly, but it also creates a dependency graph that the designers believed would be easier for users to interpret.

Enhanced versions SD had multiple versions, which incorporated semantically informed representations into the dependency annotation to different degrees. Two key versions are illustrated in (5). The BASIC representation is a typical dependency representation, in which every word is a dependent of some head. The COLLAPSED representation, however, incorporates prepositions, conjunctions and possessive markers into dependency labels, and removes them from the set of nodes. This pushes the standard even further in the direction of representing relations between content words. The function words that are pushed into the edge labels have characteristically **relational** semantics, and including them in labels is an explicit way of showing how such words signal a type of relation between content words that form phrases around them.

(5) a. Basic representation:



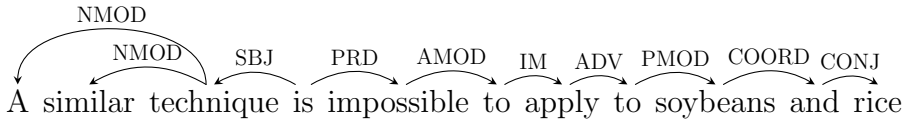
b. Collapsed representation:



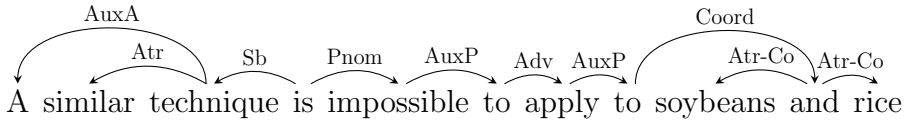
Other dependency representations for NLP

SD has been popular in NLP applications, but it competes with widely used alternatives. I will briefly describe two of those here: CoNLL (Johansson and Nugues, 2007) and Prague Dependencies (Böhmová et al., 2003). These two annotation schemes are exemplified in (6a) and (6b), respectively, in contrast to the UD annotation in (6c). (Example due to Ivanova et al. 2012.)

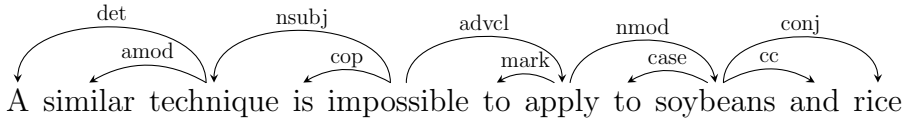
(6) a. CoNLL:



b. Prague:



c. UD:



The CoNLL dependencies have had multiple versions, first focused on multilingual dependency parsing, and later on conversion for semantic role labeling, as described in Johansson and Nugues (2007). They share important properties with SD: each dependency has a type, making the representation **typed**; each word has a single governor, making it **single-headed**; there is only one root (i.e., a word which has no governor), making the tree **single-rooted**; and there are no null elements entering dependencies. Much like SD, the CoNLL annotation standard was originally designed as a target for conversion from the Penn Treebank (PTB, Marcus et al. 1993) trees.

The relation set is, in general, similar to that of SD, with coarser labels for modifiers of nouns (collapsing determiners and adjectives, for example, in one type) and finer ones for modifiers of predicates (subdividing them into temporal, locative, etc.).

Prague Dependencies (Böhmová et al., 2003) have been one of the most prominent dependency standards in NLP. The standard is multistratal, establishing a division of labor between the surface-syntactic level, called **analytical**, and the deep-syntactic level, called **tectogrammatical**. The surface syntax layer bears the most similarity with SD and CoNLL. The Prague surface-syntactic dependencies are also typed, single-headed, single-rooted.

For the moment, the most important difference is that both Prague and CoNLL dependencies choose functional heads in places where SD chooses lexical heads (and the difference in relation to UD, the successor of SD, is even more pronounced). The consequences of this choice will be discussed further in Chapter 3, and to a smaller extent in Chapter 5. A more complete comparison of some competing ways of annotating dependency relations for NLP can be found in Ivanova et al. (2012); Rosa et al. (2014) also present a detailed comparison between SD and Prague Dependencies.

There have also been other important efforts in developing dependencies for English NLP, such as the PARC 700 Dependency Bank (King et al., 2003), a dependency-based simplification of LFG's features; or the representation of MINIPAR (Lin, 2003).

2.3 The development of Universal Dependencies

The motivation for the development of UD from SD was twofold. On the one hand, there was a need to supplement the initial phase of SD's development with a strongly data-driven approach that investigated the standard's application to a broad scope of naturally occurring data. Until 2014, there was no large-scale gold standard corpus for SD annotation; SD annotations were converted from phrase-structure trees.[4] The

[4] A deep-syntax counterpart exists; the Prague English Dependency Treebank (http://ufal.mff.cuni.cz/pedt2.0) has a manually annotated tectogrammatical layer. The surface-syntactic level is automatically produced from constituency trees. The BioInfer corpus (Pyysalo et al., 2007) has manual annotation of dependencies, but, at 33,858 tokens, it is almost eight times smaller than the corpus introduced in this section.

conversion rules, for the most part, were designed to transform the Wall Street Journal portion of the PTB (Marcus et al., 1993), and did not address challenges characteristic of other genres and registers of English. While newswire text still grounds most research into parsing for English, text from web sources is of increasing interest to NLP applications, and is significantly different from more formal registers.

On the other hand, there was a need to go beyond English. Over the years, SD was extended to several other languages, including Chinese (Chang et al., 2009), Italian (Bosco et al., 2013) and Finnish (Haverinen, 2013). These extensions share design principles and many dependency types with SD, but each includes additional relations specific to the language in question (most notably, the Chinese standard has 23 relation labels that are not in the English version) and omit others that are specific to English.

Two initiatives launched to address these needs led, together, to the development of UD: the annotation of the English Web Treebank corpus, and the development of the Universal Dependencies Treebanks. The rest of this section describes the nature of these two initiatives, and their influence on specific design decisions for UD is discussed in more detail in Section 2.4.

2.3.1 The development of the English Web Treebank

In 2012, the English Web Treebank (EWT) corpus, consisting of 254,830 tokens (16,624 sentences) of text from the web, was released by the Linguistic Data Consortium (LDC2012T13). The text was manually annotated for sentence- and word-level tokenization, as well as part-of-speech tags and constituency structure in the PTB standard. The annotation guidelines follow those used in other recent LDC Treebank releases (Mott et al., 2012): there is more structure in noun phrases and the POS tagset used is augmented. The data comprises five domains of web text: blog posts, newsgroup threads, emails from the Enron corpus, Amazon reviews and answers from Yahoo Answers.

The representation used for annotating was based on the SD standard, but included modifications motivated by the characteristics of the corpus, as mentioned

above. Annotation was bootstrapped with an automatic conversion of the EWT constituency trees to SD trees, performed with the Stanford Parser's converter tool (de Marneffe et al., 2006). Specially trained linguistics Ph.D. students (including myself) then checked the results, token by token.

Annotation procedure Annotation proceeded in phases. In the first phase, each annotator made a pass through a separate portion of the corpus and brought any difficult annotation decisions to the attention of the entire group of annotators. This phase allowed the annotators to become conscious of the difficulties of the genre, and make decisions together about how to handle them. After initial guidelines were put together, annotators moved to a round of double passes, in which different pairs of annotators independently annotated a small batch of data each, for a total of 6,670 double-annotated tokens. All disagreements were discussed within the annotator pairs, and occasionally in the larger group. These disagreements were then adjudicated.

After this initial stage of training, the group proceeded to single-annotate most of the data. The practice of flagging challenging data for discussion in the group persisted, and any decisions resulting from that process were incorporated into the annotation guidelines. Some decisions about the standard resulted in broadly applicable changes, such as conflation of dependency types. In such cases, the changes were not only incorporated in future annotations, but also implemented retroactively, in an automated fashion.

In this process, we revised the SD standard, leading to the changes and refinements presented in de Marneffe et al. (2013) and Silveira et al. (2014), which include improved guidelines for existing labels and new labels. New relations were introduced and further annotation guidelines were developed, making SD more appropriate for multiple genres and registers.

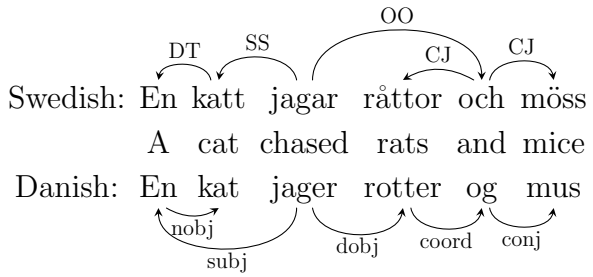
At the end of the first pass of annotations, all the guidelines produced in the process were revised by the group of annotators. When there were changes, they were, again, implemented automatically, or (in cases where manual disambiguation was needed) by searching the corpus for relevant dependency patterns and manually

making changes when applicable. An example of an automatic change relates to copulas. In SD, some verbs other than *be* were considered copulas, and annotated as such. These annotations were modified automatically after the group decided to treat only *be* as copula. In contrast, a decision to make a distinction between clauses modifying predicates and clauses modifying nouns required case-by-case revision of clauses attached to nominal predicates, to determine the clause's level of attachment. In this case, the implementation was performed manually. In this process, the corpus was made to conform with the nascent UD standard, and v.1.0 of the annotation was released as the first English datatset in the UD Treebank (Nivre et al., 2015b).

2.3.2 The need for a universal standard

As mentioned, the second challenge that UD attempts to address, in addition to SD's rigidity in the face of informal English, is the lack of crosslinguistically adequate dependency representations. Historically, dependency representations for NLP have been developed for specific languages; this has often led to very significant disparities between the representations of the same linguistic phenomena across languages. The contrast between the Swedish and Danish sentences shown in (7) (due to Joakim Nivre) is a good example of this. The Danish annotation follows the style of Kromann et al. (2004), and the Swedish, of Nivre et al. (2006).

(7)



Even though the sentence has the same structure in Danish and Swedish, the two trees share only one edge, and no labels. In Danish, the nominal *en kat* is represented with the determiner as the governor; in Swedish, on the other hand, the noun *katt* is the governor inside the nominal. Coordination is also represented differently: the tree for Swedish represents the conjunction as governor of the conjoined phrase; in Danish, the tree shows the first conjunct as the governor.

There are multiple reasons why it is useful to enforce the same annotation standard across many languages, factoring out spurious differences. At a high level, there are (at least) three ways in which such multilingual resources can be useful: for multilingual and crosslingual downstream applications in NLP; for comparative studies of language data; and for parser evaluation and learning in crosslingual settings.

The use of a common standard across languages can facilitate the development of multilingual and crosslingual systems. In a pattern-based information extraction approach, a common standard would allow dependency path patterns to be defined uniformly across data in many languages. Cortis et al. (2014) discuss the need for a multilingual parsing framework for a multilingual extension of IBM's question-answering system Watson. In Björkelund et al. (2009), a semantic role labeling pipeline uses the same dependency-tree features, defined for a common standard, across multiple languages.

Uniformly annotated treebanks can also be used for quantitative crosslinguistic research. This has been exemplified recently by Futrell et al. (2015), who used the UD treebanks to show that dependency length minimization is a widespread linguistic phenomenon and characterized it as a quantitative universal across human languages. Another example is the work of Swanson and Charniak (2014), who developed a methodology to automatically detect language transfer, leveraging on crosslingual syntactic representation. Finally, Johannsen et al. (2015) examine syntactic variation among demographic groups across several languages.

There are also advantages for parsing technology development, on two fronts. One is the evaluation front: parsing evaluations are standard-dependent, and comparing parsers' performance across standards is notoriously difficult, as argued extensively by Tsarfaty et al. (2011). (This will also be discussed in Chapter 3; see Section 3.4.3.) The lack of a homogeneous standard has obscured differences in parsing technology across languages, and the availability of multilingual treebanks can factor out annotation differences and make it clearer to what extent differences in parser performance across languages are rooted in linguistic differences.

Multilingual standards can also be useful for learning: they allow for the possibility of parser transfer, in which a delexicalized parser, trained without any word

features, is learned in a supervised fashion from annotated data in a source language, and then applied to unseen data in a target language. While it has been shown to outperform unsupervised learning, parser transfer cannot work across representations. Cross-representation results from McDonald et al. (2011) showed Portuguese to be a better source language for parser transfer into Swedish than the closely related Danish; the reason for this was that the Danish data was annotated in a very different representation than the Swedish data (as seen in (7)), thereby creating artificial differences. McDonald et al. (2013) report on a later set of parser transfer experiments with homogeneously annotated treebanks, the Google Universal Dependency Treebank (introduced below).

Recent efforts towards a universal dependency standard

Multiple projects aiming to design and implement crosslinguistically adequate annotation standards for morphosyntax, driven by the motivations discussed above, have arised in recent years. UD is, in some ways, the culmination of several such projects.

At the levels of parts-of-speech and morphology, two initiatives have been fundamental. One is the universal POS tagset of Petrov et al. (2012). In that paper, the authors proposed a tagset consisting of 12 coarse categories (such as Noun, Verb, Determiner) that exist across languages, and mapped 25 language specific tagsets to this universal set. Their grammar induction experiments show that the universal POS tags generalize well across languages. The second initiative has been the Interset interlingua for morphosyntactic tagsets (Zeman, 2008), a universal set of morphological features subsuming several existing feature sets, which was used in the first experiments with crosslingual delexicalized parser adaptation (Zeman and Resnik, 2008).

At the syntactic level, a few competing proposals appeared in the last few years. The Google Universal Dependency Treebank project (McDonald et al., 2013) was an attempt to combine the Stanford dependencies and the Google universal part-of-speech tags into a universal annotation scheme. That project released treebanks for 6 languages in 2013, and for 11 languages in 2014. The standard proposed emerged from the annotation of the treebanks: the languages were annotated independently,

with the goal of making minimal extensions to SD. Later, they were harmonized to a common denominator.

These efforts were followed by a proposal for incorporating morphology into a universal standard, due to Tsarfaty (2013), and later by the development of Universal Stanford Dependencies, which revised SD for crosslinguistic annotations in light of the Google scheme and of other adaptations of SD (de Marneffe et al., 2014). This revision was later the basis of the UD type set.

In parallel to this, the HamleDT project (Zeman et al., 2012; Rosa et al., 2014) has been an important source of harmonized annotations. The project is a large-scale effort that has harmonized treebanks in 30 languages and over a dozen representations to a single annotation style. In v.1.0 that was the Prague Dependencies style, and then in later versions it moved closer to an SD-style annotation. This required the use of structural transformations to create uniformity in the representation of coordination, the treatment of verb groups and prepositional phrases, and other points of variation. Additionally, a number of different label sets were mapped to a single set. The harmonization was done strictly automatically and cannot be considered perfect, but it still yielded a very useful resource.

The UD project merges and subsumes these efforts. (In fact, most of the authors cited in this section are active members of the core UD team, who maintain the universal guidelines recommended for all languages.) The first version of the UD annotation guidelines was released in 2013; currently the UD treebank is on v.1.3, which comprises 54 treebanks and represent the following 40 languages: Ancient Greek, Arabic, Basque, Bulgarian, Catalan, Chinese, Croatian, Czech, Danish, Dutch, English, Estonian, Finnish, French, Galician, German, Gothic, Greek, Hebrew, Hindi, Hungarian, Indonesian, Irish, Italian, Japanese, Kazakh, Latin, Latvian, Norwegian, Old Church Slavonic, Persian, Polish, Portuguese, Romanian, Russian, Slovenian, Spanish, Swedish, Tamil and Turkish.

These treebanks range in size from about 4 thousand tokens to well over 1.5 million tokens, and were developed in different ways; most were converted from existing

dependency treebanks in other representations. In addition to the UD syntactic relations, they use revisions of the universal POS tagset of Petrov et al. (2012), and the Interset morphological features (Zeman, 2008), both of which were specifically adapted for UD. Although based on its English-centric predecessor SD, UD has evolved to incorporate input from users who applied it to these multiple languages. The set of dependencies and the recommendations for their use are informed by attempts to harmonize annotations across related constructions in multiple languages.

2.4 An overview of Universal Dependencies

We have seen the major motivations for transition from SD to UD: a desire to properly accommodate a wide range of naturally occurring linguistic data, and a growing recognition of the value of crosslinguistically uniform representations. In terms of implementation, these goals are addressed in UD by means of a limited type system organized in a two-layer architecture, designed to capture both universal and language-specific phenomena while keeping them distinct. The universal layer is common to all languages, and it aims to capture phenomena at a level that highlights crosslinguistic commonalities. However, the need for parallelism with other languages often imposes a high level of abstraction on the annotation, which may be undesirable when working in a monolingual setting. For that reason, the representation is designed to be extended with language-specific relations as needed (as exemplified in Section 2.4.3 for English). This makes harmonization straightforward: the universal label that is extended must be applicable, and it can always be substituted for the language-specific label. This allows for detail that may be important for a specific language or group of languages, but difficult to port to others.

The remainder of this chapter provides an introduction to the guiding principles of UD analyses, to the universal type system and its extensions for English.

2.4.1 High-level principles

This section establishes some high-level principles underlying the UD standard, in an attempt to make more explicit what the nature of UD is, and how it relates to other frameworks for the description of syntactic structures.

UD is a nongenerative theory of syntax UD is a theory of prototypical patterns of syntax that are crosslinguistically significant. Its foundational assumptions are that languages can be analyzed in terms of a small set of prototypical distributional categories, given by the universal POS tagset; that words in these categories stand in structural relations that can be characterized with dependencies; that these dependencies can be classified according to universal prototypical properties; and that the combinations of these relations can be constrained in terms of the dependency types and the distributional types.

It is important to situate UD in the larger context of representations of syntax and their various theoretical underpinnings. In theoretical syntax today, generative models receive most of the attention. A generative model of syntax is a finite device that can generate the infinite set of sentences that constitute that language. UD is a different type of theory of syntax; it is not intended to characterize a set of sentences.[5] Nothing in UD forbids examples such as (8). A subject dependency is applicable, because the construction exhibits the the prototypical properties of the subjecthood relation in English.

(8) He accomplished. (nsubj: He → accomplished)

As such, the flexibility that allows (8) is desirable, because often in practical annotation tasks we encounter marginal or agrammatical sentences. Having a robust standard allows annotators to record as much useful information as possible, and to reject the requirement of making categorical judgments about language.

[5] In principle, UD could be viewed as a declarative generative grammar, characterizing grammatical sentences as those that can be annotated by the UD standard. However, that has not been the intention of the designers.

UD is a representation for semantics, not of semantics As an annotation standard, the main goal of UD is to provide scaffolding for semantics, giving NLP applications access to some syntactic information that is relevant for understanding-oriented tasks, such as relation extraction. Because of this, at the core of UD are predicate-argument relations.

It is crucial, however, to make a distinction between the information conveyed by this representation, and the information provided in semantic annotations. A deep understanding of natural language, supporting human-like inference on linguistic forms and interfacing with knowledge about the world, includes, minimally, a representation of the events encoded by language—the knowledge of "who did what to whom"—and goes far beyond that, comprising operations for meaning composition and a representation of the ordering and scope of those operations. Core predicate-argument relations of the "who did what to whom" type are usually considered a solid—if shallow—start towards interpretation, and are the domain of **semantic role labeling**.

The fact that semantic role labeling is a task in some ways similar to dependency parsing, due to the close relation between semantic roles and grammatical functions, is sometimes taken to mean that dependency parsing is another way of capturing semantic roles. While revealing semantic role labels is one of the important purposes that UD ultimately serves in NLP pipelines, UD is not a semantic role representation. Rather than annotate semantic arguments, UD annotates syntactic arguments, or grammatical functions.

A **grammatical function** is a recurring morphosyntactic strategy for coding the role of an argument (or modifier) with respect to different predicates. The prototypical example of a grammatical function is **subject**: a structural unit that is marked by a range of special morphosyntactic properties (in the form of linearization constraints, agreement or case marking) largely consistent across predicates. Grammatical functions are easier to learn from data than semantic roles, because they have (by definition) distinct surface realizations and are less numerous—and therefore denser. Against some views of the syntax-semantics interface (briefly mentioned below), UD does not presuppose such functions to map directly to semantic roles, or

to be semantically uniform in any way. What makes grammatical functions useful for recovering semantic roles is that a predicate maps its semantic arguments, each with a distinct role, to grammatical functions in a mostly systematic fashion, within and across languages. Even though languages code subjecthood differently, for example, there is some crosslinguistic systematicity in the way that subjects are eligible for mappings to semantic roles.

This systematic relation between syntactic and semantic arguments is certainly not completely straightforward; it has been studied in depth in what are usually called **linking theories** (Bresnan and Kanerva, 1989; Jackendoff, 1990; Van Valin and LaPolla, 1997), and it is still not completely understood. Still, many theories of syntax posit one or more levels of representation mediating the relation between morphosyntactic properties and semantic roles. This layer integrates lexical information given by the predicate and the coding of a syntactic argument to determine what semantic roles are assigned to which arguments.

UD annotates grammatical functions drawn from LFG In particular, this modular view is pursued in the paradigm of LFG (Bresnan, 1982, 2015), which inspires UD. UD's predicate-argument representation in particular draws very heavily from it.

In LFG, word order and constituency phenomena are dealt with in one module of syntax, called **c(onstituent)-structure**; a different module, called **f(unctional)-structure**, represents grammatical functions; a third module, called **a(rgument)-structure**, contains information about the semantic roles assigned by a predicate. Both c-structure and f-structure are levels of syntactic description: c-structure describes constituency properties, but f-structure can be relied on to state syntactic generalizations in terms of grammatical functions, independently of their language-specific morphosyntactic behavior. Their roles differ, however, and f-structure is a more abstract and more universal level of description, closer to semantics and directly relevant for assignment of the roles defined in a-structure.

LFG's f-structure is understood to encode grammatical functions. In Bresnan (2015), grammatical functions are defined as "classes of varying forms of expression

that are equivalent under correspondence mappings to argument structure." In LFG, arguments are assigned roles by a constraint satisfaction algorithm that maps a set of slots to a set of candidate arguments, each of which are eligible for a subset of the slots according to their morphosyntactic properties. A grammatical function is an equivalence class defined (language-specifically) by these morphosyntactic properties, and which is (crosslinguistically) eligible for a subset of argument slots. LFG allows different languages to define different ways in which a grammatical function is characterized morphosyntactically, but expects grammatical functions across languages to have the same properties with respect to argument mapping. So, for example, subjects and objects may be characterized differently by different languages, but in each case we expect it to hold that subjects will be preferred over objects for more agentive semantic roles.

Grammatical functions in LFG are not necessarily encoded in constituency grammar, as is the case in other theories, famously those in the Government and Binding (GB, Chomsky 1981). They are taken to be primitives, which is the motivation for having separate modules within syntax. This is radically different than the GB approach, in which semantic roles are taken to map directly to an abstract constituency-based representation, which is related to the surface constituency representation by a means of strictly syntactic rules and with no reference to lexical information from the predicates. In LFG, while c-structure can provide signals about the assignment of grammatical functions, such signals can also come from morphology, which lies outside the syntactic modules. So, although languages can and do individually define systematic relations between c-structure and f-structure, there is no universal, crosslinguistically valid mapping from c-structure to f-structure to a-structure. On the contrary, c-structure is understood to be a *locus* of crosslinguistic variation, while grammatical functions, however they are represented in surface forms, are stable across languages.

For semantic role assignment, it is f-structure that encodes the relevant syntactic information. Some select semantic information about how many arguments a predicate takes and what order of prominence they stand in is encoded in a-structure, information which is then used to map arguments to grammatical functions. From

this perspective, it is clear that grammatical functions are the aspects of syntax most relevant to the goals that UD aims to serve: they are crosslinguistically stable, allowing for uniform representation across many languages, and they interface with predicate-argument semantics. This is one of the premises underlying the design of UD: lexicalized dependency trees, labeled with grammatical functions, can provide information about semantic roles, because they represent both the argument-structure information introduced by the predicate[6] and the syntactic information about grammatical functions.

UD itself says nothing about the mapping from c-structure to f-structure, in any language.

In summary, at the core of clausal syntax, the UD representation annotates grammatical functions. The specific typology draws from LFG's f-structure, but it is important to realize that grammatical functions have been acknowledged to be an important level of syntactic description in many theories of language, not always by the same name. In Relational Grammar (Perlmutter, 1983), grammatical relations have a central role in syntactic analysis. In Head-Driven Phrase Structure Grammar (Pollard and Sag, 1994), valence lists specify the grammatical relations of a verb's arguments. Within the dependency tradition, Mel'čuk's criteria for labeling syntactic relations, seen in Section 2.2.2, are related to how grammatical functions can be identified.

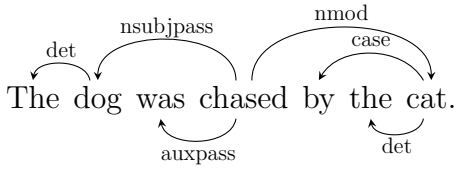
UD is robust beyond the clausal core While this discussion of grammatical functions is very focused on core clausal syntax, another important principle of UD is that a sentence should be represented as a connected graph, with information about how each word fits into the structure. This requirement makes it necessary to address syntax beyond the core clause, as is evident in a number of dependency types specifically designed for extraclausal dependents.

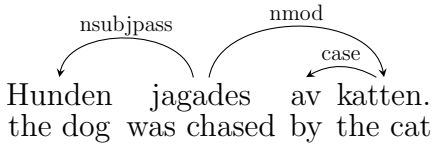
In this domain, there is less guidance from linguistic theory, where many phenomena that lie on the fringe of core clausal syntax have received limited attention.

[6]There are limits to the extent to which this information in explicitly encoded, which are discussed in Chapter 4, starting with Section 4.2.

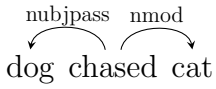
Spontaneous, natural language comprises not only the sort of examples that linguists craft to illustrate the power of compositional semantics, but also messy examples in which crisp linguistic structures mix with extraneous phrases such as contact information, speech errors, and emoticons. There are many dependency labels in UD that do not represent grammatical functions, but rather signal that the dependent's relations to the rest of the clause cannot be understood in terms of core clausal syntax, or that its semantic contribution happens not at the event representation level, but at a higher, discourse level. (Examples are discussed in Section 2.4.3.)

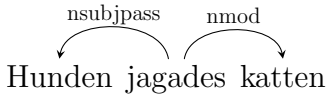
UD makes content words phrasal heads Another important principle of UD, inherited from SD, is to make content words phrasal heads, and therefore governors of function words. This design choice gives rise to an interesting property: when function words are removed from the representation, the content word nodes that remain form a connected graph—a content-word subtree. Crosslinguistically, if content words are mapped by a translation relation, the choice to promote them as phrasal heads maximizes the extent to which grammatical functions are preserved in the mapping. Example (9), due to Joakim Nivre, illustrates this.

(9) a. 

b. 

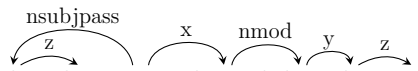
There is a parallelism between the subtrees formed by the content words (10).

(10) a. 

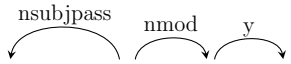
b. 

This is made possible by the fact that there is a relation between *dog* and *chased*, with no mediation from *was*. The passive marker, realized as a free morpheme and therefore a token in English, is a bound morpheme in Swedish. Similarly, if the determiner *the* was made the head of *dog*, the parallelism would be lost. This is illustrated in (11).[7] (The labels `x`, `y` and `z` stand in for dependency types that would be defined as designating the complements of auxiliary verbs, prepositions and determiners, respectively.)

(11) a. The dog was chased by the cat.



b. Hunden jagades av katten.



This can be seen as competition between syntax and morphology, as described in Section 2.4.1, and it occurs in multiple domains, within and across languages. (Bresnan, 2015, p.5) notes that "there often appears to be an inverse relation between the amount of grammatical information expressed by word structure and the amount expressed by phrase structure." Establishing relations between lexical words makes it possible to have a representational layer that remains constant across different morphosyntactic realizations of the same grammatical functions. Consequences of this design for syntactic parallelism in naturally occurring data will be shown in Section 3.4.

The UD standard enforces distinguished dependency types for the relations between functional and lexical phrasal heads, and does not allow words labeled with these types to take dependents. In light of this, we can say that functional heads have a distinguished status, which approximates their treatment in UD to that found in the foundational works of modern dependency syntax, such as the theories of Tesnière or Mel'čuk. In relation to Tesnière, we can say that the dependency types reserved

[7] In (11a), the subject of the verb is attached to the auxiliary head, while the prepositional phrase is attached to the main verb. One can argue about the attachment of the prepositional phrase in an auxiliary-head representation, as will be seen in Chapter 3, but the point about parallelism still holds.

for functional words are akin to transfer relations; in relation to Mel'čuk, we can say that they exist only in surface syntax, but not in deep syntax: that is, that they are surface elements that realize the contentful relations encoded in such dependencies as grammatical function. The flexibility of the UD type system accommodates these interpretations because it enables the representation to preserve the distinguished status of functional phrasal heads, even as it makes them governed by lexical phrasal heads. In this way, the identity of the functional heads is not lost.

UD requires hard distinctions between function words and content words Beause it assigns this distinguished status to function wods, UD requires the identification of such words. Linguistic theory dating back to Aristotle has acknowledged a pre-theoretical distinction between content and function words, supported by distinct behavior in multiple domains: function words are closed-class; they have different phonological properties; they are acquired later; they have a much smaller contribution to the meaning of the sentence, and can usually be inferred if dropped. There are clear reasons to make a distinction between content and function words.

Nevertheless, where exactly to draw that distinction is a difficult decision. It is widely accepted that lexical units exist in a grammaticalization cline, which has been described as moving from content word to function word and beyond that to clitic and affix (in the formulation of Hopper and Traugott 2003). Due to ongoing diachronic shifts along this cline, words do not always have a clear categorical status as function or content words. At the moment, UD does not present a solution to this, and requires an inevitably arbitrary line to be drawn, designating function words to be given special status by means of dedicated dependency types with distinct properties. The question of how to draw this line is pursued further in Chapter 4 for Romance languages (see in particular Section 4.5).

UD makes compromises The design of UD attempts a compromise between multiple goals. One goal, briefly investigated in Chapter 3, is to maximize the extent to which interpretation of that data is stable across languages. One more (also targeted in Chapter 3) is to produce representations that can learned for statistical parsers,

achieving a tradeoff between the expressiveness of the representation and its suitability for automatic parsing: a simple representation that can be assigned with high accuracy may be more useful to downstream applications than a more sophisticated representation which can only be assigned with very low accuracy. A third goal, discussed in Chapter 4, is to produce meaningful representations that yield linguistically interpretable data, as well as capture some generalizations formulated within a syntactic theory. Another goal, which is explored in Chapters 4 and 5 from different perspectives, is to increase the usefulness of lexicalized dependency paths for downstream NLP applications, by maximizing the extent to which the dependency tree offers information about the semantic role of an argument. Finally, yet another goal is to maintain the representation simple enough that it can be annotated consistently, and be useful to non-syntacticians.

The tension between these goals sometimes leads the principles themselves to be compromised. An example of that is how `nsubjpass`, a relation for subjects of passive clauses, is an example of the representation going beyond grammatical functions. Syntactically, subjects of passive and active clauses can be argued to have the same grammatical function, but a distinction is made in the type system to provide more useful dependency paths.[8]

2.4.2 The logic of the UD type system

UD uses a set of 40 dependency types for all languages. (These types are shown in Figure 2.1, for reference, and discussed in detail in Section 2.4.3.) This type set is designed to make some systematic distinctions among the various types of dependencies. In general, for core predicate-argument syntax, UD attempts to have labels that:

1. Distinguish different grammatical functions;

2. Express some minimal constraints on which dependency trees are well-formed, distinguishing dependents of a predicate from dependents of an entity;[9]

[8] This may be changed in future revisions of UD.

[9] In Section 3.3.3, I discuss how this is crucial in mitigating attachment ambiguity.

3. Identify boundaries between the domains of predicates;
4. Indicate whether a clause is open or closed;[10]
5. Distinguish core dependents from noncore dependents.

It is worth noting that some syntactic properties prominent in theoretical syntax are not systematically annotated. One example is finiteness: the UD label set does not distinguish between finite and non-finite clauses. The main reasons for this are that this distinction is not consistent across languages, and it does not reliably map to significant differences in interpretation. There is also no systematic distinction between adjuncts and noncore arguments, following a decision made in the PTB annotation to leave out this distinction (Marcus et al., 1993), which annotators could not make consistently.

These systematic distinctions can be described in a set of **type features**. The guarantee of a systematic distinction is that, if two syntactic relations differ at a type feature, they will not be annotated the same.[11] I will discuss seven distinctions that are made systematically by UD, and can be understood as the attribute hierarchy shown in Figure 2.1.

The structural attribute

A [+**structural**] type designates an asymmetric relation—a true syntactic dependency. This asymmetry manifests itself differently in different types of phrase. In exocentric complementation relations, the asymmetry takes the form of selection restrictions: the predicate can associate with a lexically defined number of arguments of particular types, and those arguments are dependent on it. In endocentric phrases, the syntactic behavior of the two nodes of a [+**structural**] edge indicates which one is the head: the dependent, but not the head, can be dropped without harming

[10] This distinction comes from LFG. A clause is closed if the main predicate's arguments are all realized within the clause, and open if, as happens in raising and control, the subject is obligatorily realized in a higher clause.

[11] There are currently exceptions to this, which I address in the current section.

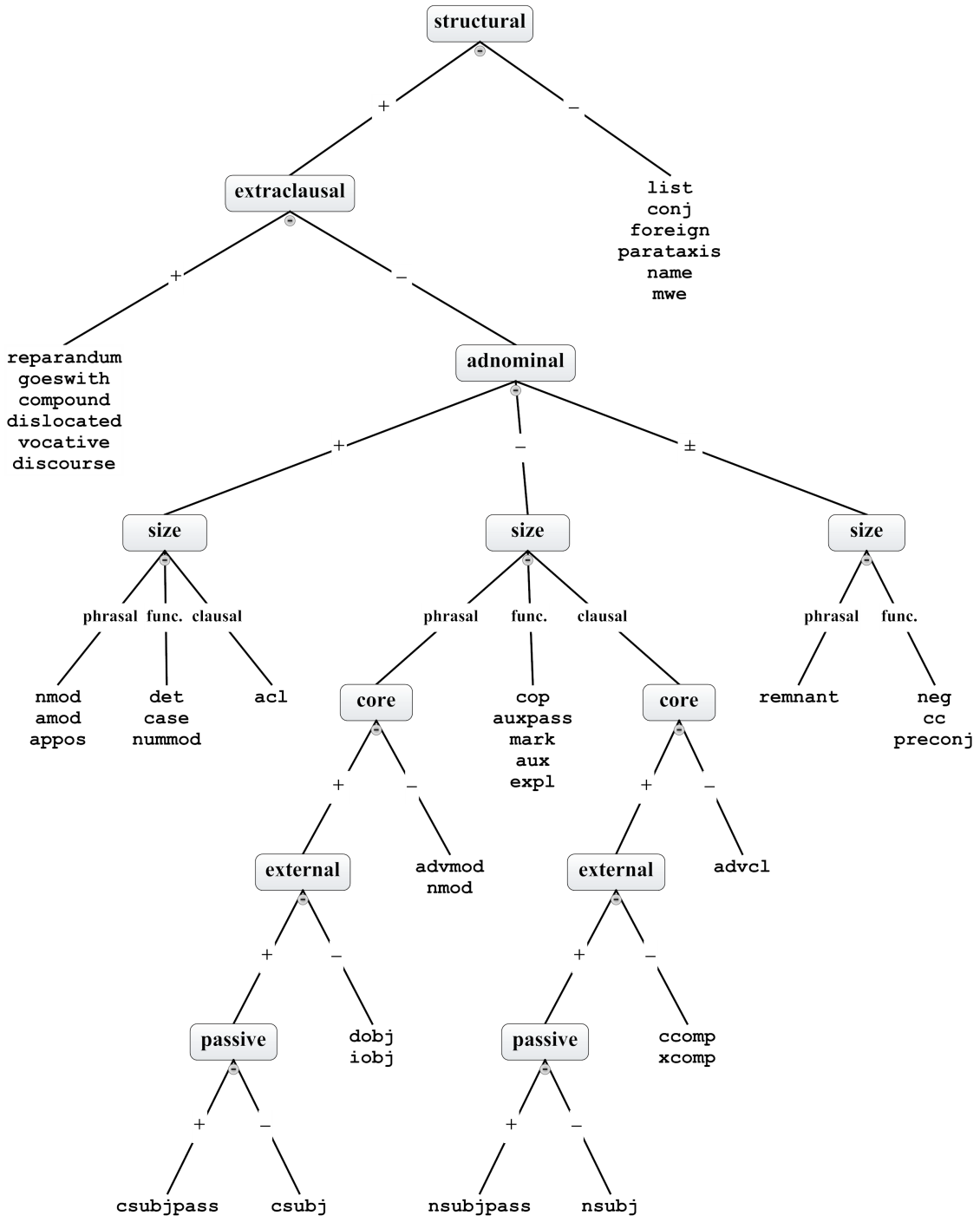


Figure 2.1: Most universal relations from UD (v.1), organized by feature signature. Not included in the table are the relations `dep`, `root` and `punct`. Thanks to Jody Greenberg for his help producing this chart.

grammaticality. Conversely, with [–**structural**] dependencies, there is no clear asymmetry; either word could arguably be treated as the head. (In these cases, we adopt conventions about what should be the head; more on this in Section 2.4.3.) The `mwe` relation (12) is a good example of a [–**structural**] feature: it creates a lexical unit *because of*, which functions as a single marker in the clause.

(12) He cried because of you. (mwe)

The extraclausal attribute

As mentioned before in Section 2.4.1, while some dependency types designate grammatical functions and form the building blocks of core clausal syntax, other dependencies are meant for attaching structures above the clausal level, or (in the case of `compound`) at the word level. This attribute separate the dependencies that contribute to form the clausal spine to other dependencies that form structure below and above that level.

The adnominal attribute

This feature distinguishes dependencies that occur within the nominal domain, attaching to a nominal head, from those that occur in the predicate domain, attaching to a predicate head. The difference between adjectival and adverbial modifiers is due to this feature.

(13) a. It was a quick run. (amod)

b. I ran quickly. (advmod)

Some dependency types are unspecified with respect to this attribute, as we will see below.

The size attribute

This attribute has three values: **functional**, **clausal** and **phrasal**, according to the size of the dependent bearing the label.

Dependency types with a [**functional**] specification indicate that the dependent is a function word, as opposed to an argument or modifier. In general, [**functional**] dependents correspond to functional heads (such as modals or complementizers) in other grammar formalisms. Determiners are a typical example, as in (14).

(14) the girl (det)

The **clausal** value is assigned if the dependent is a predicate. Dependents with the value [**clausal**] (such as `ccomp`, `acl`, `advcl`) and sub-clausal dependents stand in contrast to [**phrasal**] dependents (such as their counterparts `dobj`, `amod`, `advmod`), which are not argument-taking predicates.

The core attribute

Among [**phrasal**] and [**clausal**] dependents, we make a distinction between core arguments, and noncore dependents (some of which may also be considered arguments). By **core** dependents, we mean dependents in grammatical functions that are distinguished with special coding strategies that are not used for adjuncts. Their argument status is syntactically encoded, whereas noncore dependents have a form that does not clearly mark them as argumental or adjunctive.

Andrews (2007) argues that most languages distinguish core arguments from obliques, which can be arguments or adjuncts. The author claims that core grammatical functions can characteristically be associated with a range of semantic roles, while obliques are coded in a way that tightly couples them with certain roles. (An example of this is how prepositional phrases in English have their semantic roles restricted by the prepositional marker, while subjects can take on almost any role.)

Core arguments have a distinguished status in several syntactic theories, such as LFG (Bresnan, 1982, 2015), Role and Reference Grammar (Van Valin Jr., 1992) and Basic Linguistic Theory (Dixon, 1994). It should be noted that both Dixon and

Andrews apply this distinction only to nominal arguments. In the present work we extend it to clausal arguments: clauses that are coded as complement clauses (as opposed to as adverbial clauses) have the [+**core**] feature.

The external attribute

Within the space of core arguments, we distinguish between [+**external**] arguments (subjects) and [−**external**] arguments (objects).

The passive attribute

Finally, [+**external**] arguments are further distinguished in terms of the voice of the predicate that selects them. Subjects of passive verbs (15), be they clausal or nominal, are always annotated differently than subjects of active verbs.

(15) The window was broken by the kids (nsubjpass)

2.4.3 A brief introduction to UD dependency types

The 40 dependency types of UD are briefly introduced and exemplified in this section. In general, the discussion of relations already present in SD is less detailed; the focus here is on UD's newly introduced or reinterpreted relations. Some of the definitions in this introduction are purposefully underdetermined, because they require further data analysis to be made complete; some of this data analysis is provided later, in Chapter 4.

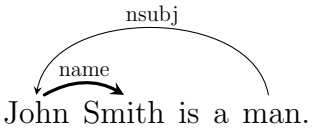
[−**structural**] dependencies

These flat dependencies have an arbitrary distinction between head and dependent. They form phrases without any clear internal structure. There are five different dependency types with this feature signature; three of these, namely `name`, `mwe`, `goeswith`, form (together with `compound`, which has a different feature signature)

the set of **word-level dependencies**: dependencies that form complex lexical units and which, in that sense, are not strictly syntactic.

The fact that these dependencies have a [−**structural**] feature does not mean that dependents with these types cannot have internal structure themselves. For example, under a dependency typed `parataxis`, we may see full-fledged clausal structure. However, the relation between the entire paratactical clause and its governor does not have structural properties.

name This dependency is used to join together multiword proper names. By convention, proper names are left-headed.

(16) 

mwe The `mwe` type, which abbreviates 'multiword expression', joins together expressions that span multiple tokens acting as a single function word. The term is often thought to encompass a family of related but linguistically distinguishable phenomena, which Sag et al. (2002) define generally as "idiosyncratic interpretations that cross word boundaries." In UD, the use of the `mwe` label is much more restricted than this definition.

Multiword expressions are divided by the same authors into fixed expressions, semi-fixed expressions, syntactically flexible expressions and institutionalized phrases. Fixed expressions, such as *by and large* or *ad hoc*, do not tolerate any internal modification, morphological or syntactic. Semi-fixed expressions, which include some idioms (such as *kick the bucket*), compound nominals (such as *car park*), and proper names, can undergo some degree of lexical variation—for example, inflection in *kicked the bucket* or *car parks*. Syntactically flexible expressions, which include verb-particle constructions (*look up*), another class of idioms[12] (such as *let the cat out of the bag*), and light-verb constructions (such as *make a mistake*). Institutionalized phrases are syntactically variable and semantically compositional, but are idiosyncratic.

[12]These would be decomposable idioms, as opposed to the non-decomposable idioms that are semi-fixed expressions. For more, see Sag et al. (2002).

In UD, the guidelines with regard to these expressions can be summarized as: "If there is structure, represent it." Any constructions that undergo any type of morphosyntactic variation are assigned a surface syntax representation. This encompasses all idioms (17), and light-verb constructions (18).

(17) take a photo (dobj: take → photo)

(18) kick the bucket (dobj: kick → bucket)

Compounds, proper nouns and verb-particle constructions are dealt with by means of specific relations. Finally, we are left with fixed expressions, which are dealt with by means of the `mwe` relation. The relation is reserved for units acting as function words, as exemplified in (19); contentful fixed expressions, such as *by and large*, are not represented with this relation.

(19) He cried because of you. (mwe: because → of)

In addition to occasional difficulties in determining whether the criteria of semantic noncompositionality and syntactic variability apply, the use of the `mwe` relation in UD is further complicated by the requirement that the unit be a function word, which is not always a clear distinction.

By convention, multiword expressions in English are head-initial.

goeswith This relation is meant to match the GW POS tag that was introduced in the revised PTB guidelines (Mott et al., 2012). This POS tag is used when a typing error is present in the data and a word is split into two or more space-separated strings. When this happens, the pieces that are tagged GW are joined to the "head" piece with the `goeswith` relation (20).

(20) I th ought you were coming. (goeswith: th → ought)

UD does not allow words with a space in them, so this dependency allows such words to be built from multiple space-separated tokens. By convention, these relations are always left-headed.

foreign This relation is used to join strings of words that are unanalyzed because they are in a language different than the main language being annotated (21).

(21) It's like my mom always says, C'est la vie.

foreign

foreign

list This relation was introduced to handle extragrammatical material which is interweaved with text in certain genres. In emails, for example, there is often contact information given in a snippet of text at the end of the message. The internal structure of such a snippet is given by the list relation (22).

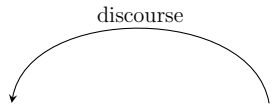
list

(22) Natalia Silveira, 7325556637

[+structural, +extraclausal]

Among [+**structural**] dependency types, [+**extraclausal**] dependencies are reserved for elements whose distribution is governed by specific syntactic properties and have internal structure, but which are not integrated in core predicate-argument syntax. Mel'čuk has a dependency type called APPEND for essentially the same purpose (Mel'čuk, 2009).

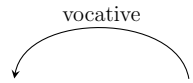
discourse The original Stanford standard did not capture aspects of colloquial writing that are not present in the newswire text in the PTB, which has driven much of parser and representation development in NLP. The EWT contains interjections, emoticons, and other discourse markers, which function as extraclausal dependents, contributing meaning at the discourse level. For these elements, we introduced the relation `discourse`. An example would be (23).



(23) Hello, my name is Vera.

As in this example, `discourse`-typed dependents are always dependent on a predicate. All words tagged INTJ in the revised PTB POS tagset (Mott et al., 2012) are labeled as `discourse`.

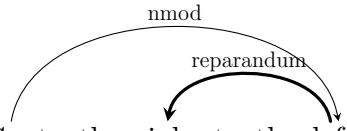
vocative As its name implies, this relation is used for vocatives (24). This type of dependent is singled out from other discourse-level relations because vocatives identify the addressee of a discourse, which can be useful information for client applications.



(24) Tracy, do we have concerns here?

reparandum Another phenomenon that occurs frequently in spontaneous language use (including, of course, speech, but also informal web text) is the occurrence and reparation of disfluencies. According to Shriberg (1994), disfluencies have two to three regions: the reparandum, the interregnum, which is an optional editing term, and the alteration, which repairs the disfluency and marks the onset of fluency.

For these cases, we introduce the `reparandum` relation, which serves to analyze the erroneous or disfluent part of the sentence that was discontinued by the speaker. Although disfluencies come in many different forms and are hard to analyze coherently, the idea is that there is usually a complete fluent structure, which the alteration fits into, and extraneous fragments consisting of the reparandum (which may be modified in the alteration, or simply abandoned) and the optional interregnum. The reparandum relation serves to fence off the disfluent fragments by making them dependent on the alteration, which ensures that the dependency tree is still connected if they are removed.



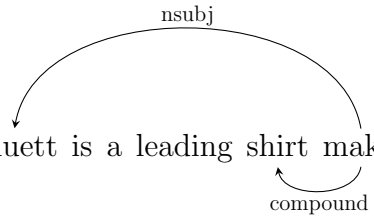
(25) Go to the righ- to the left.

dislocated Finally, one more type of element that did not have an appropriate analysis in the SD standard are dislocated constituents. These are preposed or postposed extraclausal elements, often topics, usually co-referent with an entity in the clause. Dependents of this type always have a predicate head, reflecting their high attachment.

(26) This is our office, me and Sam.

compound The `compound` type introduces word-level dependents, akin to `mwe`, but it is distinct in that a clear head can be identified. The relation is reserved for compounding, and so it always designates a direct dependent of a head, never a phrase.

(27) Cluett is a leading shirt maker.



[+structural, −extraclausal]

Most structure-forming dependencies do play a role in the predicate-argument syntax, thereby getting the feature [−**extraclausal**]. This is the class of dependencies that receives most of the attention in linguistic theory.

[+structural, −extraclausal, ±adnominal]

While most [−**extraclausal**] dependencies are specified with respect to the **adnominal** attribute, the labels discussed in this section are used both within and outside the adnominal domain.

neg Negation is a type defined by its semantics, and it applies to specific negation words that can have heterogeneous syntactic functions. The label `neg` is used not

only for verbal negation (28a), but also for negation in the nominal domain (28b), when it functions syntactically as a determiner.

(28) a. I have no pencils.

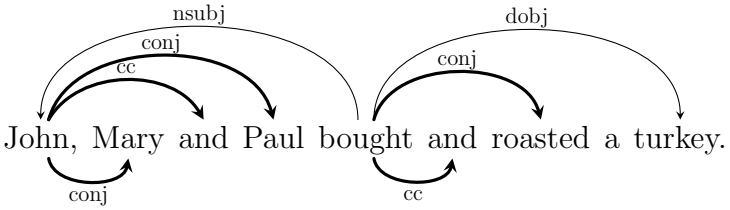
b. I do not have pencils.

In English, among other languages, these functions are realized by different lexical items.

cc This relation is reserved for coordinating conjunctions; it is exemplified in (29).

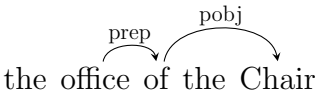
conj In UD, the first conjunct stands as phrasal head of the whole conjunction. As such, it enters dependencies with the phrase's governor, and it is the governor of any dependents attaching at the phrase level, such as *a turkey* in (29), which stands as object of *bought*. The `conj` relation is used to attach other conjuncts to the first one.

(29)

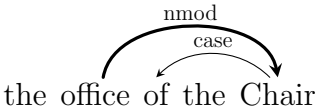


nmod In English, the most important difference between UD and SD is the treatment of prepositional phrases. Whereas in SD prepositions are heads of their complements (30a), in UD, they depend on the complements (30b).

(30) a. SD:

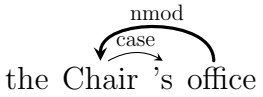


b. UD:



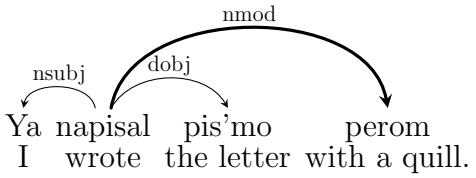
In the UD analysis, nmod labels the relation between two content words, and the preposition depends on its complement. In general, nmod labels an oblique argument or adjunct whose relation to the governor is further specified by a marker. The motivation for this change is to push the principle of making content word heads, and consequently bring out parallels between languages. Under this analysis, English prepositional phrases are analyzed in the same way as case-marked nominal phrases in other languages, factoring out the encoding difference. In fact, even in English the parallel between a case-marked nominal phrase and a prepositional phrase, in the genitive alternation, is now obvious; compare (30b) to (31) below.

(31)



When the role of the adposition in English is performed by a case marker, such as in the Russian example below (32), the use of nmod makes the two alternatives more similar.

(32) Russian:



[+structural, −extraclausal, +adnominal]

Dependents in the adnominal domain include functional dependents as well as noncore arguments and adjuncts. UD makes fine distinctions in this set. Clausal and phrasal dependents of nominals, as opposed to their counterparts for predicates, are always considered noncore dependents, and thus have no **core** attribute.

[+structural, −extraclausal, +adnominal, functional]

nummod This label is for numerical modifiers of nouns.

(33) Twenty years from now, robots will talk like humans.

det This relation is used for determiners (34a), loosely defined as a small set of function words that stand in complementary distribution and are distinct from open-class nominal modifiers.

(34) a. the boy

b. each student

case This relation is used for any function word that specifies the grammatical function of the nominal in relation to its head, as seen an previous examples and repeated below in (35). The relation is used uniformly for adpositions.

(35) a. the Chair 's office

b. the office of the Chair

Across languages, the UD analysis brings adpositions in languages like English closer to proper case markers in other languages, such as Hebrew (36). This is another instance of how preferring lexical heads can lead to greater parallelism across languages, due to competition between syntax and morphology. When case markers are tokens, as below, the analysis is exactly parallel to that proposed for English prepositions.

(36) Hebrew:

wkfraiti at hsrj
and when I saw ACC the movie

[+structural, −extraclausal, +nominal, clausal]

Clausal dependents of nominals all fall under one relation, `acl`, which is analogous to the `advcl` relation for the predicate domain.

acl In English, the canonical example of a clause modifying a noun is the relative clause. Relative clauses are a type of clausal noncore dependent in the nominal domain, characterized by the fact that the modified noun plays a role in inside the clause that modifies it. There are, however, other types of clauses that can modify nouns, either coexisting with relative clauses (as in English) or exclusively within a language (as in Warlpiri, which according to Comrie 1989 lacks a relative clause construction). Such other clauses clearly modify a noun, but do not have this characteristic relative-clause structure. An example is given in (37).

(37) His talk gave us a sense of the issues as he sees them. (acl)

SD had only a relative clause type, but due to the existence of a more general form of clausal modification of nouns, UD introduced a general relation for clausal modifiers of nouns, `acl`. In languages that have distinctive relative clauses, such as English, the language-specific relation `acl:relcl` is available.

The label `acl` is also appropriate for the representation of depictives. Depictives are predicative, so they should be represented with a [**clausal**] dependent. UD takes the view that a depictive is a dependent of the nominal that it modifies.

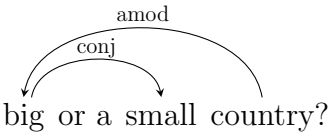
(38) a. She entered the room sad. (acl)

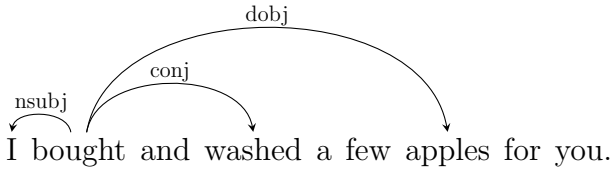
b. He painted the model naked. (acl)

[+structural, −extraclausal, +nominal, phrasal]

remnant SD was always committed to having no empty nodes, which presents a challenge for the treatment of unpronounced material. This commitment remains in UD. Three strategies are used for different constructions with unpronounced material; one of those is based on the relation `remnant`. For context, I will briefly summarize the three strategies.

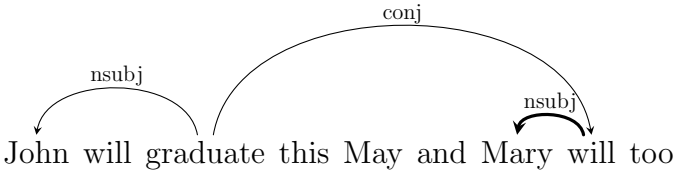
In the case of right node raising (39), we analyze the surface syntax without making reference to any unpronounced nodes.

(39) a. 

b. 

Recovering the unpronounced nodes in trees like these can be cast as a problem of conjunct propagation.

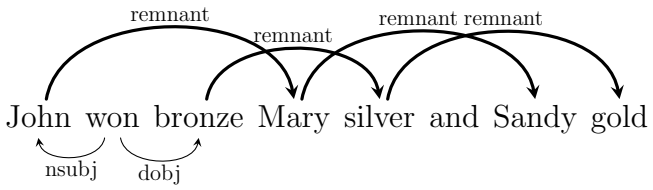
A second strategy, used for VP-ellipsis, is **promotion by head elision**, whereby a functional dependent can take the place of an elided lexical head in the structure. The stranded auxiliary material takes the role of the verb, and stands as head of the subject (40).

(40) 

However, neither of these strategies can be used when gapping is present, as in (41).

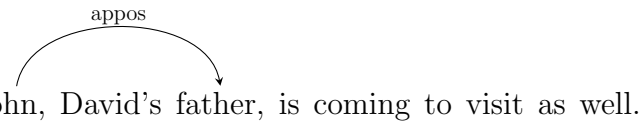
(41) John won bronze, Mary silver, and Sandy gold.

In this third case, there is no functional material that can be promoted, and there is no surface analysis that can preserve the correct relations between the arguments of each (unpronounced) copy of the predicate. For these situations, we introduce the `remnant` relation. In gapping, a **remnant** argument corresponds to a correlate in a preceding clause. The `remnant` relation connects a remnant to its correlate, signaling that the relation between the remnant and an unpronounced copy of the predicate is the same as that between the correlate and the pronounced predicate. When more than one unpronounced copy is available, all `remnant`-typed dependents corresponding to the same argument slot are chained together, as below:

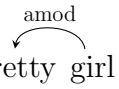
(42) 

This is done so that arguments in the same tuple can be grouped together, by their distance from the original argument slots. This is still a problematic analysis, as `remnant` is not a true dependency and it is often difficult to establish the conditions for its use clearly. It is likely that this will be revised in the future as the UD standard is further developed.

appos The `appos` relation is used for appositives, which are phrases that can usually substitute the head, and add information about it.

(43) 

amod This relation labels all modifiers of nouns that have an adjectival head.

(44) 

[+structural, −adnominal]

In the predicate domain, there are many relation types. They can be divided in two important ways. In one dimension, core arguments contrast with noncore dependents, which include noncore arguments and adjuncts. In another, dependents vary in size between functional elements, phrases, or clauses.

[+structural, −nominal, functional]

These are the functional elements that attach to predicates.

mark This relation is for the so-called subordinating conjunctions of traditional grammar, which include complementizers, prepositions and *wh*-words. These words are predicate-domain analogies to `case` dependents. They are words that introduce embedded clauses, and their semantic contribution is to clarify the relation between the matrix and the embedded material.

aux This label is used for verbal auxiliaries and modals. The application of this label is discussed in more detail in Section 4.5.

(45) You should get out of here

auxpass This label designates a special type of auxiliary: a marker of the passive voice.

(46) I was raised there.

cop It was always the case in SD that the copula was treated differently than verbal predicates. Copulas in UD are still represented as a dependent of the nominal or adjectival predicate they introduce, and this is a significant advantage for crosslinguistic parallelism. Many languages often or always lack an overt copula in constructions with nonverbal predicates; the content-head analysis ascribes the subject in the same

way whether or not a copular verb is present. Compare the examples from English and Russian in (47):

(47) a. Ivan is the best dancer. (nsubj)

b. Russian:

Ivan lučšij tancor. (nsubj)
Ivan best dancer

Even English, in nonstandard varieties, can allow the copular verb to be dropped, as in (48), an example from the EWT.

(48) Email usually free if you have wifi. (nsubj)

We extend this analysis of copulas to the constructions where a copular element is realized, such as in (49).

(49) She is great. (cop)

The use of `cop` will be discussed further in Section 4.5.

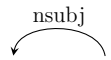
expl This is used for semantically empty material that receives a grammatical function, such as subject (50a) or object (50b).

(50) a. It is raining. (expl)

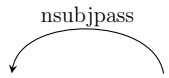
b. I hate it that we can't stay longer. (expl)

[+structural, −nominal, phrasal, +core, +external]

nsubj Nominal subjects of active-voice clauses are typed `nsubj` (51).

(51) The sky is blue


nsubjpass This type is similar to `nsubj`, but it bears the [**+passive**] feature and denotes a subject that is promoted by a regular morphosyntactic alternation, like the passive voice in English (52).

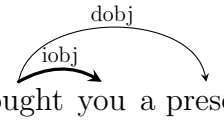
(52) The whole house got painted red.


[+structural, −nominal, phrasal, +core, −external]

dobj This type is used for direct objects, and is exemplified in (53).

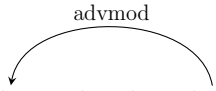
(53) We loved it.


iobj This relation is used for indirect objects (54), in languages that allow for them. Indirect objects are always a second complement; complements that occur alone are always annotated as `dobj`.

(54) I brought you a present.


[+structural, −nominal, phrasal, −core]

advmod These are noncore arguments or adverbial modifiers of predicates, headed by an adverb (55).

(55) Where do they live?


[+structural, −nominal, clausal, +core]

ccomp This relation is used for clausal internal arguments, that is, clausal complements. Specifically, it is used when the subject of the complement clause is **closed**, that is, when all the core arguments of the clause's main predicate are realized within clause boundaries (56). (This includes clauses with arbitrary control.)

(56) I always say that you have a lot of potential.

xcomp These dependents are similar to those typed `ccomp`. The difference is that a clause typed `xcomp` is **open**, lacking an internal subject; instead its subject is obligatorily identified with the subject of the higher clause. This distinction comes from LFG; see Bresnan (2015). This means that the identity of the subject is given by an argument of the matrix clause (the lowest one), and there is no other possible interpretation of that subject.

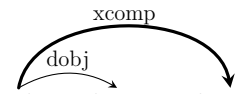
This label applies to the representation of both raising (57a) and subject control (57b); the fact that in the former construction the understood subject does not receive a semantic role in the matrix clause, while in the latter one it does, is understood to be a lexical difference encoded by the matrix predicate. As such, that difference is not reflected in the dependency representation.

(57) a. Ken seemed to like it.

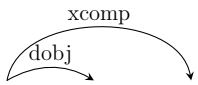
b. He loves to dance.

The same analysis extends to object control (58a) and Exceptional Case Marking (58b), which are other types of open-clause constructions. In both cases, the understood subject is identical to the direct object of the matrix verb. Only in object control does the understood subject receive a semantic role from the matrix verb, but again that difference is not reflected in the UD representation.

(58) a. I made him dance.

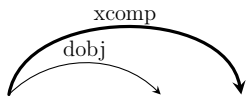


b. I believed him to be innocent.



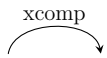
The `xcomp` relation also applies to resultative complements. We echo the view of Huddleston and Pullum (2002) in claiming that resultative phrases are always complements. This is in line with the classical LFG analysis (Simpson, 1983), where resultatives are argued to form via a lexical rule that changes the argument structure, adding a complement. So, although *blue* seems optional with respect to an ordinary use of *painted* in (59), it is an obligatory complement with respect to a resultative argument structure for the verb *paint*. In this view, the right analysis for resultatives is as `xcomp` dependents of the main predicate.

(59) He painted the house blue



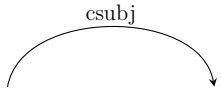
Complements of attributive verbs as exemplified in (60), are also as analyzed as `xcomp`.

(60) You look fantastic

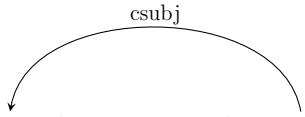


csubj By analogy with `nsubj`, clausal subjects (finite or not) of active-voice clauses are typed `csubj`.

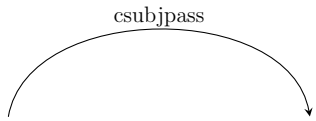
(61) a. It is normal to get nervous.



b. That they would even try this is very disconcerting.



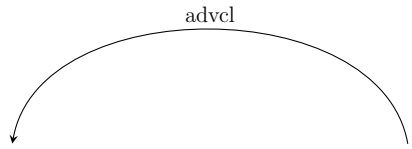
csubjpass This is similar to `csubj`, but has the [**+passive**] feature and dependencies only in passive clauses.



(62) It was believed that they had been kidnapped.

[**+structural, −nominal, clausal, −core**]

advcl This relation holds between a predicate and a clause that functions as an adverbial.

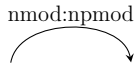


(63) Right when we walked into the room, they stopped talking.

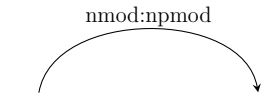
English-specific relations

Some language-specific extensions are in place for annotating English with UD. Some of these types, such as `acl:relcl`, appear in several other languages. These extensions are explicitly defined to subtype a universal label (which must be included in the label name and separated from the extension by a colon), and are expected to have all the parent label's properties.

nmod:npmod This relation is used for nominal dependents that function as adverbials and do not have an adposition or case-marker dependent, as opposed to those typed `nmod` (64).



(64) a. I paid 90 dollars a share.



b. Can you not fold them that way?

nmod:poss This relation holds between a nominal head and its possessive determiner (65).

(65) From this day on, I own my father 's gun.

nmod:tmod A bare noun phrase functioning as a temporal modifier is typed `nmod:tmod` (66). Like `neg`, this is really a semantically motivated label, meant for applications.

(66) Friday I came in a little late.

cc:preconj The `cc:preconj` relation is used for the first element of correlative conjunctions like *both . . . and*, or *either . . . or* (67).

(67) Either you fix this, or you explain to them what happened.

acl:relcl By far the most common type of clauses to modify nouns in English are relative clauses, characterized by a fronted or null relative pronoun. These are labeled with `acl:relcl` (68). The advantage of making this distinction for relative clauses is that it allows the relative pronoun, which is annotated with its role inside the relative clause, to be identified with the nominal that it corefers with. (See Section 5.5.5 for an application of this.)

(68) John, who just moved here, is gonna be sharing your office now.

The `acl:relcl` label in English is used not only for canonical relative clauses, but also for free relatives. On the surface, free relatives look exactly like interrogative subordinate clauses:

(69) a. I didn't hear where she told me to put it.

b. I put my purse where she told me to put it.

However, further probing the syntactic behavior of these two structures reveals important differences. Whereas different types of *wh*-complements can be accepted in (69a), the free relative in (69b) alternates with locative adverbials instead.

(70) a. I didn't hear what room she told me to put it in.

b. *I put my purse what room she told to put it in.

(71) a. *I didn't hear on the table.

b. I put my purse on the table.

In general, the syntactic category of a free relative is that of the *wh*-phrase in it. For this reason, we adopt the analysis proposed in Bresnan (1982) and call the clausal material dependent on the *wh*-phrase (72). The distinguishing characteristic of this representation of free relatives is that the head of the `acl:relcl` dependency is the relative pronoun itself.

(72) I put my purse where she told me to put it.

det:predet The `det:predet` relation is used when words like quantifiers co-occur with (and precede) a determiner.

(73) All the kids asked for ice cream

2.5 Conclusion

This chapter introduces Universal Dependencies, a new dependency representation specifically designed to enable crosslinguistic syntactic annotation. UD came out of revisions of Stanford Dependencies (motivated by the annotation of a new corpus,

the EWT) and a merger of efforts to develop crosslinguistically adequate annotation schemes that could be universally applicable. The discussion of UD's alternatives and predecessors situates it with respect to traditional Dependency Grammar, existing dependency representations for NLP and previous work towards a universal representation. These comparisons help highlight specific design choices that were made for UD, such as the choice to promote lexical heads, and that distinguish it from other representations such as CoNLL. This specific distinction will be investigated further in Chapter 3. In addition, I discussed the theoretical underpinnings of UD, an essential (but previously unfulfilled) requirement for justifying argumentation moves in UD-based syntactic analysis, which I explore in Chapter 4. Finally, the UD type system was briefly introduced, along with a feature-based view of its key properties; I will revisit these features, and the distinctions they induce, in Chapter 5.

Chapter 3

Producing structure: Parsing

3.1 Introduction

There is a considerable amount of research suggesting that the choice of syntactic representation can have an impact on parsing performance, in constituency (Klein and Manning, 2003; Bikel, 2004; Petrov et al., 2006; Bengoetxea and Gojenola, 2009) as well as dependency (Nilsson et al., 2006, 2007; Schwartz et al., 2012) parsing. Recently, this has led designers of dependency representations (de Marneffe et al., 2014) to suggest the use of an alternative **parsing representation** to support the performance of statistical learners.

While it is clear that, at the limit, trivializing a linguistic representation in order to make it easier to parse is undesirable—for example, by making each word depend on the previous one—there certainly exists a variety of choice points at which more than one type of design is defensible. This is evidenced by the differences among dependency representations for Natural Language Processing (NLP), which are briefly commented on in Section 2.2.3.

One such choice is between prioritizing syntactic or semantic criteria for headedness. In the dependency tradition, both types of criteria have been recognized to motivate headedness, leading to well-known conflicts (as discussed in Section 2.2.1, and also Nilsson et al. 2006) and raising the question of which criteria to prioritize. Here I investigate the representation in Universal Dependencies (UD, Nivre et al.

2016) of four syntactic constructions that are *loci* of such conflicts: verb groups, prepositional phrases, copular clauses and subordinate clauses.[1] My interest is in how the representations chosen for these constructions affect our ability to accurately parse in that standard. Relatedly, I investigate the motivation for having lexical heads, in order to determine whether we should insist on this design even if it turns out to be more difficult to parse.

For each target construction, structural transformations are defined that demote the lexical head and make it dependent on a functional head. If representing functional heads is more favorable for learning to parse but lexical heads are better for use, then it could be advantageous to use these transformations to create a parser-internal representation while preserving the choice for lexical heads in the output.

In order to address the question of how each representation fares in parser learning, I show experimental results in four conditions: with a transition-based parser for the English Web Treebank (EWT), the Wall Street Journal (WSJ) portion of the PTB, and a French treebank; and with a graph-based parser for the EWT. These experiments explore three dimensions of variation: between the two main approaches for data-driven dependency parsing (see Section 3.4.1 for a brief discussion of their differences); between two languages; and between different UD data sets for the same

[1]It is worth adding a note about two other constructions that could easily have been part of this list: noun phrases and coordinated phrases.

In other work (Nilsson et al., 2006; Schwartz et al., 2012), coordination has been studied among other constructions that present difficulties for choosing a head. Here I leave it out because it is different in nature than the constructions investigated. In this chapter, the target constructions have the characteristic of possessing a functional/syntactic head that is distinct from a lexical/semantic head. In coordination, however, while a function word is often present, it does not have a claim to syntactic head. A coordinating coonjunction does not determine the distributional properties of coordinated items; coordinated nominals still have the behavior of a nominal. The difficulty of representing coordination lies elsewhere: most often, it is an intrinsically symmetric construction (modulo agreement phenomena targeting specific conjuncts, in languages such as Arabic), while dependencies are intrinsically asymmetric relations. In the terms of Tesnière (2015) (see Section 2.2.2), we concern ourselves here with dependents that can be said to enter transfer relations, and coordinates stand in a junction relation.

As for noun phrases, whereas in theoretical syntax there has been much debate about the consequences of adopting nominal versus determiner heads for noun phrases, in dependency representations for NLP, the almost consensual choice has been for nominal heads. A famous exception is the Danish Dependency Treebank (Trautner-Kromann, 2003), as illustrated in (7) in Chapter 2. For that reason, in this chapter I leave determiners aside to focus on other issues that are more characteristic of UD.

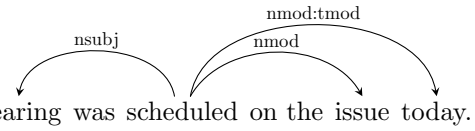
language. In each experiment, I am specifically interested in the potential usefulness of a parsing representation: if a parser for UD makes use of an internal functional-head representation, will there be significant improvements in the UD output?

In summary, I find that all of these factors influence the usefulness of defining a functional-head parsing representation, and in particular that such a representation is much more useful for some languages than others. Extending the results to four other languages, I show that this strategy can yield as much as 2% absolute improvement in labeled acuracy score (LAS), in the case of Spanish. On the other hand, I caution against naive comparisons, since LAS is in practice biased towards functional heads. I also show that, despite advantages of functional heads in many parsing setups, there are empirical reasons for preferring a lexical-head design in multilingual settings.

3.2 Related work

In Nilsson et al. (2006), the authors investigate the effects of two types of input transformation on the performance of MaltParser (Nivre et al., 2007). Those two types are: structural transformations, of the same nature of those investigated here; and projectivization transformations, that allow non-projective structures to be represented in a way that can be learned by projective-only[2] parsing algorithms, and then transformed into the non-projective representation at the end. Of interest here are the structural transformations, which target coordinated phrases and verb groups. The data and baseline representation come from the Prague Dependency Treebank (PDT) version 1.0 (LDC 2001T10). The PDT's representation of coordination is so different from UD's that the transformation does not apply.

[2]In projective dependency trees, there are no crossing arcs when dependencies are drawn as edges above the words. While most natural language structures are projective, nonprojective structures also exist. An English example (adapted from Kübler et al. 2009) is shown in (74).

(74) 

Because of such possibilities, UD trees are not guaranteed to be projective.

The verb group transformation, on the other hand, is almost identical to the AUX transformation proposed in Section 3.3.2. In the PDT, auxiliary verbs never have dependents. Other dependents of the main verb are attached to the first verb of the verb group if they occur anywhere before the last verb; otherwise, they are attached to the last verb. In the reverse transformation, all dependents of auxiliaries go back to the main verb. All the transformations reported in the paper prove helpful for the parser. In the case of verb groups, which is of particular interest here, LAS goes up slightly, by 0.14% (in a test set of 126k tokens).

Following up on the previous paper, Nilsson et al. (2007) investigates the same transformations applied to different datasets and under distinct parsing algorithms, to understand if they generalize across languages and parsing strategies. The representations for the different languages studied are similar to the PDT's representation. With respect to the structural transformations, the authors find that there are, again, small gains from converting the representations of coordination and verb groups. However, in their experiments, graph-based MSTParser (McDonald et al., 2006), unlike transition-based MaltParser, does not perform better on the transformed input.

Schwartz et al. (2012) is a systematic study of how representation choices in dependency annotation schemes affect their learnability for parsing. The choice points investigated also relate to the issue of headedness. The experiments look at functional versus lexical heads in six constructions: (1) coordination structures (where the head can be a conjunction or one of the conjuncts), (2) infinitives (the verb or the marker *to*), (3) nominal phrases (the determiner, if any, or the noun), (4) nominal compounds (the first noun or the last), (5) prepositional phrases (the preposition or its complement) and (6) verb groups (the main verb, or the highest modal, if any). Each combination of these binary choices is tested with 5 different parsers, which represent different paradigms in dependency parsing: MSTParser, Clear Parser (Choi and Nicolov, 2009), S_u Parser (Nivre, 2009), NonDir Parser (Goldberg and Elhadad, 2010) and Dependency Model with Valence (Klein and Manning, 2004). The edges in the representation are unlabeled, unlike the common practice in NLP.

The results show a learnability bias towards a conjunct in (1), a noun in (3), and

a preposition in (5) in all the parsers. Furthermore, a bias towards the modal heads in (6) and towards the head-initial representation in (4) is seen with some parsers. No significant results are found for (2). The authors also test combinations of the different headedness choices and show that gains are additive, reaching up to 19.8% error reduction.

In Ivanova et al. (2013), the authors run a set of experiments that provide a comparison of (1) 3 dependency schemes, (2) 3 data-driven dependency parsers and (3) 2 approaches to part-of-speech (POS) tagging in a parsing pipeline. The relevant comparison here is (1). The dependency representations compared are the BASIC version of Stanford Dependencies (SD, de Marneffe et al. 2006; de Marneffe and Manning 2008), and two versions of the CoNLL Syntactic Dependencies (Johansson and Nugues, 2007; Surdeanu et al., 2008). For all parsers and in most experiments (which explore several pipelines with different POS-tagging strategies), SD is easier to label (i.e., label accuracy scores are higher) and CoNLL is easier to structure (i.e., unlabeled attachment scores are higher). In terms of LAS, MaltParser has the highest score of all 3 parsers, in combination with SD, and MSTParser performs best with CoNLL.

A comparison between representations is also the theme of Elming et al. (2013), but with a view to extrinsic rather than intrinsic evaluation. The authors study how representation choice affects performance in downstream tasks, and conclude that different tasks may benefit from lexical or functional heads. This paper is reviewed in more detail in Section 5.3.

3.3 Structural transformations

The experiments in this chapter are based on transforming a UD-annotated data set by executing a series of tree-based operations that move and relabel edges. All the transformations studied here have the same underlying structure: they involve a content word which is a (phrasal) head by semantic criteria, and a function word which is a head by syntactic criteria.

In UD for English, typically these are structures such as (75a), where y is the

semantic head of a phrase, x is the functional head of the same phrase, and w is the phrase's governor.[3] In UD, the lexical head governs the lexical one, and that dependency is labeled with a dedicated type. The transformations I present reverse x and y's roles in relation to each other, and in relation to w, yielding structures such as the one schematized in (75b). They rely on the presence of dedicated dependency types for the relations between lexical and functional heads. In a dependency tree for a phrase with competing functional and lexical heads, I will call the word which is represented as head the **promoted head** (boldfaced in (75a)); that word will be attached to the governor of the construction. The other head is the **demoted head** (boldfaced in (75b)), and it will be attached to its promoted counterpart. So we have:

(75) a. $w \quad [x \quad \boldsymbol{y}]$ (functional)

b. $w \quad [\boldsymbol{x} \quad y]$ (complement)

In the simplest case, transformations of this kind can be inverted with no loss, which means the linguistic representation can be transformed for parser training, and the parser output can go through the inverse transformation for consumption by downstream applications. (This is the approach taken in Nilsson et al. 2006.) In other (common) cases, however, there may be important difficulties, which will be discussed in Section 3.3.3.

3.3.1 The CASE transformation

To illustrate in some detail, let us examine the case of prepositional phrases. Take, for example, the sentence in (76). The lexical-head representation, which UD adopts,

[3]Here I adopt the terminological convention introduced in Section 2.2.1: when the word 'head' is employed to refer to a phrasal head, I will use 'governor' to designate the word on which the head depends. This is to avoid confusion between two senses of 'head'. When there is no ambiguity in context, I will use 'head' to mean the parent of a node in a dependency tree, as is common practice in NLP.

chooses *life* as the promoted head, making *of* the demoted head, as shown in (77a). The functional representation, shown in (77b), reverses those roles.

(76) I found the love of my life.

(77) a. the love of my life

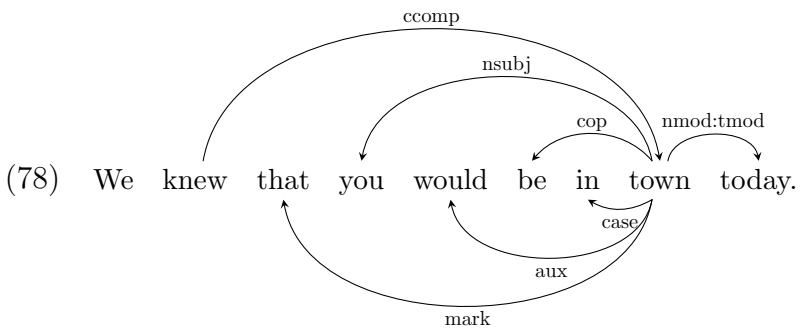
b. the love of my life

This is a particularly interesting example, because there is already evidence in the literature (Schwartz et al., 2012) that making prepositions heads—that is, adopting the functional-head representation for prepositional phrases—can yield higher parsing accuracy. This will be called the CASE transformation, because it targets the label `case`, used in UD for prepositions. (In English, that label is also used for the genitive marker *'s*, but here the transformation is not applied to that marker.) The other transformations are AUX, COP and MARK, named after the dependency types that label the functional heads they promote.

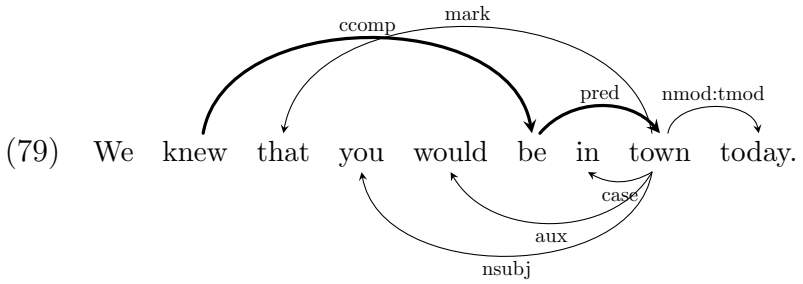
3.3.2 Other transformations

In (78) we find the UD representation of a sentence that has all the target constructions for which transformations are defined. The sentence exemplifies uses of the four labels `aux`, `case`, `cop` and `mark`. Each transformation generates a different tree for this sentence, as we will see.

It will be clear from the examples in this section that, when the functional head is promoted, the way in which the dependents of the (now demoted) lexical head are handled can have important consequences. Illustrated first are the simplest versions of each transformation, where no dependents of the demoted head are moved. In Section 3.3.3, alternatives will be discussed.



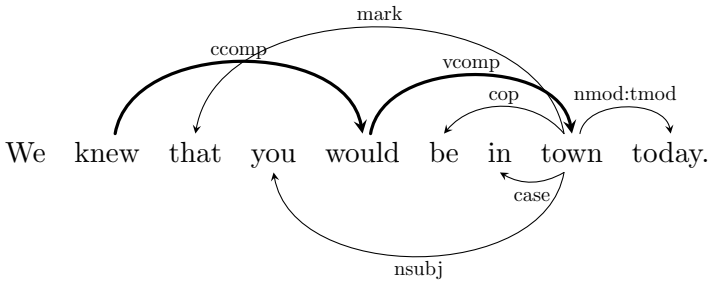
The COP transformation The label `cop` is used for the verb *be* in copular clauses. In relation to other dependency schemes, UD makes a distinctive choice here, discussed in Section 2.4.3: instead of attaching the subject and other clausal dependents to the copular verb, and making the predicate itself a dependent of that verb, the representation takes the nonverbal predicate as the head of the clause, governing the verb and predicate-level dependents. This representation allows copular clauses to be treated uniformly in languages with and without overt copulas. In (78), the predicate is a prepositional phrase, but since those are also represented with lexical heads, the head of the entire copular clause is the noun *town*. Note that even the auxiliary is attached to the predicate rather than the copular verb. The simple COP transformation, in which none of the dependents of the lexical head are moved to the functional head with its promotion, yields the tree in (79).



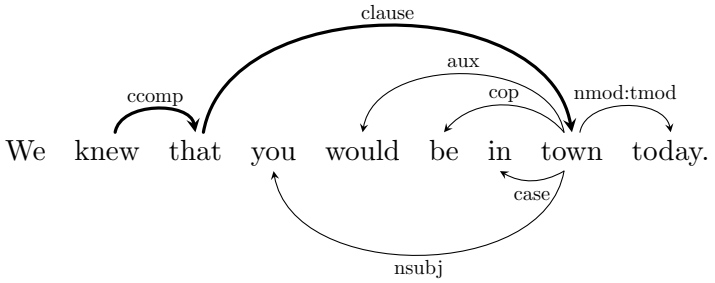
The AUX transformation In English, the label `aux` is used to attach modals and traditional auxiliaries. In the case of the auxiliary *be* in passive formation, the label `auxpass` is used, to encode voice information directly in the dependency tree. The AUX transformation is also used for `auxpass` dependencies. In order to avoid making

the transformed representation artificially easier by eliminating this voice distinction, the complements of `aux`-labeled words are labeled differently than the complements of `auxpass`-labeled words.

As mentioned above, these dependents are always attached to the predicate, which is why here the head of *would* is *town* and not *be*. The simple AUX transformation results in the tree depicted in (80).

(80) 

The MARK transformation The label `mark` is used for subordinating conjunctions in embedded clauses, and additionally for the infinitival marker *to*. It is always attached to the predicate, much like `aux`. The yield of the simple MARK transformation is illustrated in (81).

(81) 

Note that, in all cases, the labels used for the demoted head in the transformations are not part of the UD label set.

3.3.3 Handling of dependents in transformations

The examples of simplified transformations given above make it apparent that transformations can introduce undesirable nonprojectivity ((79) and (80)), and may sometimes result in representations that are linguistically objectionable—such as the adverb attachment in (79). Both of those are reasons why it may be desirable to move

the dependents of the lexical head when it is demoted. But exactly which dependents to move is an important question, due to the fact that modifier attachment in a dependency representation can be inevitably ambiguous, as shown below.

Attachment ambiguities inherent to UD

The fact that UD does not, for the most part, capture the distinction between head-level modification and phrase-level modification has important consequences. The issue is determining the level of attachment of a dependent and whether it needs to be moved or not. In the light of a theory of syntax in the style of Government and Binding Chomsky (1981), in which lexical structures are dominated by layered functional structures, one may argue that no two constituents share the same functional head. However, it is clear that the same lexical item can be the lexical head (that is, the semantically most prominent word) of multiple nested constituents. These distinctions are often very subtle and irrelevant for practical applications. While there is much debate in theoretical syntax about the attachment sites of different types of adverbs, especially in the Cartography program (Cinque and Rizzi, 2008), such distinctions have not concerned most NLP researchers.

Nevertheless, UD's radical adoption of lexical heads creates some situations where distinctions in attachment level are clear and very meaningful. The most obvious case is probably that of nominal predicates in copular clauses. In UD, we have trees like (82a) and (82b).

(82) a. She was just a little girl at the time. (nmod)

b. She was just a little girl with red hair. (nmod)

In (82a), the prepositional phrase is a modifier of the predicate. But in (82b), clearly the modifier is in the nominal domain. In UD, the head is the noun *girl*, because it is both the head of the nominal constituent, and the head of the clausal constituent (since it is the lexical head of the copula). In some cases, UD offers

an opportunity for disambiguation in the type system, by means of the **adnominal** attribute. Clausal modifiers of a nominal predicate, for example, are labeled differently if they attach at the clause level or below it. Clausal dependents of a noun are typed `acl` (83b), but clausal dependents of a predicate are typed `advcl` (83a). Prepositional phrases, nonetheless, are uniformly labeled `nmod`.[4]

(83) a. She was just a little girl when I met her parents. (advcl)

b. She was just a little girl who loved to read. (acl)

However, for other types of predicate (non-nominal and non-verbal), this distinction does not apply. With adjectival predicates, clause-level and below-the-clause attachment is indistinguishable for clausal dependents.

(84) a. I was ready for the party. (nmod)

b. I was ready before your arrival. (nmod)

c. I 'll be ready to go. (advcl)

d. I 'll be ready when you want to go. (advcl)

[4]There is a possibility that a distinction with respect to **adnominal** will be made for `nmod` in future iterations of the UD guidelines. However, this distinction does not solve similar challenges with, for example, adjectival predicates, as exemplified in (84). It may be the case that UD needs a systematic distinction between predicate-level modifiers and below-the-predicate modifiers, but it is unclear how to implement it in a precise manner.

In (84b) and (84d), the `nmod` and `advcl` edges correspond to modifiers that attach at the clause level. In contrast, in (84a) and (84c), `nmod` and `advcl` attach within the adjectival phrase. This is reflected in the contrasting effects of fronting each modifier, shown in (85).

(85) a. * For the party, I was ready (nmod)

b. Before your arrival, I was ready (nmod)

c. * To go, I 'll be ready. (advcl)

d. When you want to go, I 'll be ready. (advcl)

This pervasive ambiguity is actually a consequence of the choice to represent lexical heads. If functional heads were promoted, the clausal constituent would have a head (the copular verb) distinct from the predicate. Consequently, attachment below or at the clause level would be represented differently. The functional-head representation would create a possibility for disambiguation, as shown in (86).

(86) a. She was just a little girl at the time. (nmod)

b. She was just a little girl with red hair. (nmod)

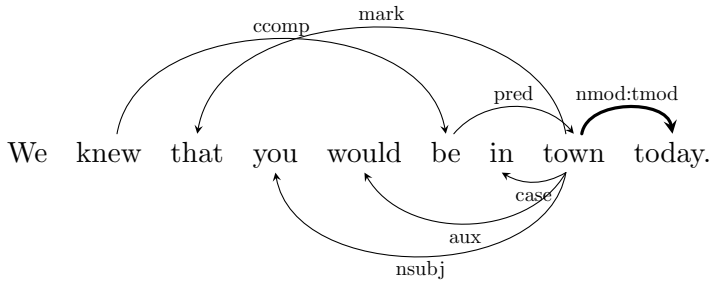
This poses a problem in the context of the transformations studied here, because when moving from lexical heads to functional heads, they go from a structurally ambiguous representation to a structurally non-ambiguous one. It is not necessarily

simple, or possible, to resolve the ambiguity in order to obtain the correct parsing representation. (The same issue arises with coordinated constituents in Nilsson et al. 2006.) The dependents of a lexical head cannot be blindly reattached to a promoted functional head in transformations, and careful handling of dependents may be necessary. In summary, there is some subtle linguistic reasoning involved in making defensible attachment choices, and this presents difficulties for automatic transformations that affect how such choices are represented.

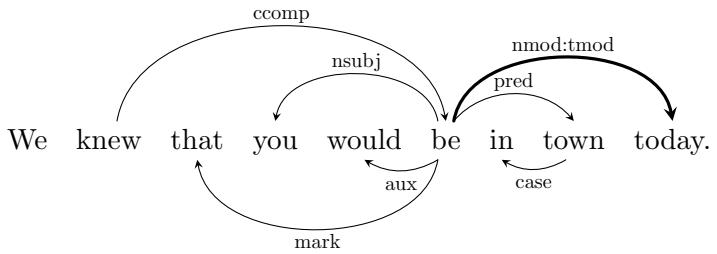
Introducing handlers

In an attempt to address these difficulties, 3 versions of each transformation were designed and tested. In the SIMPLE version, which was illustrated in Sections 3.3.1 and 3.3.2, none of the dependents of the lexical head are moved when the functional head is promoted. In the FULL version, all dependents of the lexical head are moved, except those with the [+**adnominal**] feature (`amod`, `acl`, `appos`, `det`, `case`, `nummod`). In the PARTIAL version, which is doubly virtuous in that it minimizes nonprojectivity and is closest to most current-day practice of syntax, all dependents of the lexical head which occur to the left of the functional head (roughly subjects and high adverbs) are moved when that head is promoted, and all other dependents are left attached to the lexical head. So now for each transformation P, we have P_{S}, P_{F} and P_{P}. To provide a comparison with COP_{S}, repeated below in (87a), COP_{F} and COP_{P} are illustrated in (87b) and (87c), respectively.

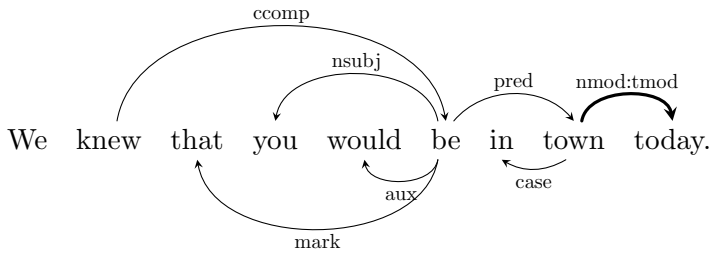
(87) a. COP_{S}:



b. COP$_{\text{F}}$:



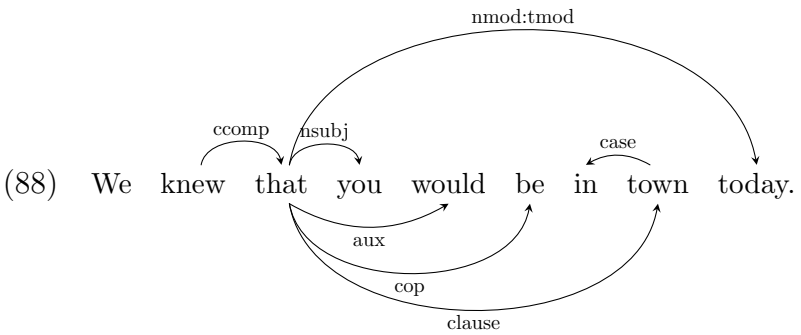
c. COP$_{\text{P}}$:



It should be noted that dependents which are known to always attach to heads rather than phrases are never moved—these are `mwe`, `compound`, `goeswith`, `name` and `foreign`. These dependents are always associated with particular tokens, not with phrases.

In COP$_{\text{F}}$, *today* is moved and becomes a dependent of *be*, the promoted head; in contrast, in COP$_{\text{P}}$, that dependent remains attached to the lexical head *town*, since it does not occur to the left of the promoted head. If the sentence were *We knew that today you would be in town*, these two transformations would have identical results.

For the higher heads, namely `aux` and `mark`, there is another important distinction between the FULL and PARTIAL handlers. In the FULL version, when a function word is promoted, any sister function words lower than it will be moved from the demoted lexical head to the promoted functional head. In the case of `mark`, for example, we would have a tree as in (88).

(88) 

Linguistically, this choice is hard to defend. In a sentence such as (88), the appearance of the lower function words is not conditioned on the presence of the higher one. If we think about these functional dependents not as prototypical dependencies but as transfer-style relations, in the manner of Tesnière (2015), then (again) it is hard to justify breaking them up. However, I will use this transformation (and others with analogous problems) in the spirit of investigating what may be helpful for parsing, knowing that the final output will be reverted to UD.

3.3.4 Recovering UD

The goal of producing an intermediary representation for parsing UD raises the question of whether this transformation is invertible. So far, we have discussed the difficulties in moving away from UD; now we turn to the difficulties of making the roundtrip. In this section, this question will be given an abstract answer, relevant for perfectly UD-compliant annotation. In the next section's discussion of the experiment data, a more practical answer, taking into account annotation errors and accommodations in the data, will be offered as a complement.

We have already seen that, when a functional head is promoted, the dependents of the lexical head which modify the entire phrase, rather than only the lexical word, must be moved to the new head. For the transformation to be invertible, it must be possible to move those dependents back. This introduces the following question about any dependent of the promoted functional head: is it the case that it modifies the function word directly, and must therefore remain attached to it, or is it the case that it modifies the phrase headed by that word, and must now be moved to the

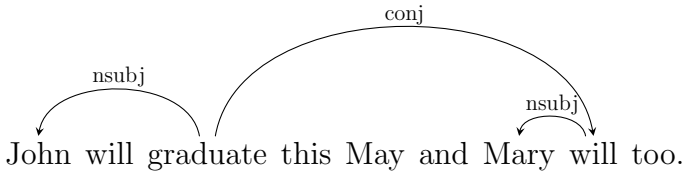
phrase's new head?

The difficulty of this question is (at least in theory) mitigated by UD's strong stance on the status of function words. In general, UD does not allow function words to take dependents, as discussed in Section 2.4.1. However, there are four exceptions to this.

Dependents of function words in UD

Multiword expressions Two exceptions are essentially irrelevant for present purposes. First, the use of the `mwe` relation to represent multiword function words is perfectly acceptable. This is not important; no transformations ever move this type of dependent, as mentioned in Section 3.3.3.

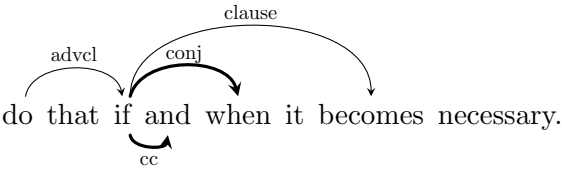
Promotion by head elision Second, function words may undergo promotion by head elision (see Section 2.4.3), in which case they essentially take on the role of a missing lexical word in a structure. This is also not important for my structural transformations, because in these situations the function words do not have dependency labels characteristic of function words, and therefore are not targeted by the transformations. Example (89), repeated from (40), illustrates this: the auxiliary *will* is promoted.

(89) 

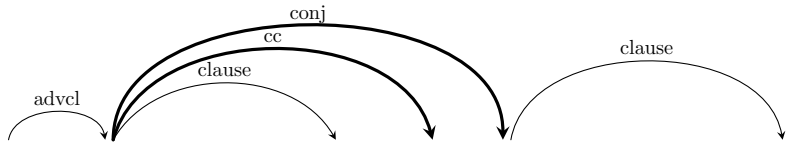
The two remaining exceptions are relevant in that they affect the invertibility of the structural transformations studied here.

Coordinated function words One is the case of coordinated function words; coordination can apply to any word category, and function words are no different. Contrast the coordination of complementizers in (90a) with the coordination between clauses headed by complementizers in (90b). The dependency trees are encoded in a MARK representation.

(90) a. I will do that if and when it becomes necessary.

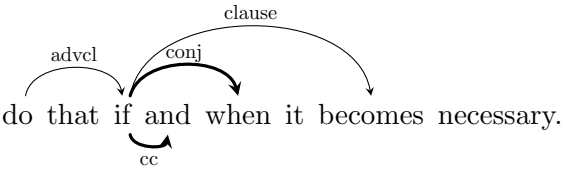


b. I will do that if it becomes necessary and when the right time comes.

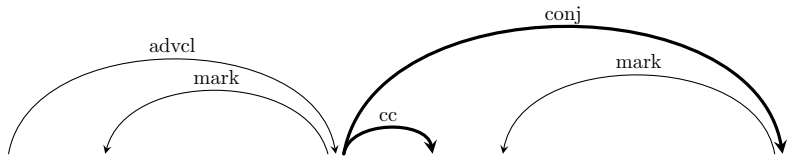


In the MARK representation, the two levels of coordination are ambiguous. This is just another example of level-attachment ambiguities intrinsic to dependency representations, but now the difference is that this particular ambiguity surfaces in a functional-head representation (90), but disappears in a lexical-head one (91).

(91) a. I will do that if and when it becomes necessary.



b. I will do that if it becomes necessary and when the right time comes.



The crucial problem here is that the dependents labeled `cc` and `conj` should not be moved going from (90a) to (91a); but they must be moved when going from (90b) to (91b). Perfectly inverting the MARK transformation, therefore, requires making this distinction. While it is possible to adopt heuristics, the distinction ultimately requires human judgment, because trees such as (90a) are systematically ambiguous between a head-level coordination analysis and a promotion-by-head-elision analysis, under which *when* would stand for an ellided clause. For this reason, there is always some uncertainty associated with moving coordinated function words to a lexical-head representation.

Light modifiers Finally, the fourth exception to the UD principle of not attaching dependents to function words is the attachment of negation and some light adverbials that can attach to `mark`-typed dependents. The `neg` relation can attach to a node of any other type, including functional types. The `advmod` relation, while generally reserved for dependents of predicates, is sometimes analyzed as dependent of a complementizer, as in (92).

(92) Just when you thought it was over, it started all over again. (advmod: when → Just)

When such an edge appears in a functional-word representation where *when* is a promoted head, it may not be possible to determine with certainty whether the adverb attaches at the clause level or at the head level, which makes a transformation into such a representation more difficult to invert.

The discussion here has been based on the UD guidelines, and applies to an idealized implementation of UD. The description of the experimental setup in Section 3.4.2 includes some consideration, both qualitative and quantitative, about the ways in which each dataset utilized introduces complications for transforming the representation in either direction.

3.3.5 Stacked transformations

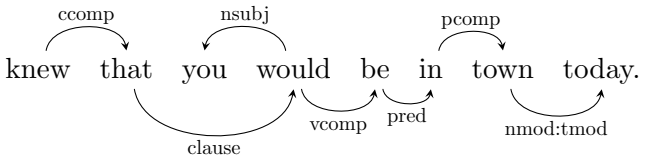
Each transformation described above affects only one type of functional head. There is no reason why these transformations should not be used together; in fact, if they are beneficial for parsing in isolation, it may very well be the case that they will be even more so in conjunction.

There are many ways to combine the different transformations; one has the option to use some or all of them, and with different dependency handlers. Additionally, applying them in different orders can yield different results.[5] Here I use an outside-in

[5]The results of stacking transformations in different orders are wildly different especially with FULL handlers. Stacking MARK_F, AUX_F, COP_F and CASE_F in that order (outside-in) results in (93a); stacking them in the opposite order gives (93b). The reason for this is that the FULL handling always moves dependents, so stacking the transformations creates a snowballing effect whereby the dependents moved in all the earlier transformations end up attached to the last head targeted.

ordering of these transformations. This usually results in a linguistically defensible representation: in a sentence using dominant English word order, subjects will attach no higher than auxiliaries, and adverbs will attach to the functional head immediately below them or to predicates. The results are exemplified in (94).

(94) We knew that you would be in town today.

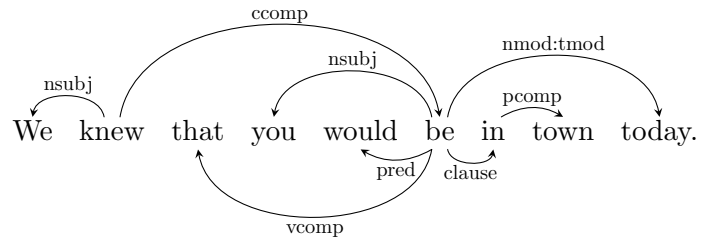


3.4 Experiments

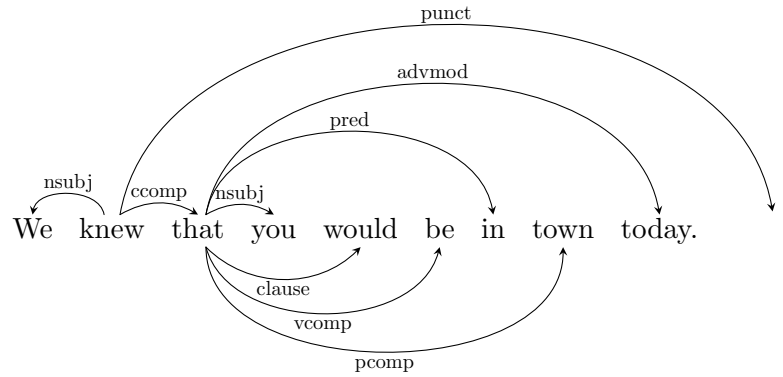
Each experiment in this chapter compares the accuracy of parsers trained on different versions of the same data: one annotated in UD, and one automatically converted to a parsing representation. The experiments are designed to shed light on two main questions: (1) which representations are useful; and (2) how that usefulness varies with same-language data sets, different languages, and different parsers.

There are four sets of experiments; in each set, 15 models trained on different

(93) a. We knew that you would be in town today.



b. We knew that you would be in town today.



representations are compared to a baseline model, holding constant a parser and a data set. The first set was performed with MaltParser (Nivre et al., 2007) on the EWT. Across different sets, 3 contrasts in in types of data and parser are provided. In order to compare results along the dimension of different languages, the French treebank from the UD project was used in conjunction with MaltParser in a second set of experiments. To provide a contrast along the dimension of parsing algorithms, a third set was produced by using MateParser (Bohnet, 2010) with the EWT data set. Finally, to show a within-language comparison of two distinct data sets, MaltParser experiments were also run with the WSJ corpus, which differs from the EWT in two important ways: it comprises a different genre of English, and it was converted automatically to dependency annotation from gold-standard phrase-structure trees, with no manual checking.

3.4.1 Parsers

The version of MaltParser used was 1.7.2.[6] For MateParser, version 3.6.1[7] was chosen, as it is a graph-based parser implementation. Graph-based parsing stands in contrast with transition-based parsing, represented by MaltParser, as another major paradigm of data-driven learning for dependency parsers. The crucial difference between these two paradigms is that, while a transition-based system is based on a state machine and learns to score transitions between states, a graph-based system learns to score dependency graphs and combines subgraphs to produce a maximum-scoring result. Kübler et al. (2009) is a good source for more details on these two parsing frameworks.

A concern with this type of experiment is that the default settings of an off-the-shelf parser may be implicitly biased towards the representation that has been typically used to demonstrate its usefulness. It is important to explore different hyperparameters and feature sets, to make sure that, in each case, a the parser model being tested is suitable to the particular representation. This is especially true in the case of MaltParser, which offers much flexibility in choice of parsing algorithms and feature templates for transition learning.

[6] http://maltparser.org/download.html
[7] https://code.google.com/p/mate-tools/downloads/detail?name=anna-3.61.jar

In the case of MaltParser, optimization was performed by use of MaltOptimizer (Ballesteros and Nivre, 2012), a tool which uses heuristics to arrive at good choices of parsing algorithm, feature set, and hyperparameters for MaltParser on a given data set. In every experiment with the parser, this tool was used on each version of the training data, with the corresponding development set used for validation. This generated 16 different MaltParser models: one for the baseline, and one for each of the three versions of the five transformations. The metric being maximized was LAS excluding punctuation, a widely used metric that seems to be the crucial measure of performance for most client applications.

There is some variety in the parser options arrived at (especially for the EWT). Some things are worth noting: the *stacklazy* algorithm dominated results for the EWT, while *stackproj* and *nivreeager* were most used in the WSJ and French data, respectively; in the EWT, pseudo-projectivization was used almost exclusively in the SIMPLE-handling representations (the one exception being COP_F), but it was used in 15 of the WSJ data sets and 12 of the French ones.

In the case of MateParser, there is no such tool, but the parameter space to explore is much smaller. Unlike the transition-based counterpart, MateParser does not offer a choice between different parsing algorithms or feature templates. There is only one hyperparameter to tune, the projectivity threshold, which is explored in some detail in Bohnet (2010). In order to find a good setting for this, I trained multiple models with different thresholds on part of the training set, and evaluated them on the development set of each corpus. These preliminary models were trained on 6000 randomly drawn sentences from each training set, with threshold values between 0.0 and 1.0, at 0.1 increments. The best threshold was then used to train the final model on all the training data, for each of the 16 representations. There was some variation in what turned out to be the best threshold for each representation, but all values were between 0.0 and 0.5.

3.4.2 Data

An important part of understanding parsing accuracy for different transformations is understanding how those transformations affect the data. Each one targets a different construction, and the frequency of those in each data set varies. Additionally, there may be annotation errors or conventions that interact with the transformations in different ways. In this section, I discuss aspects of how the 15 transformations affect each data set.

		FULL			PARTIAL			SIMPLE		
		one	*round*	*nonpr*	*one*	*round*	*nonpr*	*one*	*round*	*nonpr*
AUX	**EWT**	78.88	100.00	6.12	86.40	100.00	5.60	91.63	100.00	41.58
	French	88.16	100.00	14.68	92.07	100.00	12.55	95.26	100.00	45.88
	WSJ	84.14	100.00	0.98	89.18	100.00	0.44	93.64	100.00	49.22
CASE	**EWT**	78.22	100.00	9.81	80.82	100.00	5.26	81.63	100.00	10.55
	French	63.62	99.75	23.03	66.65	99.79	13.10	67.03	99.79	18.58
	WSJ	74.79	99.97	12.43	78.17	99.98	0.69	78.71	99.98	6.30
COP	**EWT**	88.45	100.00	6.63	91.67	100.00	6.33	95.07	100.00	31.52
	French	92.47	100.00	17.40	94.66	100.00	12.84	96.76	100.00	39.64
	WSJ	93.87	100.00	1.78	95.53	100.00	0.32	97.53	100.00	23.75
MARK	**EWT**	83.30	99.99	5.88	91.61	99.99	5.25	91.79	99.99	7.53
	French	92.97	99.93	13.33	96.54	99.94	12.91	96.57	99.94	13.46
	WSJ	87.29	100.00	0.85	93.53	100.00	0.33	93.70	100.00	3.40
ALL	**EWT**	42.67	99.93	12.42	55.31	99.99	6.66	61.87	99.99	57.09
	French	44.66	99.65	29.93	51.93	99.71	13.89	56.28	99.71	68.34
	WSJ	50.11	99.93	15.39	59.75	99.98	0.86	64.72	99.98	63.58
UD (*nonpr*)		**EWT** 5.24		**WSJ** 0.30		**French** 12.47				

Table 3.1: Statistics about parsing representations across data sets. AUX, COP, MARK, CASE, and ALL designate the heads targeted in the experiment; FULL, PARTIAL and SIMPLE are the dependent-handling strategies used. For each transformation, *one* is the percentage of dependency edges *unchanged* in a one-way transformation, *round* is the percentage of edges *unchanged* in a roundtrip transformation, and *nonpr* is the percentage of nonprojective edges in each representation. The baseline percentage of nonprojective edges in UD, which does not depend on transformations, is given in the last row.

EWT

The EWT consists of manually produced UD annotation for about 254k tokens. The annotation process and the provenance of this data is described in detail in Section 2.3.1. The version used here was v. 1.2 (Nivre et al., 2015a).

Data characteristics that affect invertibility In terms of the invertibility of the transformations, the EWT is the most suitable dataset. The reason for this is that, because I was directly involved in the production of the EWT, I personally fixed many annotation errors that affected the output of the structural transformations studied here. Annotation errors causing function words to have undue dependents were mostly cleaned up, although a handful remain in v. 1.2.

The EWT does, however, make a systematic and purposeful exception to the principle of not attaching dependents to function words, in addition to those already allowed by the UD guidelines. This additional exception applies in the case of sentences such as (95).

(95) Up to 40 rockets had been fired, weeks after the military withdrew from the territory.

In this case, we understand that *weeks* modifies *after*, by quantifying *how long after* the withdrawal event the firing of rockets took place.[8]

There are a few such examples in the corpus, and they are not correctly recovered in roundtrip transformations that make `mark` dependents heads and then again dependents. This is a very small source of inversion errors in the EWT data, as we will see next.

[8]This analysis is based on the following observations: The word *weeks* can be omitted (96a); it can occur in the presence of the adverbial *after* (96b), not requiring the entire adverbial clause; however, it cannot stand alone (96c).

(96) a. Up to 40 rockets had been fired, after the military withdrew from the territory.

b. Up to 40 rockets had been fired, weeks after.

c. * Up to 40 rockets had been fired, weeks.

Impact and invertibility of transformations The columns labeled *one* in Table 3.1 show the percentage of analyzed tokens (that is, token-label-head triples) in the training data that are *unchanged* by a transformation, for all 15 transformations; these numbers were obtained with the same evaluation script that is used to measure parsing accuracy, so they do not consider punctuation.

These counts make it clear that, in the case of CASE and MARK, there is little difference between the PARTIAL and SIMPLE transformations. That is because in these transformations, the corresponding lexical heads are unlikely (in English) to have dependents which occur to the left of the functional head.

Table 3.1 also shows the proportion of changes that are successfully recovered after each transformation, in the columns labeled *round*. For this purpose, the entire data set was transformed and then inverted. The numbers reported are an evaluation of the transformed-then-inverted data with respect to the gold standard in UD. This is exactly what we would have if a parser trained on a transformed representation achieved 100% LAS, and then its output was transformed back to UD and evaluated against the original gold standard. It is, in that sense, an upper bound on the post-inversion parser evaluation. These results create a picture of the extent to which limitations of invertibility can compromise the usefulness of using a parsing transformation, and it is clear that such limitations have a very limited impact on the results.

The transformations are also very different in terms of how much non-projectivity they introduce. Columns labeled *nonpr* in Table 3.1 show how that proportion changes with each transformation, which helps understand their performance. (These measures were obtained with MaltOptimizer.) In the transformations with SIMPLE handling, which do the least to avoid non-projectivity, a very high proportion of edges can become non-projective, and this degrades parser accuracy.

French UD treebank

The French UD treebank was automatically converted to UD from the French treebank v. 2.0, introduced in McDonald et al. (2013). The original data was annotated

manually in the SD style, and then harmonized with another three manually annotated and two automatically produced treebanks in other languages. The text comes from reviews of businesses, blogs, news stories, and Wikipedia entries. The conversion to UD was performed mostly automatically, with heuristic rules. The raw data has some modifications in relation to the original release: sentences with missing words were fixed, and the train/test/dev split was modified. The version used here was v. 1.2 (Nivre et al., 2015a).

Data characteristics that affect invertibility Not all the annotation conventions adopted for the French data align exactly with those used in the EWT. In addition, as mentioned above, I worked on eradicating annotation errors in the EWT that affected these transformations. No such step was taken for the French data, so it is only natural that there are more places in the French annotation where structural transformations fail.

Some examples of this are presence of more than one `case` dependent on either side of a head (97a), and a range of adverbs attached to `mark` nodes (97b).

(97) a. reproduite par Leonardo da Vinci

b. tout en conservant sa prononciation

Impact and invertibility of transformations Table 3.1 quantify the percent of tokens unchanged by each one-way structural transformation and their roundtrip counterparts, as well as non-projectivity. The patterns seen there are very similar to those occuring in the EWT data.

WSJ

This data set was produced by converting the WSJ constituent trees to UD with the Stanford converter (de Marneffe et al., 2006). As such, it is very consistently annotated, but it also contains some systematic errors not present in the EWT.

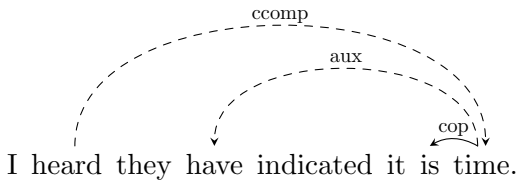
Data characteristics that affect invertibility Most errors result from having two `case` dependents on either side of a head, which the transformation does not recover.[9]

Impact and invertibility of transformations The changes resulting from each one-way structural transformation and the invertibility of those changes are quantified in Table 3.1. The same table shows that, in spite of very little non-projectivity in the WSJ data set (which is produced automatically by a converter that has very few rules yielding non-projective dependencies), it is still the case that the SIMPLE representations create a lot of non-projectivity.

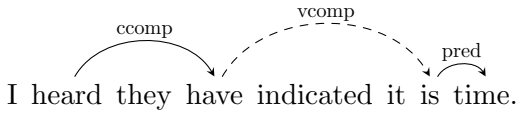
3.4.3 Evaluation methods

There are well-known problems with evaluating parsers across annotations, discussed extensively in Tsarfaty et al. (2012). An important problem for us is that some parser errors are penalized differently in a lexical-head or a functional-head representation. Consider the pair of wrongly parsed sentences in (98). The parser errors are the dashed edges.

(98) a. UD:



b. ALL$_P$:



In the ALL$_P$ parse, there is one incorrect edge; in the lexical-head version, there are two. The error is conceptually the same: both parses lead to the interpretation

[9]This is an implementation issue; it is possible to recover the structures correctly. But since this is not a dependency pattern that is expected in UD-compliant annotation of either English of French, I did not put time into handling this corner case.

that *the speaker has heard that is it time*. The lexical head of the complement of *heard* is identified as *time* and not *indicated*. With functional heads, the `ccomp` edge does not touch the wrong lexical head, because the head of the complement clause is taken to be the auxiliary. In UD, the head of the complement clause is the lexical verb, so the `ccomp` edge touches it.

A misidentified functional head could similarly be double-counted in a functional-head representation, but in practice that is a much less common type of error. Given typical error patterns, functional-head representations are more forgiving of dependency parsers, and the same parse can have significantly different accuracy scores in a lexical-head versus functional-head representation. (In fact, this is seen in a set of experiments from Tsarfaty et al. 2012.)

To discount this, I interpret the score of each functional-head representation with respect to a comparable baseline. The comparable baseline is established by applying a transformation P to the output of a parser trained on UD and to the gold standard, and then evaluating the transformed parse against the transformed gold standard to set the baseline performance. This allows us to isolate the effect of the transformation on the learning from any biases in the evaluation metric.

Additionally, since my focus here is on investigating strategies that may improve parser performance for UD, rather than guiding the design of a new representation, results on UD itself are of special interest. These are obtained by transforming the output of a parser with the inverse of the transformation applied to the training data, and comparing that to the UD gold standard.

To summarize, each model is scored in two ways, sketched in Figure 3.1. For each parsing representation P, the P-native model is evaluated against a gold standard transformed into P, $gold_P$, and against the original gold standard, $gold_U$. These are the LAS_P^P and LAS_U^P scores, respectively. The UD-native model is also evaluated twice: against $gold_U$, receiving a LAS_U^U score, and against $gold_P$, obtaining accuracy LAS_U^P. For the LAS_U^P and LAS_P^U evaluations, the parser output has to be converted into the gold standard's representation. In order to understand how the UD-native model compares to the P-native model, I compare LAS_P^P to LAS_P^U, and LAS_U^P to

LAS_U^U. A positive difference $LAS_P^P - LAS_P^U$ means that training a parser on representation P is beneficial when that parser is evaluated against a P-represented gold standard. A positive difference $LAS_U^P - LAS_U^U$ means that P is beneficial for learning even when the parser is evaluated in UD.

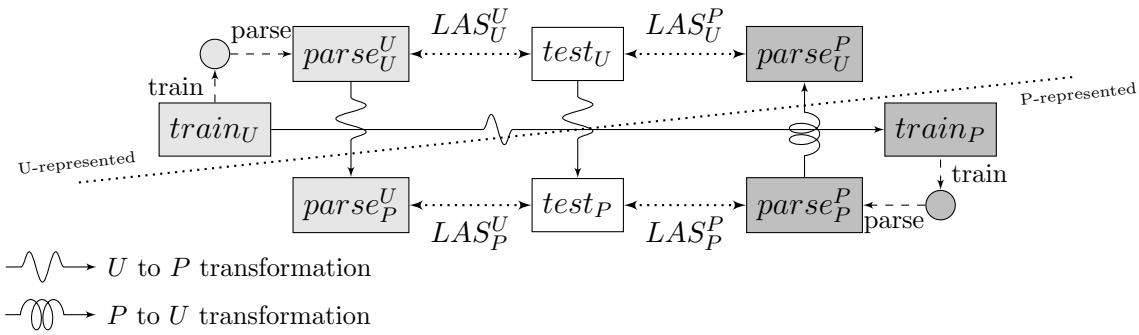


Figure 3.1: A diagram of the evaluation methodology described in Section 3.4.3. Let U be the baseline representation (UD), and let P be a parsing representation. Then for dataset *data*, $data_U$ is its representation in UD, and $data_P$, in P. If *data* was produced automatically by a parser, $data^U$ was produced by a U-native parser model, and $data^P$, by a P-native one. The light-shaded blocks comprise the U-native parser's pipeline; the dark-shaded blocks, the P-native parser's one.

3.4.4 Evaluation results

Results are shown in Table 3.2. The significance threshold is 0.05, with a Holm-Bonferroni adjustment for 15 experiments, within the experiments relative to each data set.

EWT and MaltParser Most significant differences to the baseline are negative. The exception is ALL_P, but the gain in LAS_P^P is lost in LAS_U^P. All in all, there is no significant gain for producing UD.

A different parser: Mate Mate's accuracy varies more with the representation than MaltParser's. There are significant positive differences in LAS_U^P, with MARK_P and ALL_P, reaching 0.65% in the latter case.

		FULL			PARTIAL			SIMPLE		
		LAS_P^U	LAS_P^P	LAS_U^P	LAS_P^U	LAS_P^P	LAS_U^P	LAS_P^U	LAS_P^P	LAS_U^P
AUX	**Malt**	84.94	−0.63*	−1.03*	85.0	−0.15	−0.38	84.95	−0.59	−0.66
	Mate	85.8	−0.47	−0.83*	85.95	−0.4	−0.51	85.84	−1.06*	−1.02*
	French	76.38	0.25	−0.3	76.48	0.81	0.55	76.43	0.49	0.27
	WSJ	89.97	−0.01	−0.24	90.01	0.15	0.01	89.99	−0.51*	−0.63*
CASE	**Malt**	85.09	−0.61	−0.83*	85.18	−0.07	−0.14	85.18	−0.04	−0.09
	Mate	85.73	−0.3	−0.58*	85.92	−0.04	−0.17	85.92	−0.28	−0.32
	French	75.52	−0.74	−1.16	76.22	0.31	0.17	76.29	0.61	0.22
	WSJ	89.95	−0.70*	−0.97*	90.06	0.42*	**0.35***	90.08	0.3	0.24
COP	**Malt**	85.15	0.0	−0.54	85.23	0.04	−0.34	85.02	−0.53	−0.74
	Mate	85.93	0.25	−0.39*	86.16	0.52*	0.35	85.9	−0.83*	−0.87*
	French	77.15	0.81	0.16	77.36	1.12*	1.0	76.72	1.06	1.0
	WSJ	90.14	0.0	−0.26	90.2	0.19	0.03	90.05	−0.42*	−0.48*
MARK	**Malt**	84.62	−0.45	−0.77*	84.96	0.13	0.14	84.96	0.33	0.26
	Mate	85.28	−0.08	−0.73*	85.76	0.60*	**0.56***	85.76	0.3	0.22
	French	75.51	0.29	−0.64	76.35	1.15	1.03	76.35	1.36	1.21
	WSJ	89.81	0.04	−0.27	89.98	0.33*	**0.28***	89.98	0.51*	**0.47***
ALL	**Malt**	85.07	−1.26*	−2.25*	85.63	0.38	−0.31	85.37	0.2	−0.13
	Mate	85.65	−0.77*	−2.07*	86.61	0.93*	**0.65***	86.21	−0.29	−0.41*
	French	75.56	0.08	−1.58	77.1	2.06*	**1.63***	76.53	1.69*	1.13
	WSJ	89.98	−0.95*	−1.61*	90.34	0.66*	**0.49***	90.2	0.49*	0.11
UD (LAS_U^U)		**Malt** 84.92		**Mate** 85.72		**WSJ** 89.97		**French** 76.42		

Table 3.2: Results across data sets. AUX, COP, MARK, CASE, and ALL designate the heads targeted in the experiment; FULL, PARTIAL and SIMPLE are the dependent-handling strategies used. The metric is always LAS. The LAS_U^U baseline accuracy, which does not depend on transformations, is given in the last row. LAS_U^P and LAS_P^P are represented as differences relative to LAS_U^U and LAS_P^U, respectively. Differences marked with * are significant at the $p < 0.05$ level, with a Holm-Bonferroni adjustment for each data set. Positive significant differences in LAS_U^P, which correspond to gains from using a parsing representation, are bold-faced.

A different data set: WSJ In this second English data set, results are again divided, but half of the transformations offer gains that carry over to LAS_U^P; the highest is 0.49%.

A different language: French All significant differences are positive and carry over to UD. Strikingly, the highest improvement in LAS_U^P is of 1.63%.

3.5 Discussion

This section addresses trends in the results and parser error patterns in some experiments with significant differences.

An interesting generalization is that the significant results tend to be consistent: if a parsing representation brings a significant difference with one data set, other significant differences have the same sign. The only arguable exception to this is ALL$_S$, which creates a positive difference in LAS_U^P for French and the WSJ, but a negative difference in LAS_P^P for Mate.

As a rule, PARTIAL handling is the best strategy for moving dependents; FULL handling (which greatly increases nonprojectivity) is almost always harmful.

LAS_P^U scores are consistently higher than LAS_U^U, which shows that simply moving a UD parse to a functional-head representation creates a nominal increase in accuracy, as discussed in Section 3.4.3.

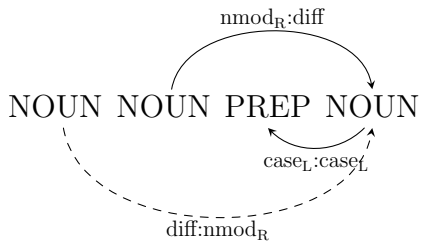
Relatedly, LAS_U^P scores are lower than LAS_P^P scores, due to parser errors being double-counted, in the manner described in Section 3.4.3. Similarly, errors can be propagated when a dependent correctly attached to a functional head is moved to a wrongly identified lexical head. For example, if a copula is head of an incorrect predicate, then a subject that was correctly attached to that copula will be moved to the wrong predicate in the conversion to UD, introducing an error.

3.5.1 Characterizing errors

In order to characterize parsing errors made on the different representations of the EWT, I used a graph mining approach for error analysis, following Kirilina and Versley

(2015). Each dependency tree was transformed to produce two graphs: one in which the nodes were universal POS tags (instead of words) and another in which the nodes were unlabeled. Dependency labels were also extended with the direction of the dependent (to the left or to the right of the head). The edge labels in the resulting graphs are a concatenation of the parser-assigned label and the gold label, substituting `none` when the edge is present in one tree but not the other. The resulting graphs, which represent the parse for the entire corpus annotated with its errors, were mined for frequent subgraphs with gSpan (Yan and Han, 2002).

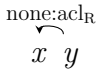
(99)



In (99) we have four tokens in a dependency graph. Each edge has two labels here, joined by a colon. The first side refers to the output of the parser, and the second side refers to the gold standard. Words are represented by their POS tags, so that frequencies can be aggregated for different words of the same POS tag.

We see that in this example, parse and gold standard agree as to the attachment of the preposition. However, the label `diff:`nmod$_R$ on the last noun indicates that, while both parses use the same dependency type for the label, this edge exists only in the gold standard (represented by the second part of the label), and the parser output contains a similar edge with a different governor. Conversely, the edge labeled nmod$_R$`:diff` exists only in the parser output, while the gold standard contains a similar edge with a different governor.

(100)



Example (100) shows a single-edge error, not constrained by POS tags: where the gold standard shows an `acl` edge between two tokens, the parser output recovers no

edge at all. The label `none` on the parser side is different than the label `diff` above: it indicates that the edge going into y in the parser output not only has a different governor, but also a different label.

Because of the way the graphs are constructed, these subgraphs represent patterns of errors over dependency labels or dependency labels and POS tags. I then compared the frequency of error patterns in the UD baseline and in the output of a parser after its transformation back to UD. I chose some intriguing significant results to perform error analysis on: the performance of ALL_F and ALL_P, which yield, respectively, the best and worst results on the UD representation in all sets of experiments; and the contrast between CASE_F and CASE_P in the WSJ set, where the former has negative impact, but the latter, positive.

Errors in ALL_P This is the best-performing transformation in all data sets. French improves in unique ways: root identification improves by much more than in the English data (from 193 correct edges in the baseline to 244). The attachment of `nmod` also improves from 51 errors in the baseline to 25.

In EWT $\times$ Mate, root identification is no longer a noticeable source of performance differences, but the baseline's 245 `nmod`-attachment errors fall to 216. The system is slightly better with the `nsubj` type, but makes more errors with `nsubjpass`.

In the WSJ, MaltParser produces more extraneous roots (words not attached by the end of the parse) in the ALL_P-native version: 369 to the baseline's 198. Despite this, gains distributed among a few types of errors amount to overall improvement.

Errors in CASE_F and CASE_P Previous literature indicates that making prepositions heads improves accuracy, but these two results show a more complex picture. In the WSJ, CASE_F hurts PP attachment: there are 658 such errors in the CASE_F-native parse, but 591 in the baseline. Prepositional complements are also more often wrong.

With the CASE_P transformation, the difference in `nmod`-attachment errors shrinks, and there is an improvement in `advmod`-attachment.

3.6 More languages

Since language comes out as the most important dimension of variation in determining whether or not a parsing representation is useful, I ran additional experiments with 4 more languages, focusing on the ALL$_{\text{P}}$ representation, which consistently has the best results in the previous experiments. The languages chosen were German and Swedish, from the Germanic family, and Italian and Spanish, from the Romance family. These experiments were run with MaltParser, optimized with MaltOptimizer (as described in Section 3.4.1), on the data from v. 1.3 of the treebanks (Nivre et al., 2016). The results are on Table 3.3.

	ALL-P		
	LAS_U^U	LAS_P^P	LAS_U^P
German	73.02	72.54	-1.77^*
Italian	88.12	88.62	-0.01
Spanish	79.58	82.50	**2.02***
Swedish	83.38	84.05	-0.21

Table 3.3: LAS_U^U, LAS_P^P and LAS_U^P for P=ALL$_{\text{P}}$ in four additional languages.

These are intriguing results: there is a large gain (2.02% in LAS_U^P) for Spanish, which is even larger than previously seen for French. In Italian and Swedish, the story looks similar to what we already saw for English: there is a gain in LAS_P^P, but it is lost when moving back to UD. In German, however, LAS_P^P is actually lower than LAS_U^U, and the roundtrip score LAS_U^P is even worse, 1.77% lower. This means that functional heads are worse for parsing in German, and the LAS_P^P score shows that this is the case even before we attempt a roundtrip conversion to UD.

This points to a large degree of language-to-language variation in whether a parsing representation is a suitable strategy. It is not clear how to predict that variation. French, Italian and Spanish are structurally quite similar, and yet the result we see for Italian is distinct. This may be due to differences in annotation; as will be discussed in detail in Section 4.5, the three languages make different choices about what verbs count as auxiliaries and copulas, with Spanish being the most liberal one and Italian

the most conservative. It may be that the use of a range of auxiliaries and copulas in Spanish is what determines when a parsing representation is useful: treating verbs as functional dependents works better when a small number of verbs are predictably labeled `aux` or `cop`, and this happens more consistently in Italian than French or, especially, Spanish.

That does not explain, on the other hand, why performance in German degrades with the functional-head representation, and more error analysis is needed to clarify this point.

3.7 The case for lexical heads

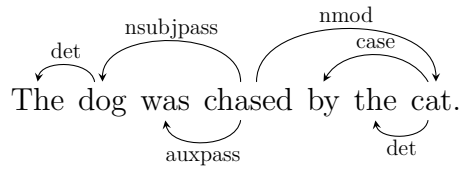
So far we have discussed mostly the idea of a parser-internal representation, taking for granted the idea that the outside-facing representation that we want to use in our NLP pipelines should be lexical-head-centered, as UD proposes. But functional-head representations are certainly not without merit. Representing functional heads brings appealing syntactic properties for language description. Osborne (2015) makes a good case that, in English, mostly choosing function words as heads produces a dependency tree such that all the constituents in the sentence appear as subtrees of the dependency tree, whereas lexical-head trees do not exhibit this property. This may very well be true across languages. The present chapter raised the additional argument that functional-head representations tend to be easier to parse, and that those gains cannot always be ported back to UD. They can also be less ambiguous, as shown in Section 3.3.3. Can we still defend the choice for lexical heads?

As argued in Chapter 2, languages vary in whether they use free or bound morphemes to express grammatical meanings. This has been reflected in historical approaches to Dependency Grammar: both Tesnière and Mel'čuk give special status to these free morphemes, function words, because across languages they stand in alternation with bound morphemes. In UD too, function words have special status: they are labeled with dedicated dependency types that identify them as functional heads, and they do not have their own dependents (for the most part). This approximation between function words and bound morphemes brings about a property that

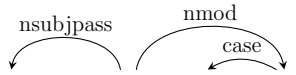
favors lexical-head representations over the functional-head alternatives: parallelism between languages is maximized. This has been an important motivation for the choice to represent lexical heads as governors of functional heads in UD, along with a belief that dependency trees with lexical heads are more useful downstream.

The notion of crosslinguistic parallelism between syntactic structures across languages is not always very precise. An illustration of this parallelism, repeated from (9) and due to Joakim Nivre, can be seen in the following Swedish-English pair.

(101) a. The dog was chased by the cat.

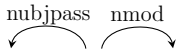


b. Hunden jagades av katten.

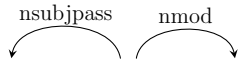


There is a parallel between the subtrees formed by the content words.

(102) a. dog chased cat



b. Hunden jagades katten



Whether we make a choice for lexical or functional heads, the dependency trees of the two sentences are not isomorphic under a word-to-word alignment. However, if we consider only the content heads in the sentence and align them across the two languages, then we see that this translation is dependency-preserving in UD. This is important because the content words and the relatons between them have the most to contribute for interpretation, especially the type if interpretation that current NLP applications focus on. (Circling back to Mel'čuk 1988, we can think of this as a deep syntactic structure in which function words are not represented.) The advantages of functional-head representations are not lost, because UD trees preserve information about the identity of functional heads, which allows almost all the same constituency information to be recovered.

3.7.1 Measuring parallelism

While the argument of crosslinguistic parallelism has been used before to justify the choice of lexical heads, so far it has relied on artificial and simplified examples, while the reality is that crosslinguistic correspondences are in practice very complex. There are no systematic studies of whether and how this parallelism arises in naturally occurring data, at any scale. This section offers such a study, at a small scale, along with a qualitative analysis of its results.

I produced a short gold-standard parallel corpus of dependency-parsed Spanish and English text, with a random sample of 50 sentence pairs from Europarl (Koehn, 2005). I hand-corrected parses on both sides and word alignments between them, produced with automatic tools, and then converted these hand-corrected annotations from UD with the ALL_P transformation.

In order to make a comparison, we need a measurable notion of parallelism. It seems that the notion of dropping function words from the tree, as illustrated in (102), is biased, because it already presupposes that function words contribute less and should be treated differently. One typical and very important way of using dependencies in NLP pipelines is to use **dependency paths** between words as features for the possible relation between those words. I propose to measure parallelism in a way that touches on this practical use of dependency paths: the parallelism between the dependency structures in two languages is the similarity of the dependency paths between nominals aligned across those languages.

In more detail, I identified all the nominals in each sentence as the words labeled `nsubj`, `nsubjpass`, `dobj`, `iobj`, or `nmod`. For each pair of aligned words in each pair of sentences, I then identified the pairs such that both sides were nominals. These pairs of words with nominal labels are considered **aligned nominals**. Then, for each unordered pair of aligned nominals, I extracted the unlexicalized[10] dependency path between the words in the source language side and the unlexicalized path between the words in the target. I restricted the length of the paths with a parameter *level*. The value of *level* is the minimum distance between a nominal and its lowest common

[10]Another way to implement this metric would be to use the word alignment to align lexicalized dependency paths.

ancestor with the other nominal. (Nominals that are far apart in the tree are much less likely to be interesting for tasks targeting relations between entities.) The question then is whether the path between those nominals in the source language is identical to that between their aligned counterparts in the target language. Having identical paths can be useful in a setting where a system is learning from multilingual data, for example.

A note on parallelism between translations Before exploring the results, it should be noted Europarl is a corpus of natural translations, and as such they are not always literal translations that lend themselves neatly to parallel structure assignment, or even to word alignment. Even in this small example of 50 sentence pairs, there are several examples of significant structural differences between the two languages that are imposed not by grammatical differences, but by different discoursive strategies on the part of the authors. An interesting example is the sentence pair in (103).

(103) a. Me van a permitir que me detenga un momento sobre esta cuestión en la forma como ha sido propuesta por la Comisión para desarrollar una cooperación reforzada.

b. I would like to spend a little time on the Commission's proposals for developing enhanced cooperation.

In this example, there is a contrast between *me van a permitir* and *I would like to*; in the Spanish sentence, the speaker appeals to the audience for permission to detain themselves on an issue; in the English counterpart, the speaker simply states that they would like to do so. Additionally, where the English sentence has *on the Commission's proposals*, the Spanish one elaborates: *en la forma como ha sido propuesta por la Comisión*. A literal translation of the Spanish sentence would be closer to (104).

(104) You will allow me to detain myself for a moment on this issue of the way which was proposed by the Commission for developing enhanced cooperation.

3.7.2 Results and discussion

The results for 3 values of *level* are given in Table 3.4. UD does in fact perform better than the functional head representation on this metric, although it can sometimes introduce differences between paths that would be identical in the functional head representation, as I will discuss below. Even though Spanish is structurally very close to English, it is still the case that there is roughly 30% to 40% more parallelism with UD than with ALL_P.

level	*% identical in UD*	*% identical in* ALL_P	*total pairs*
1	65.5	50.9	110
2	44.4	31.9	270
3	34.9	24.5	384

Table 3.4: Percentage of nominal-to-nominal paths that are identical in the source and target language. The column *level* is a restriction on the maximum distance between a nominal on either side of the path and the two nominals' lowest common ancestor.

I examined the results for the case of $level = 3$. The mismatches in paths come from exactly 7 sentence pairs in the corpus, which are given in (105) through (111). (Some paths are discussed in more detail below; boldfaced words appear in those paths.)

(105) a. Los **talibanes utilizan** la **religión** como pretexto para quitar les todos los derechos a las mujeres.

b. The **Taliban are using religion** as a smoke screen to strip away all women's rights.

(106) a. Hemos reaccionado con gran rapidez y hemos enviado una señal inequívoca de nuestra intención de responder a cualquier **daño** que **pudiera ocasionar** la **aplicación** de la ley a intereses europeos.

b. We moved very quickly to give a very strong signal of our intention to respond to any **damage** which the **application** of legislation **could cause** to European interests.

(107) a. Esto muestra una extraordinaria señal de **solidaridad** de los Estados miembros existentes **para con** un **país** candidato pequeño.

b. This shows a remarkable sign of **solidarity** from the existing Member States **towards** a small **country**.

(108) a. La autoridad presupuestaria **será informada cuando** el perfil ejecutivo se **aparte** significativamente **de** el perfil propuesto.

b. The budgetary authority **will be notified when** the implementation profile **deviates** considerably **from** the proposed profile.

(109) a. Terminaré con esta observación y agradezco a todos aquellos de entre ustedes que han contribuido a hacer de este debate sobre el complemento financiero a el Cuarto Programa Marco, un debate de tanta calidad.

b. I shall finish on that note, and I thank all of you who have contributed to the high quality of this debate on the supplementary financing of the fourth framework programme.

(110) a. Queremos, en definitiva, que Europa pueda presentar se con una sola voz en todo lo que es el mercado interior y actuar como modelo—por qué no—en el concierto internacional.

b. To sum up, we want Europe to speak with a single voice with regard to the internal market and to act as a model—why not?—in international relations.

(111) a. Ni en América Latina, ni en Asia, ni en el conjunto de América se dan cita estos tres elementos.

b. Not even in Latin America or Asia or the whole of America do these three factors exist side by side.

Many of the differences arise where an English verb selects and auxiliary while the Spanish correspondent has an equivalent bound morpheme. This occurs, for example, in (105); the unlexicalized paths for one pair of nominals are shown in Figure 3.2. When the auxiliary *are* gets between the subject and the predicate in the ALL$_{\text{P}}$ representation, the paths from subject to object are no longer the same in the two languages.

$$\text{religión} \quad \xleftarrow{dobj} utilizan \xrightarrow{nsubj} \quad \text{Talibanes}$$
$$\xleftarrow{dobj} utilizan \xrightarrow{nsubj}$$

$$\text{religion} \quad \xleftarrow{dobj} using \xrightarrow{nsubj} \quad \text{Taliban}$$
$$\xleftarrow{dobj} using \xleftarrow{vcomp} are \xrightarrow{nsubj}$$

Figure 3.2: UD and ALL-P unlexicalized dependency paths, respectively, between *religion/religion* and *Talibanes/Taliban*, respectively, in example (105).

However, sometimes such differences are an artifact of the choice of how to distribute phrase-level dependents between lexical and functional heads, discussed in Section 3.3.3. For example, in (106), there are verb groups in both sentences: *could cause* and *pudiera ocasionar*. The paths between *daño/damage* and *aplicación/application*, shown in Figure 3.3, are only different because the Spanish subject *aplicación* is post-verbal and remains attached to the lexical verb, while the English subject *application* is pre-verbal and becomes a dependent of *could*.

Another less obvious source of differences in parallelism comes from nested functional heads. When functional heads are stacked over the same lexical head, the level of stacking may be different in the two languages, which will be reflected in the parallelism metric. Even though, in (107), both *para con un país candidato pequeño* and *towards a small candidate country* are prepositional phrases, in English there is

daño $\xrightarrow{acl:relcl}$ *ocasionar* $\xrightarrow{nsubj}$ aplicación

daño $\xrightarrow{acl:relcl}$ *pudiera* $\xrightarrow{vcomp}$ *ocasionar* $\xrightarrow{nsubj}$ aplicación

damage $\xrightarrow{acl:relcl}$ *cause* $\xrightarrow{nsubj}$ application

damage $\xrightarrow{acl:relcl}$ *could* $\xrightarrow{nsubj}$ application

Figure 3.3: UD and ALL-P unlexicalized dependency paths between *daño/damage* and *aplicación/application* in example (106).

a single prepositional head, and in Spanish there is a complex preposition or nested prepositions. I assigned the Spanish prepositions a nested analysis. In that case, the paths between *solidariedad/solidarity* and *país/country* are only the same in UD, as can be seen in Figure 3.4, due to the flat analysis of stacked functional heads. If we analyzed *para con* as a complex preposition (with the `mwe` relation), the paths would still be the same in the ALL_P representation.

solidariedad $\xrightarrow{nmod}$ *para* $\xrightarrow{nmod}$ $\xrightarrow{pcomp}$ *con* $\xrightarrow{pcomp}$ país

solidarity $\xrightarrow{nmod}$ $\xrightarrow{nmod}$ *towards* $\xrightarrow{pcomp}$ country

Figure 3.4: UD and ALL-P unlexicalized dependency paths between *solidariedad/solidarity* and *país/country* in example (107).

However, sometimes nested functional heads do not have an alternative analysis, as is the case of *will be* in (108) (shown in Figure 3.5).

autoridad $\xleftarrow{nsubjpass}$ *informada* $\xrightarrow{advcl}$ *aparte* $\xrightarrow{nsubj}$ perfil

autoridad $\xleftarrow{nsubjpass}$ *sera* $\xrightarrow{vpass}$ *informada* $\xrightarrow{advcl}$ *cuando* $\xrightarrow{clause}$ *aparte* $\xrightarrow{nmod}$ *de* $\xrightarrow{pcomp}$ perfil

authority $\xleftarrow{nsubjpass}$ *notified* $\xrightarrow{advcl}$ *deviates* $\xrightarrow{nsubj}$ profile

authority $\xleftarrow{nsubjpass}$ *will* $\xrightarrow{vcomp}$ *be* $\xrightarrow{vpass}$ *notified* $\xrightarrow{advcl}$ *when* $\xrightarrow{clause}$ *deviates* $\xrightarrow{nmod}$ *from* $\xrightarrow{pcomp}$ profile

Figure 3.5: UD and ALL-P unlexicalized dependency paths between *autoridad/authority* and *perfil/profile* in example (108).

Overall, it is clear that, even in this pair of very similar languages, the differences in

realization of grammatical elements as bound or free morphemes can lead to significant differences in parallelism of dependency trees. Even though functional heads have some advantages, as shown in this chapter, there is still a clear case for lexical heads when the goal is to preserve crosslinguistic parallels.

3.8 Conclusion

In this chapter, I motivated and presented a series of experiments designed to show the impact of the design of dependency representation on the accuracy of a parser. The issue investigated was, specifically, the choice of lexical versus functional heads in dependency structures—a design choice that sets UD apart from other popular dependency representations for NLP. The experiments covered two parsers, two languages, and two data sets for the same language, to create a more complete picture of the results.

In the process, I discussed in detail the transformations designed and the process of applying them: how each structural transformation affects the data, what challenges need to be addressed in order to arrive at a satisfactory procedure, and some differences in expressivity between the alternatives. I also showed that evaluating representations against each other is nontrivial, as has been noted in the literature (Tsarfaty et al., 2012). In the case of choosing between lexical- and functional-head representations, the LAS metric is in practice biased toward functional heads, as seen in several data sets from the fact that scores go up simply by transforming the data to be evaluated to a functional-head representation. We can expect parser accuracy on UD to be in general lower than on, for example, CoNLL, which mostly prefers functional heads; some of the accuracy difference is only nominal. This is consistent with findings in Ivanova et al. (2013) and Tsarfaty et al. (2012).

In general, the possibility and extent of accuracy gains is influenced by data choice, parser choice, and especially language. Gains of 1.63% in LAS for French, and of 2.02% for Spanish were obtained by using a parser-internal representation. In English, we observed that gains with MateParser were larger than with MaltParser, which raises the question of whether gains for French and Spanish might also be larger with

a graph-based parser. For users of UD, a parsing representation with functional heads is worth considering as a simple way to improve results.

This chapter also addressed the motivations for UD's lexical-head design, showing with a small experiment that this design does promote parallelism between languages. This shows that, even functional heads may be better for parser accuracy in some languages, lexical heads are still preferable in a multilingual setting, with a parser-internal representation when appropriate.

Chapter 4

Representing structure: Romance syntax

4.1 Introduction

This chapter focuses on the expressiveness of Universal Dependencies (UD, Nivre et al. 2016) as a linguistic representation, with special emphasis on its ability to represent predicate-argument relations. For that purpose, it presents an analysis of three aspects of Romance syntax and their representation in four UD v.1.2 treebanks (Nivre et al., 2015a): the annotation of SE, an enigmatic but ubiquitous morpheme that plays a controversial role in argument realization; the annotation of Romance complex predicates, which present a syntax/semantics mismatch that threatens to undermine the usefulness of UD's simplified representation; and the use of the labels `cop` and `aux`, which invite us to consider how the need for parallelism and the commitment to surface structural properties can be reconciled in making crosslinguistic recommendations. All of these are *loci* of inconsistency among the treebanks I examined, which shows that more attention needs to be given to these challenges.

These analyses address two crucial high-level questions for the representation, both directly related to important ways in which we expect UD to be practically useful: by serving as a source of clues about predicate-argument relations and by promoting crosslinguistic parallelism.

The first question, which is touched on in the discussion of complex predicates and, naturally, of the labels `cop` and `aux`, is how to make consistent decisions across languages about what counts as a function word. This is essential for promoting the kind of parallelism that we examined in Section 3.4, and yet it is very difficult to accomplish.

Relatedly, I ask to what extent we can reconcile UD's commitment to surface syntax—a requirement for producing UD automatically—and its inevitable concern for semantics—which in many cases is what Natural Language Processing (NLP) applications ultimately care about—when complications arise at the syntax-semantics interface. In these two cases, the relation between grammatical functions and semantic roles may be less than straightforward; by sketching and arguing possible analyses, I shed some light on the potential and the limitations of UD in that domain.

The remainder of the chapter is divided into four parts. The first section discusses the assumptions and criteria that will be used for deciding between alternative UD representations for a construction. The next two tackle SE and complex predicates. I give a descriptive introduction of the syntactic properties of these constructions, followed by a discussion of how they are annotated in the UD treebanks; drawing from discussions in the syntax literature, I argue for specific proposals for making the annotations consistent, and comment on the limits of the representation. The fourth part deals with the use of two functional labels from the UD type system in the four treebanks. The discussion is tied back to complex predicates, and I show how decisions about these labels have global implications for the four languages, ultimately making a recommendation for their crosslinguistic treatment.

4.2 Notes on analysis

This section discusses the analytic principles that guide the argumentation in the remainder of the chapter. Not all familiar methods of syntactic argumentation are necessarily relevant for UD. As discussed in Section 2.4.1, UD is not a generative grammar such as Government and Binding (GB, Chomsky 1981) or Lexical-Functional Grammar (LFG, Bresnan 2015). There are many commitments, both in terms of predictive

power and in terms of expressiveness of the representation, that UD does not explicitly make. Notably, UD is not a framework for modeling grammaticality judgments. This leaves room for deciding which phenomena need to be represented by UD, and some of the proposals in this chapter hinge on the assumption that certain properties do not need to be reflected in the UD representation.

4.2.1 From dependency types to semantic roles

As explained in Section 2.4.1, UD is meant to provide scaffolding for semantics, capturing syntactic information that NLP applications can use in understanding-oriented tasks. In the current chapter, I focus on the way this is implemented in the predicate-argument domain.

Predicate-argument relations are crucial for relation extraction, and in Linguistics they are often typed by what are called **semantic roles**. While UD does not encode them directly, these semantic relations are usually understood to have a tight (albeit not fixed, from the point of view that we take here) connection with the syntactic relations that UD does encode. This connection, described by what are called **linking theories** (e.g. Bresnan and Kanerva, 1989; Jackendoff, 1990; Van Valin and LaPolla, 1997), is organized by an individual predicate: each predicate's lexical entry is associated with some information about how the syntactic arguments of that predicate are mapped to the semantic roles that the predicate assigns. This information is the predicate's **argument structure**. The syntactic arguments are differentiated by their **grammatical functions**, and so we claim that a grammatical function realizes a semantic role in the context of a particular argument structure.

An approximation of linking in UD

In order for UD to serve as scaffolding for interpretation, it is important that, as much as possible, the semantic role assigned to an argument by a predicate be recoverable from the dependency trees, as predicted by linking theories. In UD, we can expect a semantic role to be inferrable from the triple (GP, GF, SF): **governing predicate**, grammatical function and **subcategorization frame**.

The governing predicate The first element, the identity of the predicate itself, reflects the lexicalist approach of UD: there is no requirement that semantic roles be determined from syntactic information alone. This assumption allows us to dispense with the notion of a direct correspondence between syntactic configurations and semantic roles that is consistent for all predicates; it also captures the expectation that information about predicate-argument relations can largely be apprehended from the surface of language. The identity of the predicate can account for contrasts such as the one illustrated in (112); the subjects of the two sentences do not have the same semantic roles, but we can predict which roles they receive from the differences between the two predicates.

(112) a. I ran.

b. I shivered.

The grammatical function The second element necessary for identifying semantic roles, grammatical functions, are the (core and noncore) argumental labels of the UD dependency type; for a multiargument predicate, grammatical functions serve the crucial purpose of coding the mapping between roles and arguments, which cannot be learned without them. I will consider as grammatical functions the core argument labels, but also the noncore labels `nmod` and `advcl`, augmented with the identity of their functional heads (prepositions or complementizers). Those functional heads also code role assignment, by introducing relational meanings that explain the argument's function with respect to the predicate. Grammatical functions establish the contrast between the two arguments of *killed* in (113): the argument marked as subject receives an Agent role, and the argument marked as object receives a Patient role.

(113) He killed the president.

The subcategorization frame Finally, the third element in the triple, subcategorization frame, makes an implicit reference to the underlying argument structure of the predicate. The observed subcategorization frame of the predicate is an important hint about that underlying structure. The contrasting roles of the subjects in (114)

can be distinguished when we take into account the subcategorization frame, which reveals differences at the argument structure level.

(114) a. Jack broke the window.

b. The window broke.

The subcategorization frame, for our purposes, will consist of the subset of dependents marked with grammatical functions that are selected by the predicate in quesiton. These dependents can include any core arguments, and also the `expl` label, because it signals that a particular syntactic position is not available for mapping. So, for example, in (114), we have two frames: $\langle nsubj, dobj \rangle$ and $\langle nsubj \rangle$.

There are well-known limits to the use of this triple, illustrated by (115). In this case, the subcategorization frame cannot distinguish between different possible underlying argument structures: one in which the predicate takes a Theme and one in which it takes an Agent. In both cases, the sole argument ends up realized in the same subcategorization frame: as a subject.

(115) a. The potatoes are cooking.

b. The chef is cooking.

My goal will be to offer annotation standards whereby this principle can be maintained whenever possible, that is, whenever the subcategorization frame is not inherently ambiguous with respect to the underlying argument structure. My working assumption here is that UD limits itself to representing a version of grammatical functions, as discussed in Section 2.4.1, but that some argument structure operations with morphological reflexes can also be encoded in the system.

4.2.2 Choosing between analyses

In discussing different alternatives for how to represent a construction, I take into account the fact that any use of a given label has implications about the properties attributed to that label. While UD does not model grammatical judgments, it does

model syntactic relations between structures; these relations, each with its characterizing properties, are drawn from a limited set that is meant to cover all observed structural relations across languages.

In order for the representation to support semantic interpretation and syntactic investigations, it is important that these types be informative. One important question to ask about any proposal to represent a construction is to what extent it preserves or challenges viable generalizations about the dependency types it employs. Diluting generalizations about types may not make UD less universal, but it makes it less useful.

In this chapter, the dependency types that I will be most concerned with will be those relevant for calculating semantic roles. Some important properties of these are outined next.

Some properties of key dependency types

Certain dependency types deserve special consideration when characterizing each element of the triple (GP, GF, SF). It is worth considering exactly what dependents should be taken into account for calculating semantic roles from dependency trees.

Characterizing the governing predicate In the simplest case, the governing predicate of an argument is merely its governor in the dependency tree. However, UD acknowledges the existence of complex lexical units, and this implies that a governing predicate can be, more than simply a token, a subtree consisiting of specific dependents.

In this case, all dependents that attach at the word level, forming complex lexical units (which include `mwe` and `compound`) should be considered when determining the identity of an argument's GP.

In addition, any dependents that signal argument structure operations with morphosyntactic reflexes should be included here. This is the case of `auxpass`, which is a reflex of an operation that systematically reorganizes argument structure for a productive class of predicates.

Characterizing grammatical functions For simplicity, this brief discussion is restricted to the nominal domain; in general, this chapter will not tackle clausal arguments.

In UD, grammatical functions of nominals can be core dependents, labeled `nsubj`, `nsubjpass`,[1] `dobj` or `iobj`, or argumental obliques, labeled `nmod`. In the case of the core dependents, the label itself is enough information for linking; but for oblique dependents, any `case` dependents under the argument must be included in the calculus as part of the grammatical function information. This is because these dependents encode relational information that is needed to identify the function of obliques with respect to a predicate.

Characterizing subcategorization frames The subcategorization frame of the predicate, which is a proxy for the predicate's argument structure, includes the list of the predicate's grammatical functions and additionally the dependency `expl`. This label, while not a grammatical function in itself, stands as a wildcard for any element that has the morphosyntactic properties associated with a particular grammatical function but does not receive a semantic role. Only a limited set of function words, licensed by certain predicates, can receive this label.

Excluding other functional dependents Another important characterization for this analysis is that of the `aux` and `cop` labels. These functional elements are expected to display a defining set of properties across languages: (1) they do not introduce new actions or states; (2) they add information that is grammaticalized in the language; and (3) they cannot have modifiers that do not also modify their heads. This has two important consequences: these types of dependents cannot themselves have dependents, and they cannot alter semantic role assignments.

Note that this excludes `aux` and `cop` from the calculus of semantic roles, which will be crucial in the discussion of complex predicates. This is an important generalization for UD because, in order for the idea of parallelism between lexical words,

[1] This is actually not a distinct grammatical function in English or Romance, and this label is likely to be removed from the UD type set in future revisions.

as developed in Section 3.4, to be meaningful for relation extraction, we have to assume that the functional-typed dependents which are factored out from those parallels do not interfere with predicate-argument relations. If they do, then the dependency triples common across languages cannot be assumed to correspond to the same predicate-argument relations.

4.2.3 Data

My data for the analysis of consistency comprises French, Italian, Portuguese and Spanish. Both the Spanish (423k tokens) and French (390k) treebanks originated from the Google Universal Dependencies project McDonald et al. (2013) and are composed of texts from blogs, news, web reviews and wikis. The Italian treebank (252Kk), which is a conversion of the Italian Syntactic-Semantic Treebank (Montemagni et al., 2003), also has news and wiki text, in addition to legal-domain documents. The Portuguese treebank (226Kk), a conversion of the data in Bosque (Afonso et al., 2002), mixes Brazilian and European Portuguese and is made up exclusively from news text. All treebanks were converted from other dependency representations, and all but Portuguese received some amount of manual checking.

4.3 Se clitics

It remains a thorny issue in Romance syntax that, across Romance languages, a single morpheme takes seemingly different roles in a range of constructions that are not obviously related, displaying a consistent versatility that resists a unified analysis. These morphemes, which I will call SE clitics (*se* in French, Portuguese and Spanish, *si* in Italian), have, for the most part, the same properties in these four languages, as well as other Romance languages. There are also strong parallels with Slavic (see Rivero, 2001; Medová, 2009; Teomiro García, 2010). In the face of their recurrence and consistent behavior across languages, the appeal of a unified analysis of such clitics is undeniable. Nevertheless, there is currently no consensus on whether, in any one language, this morpheme consists of a single linguistic entity entering multiple

constructions with different functions, or (if not) on how the different homophonous entities may be related.

The full scope of this issue has filled more than one dissertation in theoretical Linguistics, so its brief introduction here will inevitably be incomplete. My goal is to show the treatment of these constructions in the UD v.1.2 corpora, and make a proposal for treating them consistently in the future.

I introduce the issue with a concise summary of important facts about reflexive clitics in the four target languages, in Section 4.3.1, focusing on the constructions that are common to most of these languages.[2] This description will divide the SE constructions into five types which appear in different classifications (Rivero, 2001; Dobrovie-Sorin, 2006; González Vergara, 2006; Medová, 2009; Teomiro García, 2010; Mendikoetxea, 2012): true reflexive/reciprocal; inherent reflexive; inchoative; impersonal; passive/middle. I review the current treatment of SE in the UD treebanks, showing how different interpretations of the guidelines (and different conversion strategies) lead to different implementations of these constructions. I then make a proposal for representing each construction in UD, with reference to the major directions that influential analyses of these phenomena have taken, and concentrating the discussion on what the analysis implies for the characterization of any dependency types involved.

4.3.1 A brief description of SE

Background Because SE displays properties of clitics and reflexives, some preliminary considerations about both paradigms in Romance are in order.

All Romance languages have an object clitic paradigm, representing inflections for person and number, as well as gender in the 3rd person. The 1st- and 2nd- person clitics also function as reflexives; but in the 3rd person, there seems to be a distinguished reflexive clitic, SE, without number or gender marking. This is exemplified in Table 4.1 for Portuguese. SE clitics are not case-marked in any of our four languages.

In addition to these clitics, each language has distinct reflexive and reciprocal tonic

[2]This excludes, for example, applicative SE in Italian and Spanish, or aspectual SE in Spanish.

	Accusative	Dative	Reflexive
1SG	me	me	me
2SG	te	te	te
3SG	o/a	lhe	**se**
1PL	nos	nos	nos
2PL	vos	vos	vos
3PL	os/as	lhes	**se**

Table 4.1: Portuguese object clitics

pronouns, shown in Table 4.2. (The tonic reflexives will be glossed as REFL.STR, in contrast with the clitics glossed REFL.)

	3SG Tonic Refl.	Tonic Rec.
French	soi-même	l'un l'autre
Italian	se stesso	l'uno l'altro
Portuguese	si mesmo	um ao outro
Spanish	sí mismo	el uno al otro

Table 4.2: Tonic reciprocal and 3rd person tonic reflexive pronouns.

SE participates in these paradigms, alternating with clitic and tonic reflexives; but its distinctive feature is that, in some constructions, takes on a different role that cannot be realized by the other elements in these paradigms, as we will see next.

Anaphoric and impersonal SE The SE clitic participates in various constructions with different sets of predicates. Following Mendikoetxea (2012), I will divide these constructions into two types. In the first type, the predicate takes a canonical subject of any person and number, and SE alternates with the object clitics, according to the person and number of the subject. These comprise the true reflexive/reciprocal, inherent reflexive and inchoative constructions, for which I will adopt the term **anaphoric**.

In other constructions, either there is no overt canonical subject, or the subject's person feature is restricted to 3rd person. The number feature of the subject can vary—but that does not trigger overt agreement because SE itself is unspecified with

respect to number. In this case, SE does not alternate with object clitics. An arbitrary human agent is understood, but it cannot be expressed overtly. These consist of impersonal, passive and middle SE constructions, and they will be called **arbitrary**.

Anaphoric SE

True reflexive/reciprocal In the true reflexive/reciprocal construction, illustrated in (116), the predicate has a transitive counterpart, and SE alternates with 1st- and 2nd-person object clitics, agreeing with the subject, as demonstrated in (117) for Portuguese. These uses of SE are characterized by a reflexive interpretation.

(116) a. Spanish:
María se critica.
María SE criticizes
Maria criticizes herself.

b. Italian:
Maria se guarda.
Maria SE guards
Maria guards herself.

c. French:
Luc se lave.
Luc SE washes
Luc washes himself.

(117) a. Portuguese:
Pedro se lavou.
Pedro SE washed
Pedro washed himself.

b. Eu me lavei.
I REFL.1SG washed
I washed myself.

The clitics also alternate with the tonic reflexive or reciprocal anaphors, as shown in (118). However, plural SE, shown in (118a), as well as the plural object clitics, are ambiguous between a reflexive and reciprocal reading, while the tonic reflexive (118b) and the tonic reciprocal (118c) forms are unambiguous in this respect.

(118) a. Portuguese:
Os meninos se lavaram.
The boys SE washed.PL
The boys washed themselves. OR The boys washed each other.

b. Os meninos lavaram a si mesmos.
The boys washed.PL to REFL.STR.3PL.M
The boys washed themselves.

c. Os meninos lavaram uns aos outros.
The boys washed.PL REC.3PL
The boys washed each other.

Different authors have argued these clitic-hosting verbs to be unergative or unaccusative, as we will discuss in Section 4.3.3.

Inherent reflexives These verbs, illustrated in (120), differ from true reflexives in that they have no transitive alternation. Furthermore, there is no alternation of SE with strong reflexives, and no possibility of a reciprocal interpretation with plural subjects. Inherent reflexives are similar to true reflexives in that the 1st- and 2nd-person object clitics alternate with SE, as shown in (121).[3] Some inherent SE verbs

[3]Discussing Spanish, González Vergara (2006) notes that it is sometimes hard to draw a line between inherent and true reflexives. For example, Portuguese *levantar-se* means to stand up; it does not take a tonic reflexive anaphor (119a), but it appears to have a transitive counterpart (119b):

(119) a. Portuguese:
Eu levantei a mim mesma.
I lifted to REFL.STR.1SG.F

b. Eu levantei a cadeira.
I lifted the chair
I lifted the chair.

also take additional arguments, which must be realized as obliques, such as *de Marie* in (120c).

(120) a. Spanish:
Juan se desmayó.
Juan SE fainted
Juan fainted.

b. Italian:
Giovanni si sbaglia.
Giovanni SE mistakes
Giovanni is wrong.

c. French:
Jean s' est souvenu de Marie.
Jean SE has remembered of Marie
Jean remembered Marie.

(121) a. Portuguese:
Pedro se confundiu.
Pedro SE confused
Pedro was confused.

b. Eu me confundi.
I REFL.1SG confused
I was confused.

The subjects of inherent reflexives are usually argued to be internal arguments, as evidenced by unaccusative behavior. This will be discussed in Section 4.3.3.

Inchoative SE Like true reflexives, inchoative SE predicates such as those in (122) alternate with a transitive counterpart. Inchoatives alternate with causative verbs (123). Like inherent reflexives, they cannot take a strong anaphor (124a) and do not allow a reciprocal reading (124b).

(122) a. Portuguese:
O vaso se quebrou.
The vase SE broke
The vase broke.

b. French:
La branche s' est cassée.
The branch SE is broken
The branch broke.

c. Spanish:
El cristal se rompió.
The glass SE broke
The glass broke.

d. Italian:
Il vetro si rompe.
The glass SE breaks
The glass breaks.

(123) Portuguese:
Eu quebrei o vaso.
I broke the vase
I broke the vase.

(124) a. * O vaso quebrou a si mesmo.
The vase broke to REFL.3SG.M

b. Os vasos se quebraram.
The vases SE broke
The vases broke. NOT The vases broke each other.

Like inherent SE predicates, these verbs are usually considered unaccusative. Medová (2009) notes that, while inherent reflexives can take human arguments, inchoatives normally take inanimate arguments.

Arbitrary SE

Passive and middle SE Romance has two passive constructions, sometimes referred to as the analytical (or periphrastic) and the synthetic (or SE) passives. The **analytical** passive (shown in (125)) parallels English passives and requires an auxiliary.

The **synthetic** or SE passive (**??**) also promotes an internal argument to subject, but rather than being marked by an auxiliary, it is marked with SE. As such, the passive SE construction has a transitive counterpart. This passive cannot take an agent phrase (127a), contrary to analytical passives. Passive SE is also incompatible with agentive adverbs (127b) and purpose clauses. The agent is interpreted as an indefinite human subject.

(125) Portuguese:
As maçãs foram comidas pelas crianças.
The apples were eaten by the children
The apples were eaten by the children.

(126) a. Spanish:
Se comen las manzanas.
SE eat.PL the apples
Apples get eaten.

b. Italian:
Le materie letterarie si studiano in questa universitá.
The literary subjects SE study.PL in this university
Literary subjects are studied in this university.

c. French:
Les pommes se mangent en hiver.
The apples SE eat.PL in winter
Apples get eaten in winter.

(127) a. Portuguese:
* Comeram -se maçãs pelas crianças.
Eat.PL -SE apples by the children

b. * Comeram -se maçãs com vontade.
Eat.PL -SE apples gladly

The middle SE construction is syntactically identical to the SE-passive, at least on the surface: it is unaccusative and incompatible with the expression of an agent. However, it has a stative, generic interpretation (as opposed to eventive and episodic), which in some languages must be triggered by an adverb.

(128) a. Portuguese:
Grego se traduz facilmente.
Greek SE translates easily
Greek translates easy.

b. French:
Le grec se traduit facilement.
The Greek SE translates easily
Greek translates easy.

c. Spanish:
Estas manchas no se quitan con nada.
These stains not SE come out.PL with nothing
These stains don't come out at all.

d. Italian:
Questo vestito si lava facilmente.
This dress SE washes easily
This dress washes easy.

Many authors consider the middle-SE construction simply a different reading of the other impersonal SE constructions; Mendikoetxea (1999) describes a middle-passive SE and a middle-impersonal SE, which would be the middle readings of the passive and

impersonal constructions. Here I will not discuss a separate syntactic representation for middle SE.

Impersonal SE The impersonal construction is very different than the other SE constructions. Multiple types of predicate can enter it: not only transitive verbs (129a), but also unergative (129b), unaccusative (129c), and passivized verbs (129d), and non-verbal predicates with a copula (129e). When the verb is transitive, its internal argument receives accusative case and does not trigger verb agreement, unlike in the passive construction. This is shown in (129a), where *cambios* does not agree with *observa*.[4]

(129) a. Spanish:
Se observa cambios en la economia.
SE observe.SG changes in the economy
Changes are observed in the economy.

b. Portuguese:
Aqui em casa se dorme cedo.
Here at home SE sleep.SG early
Here at home everyone sleeps early.

c. Nesse país se chega sempre tarde.
In this country SE arrive.SG always late
In this country people are always late.

d. Italian:
Spesso si è traditi dai falsi amici.
Often SE is betrayed.SG by false friends
One is often betrayed by false friends.

[4]This is a *locus* of crosslinguistic variation; some varieties of Romance languages, and some Slavic languages, do not allow the impersonal (that is, non-agreeing) construction with transitive verbs, leaving only the agreeing (passive) construction. Some authors give them a unified view, such as Teomiro García (2010); Medová (2009). A side note: in Brazilian Portuguese, the impersonal SE construction with transitive predicates is frequent but stigmatized; prescriptive grammarians preach agreement between the verb and the internal argument.

e. Non si è mai contenti.
Not SE is anymore happy
One is not happy anymore.

French does not have this construction; in some analyses, such as Manzini (1986), this is predicted by using the possibility of null subjects as a requirement for the construction to be licensed. In fact, many authors (e.g. Dobrovie-Sorin, 1998) view the SE clitic in this construction as corresponding to French *on*, English *one*, German *man*.

Impersonal SE constructions in Romance have a peculiarity: they are impossible in most non-finite contexts (Burzio, 1986; Cinque, 1988). They cannot appear at all under control verbs (Burzio, 1986), and under raising verbs, they can occur only if the infinitive is a transitive or unergative verb (as observed by Cinque, 1988).

4.3.2 SE in the v.1.2 treebanks

Each of the four Romance languages under analysis here makes different choices for the representation of the SE clitic in the v.1.2 treebanks, which reflects the fact that the existing guidelines do not give sufficient attention to the syntactic properties involved in these phenomena. In this section, I present a brief summary of the strategies adopted in each language. The examples are simplified from real corpus examples. Table 4.3 presents the frequency of each label for the morpheme SE in each treebank.

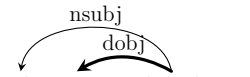
French		Italian		Portuguese		Spanish	
1346	dobj	1504	expl	841	dobj	4790	iobj
5	nsubj	619	expl:impers	317	nsubj	9	dobj
1	nsubjpass	165	dobj	217	iobj	2	mark
1	compound	21	iobj	27	auxpass:reflex	2	nsubj
1	det			17	dep	2	root
				8	nmod	1	compound
				1	advcl	1	det
						1	parataxis

Table 4.3: Counts of different dependency types used for SE in the four treebanks.

SE in French

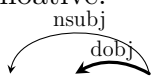
The label counts for the word *se* in the French v.1.2 corpus are shown in Table 4.3. Discarding the very infrequent labels, we can see that, essentially, the clitic is uniformly annotated as `dobj` (130), even when it occurs with a clearly monoargumental predicate: for example, the case of SE *suicider* ('to kill oneself') in (130c).

(130) a. True reflexive:



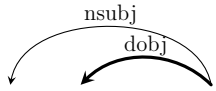
Il se ridiculise.
He SE ridicules
He ridicules himself.

b. Inchoative:



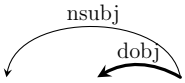
Elles se perdent en cherchant le Petit Trianon.
They SE lose.PL searching for the Petit Trianon
They got lost looking for the Petit Trianon.

c. Inherent:



Maria se suicide.
Maria SE commits suicide
Maria kills herself.

d. Passive:

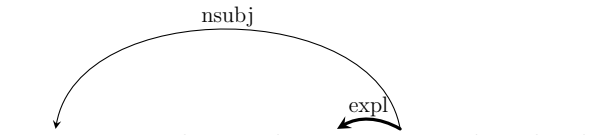


Le massacre se produisit dans les jours suivants la bataille.
The massacre SE produces in the days following the battle
The massacre happened in the days after the battle.

SE in Italian

In Italian, the annotation of *si* is more nuanced. Four labels are used, as shown in Table 4.3. The internal argument labels `dobj` and `iobj` are not used very consistently for any of the constructions, but `dobj` seems to be used mostly for passive SE, and `iobj`, mostly for true reflexives. The language-specific label `expl:impers` appears on inchoative, passive and impersonal SE, but not exclusively—`expl` is also used for these, as well as for true reflexive and inherent SE.

(131) a. True reflexive:



Bossi covers himself in ridicule when he speaks.

b. Inchoative:

The article does not apply in this case.

c. Inherent:

(They) will continue to behave in this way.

d. Passive:

The salon will open in four days.

e. Impersonal:

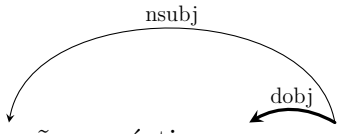
Ma già si parla di epurazioni.
But already SE speaks of investigations
But there is already talk of investigations.

SE in Portuguese

As in Italian, in this corpus we find a range of annotations of SE constructions. Most utilize the core argumental labels, but a language-specific label, `auxpass:reflex`, is also introduced. The counts[5] are shown in Table 4.3.

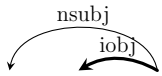
The use of `nsubj` is reserved for impersonal SE—although, due to ambiguity, that is sometimes indistinguishable from passive SE—while `dobj` and `iobj` apply mostly in true reflexives, but sometimes also in other SE constructions. The language-specific `auxpass:refl` is reserved for passive SE.

(132) a. True reflexive:



Gravações acústicas se encaixam com o nosso tipo de som.
Recordings acoustic SE fit.PL with the our type of sound
Acoustic recordings fit in with our type of sound.

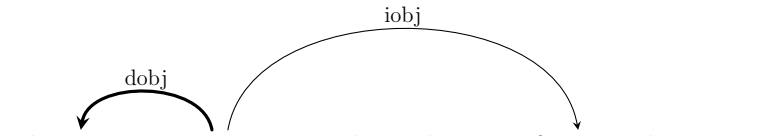
b. Inchoative:



A praia se torna exclusiva dos passageiros.
The beach SE becomes exclusive of the passengers
The beach becomes passenger-exclusive.

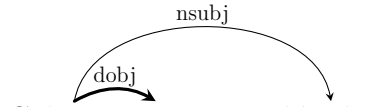
c. Inherent:

[5]These numbers refer to the word *se* when UPOS-tagged *PRON*, because *se* is also a complementizer in Portuguese.



He takes possession of all types of fish.

d. Passive:



New doubts are put forward.

e. Impersonal:

It puzzles me that one works for this.

SE in Spanish

The situation of Spanish is similar to that of French: discarding some inconsistencies, SE is uniformly labeled as an internal argument—but the label used is `iobj` rather than `dobj`. The complete counts are shown in Table 4.3.

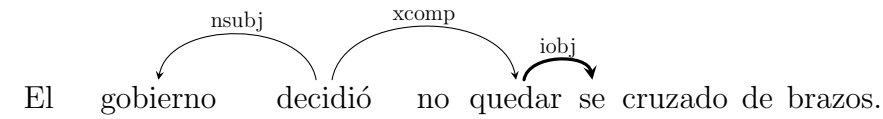
(133) a. True reflexive:

The CNT removes itself from the committees.

b. Inchoative:

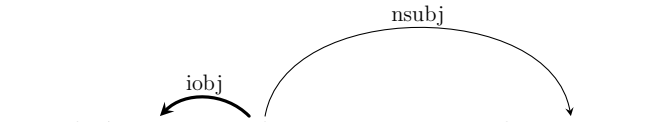
East Milton SE finds located in the coordinates
East Milton is located in these coordinates.

c. Inherent:



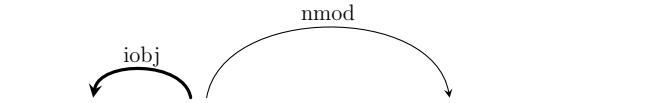
The government decided.SG not stay SE crossed of arms
The Israeli government decided not to keep its arms crossed.

d. Passive:



Also SE visualize numerous dips
Numerous dips were also seen.

e. Impersonal:



Not SE adhered.SG to the passport biological
One did not adhere to the biological passport.

4.3.3 Existing literature

A fully comprehensive survey of the literature on this topic would be well out of the scope of the present discussion, especially considering that much of that literature builds on framework-specific theoretical assumptions with no bearing on the UD representation. Rather than attempt such a survey, I will focus on aspects of existing proposals that are directly relevant to the distinctions that UD can express. For a

thorough bibliographic review, the reader is encouraged to refer to Medová (2009) and Teomiro García (2010). Here, I will briefly outline some major divisions between different analyses, intending to convey that there are important contested issues; in Section 4.3.4 I will draw from that literature to motivate specific decisions about representing SE in UD.

The first essential question about the analysis of SE constructions is whether the same linguistic element participates in all of them. Unified analyses exist: examples are Manzini (1986); Kayne (1988); Mendikoetxea (1997); Medová (2009). Those are, however, the exception. By and large, in synchronic terms, most authors view SE as a set of homophonous elements.

How many elements are in that set, and how they divide up the space of SE constructions, are also hotly contended questions: for example, Burzio (1986) attempts to unify the passive, middle, inchoative and inherent uses of SE as markers of unaccusativity. Dobrovie-Sorin (2006) unifies the anaphoric uses of SE with the passive and middle uses, but argues that impersonal SE is a separate linguistic entity. Cinque (1988) describes two types of SE, one which is argumental and appears in true reflexive/reciprocal and in some impersonal constructions, and another which appears in inchoative and inherent constructions, and other impersonal constructions.

Another question is whether SE is an element introduced in the syntax, or a marker of a lexical operation. Grimshaw (1982) provides a (superficially) unified analysis for French SE (which cannot be impersonal) by designating it a grammatical marker of different valency operations taking place in the lexicon. Wehrli (1986) proposes a similar description. Both, however, require stipulating the distribution of SE, as they cannot explain why the same marker is used for different operations. Even more fundamentally, taking SE out of the syntax predicts that it should only appear where selected by its governing predicate. However, SE also appears in Exceptional Case Marking (ECM) positions. If SE arises in a pre-syntax operation, before the predicate is combined with an ECM complement, this cannot be explained.

4.3.4 A proposal for SE in UD

Different UD representations are necessary Clearly SE is an element pertaining to core predicate-argument syntax: it is associated with verbal predicates, appears to alternate with argumental clitics, and participates in argument-structure alternations. It will be helpful to keep in mind that, as discussed in Section 2.4.1, UD dependencies in the predicate-argument domain are meant to label grammatical functions, marking syntactic distinctions that allow semantic roles to be recovered from the triple (GP, GF, SF), introduced in Section 4.2. The labeling of SE should preserve differences in semantic role mapping as much as possible.

Another important remark is that, in extant theories of SE, a unified analysis of the subtypes of constructions invariably hinges on the availability of sophisticated formal machinery: multiple levels of representation (such as Alsina, 1996a) or reference to a derivation history (such as Manzini, 1986). The representation available in UD is much simpler in nature; the only representational tools for predicting differences at the syntax-semantics interface are the labels assigned to the predicate's arguments.

Considering the simplicity of the UD toolkit, and the fact that even the availability of very sophisticated theoretical machinery has not enabled a consensual unified analysis of SE constructions, it follows that committing to a unified UD representation for these constructions is likely to be futile. Moving forward with this assumption, I will approach each SE construction separately, and make a proposal to represent it. The possibility of unification will be considered only at the end.

True reflexive SE

Intuitively (as a Portuguese native speaker), true reflexives seem to be the canonical use of SE—witness the fact that SE is often referred to as a reflexive clitic, despite its wide range of uses. True reflexive predicates assign two semantic roles, and Medová (2009) notes that there are two ways in which reflexivity can be implemented: either the predicate is somehow marked as reflexive, so that it takes a single syntactic argument, or one of the syntactic arguments is marked as necessarily identical to the other one.

At first glance, the alternation of SE with internal arguments under true reflexive predicates suggests an analysis of such predicates as transitive, with SE taking the place of internal argument. Burzio (1986); Manzini (1986); Cinque (1988) all argue that true reflexive SE is argumental in nature, with different nuances: Burzio claims that SE forms a chain with an empty category in argument position, which is also his analysis for Romance object clitics; Manzini, that it is an argument clitic and an anaphor; Cinque, that it is base-generated in object position. In these analysis, SE is theta- and Case-marked, and therefore believed to receive a semantic role.

The difficulty in maintaining a transitive analysis for true reflexive SE is that the construction patterns with intransitive rather than transitive predicates with respect to many tests. (Mendikoetxea, 2012, p. 489) gives the following summary:

> Arguments for unaccusativity include, among others, Italian auxiliary selection and participle agreement (Burzio, 1986), French causatives (Grimshaw, 1982, following Kayne, 1975), nominalized infinitives in Italian (Zucchi, 1993) and Spanish (Mendikoetxea, 1997), participial constructions and arbitrary subjects in Catalan (Alsina, 1996b) and Spanish (Mendikoetxea, 1997), and missing causes in causative constructions in Spanish (Mendikoetxea, 1997). Evidence for the transitive-like properties of CL-REFLs is found in the ungrammaticality of partitive clitics in Italian (Burzio, 1986) and Catalan (Alsina, 1996b), the fact that the subject cannot be realized as a bare NP in Spanish (Mendikoetxea, 1997), subject-oriented secondary predicates in Catalan (Alsina, 1996b) and, crucially, reflexive constructions with dative clitics in Romance languages like Spanish, Portuguese, and Italian (Dobrovie-Sorin, 2006; Labelle, 2008).

On the other hand, for Reinhart and Siloni (2004), SE is also a morphological marker of a syntactic reflexivization operation, but one that maps two roles to the external (and only) argument. This predicts unergative behavior for true reflexives. Dobrovie-Sorin (1998, 2006) present a similar account in which SE is a Case-marked non-argumental clitic that forces the coindexation of the subject and object positions, also predicting unergativity. Alsina (1996b) predicts properties of both unergativity

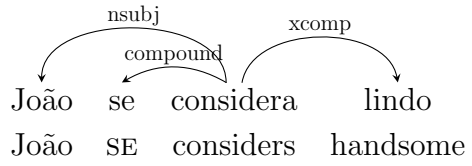
and unaccusativity by developing an LFG account in which a single syntactic expression is assigned two semantic roles, via binding in argument structure.

Finally, Grimshaw (1982) and Wehrli (1986) both propose that SE in true reflexives is a marker of a lexical reflexivization operation.

In order to decide how to represent true reflexive SE in UD, I will now consider how different analyses can be roughly "translated" into the standard.

`compound` is ruled out Starting with the marker approach from Grimshaw (1982) and Wehrli (1986), I propose that in UD it would correspond to representing true reflexive SE as a word-level dependent, such as `compound`. The problem which these analyses pose for representing ECM, as discussed in Section 4.3.3, does not arise for UD. As shown in (134), ECM predicates such as *considerar* can take open complements while maintaining a generalization about their lowest core argument being the controller (*João*, in (134)).

(134) Portuguese:



João considers himself handsome.

What a `compound` label does not capture is the fact that SE in true reflexives alternates with object clitics. It can be linearized before or after the verb, and it agrees with the subject. Neither of those properties is expected of other `compound` dependents in Romance. Since we cannot extend the `compound` analysis to the other reflexive clitics, the parallelism between those clitics and SE is lost.

`expl` is possible For translating the syntactic non-argumental approaches, `expl` would be the most natural choice; the single full nominal argument would be represented with a core argumental label.

Revisiting the analytical principles laid down in Section 4.2, one important question to raise about this option is how it allows semantic role assignment to be correctly

recovered. In the `expl` analysis, we would assign a subject label to the fully expressed argument. This subject then has to receive two semantic roles from the predicate. There is no express constraint against this, since the subcategorization frames would be different: $\langle nsubj, expl \rangle$ in the SE usage, and $\langle nsubj, dobj \rangle$ in the transitive usage. In ECM constructions, the embedded predicate would be labeled `xcomp`, implying that the external argument is to be found in the matrix clause. In the case of true reflexives, for lack of an object in that clause, the matrix subject would correctly be taken to be that external argument, allowing the semantic roles to be recovered.

(135) João se considera lindo
nsubj, expl, xcomp

`dobj` is possible An alternative to the `expl` analysis would be to take an argumental approach, in which both the canonical subject and the SE clitic receive core argumental labels. The most natural option would be to label SE `dobj`; labeling it `nsubj` and making the full nominal `dobj` would not be appropriate because that nominal agrees with the verb and has the canonical properties of the grammatical function of subject in Romance. This holds even if we assume that the predicate is unergative, because the distinction between unergatives and unaccusatives does not change the assignment of dependency types to the predicate's arguments.

The subcategorization frames of the transitive and reflexive uses of verbal predicates would both be $\langle nsubj, dobj \rangle$, which means both would be interpreted as assigning a role to the `dobj` dependent. SE would be interpreted as a role-bearing internal argument. This leaves open the question of how SE is identified with the subject, but that problem exists for strong reflexives, independently of the analysis of SE; we can leave it up to coreference, which UD does not attempt to represent. There is no problem for representing ECM constructions: the presence of an object in the matrix clause forces the external argument of the `xcomp`-labeled complement to be identified with that object, and the object is identified with the matrix subject by coreference.

(136) João se considera lindo
nsubj, dobj, xcomp

Inherent SE

The defining property of inherent SE is that the clitic is obligatory with these predicates: they have no alternations without it.

As with true reflexives, there is some disagreement about whether the predicates are unaccusative or unergative. According to Teomiro García (2010), for example, they are unaccusative. However, Otero (1999) argues that there are two subclasses of inherent reflexive verbs in Spanish: unergative and unaccusative.

For Burzio (1986), the clitic is simply a morphological marker. This is also his analysis of inchoative SE, and the only difference is that inherent SE verbs do not participate in a causative alternation. Reinhart and Siloni (2004) take the same direction, adopting a marker analysis for both inchoative SE and inherent SE, but noting that a potential causative entry corresponding to these verbs is not lexicalized.

There are also argumental analyses of inherent SE. Masullo (1999) takes inherent SE to be an argument coindexed with the subject, as an expression of internal causation or inanimate external causation.

Because there is only one argument realization pattern for these predicates, there are fewer constraints with respect to the choice of label for annotating SE in UD. The typing of SE does not have to signal the difference in subcategorization frame with respect to a transitive counterpart.

`compound` is ruled out One possible choice at first glance would be to use the `compound` label, creating a complex unit. One important objection raised before for true reflexives—the fact that the analysis does not easily extend to the reflexive clitics that alternate with SE in this construction—is weaker here, because, since inherent SE predicates are monoargumental, it would not be crucial to represent the other reflexive clitics as arguments. However, there are still problematic aspects to the use of this label: as I argued for true reflexives, the fact that the clitic can linearize on either side of the verb and the fact that it agrees with the subject are both properties that are not expected of `compound` elsewhere. Additionally, `compound` dependents are meant to be optional modifiers, and their governors should be able to stand by themselves (usually with a different meaning). That is not the case with inherent

SE, since these verbs require the clitic. All in all, using `compound` would dilute generalizations we can otherwise make about compounding in Romance.

`expl` is possible Since these predicates are monoargumental (at least on the surface), SE should not receive an argumental label. The label `expl` is defensible, if we consider the fact that SE alternates with object clitics, indicating it has the same grammatical function, although it bears no referent.

Interlude: passive, inchoative and impersonal SE

Before moving on to other SE constructions, I want to point out that there are very important surface similarities between some of the SE constructions. Take (137).

(137) Portuguese:
Abre -se a porta.
Opens -SE the door
The door opens. OR The door is opened. OR Someone opens the door.

This can be seen as an inchoative SE construction with the verb *abrir* or as a passive SE construction, in which case the agent is interpreted as human. In both cases, the semantic role assigned to *porta* is the same; while there is a difference in interpretation, it rests on the conception of an unexpressed agent. An impersonal SE analysis is also consistent with this sentence, as confirmed by the fact that pluralizing *porta* does not necessarily require the verb to agree (138). The semantic role assigned to that argument would still be the same as in (137).

(138) Portuguese:
Abre -se as portas.
Opens -SE the doors
One opens the doors.

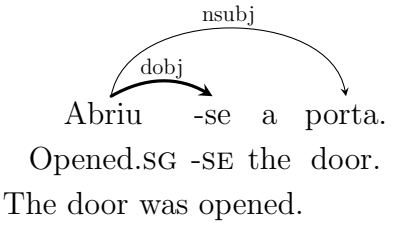
This is the most extreme type of ambiguity: the predicate admits an inchoative interpretation, which not all predicates do, and the singular subject makes it impossible to rule out impersonal SE. Other examples may be less ambiguous. But in

general, with predicates that enter the anticausative alternation and have singular subjects, these constructions can be difficult to distinguish,[6] and may not be distinct with respect to any syntactic property that UD aims to represent. This makes a unified analysis favorable for parsers and annotators.

Passive and middle SE

Because passive SE is incompatible with the expression of an agent, some authors have proposed analyses in which it bears the external theta-role. In Belletti's (1982) seminal account of Italian, passive SE is generated in a functional head position and "absorbs" accusative Case, analogously to passive morphology. It receives the external theta-role and bears nominative Case. (This stands in contrast with the author's analysis of impersonal SE, given below.) For Burzio (1986), SE is a subject clitic that forms a chain with an empty category in object position; that chain bears accusative Case and the external theta-role. For Raposo and Uriagereka (1996), passive SE is the external argument.

`dobj` is ruled out Adopting the argumental approach in UD is difficult. Clearly there is a full nominal argument in passive SE constructions that needs a core argumental role; because it (most often) occurs in canonical subject position, there is no syntactic motivation for applying an internal argument label. This means that if SE is annotated with an argumental label, there would be another dependent bearing a subject label, and SE would have to receive the `dobj` label.

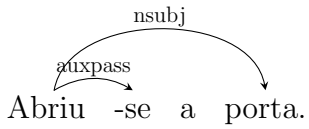
(139)


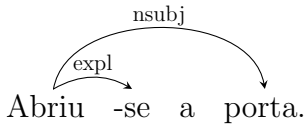
This would make the subcategorization frame of the predicate identical to the frame of the transitive alternation: $\langle nsubj, dobj \rangle$. We also have the same predicate, in

[6] To provide a personal note on this, when looking through annotated data to provide examples of the treatment of each construction in the treebanks, I found many examples quite difficult to classify into one or another construction, even in my native language.

the same surface form. Considering this, the principle that a semantic role should be recoverable for a core argumental dependent with respect to the triple (GP, GF, SF) cannot be maintained for this example, because the `nsubj` dependent of *abrir* clearly receives different roles in the transitive use and in the passive SE use; similarly for the `dobj` dependent.

`expl` or `auxpass` are both possible With that possibility rejected, we turn to functional categories for SE. We could potentially adopt `auxpass`, as currently done in the Portuguese treebank, to signify that this is a marker of a passivization operation (140a); or `expl` (140b), to signify that this is a semantically empty clitic that alternates with object clitics in the same position. Both create a subcategorization frame distinct from that of the transitive alternation; in (140a), the frame is $\langle nsubj \rangle$, and in (140b) it is $\langle nsubj, expl \rangle$.

(140) a. 

b. 

The fact that the agent of SE passives is constrained to a human agent is lost in this representation, because there is no explicit external argument in the representation to which this property can be attributed.

Inchoative SE

`dobj` is ruled out In the case of inchoative SE, semantically, we have a predicate that takes a single internal argument. These constructions alternate with causative transitive counterparts, but are themselves usually understood to be monoargumental. (Nevertheless, see Masullo 1999 for an analysis of inchoative SE as an external argument.) Based on this, I rule out argumental approaches using core argument labels, by the same rationale applied to passive SE above.

By some accounts, inchoative SE arises as the result of a lexical process. Burzio (1986) makes this argument, ascribing no syntactic role to the clitic and calling it a marker of an operation that takes place in the lexicon. Reinhart and Siloni (2004) also describe inchoative SE as arising from a lexical process, of which SE would be a morphological marker. This is also the approach of Grimshaw (1982) and Wehrli (1986). Manzini (1986) also considers that in the inchoative construction, SE attaches in the lexicon rather than in the syntax.

Cinque (1988) considers inchoative SE a syntactic element, but not an argumental one (unlike true reflexive SE). He considers that, in these constructions, the clitic "suspends" the external theta-role.

`compound` is ruled out Once more, a possibility here would be to consider SE a part of the predicate itself, in the same way that particles are word-level dependents of verbs in the case of particle verbs. In that sense we could use the `compound` label, by analogy with particles, to capture a difference in the predicate itself, rather than the subcategorization frame. This would reflect the view of inchoative SE as a marker of a lexical operation.

The disadvantages of this approach are the same ones raised for inherent SE, with one exception that does not apply to inchoative SE: while the clitic is obligatory for inherent reflexives, in the present case it is a marker of a productive argument alternation. In that sense, the clitic is closer to the "optional modifer" characterization of the `compound` label; but the important difficulty of extending the analysis to object clitics persists, as well as the optionality of the linearization and the unexpected agreement with an element external to the compound.

`expl` is possible Another possibility would be to say SE interacts with the argument structure by blocking a grammatical function; in this case, because it alternates with argumental clitics, we could use the `expl` label. This would make the decausative subcategorization frame $\langle nsubj, expl \rangle$, distinguishing it from the transitive one, as is desirable.

Impersonal SE

The case of impersonal SE is very different. Impersonal SE is incompatible with the expression of an agent; it appears in place of an external argument. (It is impossible with zero-argument predicates.) Like the passive, this construction places no restrictions on the type of predicate that can appear in it. Verbs with different subcategorization frames and even nonverbal predicates enter it.

As is the case with other SE constructions, the analyses of impersonal SE are divided. Going back to Belletti (1982), we find an argumental analysis in which impersonal SE is described as a clitic in INFL position. This is also Belletti's account of passive SE, but here SE absorbs nominative rather than accusative Case, leaving the latter free to be assigned to a direct object. SE bears the external theta-role. Burzio (1986) considers impersonal SE to be a subject clitic, forming, with an empty subject, a chain that bears Case and a theta-role. Manzini (1986) incorporates impersonal SE to her unified analysis, and claims that it is an argument clitic bearing a theta-role. Semantically, it introduces a free variable in the sentence. Rivero (2001) presents diagnostics from Spanish showing that nominative Case is not available for other arguments in impersonal-SE constructions, as well as diagnostics from Italian showing that SE can bind anaphors, including long-distance and reciprocal anaphors. With this, the author suggests that SE itself receives that Case.

There are also nonargumental analyses, in which the argument position must be occupied by a null pronominal of some kind; in Mendikoetxea (2008a, b), this element is referred to as a G(eneric)-PRO, which is only present under certain feature configurations on the T head. Otero (1986) uses PRO-arb to the same effect.

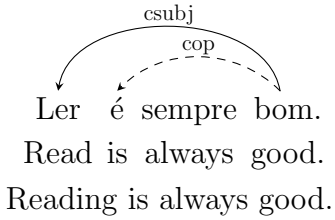
Cinque (1988) divides impersonal SE into two types of constructions and adopts distinct argumental and non-argumental analyses for each type.

`nsubj` is possible It is true that SE contributes to the interpretation of the external argument, because it restricts that argument to be a [+human] entity, similarly to constructions with arbitrary control in English, or the pronoun *one*. In that sense, for UD, we can say that impersonal SE is argumental; syntactically it stands in complementary distribution with canonical subjects, and semantically it places restrictions

on the external argument. This makes `nsubj` a natural choice.

`expl` is possible However, the constraint to have an arbitrary human agent is also placed by the presence of arbitrary null subjects, which (in Romance as well as English) are represented in UD by the lack of a subject dependency: the implicit arbitrary subject of *ler* in (141) does not correspond to any dependents.

(141) Portuguese:



In light of this and of the non-argumental analyses, there is an argument for representing impersonal SE as a semantically empty subject, typed `expl`. This approximates these structures to arbitrary control clauses.

A unified account

I argued against `compound` in the case of true reflexives, inherent and inchoative SE, based on the flexibility of the lineatization of SE in these cases, and the fact that such an analysis cannot easily be extended to the object clitics that alternate with SE in these constructions. I also presented objections to `dobj` in the case of passive and inchoative SE, due to the absence of surface properties supporting an argumental treatment of the clitic.

I argued that the `expl` label is a possible choice for true reflexive, inherent, inchoative, passive/middle and impersonal SE: it captures the fact that SE alternates with object clitics in true-reflexive, inherent and inchoative constructions, and with nominal subjects in impersonal ones. (There is no such alternation with arguments for passive SE.) I also offered some competing alternatives: `dobj` in the case of true reflexives, supported by semantic roles and the syntactic alternation with object clitics; `auxpass` in the case of passives, as the morphological evidence of passivization;

and `nsubj` in the case of impersonals, supported by the alternation with nominal subjects and the constraint on the [+human] feature of the subject.

The fact that the inchoative, passive/middle and impersonal SE constructions can be difficult to distinguish (which is bound to be a source of uncertainty for both automatic parsers and human annotators) makes a unified representation desirable. Little is lost with this underdetermination, because, as I showed with respect to (137), the semantic roles assigned by the predicates are the same in the ambiguous cases.

With this, I propose the use of `expl` for these constructions, which has the following advantages: in each case, the `expl` label captures the aforementioned alternations with argumental elements; overall, an argument with the same grammatical function in the same subcategorization frame is assigned the same semantic role; finally, there is no need to make possibly subtle semantic distinctions in order to annotate or parse the syntactic structure.

The case of inherent SE presents, in a way, the lowest stakes: differentiating subcategorization frames is not a concern, and ambiguity does not arise to pose annotation challenge. In this case, `expl` is also appropriate, mostly because it is a better representation than `compound` for an element that alternates with other reflexive clitics.

In the case of true reflexive SE (which is easy to distinguish from the other constructions, from the perspective of the annotator), I argue for the use of `dobj`. While both the non-argumental and argumental analyses can work satisfactorily in terms of semantic role assignment, as outlined, a crucial argument in favor of the argumental option is parallelism with other reflexive constructions—both the tonic reflexives within Romance, and other reflexive paradigms in other languages, such as *-self* reflexives in English. This has the advantage of associating a semantically diargumental predicate with a diargumental subcategorization frame.

This analysis of true reflexives fails to capture the fact that these reflexive predicates do not pattern with transitive verbs. Nevertheless, many such paradigmatic properties are not captured by UD; no annotation differences distinguish unergative from unaccusative predicates, for example. With that, I propose that maximizing the

extent to which the annotation of true reflexive SE is parallel to other reflexive constructions in other languages and with the predicates' transitive alternation is more important than attempting to precisely represent a paradigmatic difference between transitive and intransitive verbs.

4.4 Complex predicates

As we know, one of the key ideas motivating the design of UD is that there is a set of linguistic properties that all languages tend to express, but that appear with different morphosyntactic encodings: they can be expressed synthetically, by means of bound morphemes, or analytically, by means of function words. UD is designed to allow dependencies that represent argumental relations between content words to be annotated in the same way in every language, regardless of the way grammatical meanings are encoded. (Examples of this are discussed in Section 3.4.) This is accomplished without requiring a tokenization strategy to make synthetic and analytical encodings more similar.

The idea is simple: if there is a universal set of grammatical meanings, clearly distinguishable from lexical meanings; and if languages make a binary choice between expressing those meanings as function words or bound morphemes; then we can represent function words as dependents of their lexical heads, which makes the treatment of these universal grammatical meanings more similar across languages by creating an analogy between a dependent function word and a bound morpheme. In order for this to be possible, a hard line has to be drawn between content words and function words, as stated in Section 2.4.1.

That is the theory, but as is often the case, in practice the theory is different. There is no clear-cut line between grammatical meanings and lexical meanings that can be drawn across languages. Even in a single language, distinguishing a function word from a content word may be very difficult. This section tackles one area of such difficulty: the formation of verbal complex predicates in Romance, and the problems it presents for the distinction between content and function words in our four Romance treebanks, as well as for the representation of arguments.

The formation of complex predicates is a syntactic phenomenon in which multiple predicates behave as a single unit with respect to some syntactic processes, with one or more verbs becoming seemingly transparent. Mohanan (2006, Section 1) defines a complex predicate as "a construction in which two morphs, both of which are semantically predicative, jointly head a single syntactic clause." This (purposefully broad) definition highlights the characteristic feature that makes complex predicates difficult for UD: a mismatch between syntactic and semantic properties, creating a conflict between UD's goals of representing surface properties and providing scaffolding for semantics.

UD provides two types that are specifically meant for relations in which we find two morphemes jointly heading (in loose terms) a clause: auxiliaries and copulas, which have a tight syntactic association with their respective lexical heads. This is represented by special functional types in UD, `aux` and `cop`. In Section 2.4.1, I characterized these and other functional types in UD as akin to Tesnière's (2015) **transfer**, a symmetric syntactic relation in which a function word joins a lexical word to form a complex unit. Because of this, these types of dependents cannot have autonomous modifiers or make contributions to semantic role assignment, as explained in Section 4.2. Complex predicates have some characteristics that suggest a transfer-style analysis, but they can involve verbs that do not readily lend themselves to be classified as function words, and there are strong reasons to believe they should have dependents that are not shared by their governor.

Because the notion of complex predicate is very broad, the properties that characterize it vary from language to language, as does the set of verbs that can form complex predicates. In Romance, I adopt clitic climbing as the defining property of complex predicates, following Moore (1991), Manning (1992) and Abeillé and Godard (2003), among others.

4.4.1 A brief description of Romance complex predicates

Normally in Romance languages, object clitics attach to the verb that selects them; the choice of enclisis or proclisis varies according to several conditions. If that verb is

itself selected by a higher verb, the clitics cannot be placed before that higher verb; they have to be adjacent to the lower verb.

(142) Spanish:
Maria insistió en comer las.
Maria insisted in eat.INF ACC.3PL
Maria insisted on eating them.

However, there is a class of verbs that can in fact host clitic arguments selected by their complement predicates.

(143) a. Italian:
Maria le vuole mangiare.
Maria ACC.3PL wants eat.INF

b. Spanish:
Maria las quiere comer.
Maria ACC.3PL.F wants eat.INF

c. Portuguese:
Maria as quer comer.
Maria ACC.3PL.F wants eat.INF
Maria wants to eat them.

This phenomenon is known as clitic climbing. Bok-Bennema (2006) observes that clitic climbing is allowable only with a restricted set of verbs, and as such is the exception rather than the rule. It is not possible, for example, with Spanish *insistir* (144).

(144) Spanish:
* Maria las insistió en comer.
Maria ACC.3PL insisted in eat.INF

I will call verbs that allow clitic climbing **trigger verbs**, and refer to the clitics as **long objects**. *Want*-type verbs are not trigger verbs in French (145), but other verbs are (such as *faire* in (147)), as we will see in the next section.

(145) French:

* Marie les veut manger.

Marie ACC.3PL wants eat.INF

When trigger verbs are nested, they are all transparent to the clitic, which is free to climb to the highest verb while being selected by the lowest. This is illustrated in (146), where the two clitics *te* and *lo* are arguments of *mostrar*, 'to show'.

(146) Spanish:

Te lo quiero tratar de terminar de mostrar mañana.

DAT.2SG ACC.3SG want take care of finish.INF to show tomorrow

I want to take care of finishing showing it to you tomorrow.

While no nominal arguments can intervene between the verbs when clitic climbing occurs, other elements can appear in that position, such as some adverbials.

(147) French:

On ne le fera pas lire.

We not it make.3SG.FUT not read.INF

We will not make you read it.

Clitic climbing must always affect all clitics in a cluster together. In fact, this is a facet of a more general trend: all monoclausal properties of a complex predicate must manifest together or not at all. I will return to this in Section 4.4.3.

(148) a. Spanish:

Quiero dár te lo.

Want give DAT.2SG ACC.3SG

b. Te lo quiero dar.INF.

DAT.2SG ACC.3SG want give.INF

I want to give it to you.

c. * Te quiero dar lo.

DAT.2SG want give.INF ACC.3SG

d. * Lo quiero dar te.

ACC.3SG want give.INF DAT.2SG

Different types of trigger verbs

All trigger verbs are not always treated together in the syntax literature, and I will not treat them all together here. Canonical auxiliaries, for example, have the properties of trigger verbs, but are often considered syntactically distinguished. Because of the heterogeneity of this group, it is useful to think about the range of trigger verbs in terms of the properties that divide them into subgroups. For that purpose, I present two descriptive classifications here: one focused on surface behavior, and another on the properties of the external argument inside the trigger verb's complement. Taking a broad view of these different groups now will make the relation between different types of trigger verbs more clear, and will be useful for thinking about how the representation of complex predicates relates to the use of the functional labels `cop` and `aux`, a topic that will be revisited in Section 4.5.

Abeillé and Godard (2003) divide Romance verbs that allow clitic climbing into four descriptive classes, based on distinctive syntactic properties: auxiliaries, restructuring verbs, causatives, and attributive verbs.

Auxiliaries are defined, for Abeillé and Godard, by two properties: clitic climbing is obligatory (that is, clitics must attach to the auxiliary and cannot attach to the lexical verb) and their subject is always identified with the main verb's subject.

The class of **restructuring** verbs—which tends to get most of the attention when it comes to Romance clitic climbing—comprises modal, aspectual and movement verbs with optional clitic climbing and, again, subjects obligatorily identified with the matrix verb's.

"**Causative**" verbs (comprising verbs of influence but also of perception) are characterized for taking a nominal complement that is identified with the subject of their infinitival complement, and for obligatory clitic climbing—as long as the lower external argument does not appear between the two verbs. Causatives also allow the lower external argument to cliticize on the higher verb, as a dative if the lower verb is transitive and as an accusative otherwise.

Finally, **attributive** verbs, which comprise a set of raising and ECM verbs with nonverbal complements, also allow optional clitic climbing.

Here I will focus on restructuring verbs and causatives, which take infinitival

complements that can govern core dependents.

Moore (1991) offers another way of classifying Romance trigger verbs, based on the external argument of the lower verb. At a high level, they are divided into **clause reduction** triggers and **clause union** triggers, following Aissen and Perlmutter (1983). Clause reduction triggers can be raising, subject-control, or object-control verbs; clause union triggers are causative and perception verbs, which Moore argues select ECM complements.

This classification cross-cuts the alternative proposed by Abeillé and Godard: auxiliaries are a subset of raising triggers; restructuring verbs are divided between raising and subject control triggers; causatives comprise all clause union triggers and some object control triggers; finally, attributive verbs include raising and object control triggers. The high-level division between clause reduction and clause union reflects an analytical division that has long featured in the literature of the Chomskyan tradition, as we will see below in Section 4.4.3.

Other monoclausal properties of Romance complex predicates

There are other behaviors beyond clitic climbing that characterize Romance complex predicates. These additional behaviors are observed only in verbs that allow clitic climbing, but the implication is unidirectional: not all triggers of clitic climbing have these other monoclausal properties.

In Portuguese, Italian, and Spanish (but not French), the analytic passive (formed with an auxiliary) can also cross apparent clausal boundaries. When a trigger verb is passivized, the subject of the passive can be, rather than an object selected by the trigger verb itself, an object selected by the verb's complement, such as *questo libro*, which is an argument of *leggere* (and not of *fatto*), in (149).

(149) Italian:

Questo libro è stato fatto leggere a Mario.
This book is been made read to Mario.
This book was made to be read by Mario.

Long passive formation is allowed with causatives (with restrictions in Spanish,

more widely in Italian and Portuguese) and with some aspectual restructuring verbs in Spanish and Italian (Abeillé and Godard, 2003).

In addition to analytic passives, SE passives (see Section 4.3.4) also have a long counterpart under trigger verbs. In long SE passives, illustrated in (150), an object of the trigger verb's complement can be promoted to subject of the matrix verb, which receives a SE clitic to form a synthetic passive. Besides occurring in Italian, Portuguese and Spanish, long SE passives are also observed in French with causative triggers (with some restrictions).

(150) Portuguese:
Estas camisas podem -se passar facilmente.
These shirts can -SE iron.INF easily
These shirts iron easy.

An additional property of trigger verbs is the possibility of *tough*-movement over the trigger verb (151). Unlike in English, in Romance this movement is normally clause-bound; it can, however, happen over trigger verbs.

(151) French:
Un text impossible à faire comprendre rapidement.
A text impossible to make understand.INF quickly
A document impossible to make one understand quickly.

Just as all clitics must climb if one clitic climbs, it is also that case that clitics must climb in the presence of a long passive, as noted by Manning (1992) and illustrated in the contrast in (152). The oblique *a los dueños* is doubled with the clitic *les*; because the long passive is formed, that clitic must appear attached to *están*, even though semantically it is associated with *pintar*.

(152) a. La casa les está siendo terminada de pintar a los dueños.
The house DAT.3PL is being finished.SG.F of paint.INF for the owners.
Painting of the house is being finished for the owners.

b. * La casa está siendo terminada de pintar les a los dueños.
The house is being finished.SG.F of paint.INF DAT.3PL for the owners.

All these special syntactic properties characterizing trigger verbs are occurrences of normally clause-bounded phenomena in what on the surface look like biclausal (or even multiclausal) structures. Manning (1992) also argues that ordering and case marking of arguments under causatives, as well as adverb scope, further support a monoclausal analysis of these complex predicates. Aissen and Perlmutter (1983), Manning (1992) and Bok-Bennema (2006) all note that the fact that these properties correlate with clitic climbing is evidence that the right explanation must go beyond granting clitics the ability to climb. If the arguments selected by the lowest verb are considered to also be arguments of the highest verb, all of these properties are predicted straightforwardly.

4.4.2 Romance complex predicates in the v.1.2 treebanks

As was the case with SE, the four Romance languages investigated here represent complex predicates in different ways in the v.1.2 treebanks, indicating a need for improved guidelines. Here I summarize the strategies adopted in each language. The examples in this section are simplified from real corpus examples.

Because identifying clitic climbing requires human interpretation,[7] I relied on performing high-recall searches and scanning the results to spot valid examples. My comments on the different analyses are based on manual inspection of the results of these searches.

The searches were centered around specific verbs that exemplify different classes of triggers. These target verbs, taken from Abeillé and Godard (2003), are given in Table 4.4, along with the frequency of each verb in its respective treebank.

Simply searching for the lemmas would result in a very large sample of sentences that do not feature monoclausality properties, so I narrowed down the search space by looking for a combination of the lemma with an infinitival complement. However, the relation between those two verbs is not always represented in the same way: the trigger verbs are sometimes governors and sometimes dependents of their infinitival

[7]This is because a clitic or a passive subject associated with the matrix verb do not necessarily correspond to the internal argument of the complement verb. In fact, clitics in that position are quite common, but they correspond almost invariably to the external argument of the lower verb.

	French		Italian		Portuguese		Spanish	
Causative	faire	1030	fare	717	fazer	543	hacer	807
Perception	voir	244	vedere	459	ver	180	ver	309
Restructuring (control)	-		volere	220	querer	158	querer	190
Restructuring (raising)	-		potere	1068	poder	505	poder	1021

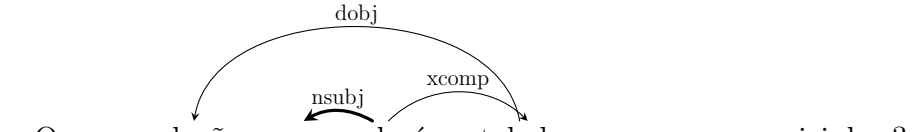
Table 4.4: Lemmas targeted in the search for complex predicate formation, with respective frequencies. For the French corpus, which did not include lemmas in v.1.2, I used the TreeTagger lemmatizer (Schmid, 1995). The gaps in the lists reflect the fact that modern French lacks restructuring verbs.

complements. I first searched for each lemma to understand if it was most commonly represented as a dependent or as a governor of the lower verb; then I performed finer searches accordingly and inspected the results to find evidence of complex predicate formation. These inspections were sample-based: I looked at no more than 100 examples of each verb.

Complex predicates are rare in Romance My investigation of the treebanks suggests that the appearance of these monoclausal properties under trigger verbs is rare, at least with most triggers. In general, clitic climbing seems very rare with the object clitics; I found one or two such examples per verb, if any. It is more common with SE, presumably because object clitics are in competition with full nominals, while SE is not. Nevertheless, even climbing of SE is rare.

In both Spanish and Portuguese, long SE passives are at first glance noticeably more frequent with *poder* than other verbs. However, with singular agreement (which occurs more often), they cannot be distinguished from impersonal SE, as explained in Section 4.3.4. One such example from the Portuguese treebank is given in (153).

(153) Portuguese:



What relationship SE can establish.INF with its neighbors?

What kind of relationship can be established with its neighbors?

If we understand this to be an impersonal SE construction, then it cannot be argued that an internal argument selected by *estabelecer* cliticizes on the matrix verb. In that understanding, we would say that the clitic *se* acts as an arbitrary subject of *poderá*. However, if *se* is a passive marker, then we have a long passive, because the subject *que relação* is the internal argument of *estabelecer* and the marker cliticizes on *poder*. The only way to be sure would be if we had a plural subject (*que relações*), as discussed in Section 4.3.4: in an impersonal construction, the verb *poderá* would remain singular even with a plural subject, but in a passive construction, it would agree with the subject.

French In (154), the complement of the causative is represented with the label `xcomp`. The clitic SE, bound by the subject of the main clause, is the internal argument of *entendre*, but it is hosted by the causative.

(154) Afin de se faire entendre , ils effectuent des actes terroristes.
In order to SE make hear.INF , they carry out some acts terrorist
In order to make themselves heard, they carry out terrorist acts.

I did not find examples of monoclausal behavior with *voir*.

Italian In the Italian treebank, there are examples of clitic climbing with the causative *fare*, as well as with *vedere*. In (155a), *si* is the internal argument of *accetare*; in (155b), *si* is an oblique argument of *restituire*.

(155) a. Long object

Aggiunse la particella nobiliare per far si accetare.
Added.3SG the particle noble to make.INF SE accept
He added the nobility title to make himself accepted.

b. Long oblique

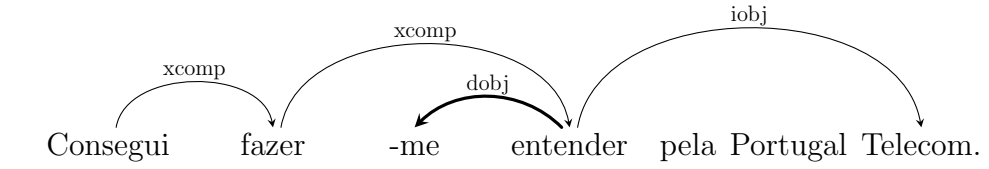
When did Panama see the Panama Channel returned to itself?

The restructuring verbs *podere* and *volere* are both labeled `aux`. I did not find examples with evidence of restructuring under these verbs.

Portuguese I found an example of clitic climbing under *fazer*, shown in (156a), where *me* is the internal argument of *entender*. I found no evidence of monoclausality with *ver* in Portuguese. Under *querer*, example (156b) shows an oblique clitic having climbed up from *dar*.

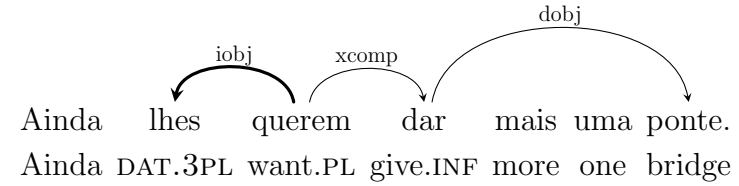
These examples 156 show that Portuguese is the only treebank in which long objects are dependents of the verb that assigns them a semantic role: *me* is the object of *entender* in (156a), and *se* is the object of *acoplar* in (156c). The exception is the *lhes* in (156b), which is not a dependent of the verb selecting it, *dar*. Under *poder* there is more evidence of monoclausality, possibly including the ambiguous example introduced in (153) above.

(156) a. Long object



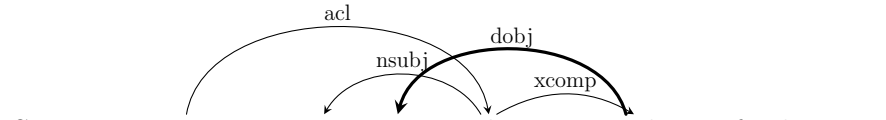
I managed to make myself understood by Portugal Telecom.

b. Long oblique



And they want to give them another bridge.

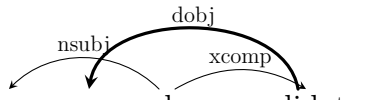
c. Long SE passive



(These) are instruments that can easily be attached.

Interestingly, I also found an unexpected example of inherent SE cliticizing on *poder*, given in (157).

(157)



More people can apply.

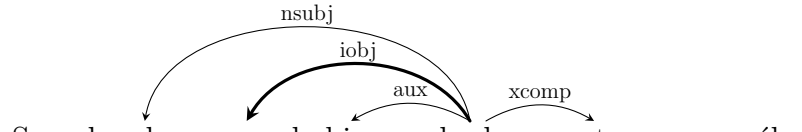
Spanish I was able to find a few examples of clitic climbing with the causative verb *hacer* in Spanish, as shown below in (158a)-(158c). All of these use the clitic SE; the clitic is bound by the subject and represents the internal argument of the complement verb. The label `xcomp` is used to mediate the complementation under the causative verb.

There were no examples of monoclausal properties under *ver*.

In this treebank, *poder* is annotated as an auxiliary, so it is represented as a dependent of its complement. As in Portuguese, complex predicate formation seems to be more frequent with this verb. A long SE passive (or possibly a SE impersonal, since the example is ambiguous) is shown in (158d).

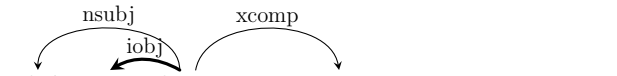
Example (158e) shows a long oblique, *me*, which is associated with *devolver*.

(158) a. Long object



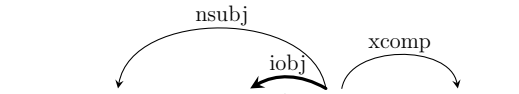
His men had had themselves killed for him.

b. Long object



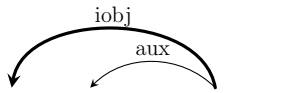
Idrís SE made recognize.INF as emir
Idrís made himself be recognized as emir.

c. Long SE passive



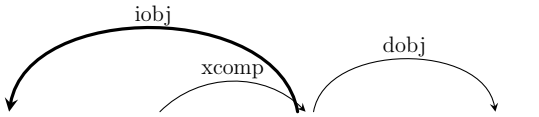
The emirs not SE made wait.INF
The emirs did not make themselves waited for.

d. Long SE passive



Assignments that until this moment SE can.PL realize.INF
Assignments which until now could not be realized.

e. Long object:



Not DAT.1SG wanted.PL return.INF the deposit
They didn't want to give me the deposit back.

4.4.3 Existing literature

The different classes of trigger verbs have been approached differently in most of the syntax literature, due to their distinct properties. In this brief review of key insights from theoretical syntax, I will make a division between restructuring and causative triggers.

Restructuring triggers

As described in Abeillé and Godard's (2003) classification in Section 4.4.1, one type of complex predicate formation between verbs is restructuring, characterized by optional clitic climbing. In this sense, restructuring verbs are fundamentally distinct from both auxiliaries and causatives, under which clitic climbing is obligatory.[8]

This optionality under restructuring triggers is suggestive of two types of infinitival complementation. That raises a fundamental question with respect to these verbs: whether the possibility of two types of complement is due to alternating subcategorization frames, or to a syntactic process that derives one type from the other. (See Wurmbrand 2006 for a broad survey of both types of proposals.) In the first case, complex predicates are monoclausal, because the complement under the trigger verb does not include a clausal boundary. In the second case, complex predicates are biclausal (at least at the start) and acquire monoclausal properties by means of a derivation.

Biclausal approaches The term 'restructuring', which I use here descriptively, is due to Rizzi's (1978) pioneering account of such constructions, in which the two types of complements are derivationally related by application of an operation called restructuring. That is an example of a biclausal approach: it accounts for restructuring constructions as biclausal at one moment of the derivation history, and derives their monoclausal properties via a structure-changing operation that renders the clausal boundary transparent. This type of approach has been abandoned in the GB framework for theory-internal reasons.

Head-movement approaches are another type of biclausal account, exemplified by Kayne (1989). In this type of analysis, a verbal head moves from within the infinitival, which is represented as a full-clause complement, to the matrix verb. This movement forms a complex head, disrupting and reorganizing government relations. This type of head movement cannot be implemented in overt syntax, because it predicts that the complex head should be continuous and, as mentioned in Section 4.4.1, Rizzi (1982) shows that adverbs can intervene between the two verbs. This has led authors to one of

[8]But if the causee argument intervenes between the two verbs, clitic climbing is blocked.

two solutions: either the head movement of the infinitival does not have phonological consequences (which can be achieved by several implementations, including covert movement), as argued in Kayne (1991) and Gonçalves (1998); or the movement is followed by movement of the causative to a higher head (separating the verbs again), as proposed by Guasti (1992) and Roberts (1997).

These types of proposals are difficult to adapt to UD. As was argued before in Section 4.3.3, the notion of a derivation history cannot be meaningfully translated to UD, which is a static representation. The head-movement approach has the additional problem of constituting a syntactic phenomenon with no phonological consequences—which, again, is not a viable or meaningful construct in UD, a representation of surface syntax.

Monoclausal approaches A different perspective on restructuring is to say that such constructions are monoclausal in all their representations, and that the complements of restructuring triggers are smaller than full clauses. Wurmbrand (2006) notes that a monoclausal approach can be instantiated in two different ways, in a Minimalist paradigm: either the trigger verb is a functional head, or it is a lexical head. The former has been argued by Cinque (2004) to be true across all restructuring configurations; Wurmbrand herself, on the other hand, has proposed a mixed approach (2001) in which restructuring can be functional, with some triggers realized as heads on the clausal spine, or lexical, with other triggers taking small, subclausal complements. Many monoclausal approaches (Chierchia, 1984; Wurmbrand, 2001) also tie the size of the complement in syntax to its status in the semantic derivation, proposing that differences between (for example) a proposition and an event are reflected in categorial syntax.

The class of restructuring triggers In addition to disputes about structural properties, another source of divisions in the literature on restructuring has been the question of how the property of being a restructuring trigger comes about. There is significant variation in this set of triggers across languages and across speakers, but a degree of semantic cohesion is undeniable. Wurmbrand (2001) suggests that

classes of infinitival-taking predicates can be organized in an implicational scale of restructuring, such that, for a given language, accepting restructuring in one class of predicates implies accepting it for the lower classes as well. This scale is reproduced from Wurmbrand (2006) in Table 4.5.

Type of verb	Grade of restructuring	Degree of restructuring
Modal verbs	Generally among restructuring predicates	Highest
Aspectual verbs	Generally among restructuring predicates	↓
Motion verbs	Generally among restructuring predicates	
Causatives	Generally among restructuring predicates	
try, *manage*, *dare*	Some degree of restructuring	
(Other) irrealis, implicative verbs	Minimal degree of restructuring	
Propositional verbs	Generally not among restructuring predicates	
Factive verbs	Generally not among restructuring predicates	Lowest

Table 4.5: Degree of restructuring observed, crosslinguistically, in different semantic classes of verbs (Wurmbrand, 2006).

The tension between cohesiveness and variability in the group of restructuring predicates has led to divisions in the literature as well; some authors, such as Aissen and Perlmutter (1983) and Kayne (1991) have argued that the ability to restructure is determined lexically, and therefore highly idiosyncratic and variable; which others, such as Wurmbrand (2001) and Cinque (2004) have described it as semantically motivated and thus highly regular and predictable. (Again, see Wurmbrand 2006 for a thorough survey of positions on this issue.)

Triggers with additional internal arguments

Causatives These verbs have two important characteristics that set them apart from restructuring verbs, as made clear by the classification proposed by Abeillé and Godard (2003): first, clitic climbing out of their complements is obligatory (when the causee argument does not intervene between the verbs); and second, the external

argument of the complement is not identified with the external argument of the matrix verb. Another fact important to the description of causatives is that their complements come in two flavors, known since Kayne (1975) as *faire-infinitiv* (FI) and *faire-par* (FP): in FI, the apparent subject of the complement is marked with the language's equivalent of the preposition *à* from French; and in FP, the subject is marked by (the equivalent of) the preposition *par*. Evidence from binding properties and optionality shows that in FI constructions the logical subject of the infinitive has argumental status, while in the FP construction it does not.

Guasti (2006) shows that there is a relative consensus in the GB framework on the bare-lexical nature of the causative complement, that is, on the absence of functional projections above the lexical verb in the complement. This position would explain why such complements lack a number of properties associated, in that theoretical framework, with the presence of functional projections, such as the ability to host a preverbal subject, negation, or auxiliaries, as well as the possibility of *wh*-movement out of the complement and the assignment of accusative case without participation of the higher verb. The alternation between FP and FI is considered an alternation between a VP or a vP complement, respectively (Guasti, 2006). The crucial difference, in more theory-agnostic terms, is that the latter type of complement has a position for the external argument of the predicate.

The monoclausal properties of causatives, whereby the arguments selected by the complement verb seem to be governed by the causative, have been described by a number of mechanisms, with much attention devoted to reanalysis (Rouveret and Vergnaud, 1980; Manzini, 1983). Another account (Guasti, 1993) appeals to incorporation, which has also been used as a device for modeling morphological causatives (Baker, 1988). Under this account, a single thematic grid is formed from the two verbs as a result of the incorporation. Because incorporation is inconsistent with the separability of the causative from its complement, the proposal pairs incorporation of the infinitive verb into the causative with excorporation of the causative to a higher functional head. (This is similar to the head-movement accounts of restructuring verbs that we saw above.)

Object-control triggers The class of object-control triggers is still somewhat puzzling, and has received comparatively little attention. They share with causative triggers the property of nonidentity between the external argument of the trigger and the complement; however, clitic climbing under object-control triggers is optional, even when the object of the matrix verb intervenes between it and the infinitive.

Moore (1998) and Nishida (2013) point out that Rizzi's original account of restructuring is compatible with the properties these predicates; ironically, accounts designed to replace it (specifically Cinque, 2004) do not. The description of restructuring verbs as lexical heads cannot be maintained for a verb that selects an internal argument and assigns it a theta-role. Cinque acknowledges this but circumvents the issue by appealing to a suggestion from Kayne (1975) that these constructions are actually instances of the causative construction, which would mean that object control triggers uniformly take VP complements.

However, Moore (1998) argues that object control triggers and causative triggers are distinct. His argument is based on three sets of facts: differences in the expression of the causee and of the controller (which according to Kayne's hypothesis would have the same structural position); evidence against a constituent dominating the controller and the embedded complement; and the absence of an alternation in the style of the FP/FI alternation observed in causatives. This author remains agnostic as to the nature of the infinitival complement.

4.4.4 A proposal for Romance complex predicates in UD

When we examine the relation between semantic roles and grammatical functions in clitic climbing and long passives, we see that the argument realized as long object (or long passive subject) in the matrix clause receives a role from the lower verb. The underlying deep relations that reflect the semantic role assignment conflict with the apparent surface relations that reflect properties of grammatical functions. The crucial question for constructing an analysis of these constructions is to what extent the dependency analysis can be made to reflect semantic roles, by the mechanism explained in Section 4.2.

Because the different classes of triggers have different subcategorization frames and different argument structures, I will discuss them separately, from the most challenging case (for UD) to the simplest. Repeating the structure of Section 4.3.4, in each case, I offer a quick sketch of some viable possibilities; after surveying different types of triggers, I consider the question of whether and how to unify them.

Object-control triggers

The first inescapable choice to be made about trigger verbs is whether they are heads of their complements (159a), or (as is currently often the case in the treebanks) dependents of their complements (159b).

(159) a. Spanish:

x

Marta no te lo permitió comprar.
Marta not DAT.2SG ACC.3SG.M allows buy
Marta did not allow you to buy it.

x

b. Marta no te lo permitió comprar.

Let us first consider the possibility of making object-control triggers dependents of their complements, as illustrated in 159b. There are essentially two ways in which this could be implemented, in principle: with the `compound` label (roughly reflecting a head-movement analysis) or with the `aux` label (reflecting a description of trigger verbs as functional heads).

`aux` and `compound` are ruled out There is a powerful objection to this path, which is also the reason why Cinque's (2004) analysis of restructuring verbs as functional heads cannot (and is not meant to) be extended to causatives.

Consider example (160), where *te* is annotated as `iobj`. If we represent *permitió* as a dependent labeled with `aux` or `compound`, it would be impossible for it to

directly be assigned an object, by the definition of these dependency types. With `compound`, we could consider stating that the two verbs form a single unit that has a joint theta-grid. While that is a viable option, it completely misses the generalization about how the roles assigned by this complex unit can be predicted from its component parts (the trigger verb and the embedded verb), creating unnecessary sparsity in the data about predicate-argument relations. Additionally, the use of `compound` here would also hurt a generalization about adjancency of heads in compounding, which could hold in Romance otherwise.

With `aux`, as explained in Section 4.2, the verb represented as an auxiliary is not expected to contribute to semantic role assignment, which makes the label clearly not appropriate for *permitió*. The only viable representation for *te* is as an object of *permitió*: both its syntactic properties, as an object clitic hosted by *permitió*, and its semantic role assignment point to that choice. That verb must then have a predicate label that allows it to have a `iobj`-labeled dependent, as shown in (160).

(160) Marta no te lo permitió comprar. (iobj)

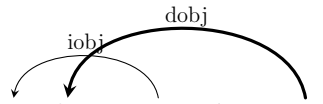
`xcomp` is possible After ruling out both `compound` and `aux` for labeling the structure in (159b), the alternative is to represent these trigger verbs as governors of their complements, as in (159a). In this case, the dependency would receive the label `xcomp`, designated for predicates with bound subjects. The difficulty then is in determining what governors and grammatical functions will be assigned to the other arguments of the complex predicate: *Maria* and *lo*.

Long objects may be attached to the higher verb One obvious thing to do would be to use the labels `iobj` and `dobj` for *te* and *lo*, respectively, and make them dependents of *permitió*, as shown in (161). With this, we uniformly attach clitics to their phonological hosts—with which they are, most often, syntactically and semantically associated.

(161) Marta no te lo permitió comprar.

This is compatible with the principle of representing systematic surface-syntactic generalizations, because the clitics are made dependents of the verb that hosts them. On the other hand, it is clearly at odds with the idea that semantic roles should be recoverable from the triple (GP, GF, SF), because the identity of the predicate *comprar* is needed to understand the role of *lo*. The role for the direct object in this case would be assigned by the complement verb, requiring more information than is available in this triple.

Long objects may be attached to lower verb The first alternative to (161) would be to use the `dobj` label based on semantic criteria, and attach both *lo* and *te* to the verbs that select them (162). (This is the route that was taken in some examples from the Portuguese treebank, as exemplified in Section 4.4.2.) This solves the role identification problem, but it goes against the goal of capturing surface-syntactic generalizations.



(162) Marta no te lo permitió comprar.

The second alternative would be to introduce a distinct (new) label, `x`, for the complement, one that would be defined so as to allow the `dobj` dependent of the object-control trigger to be interpreted as internal argument of the lower verb (163).

(163) Marta no te lo permitió comprar.

Long passives complicate this picture further. Because the examples with object-control triggers are marginal at best, according to Moore (1998), I turn to causatives to discuss long passives.

Causative triggers

If we consider clitic climbing in causatives (164), the picture is very similar to that seen with object-control triggers.

(164) Portuguese:

Eu o mandei arranjar a João.

I ACC.3SG.M ordered fix.INF to João

I ordered João to fix it.

The arguments against representing object-control triggers with `aux` or `compound` labels hold for causative triggers as well. Both *eu* and *João* in (164) receive a theta-role from *mandei*, but `aux` dependents should not assign roles to the core dependents of their lexical governors. An analysis with `compound`, on the other hand, would require us to posit a joint thematic grid, failing to capture a generalization about how the assigned roles can be predicted from the causative verb and from its complement predicate. It is also still the case, as discussed for object-control triggers, that there are problems for attaching the clitic *o* to either verb: attaching it to the higher verb misrepresents syntactic properties, and attaching it to the lower verb misrepresents semantic properties.

The new datum introduced by causatives is that the availability of long passives presents a stronger argument against the possibility of attaching the "long argument" to the lower verb.

Long passive subjects must be attached to higher verb As mentioned in the previous section, one option would be to attach the long object to the lower verb; this is undesirable because it is a failure to represent a general surface-syntactic relation (clitics attach to the verb that governs them) in favor of a semantic one. The failure to capture a syntactic generalizaton seems unacceptable in the case of long passives. Consider this example of a long analytic passive with a FI construction:

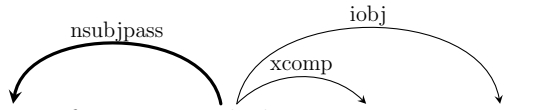
(165) Portuguse:

Os carros foram mandados arranjar a João.

The cars were ordered fix.INF to João.

It was ordered that João fix the cars.

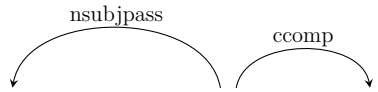
If we decide to attach *carros* to *arranjar*, this would mean rejecting a powerful generalization about verb agreement as a hallmark of subjecthood in Romance. I will rule out this option in order to preserve the representation of grammatical functions. The subject must be attached to the higher verb, as shown in (166); but there is no mechanism for it to be recovered as the internal argument of the embedded verb.



(166) Os carros foram mandados arranjar a João.

FI and FP take `xcomp` and `ccomp` respectively The difference between the FI and FP constructions should correspond to a difference in complement labels. If *a João* is an oblique argument of *mandados*, as has been suggested for FI (Kayne, 1975) and represented in (166), then the external argument of the `xcomp` dependent is identified with that oblique, as the lowest syntactic argument of the complement's governor verb. In FP, on the other hand, we find a closed embedded clause with an arbitrary subject (167), which should be represented as `ccomp`.

(167) Portuguese:



Os carros foram mandados arranjar.

The cars were ordered fix.INF

It was ordered that the cars be fixed.

Restructuring triggers

The arguments put forth above for clause union triggers (both object-control or causative) apply to some extent to restructuring verbs. Control restructuring verbs,

such as *volere/querer/querer*, assign a theta-role to the subject and thus resist an `aux`-based analysis. Additionally, there is an important argument from Wurmbrand (2001) against that type of representation. The argument comes from the fact that a sentence such as (168) is ambiguous.

(168) Italian:

Lui la vuole sposare di novo
He ACC.3SG wants marry.INF again
Again he wants to marry her. OR He wants to marry her again.

The two interpretations of the sentence can be distinguished by attachment of the adverb. In UD, this is evidence against labeling *vuole* as `aux`, because, if we did, it would violate the guidelines to assign it an adverbial dependent. This also applies for clause union triggers.

`aux` is possible for some restructuring verbs There is however, a subset of restructuring verbs that raises new questions, because the same arguments do not necessarily apply. Raising verbs that are not amenable to this type of modification offer no clear arguments against the use of `aux`. Treating such verbs as auxiliaries would greatly simplify the representation of all monoclausal properties of these constructions.

A unified account

My arguments can be summarized as follows: (1) a transfer-style representation for complex predicates, with `aux` or `compound`, is not appropriate for most trigger verbs, because they can clearly have dependents; (2) the case of long passive subjects makes it clear that the surface syntax does not support making long arguments dependents of the lower verb and (3) when a long object or passive subject is attached to the matrix verb, its relation to the lower verb is lost, under the assumptions discussed in Section 4.2.

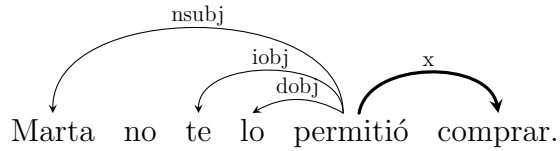
Essentially, my claim is that there are no dependency types in UD that can be used in such a way that all the semantic role assignments under Romance complex

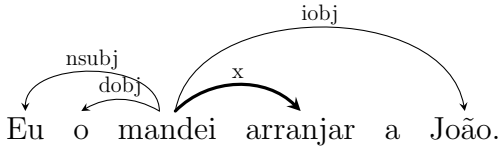
predicates in UD can be correctly inferred from the dependency annotation; the best we can do with the existing labels is to mark some of those assignments.

Two options seem viable: we can either give up on the idea of representing the link between the internal argument and the embedded verb; or we can use a new complementation label to indicate complements whose internal argument may be attached to the matrix clause (be it as `nsubjpass` or `dobj`).

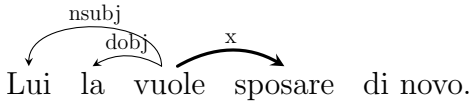
Creating new labels is not warranted for Romance I have argued that there is an important shortcoming in the use of complement labels inside complex predicates: one semantic role assignment relation, from the embedded verb to a long argument, is not recoverable. To the extent that we think of UD as a representation "for semantics," as argued in Section 2.4.1, this is a significant problem. In general, we hope to be able to recover sematic role assignments from surface-syntactic properties.

If we introduced a new label, the examples discussed above would have representations such as the ones sketched below in (169). (Glosses and translations are given in the original examples: (159a), (164), (165) and (168).)

(169) a. 

b. 

c. 

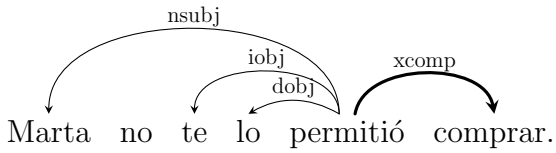
d. 

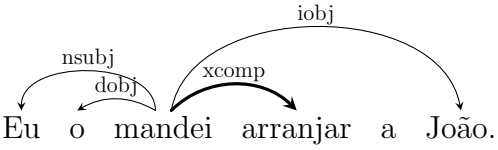
In these examples, we introduce a new label `x` that (not unlike `xcomp`) comes with an algorithm for finding the identity of the labeled predicate's arguments: the internal

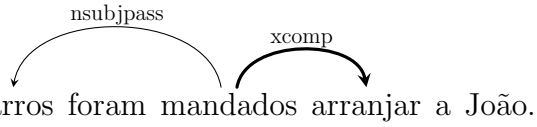
argument of the verb labeled `x` is always the `dobj` or `nsubjpass` dependent of the higher verb; its external argument is the lowest remaining argument of the higher verb, which can be `iobj` or `nsubj`.

In addition to `x`, a counterpart to `ccomp` would also need to be introduced, to correctly account for FP constructions.

However, after examining these four Romance corpora, I believe that complex predicate formation as described by theoretical syntax is very rare with most trigger verbs. The instances where arguments would be missed or misindentified are probably in the low double digits in these corpora of hundreds of thousands of words, and adding two labels to a set of 40 in order to account for them seems unwarranted. With this in mind, my recommendation is to adopt the representations in (170). In these trees, the internal argument of the predicates labeled `xcomp` cannot be identified, but the properties of each label used are preserved.

(170) a. 

b. 

c. 

d. 

Nevertheless, because we know complex predicates are a widespread phenomenon, these arguments are worth revisiting in the future to evaluate the representation of complex predicates across languages. If UD fails to account for common constructions

in other languages, the introduction of new universal labels should be reconsidered, with these facts about Romance in mind.

Some modal and aspectual verbs seem to form complex predicates more often, and those are also some of the verbs for which none of the arguments I presented against the `aux` representation are applicable, such as *potere/poder*. I will revisit such verbs in Section 4.5.3.

4.5 Identifying functional elements

This section continues to examine the challenge of drawing a line between function and content words in French, Italian, Portuguese and Spanish, and of ensuring that semantic roles are recoverable, whenever possible, from the (GP, GF, SF) triple.

In addition to being a recognized *locus* of linguistic variation Rizzi and Cinque (2016), function words are particularly important for syntactic parallelism in UD, as was shown in the analysis of an English/Spanish parallel corpus in Section 3.4. Demoting functional heads can help accomplish parallelism between syntactic and morphological strategies for encoding grammatical meanings, but only if functional heads are defined uniformly across languages that use the syntactic strategy. If there is no agreement about what should be a content head, even in languages as similar as the four Romance languages I approach in this chapter, the content-head design misses its mark in terms of crosslinguistic parallelism.

Although the subject matter is a direct extension of the discussion of complex predicates—we even encounter some of the same verbs—the approach taken in this section is fundamentally different: rather than focus on a particular set of constructions, I look into the current use of the labels `aux` and `cop` in these treebanks. As before, some clearly inappropriate uses of the labels are ruled out on the basis of argument-structure properties; for the defensible uses that remain, I argue that the decision, though to some degree arbitrary, must be taken consistently for copulas and auxiliaries; and make a recommendation of where to draw the line.

4.5.1 The cop label

One distinguishing characteristic of Stanford Dependencies (de Marneffe et al., 2006; de Marneffe and Manning, 2008) was always the treatment of copulas as function words (as discussed in Section 2.4.3), and this now remains a distinguishing characteristic of UD. This has been in part motivated by the possibility, and sometimes the obligatoriness, of copula drop in many languages: the World Atlas of Language Structures lists 175 out of 386 surveyed languages as permitting zero-copula constructions with predicate nominals (Stassen, 2013).

Pustet (2003) notes that the concept of copula is rarely defined; when it is, the definition can include functional, morphosyntactic and syntagmatic properties, but most ubiquitous is the property of being semantically empty, which also explains why some languages make it optional.[9] She offers the following, purposefully vague definition (p. 5):

> A **copula** is a linguistic element which co-occurs with certain lexemes in certain languages when they function as predicate nucleus. A copula does not add any semantic content to the predicate phrase it is contained in.

In many languages, including Romance languages, the copula is a verbal element that serves as a "hitching post" for verbal inflectional categories such as tense. In this sense, it is undeniable that a copula can carry some meaning, as shown in the contrast (171), but the intuition is that this is limited to "grammatical meanings".

(171) a. She is a student.

b. She was a student.

Pustet (2003) points out that this distinction between grammatical and lexical meanings is difficult and does not stand to language-independent examination. Expressing meanings closer to the lexical end of that (language-specific) spectrum we find so-called **semicopulas**: verbs which, like copulas, cannot stand as a predicate

[9] In fact, the central question Pustet raises is why copulas exist at all.

nucleus, but add meanings that are not normally encoded grammatically in the language.

In light of these typological difficulties, Table 4.6 presents the distribution of lemmas labeled as `cop` in our four Romance treebanks, with a cut-off frequency of 5.

French	Spanish	Portuguese	Italian
4862 être	5190 ser	2269 ser	2767 essere
232 devenir	353 estar	377 estar	
91 appeler	80 llamado	176 como	
69 nommer	66 encontrar	93 ficar	
49 rester	48 hacer	41 parecer	
38 dire	37 llamar	31 ir	
25 élire	36 quedar	29 considerar	
21 classer	27 considerar	14 que	
20 signifier	26 ver	14 do que	
20 intituler	26 parecer	13 continuar	
20 constituer	24 resultar	9 servir	
19 terminer	23 denominado	7 tornar	
17 représenter	19 volver	7 permanecer	
14 faire	15 nombrar	7 enquanto	
13 désigner	14 significar	6 quando	
9 sembler	12 titulado	6 chamar	
9 rendre	12 mantener		
8 surnommer	11 denominar		
8 naître	10 permanecer		
8 avérer	9 nombrado		
7 tomber	8 elegir		
7 promouvoir	7 hallar		
7 demeurer	6 considerado		
7 déclarer			
7 baptiser			
6 montrer			

Table 4.6: Lemmas labeled `cop` more than 5 times in each treebank, with respective frequencies.

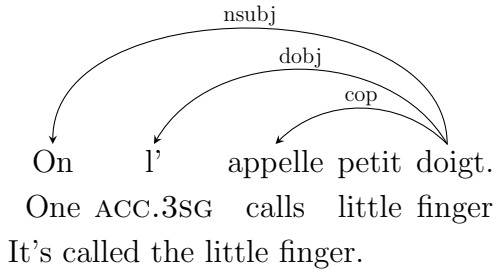
The clear pattern is that the French and Spanish treebanks have very extensive lists of `cop` lemmas; Portuguese has a more limited (although still varied) list, and

Italian takes the direction we adopted in the English treebank, which is to allow a single verb to be labeled as a copula.

Many of these verbs are included in what Abeillé and Godard (2003) name "verbes à attribut du sujet" (subject-attributive verbs, such as *être* and *ser*, but also *sembler*, *quedar*) and "verbes à attribut de l'objet" (object-attributive, such as *considérer* and *llamar*), which make up the fourth category from the authors' classification of trigger verbs, discussed in Section 4.4.1. These verbs select a complement which predicates on the subject or the object of the matrix verb, respectively, and they can form complex predicates.[10]

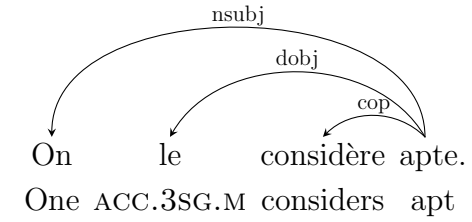
Object-attributive verbs cannot be labeled `cop` I argue that we can easily discard the object-attributive verbs as unsuitable candidates for `cop`-based analyses. This is because a structure in which these verbs are treated as `cop`-dependents lacks the necessary articulation for representing the relation between the object predicate and the object that it predicates on. This is exemplified by the trees in (173), from the French treebank:

(173) a. French:



[10] As I mentioned before in Section 4.4.1, the clitics that climb out of these attributive predicates are noncore dependents. An example would be Portuguese example (172) in which *lhe* is semantically associated with *fiel* and stands in for a prepositional phrase such as *à sua esposa*,'to his wife'.

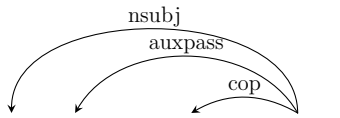
(172) Portuguese:
O João era -lhe fiel.
João was -DAT.3rd.sg faithful.
João was faithful to him/her.

b.


One ACC.3SG.M considers apt
It is considered apt.

Structures such as these put us in the awkward position of representing words such as *apte* as two-place predicates. While we might say that such two-place predicates in conjunction with the `cop` dependent are formed, this argument requires that `cop` dependents be included in the calculation of semantic roles, against our generalizations about this label. Even if we wanted to allow for that, this strategy introduces unnecessary sparsity by requiring that the attributive verb and its complement be considered together for semantic role assignment. (This objection was raised in Section 4.4.4 against assigning the `compound` label to clause union triggers.)

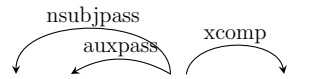
The situation is slightly better when these object-attributive verbs are used in the passive, as shown in (174), and in fact some of the verbs on the list are only labeled `cop` in the passive voice.

(174)


He was considered apt
He was considered apt.

I do not see any reason to adopt fundamentally different structures for the same verb in different voices, when we can reuse the solution adopted for object control triggers to correctly account for active and passive voice, as shown in (175).

(175) a.

b. Il était considéré apte

For the same reason, verbs such as *ver* and *hacer* in Spanish, or *faire* in French, also found in Table 4.6, are not good candidates for a `cop`-based analysis.

The remaining verbs in French, Portuguese and Spanish are subject-attributive raising verbs that are not obviously inadequate for this analysis. These verbs share important characteristics with copulas; in fact, they are the semicopulas mentioned above. They certainly add meaning—these are verbs similar to English *become*, for example; at the same time, we know this is not the type of meaning that is realized as verbal affixes in these languages, which means that in a language-specific classification, there would not be function words. The decision about whether these verbs should be considered copulas hinges on where we want to draw the line between grammatical and lexical meanings in Romance. It is no coincidence that some of these verbs, such as *parecer* and *quedar*, also appear annotated with `aux`.

4.5.2 Interlude: the relation between `cop` and `aux`

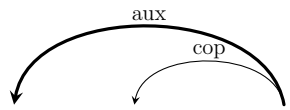
The concept of a functional label in UD is, as we know, based on the concept of functional or grammatical meanings. When we use the `aux` label for verbs that carry certain grammatical meanings, that means we must be willing to accept them as meanings that can be expressed by a copula. The inevitably arbitrary line between grammatical and lexical meanings, once drawn, should be as consistent as possible. Concretely, consider the paradigm in (176).

(176) a. Isso continua a ser um problema.
This continues to be.INF a problem
This continues to be a problem.

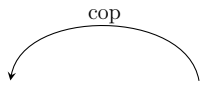
b. Isso continua um problema.
This continues a problem
This continues to be a problem.

c. Isso continua a me irritar.
This continues to me annoy.INF
This continues to annoy me.

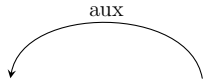
I see no reason to treat these instances of *continuar* as different lexical elements. Starting from that premise, my claim is that there are two consistent choices here: we can say that *continuar* is a grammatical element, leading to analyses in (177), or we can say that it is a lexical element, leading to analyses in (178).

(177) a. 

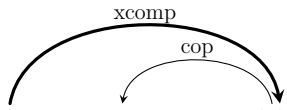
Isso continua a ser um problema.

b. 

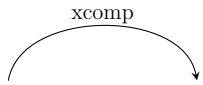
Isso continua um problema.

c. 

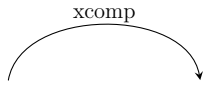
Isso continua a me irritar.

(178) a. 

Isso continua a ser um problema.

b. 

Isso continua um problema.

c. 

Isso continua a me irritar.

Having shown that these two problems are inextricably linked, I now turn to a description of the use of the label `aux`, before offering recommendations for the two labels together.

4.5.3 The **aux** label

As with copulas, it is difficult to provide a crosslinguistic definition of an auxiliary verb. Very roughly, we expect to see verbs that express certain grammatical distinctions such as person, number, tense, aspect, or voice, and that have some syntactic distinction from verbs that express lexical meanings. The semantic component of the distinction helps ensure parallelism in the use of the label across languages; the syntactic component helps ensure coherence within each language. In practice, however, this is still largely unsatisfactory when we consider many languages together.

In Romance, the inventory of `aux`-labeled lemmas in v.1.2 of each corpus,[11] with a cut-off frequency of 5, is given in Table 4.7.

French		Spanish		Portuguese		Italian	
3347	avoir	1855	haber	307	ser	1867	avere
1382	être	898	poder	33	ter	1603	essere
669	pouvoir	293	estar	22	vir	989	potere
305	devoir	179	deber	9	continuar	484	dovere
127	aller	135	ir	8	estar	154	volere
80	falloir	108	tener	6	passar	98	stare
75	vouloir	84	comenzar			32	venire
32	sembler	73	ser			25	andare
31	suivre/être[12]	70	volver				
15	venir	69	seguir				
15	souhaiter	63	soler				
8	voir	53	empezar				
8	rester	34	llegar				
7	savoir	25	venir				
		24	acabar				
		23	pasar				
		20	dejar				
		12	continuar				
		9	terminar				

Table 4.7: Lemmas labeled `aux` more than 5 times in each treebank, with respective frequencies.

[11] With small manual corrections for clear lemmatization errors.

The obvious trend is that the treebank that uses the label most conservatively is Portuguese, and the least conservative is Spanish. In fact, in the conversion of the Portuguese treebank from its original format, it was not possible to recover most auxiliaries (Dan Zeman, p.c.). That explains why the frequency of the label for lemmas such as *ser* and *estar* is so reduced compared to the corresponding lemmas in Spanish, for example, despite the similar size of the treebanks. Due to that historical accident, I will largely dismiss the decisions made in the Portuguese treebank about this, because they were artificially constrained by the information that was in the original treebank and the operations defined in the conversion process.

This list immediately raises some important questions. First: to what extent are we willing to compromise syntactic criteria in favor of semantic ones? Specifically, should the possibility of restructuring be a requirement for the adoption of the `aux` label? One high-impact consequence of such a requirement would be immediately ruling out most of the current and potential uses of the label in French, which lacks restructuring verbs in general. Second, already raised in (4.4.4): should all periphrastic aspect constructions amenable to restructuring be analyzed with the `aux` label? This would license a whole host of aspectual verbs in Spanish, many of which have non-restructuring correspondents in other languages, especially Portuguese.

4.5.4 A unified analysis

My position here will be a cautious one, in favor of limiting the range of copulas and of auxiliaries in UD for Romance. This position is based on the assumption that uniformly taking the conservative position is more likely to lead to crosslinguistic parallelism (of the form presented previously in Section 3.4) than uniformly taking the more liberal position. This is what we can expect if it is the case that grammatical meanings across languages are arranged (at least roughly) along an implicational scale, so that there is a core of basic meanings that are expressed by grammatical means in most languages.

The four Romance languages in question have a clear "default" copula which is semantically empty: *être* in French, *essere* in Italian, *ser* in Portuguese and Spanish.

In addition to those four, two verbs should also be labeled `cop`: *estar* in Spanish and Portuguese.

Ser* and *estar The *ser*/*estar* distinction in Spanish and Portuguese seems like one of few clear examples where we can argue that the "added meaning" introduced is clearly of a grammatical nature. As a first approximation, *ser* is used to predicate permanent characteristics of the subject, while *estar* predicates non-permanent properties. This pair has often been claimed to lexicalize the stage-level/individual-level distinction, but a competing analysis is that it encodes an aspectual distinction (Camacho, 2012). Roby (2009) argues (discussing primarily Spanish) that the distinction encoded in this pair corresponds to a distinction in an aspectual feature, [±Perfective], and that the same feature distinguishes the imperfect and preterite past tenses in these languages, which are differentiated morphologically. That would make these Spanish and Portuguese pairs perfect illustrations of the concept of copulas predicate-forming devices that add only what meaning is normally grammaticalized in the language.

With that, my list of copulas is given below.

French	Italian	Portuguese	Spanish
être	ser estar	ser estar	essere

Table 4.8: Recommended uses of the `cop` label in Romance.

I take the same conservative stance with respect to auxiliaries. The verbs with obligatory clitic climbing, which tend to encode tense and aspect distinctions, are clear candidates for the label. But that leaves open the status of many restructuring verbs.

The set of raising restructuring verbs is an excellent example of the sort of crosslinguistic variation that make drawing this line inevitably arbitrary. While there may be a good case syntactically for considering raising restructuring verbs in Italian, Portuguese and Spanish as functional elements to be labeled with `aux` (as I discussed in Section 4.4.4 and as paralleled by the analysis of these verbs in Cinque 2004), the French counterparts of those verbs do not restructure. This leaves us with no

morphosyntactic argument that can support making a distinction of these verbs as functional elements in French.

Ultimately, I would argue that making a small number of exceptions in French is a good example of a desirable compromise: by letting a semantic criterion "sneak in," we can increase the degree of structural parallelism across these four languages, without seriously undermining any important generalizations about the label. It is important to add that the inclusion of semantic criteria here does not mean that they are prioritized higher than syntactic criteria: as discussed above, the use of the `aux` label is compatible with the properties of these verbs and with the standing generalizations about the label. With that in mind, I propose annotating as `aux` the list of verbs given below.

French	Italian	Portuguese	Spanish
être	ser	ser	essere
avoir	haber	haver	avere
pouvoir	poder	poder	potere
devoir	dever	dever	dovere
	tener	ter	
	estar	estar	

Table 4.9: Recommended uses of the `cop` label in Romance.

This recommendation clearly reflects the intuitions of the treebank annotators: the first four rows of Table 4.7 contain the same verbs in each language, except Portuguese, in which the identification of auxiliaries is very limited.

4.6 Conclusion

This chapter is a first foray into the problems of applying UD, a historically English-centric representation, to different (though still very close) languages with new challenges, and of weighing tradeoffs between conserving the small set of universal relations and adequately accounting for language-specific phenomena. I developed a proposal for the annotation of Romance SE, pointed out UD's limitations in the face

of complex predicates, and argued for a function-word analysis for a consistently small set of verbs across the four languages.

I reviewed some concrete recommendations for the development of guidelines for Romance languages, but it may be that the most important products of this chapter are two case studies: one, on the difficulty of drawing a line between grammatical and lexical meanings, with the ultimate conclusion that the only way forward is to make a practical compromise; and a second, on a paradigm for syntactic argumentation that is pertinent to UD, which developed as I grappled with Romance SE and complex predicates, trying to understand how previously proposed analyses relate to our framework and which properties of the representation are most important.

Chapter 5

Utilizing structure: Event extraction

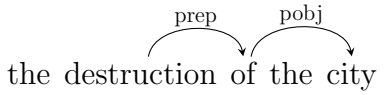
5.1 Introduction

The main concern of this dissertation is studying how design choices made for a syntactic representation for Natural Language Processing (NLP) have consequences for the representation's usage in NLP pipelines. Many decisions about what a dependency representation should be—such as the level of granularity of dependency types—have to be made with an eye to the representation's usage, rather than settled by linguistic arguments. These decisions are made based on high-level linguistic goals, but also to serve practical implementation constraints. I take Universal Dependencies (UD, Nivre et al. 2016) as a starting point, and assume as a fundamental premise that the dependency trees are often not of interest in themselves, but rather are a means to an end—or, more precisely, to multiple ends. This chapter directly investigates one such end, asking the question: is there any way in which we can make UD more suitable for use in downstream NLP applications?

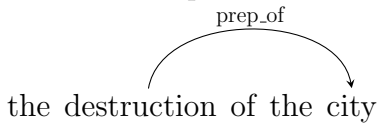
The idea of adapting the dependency representation for downstream performance was foreshadowed in de Marneffe et al.'s (2006) proposal for Stanford Dependencies (SD). That paper introduces a number of versions of SD. The BASIC dependencies, similar to UD, form trees: each word except the root has exactly one head. But

additionally, there were COLLAPSED and CC-PROCESSED dependencies, creating the contrast shown in 179. These alternatives, briefly introduced in Section 2.2.3, were post-processed to be closer to a semantic representation, at the expense of the guarantees that the ouput is a tree and that every word is included in the dependency graph. In the COLLAPSED representation, prepositions and conjunctions are removed from the dependency tree and incorporated into edge labels, which creates more edges between content words. In the CC-PROCESSED version, a dependent with other conjunct dependents has its governor propagated to those dependents.

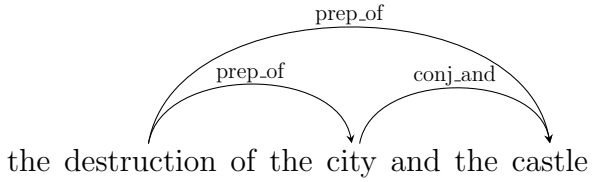
(179) a. BASIC representation:



b. COLLAPSED representation:



c. CC-PROCESSED representation:



The idea behind the use of these modified representations is that sometimes the surface-syntactic relations captured by BASIC dependencies are not isomorphic to the semantic relations that matter for downstream applications. The modified representations address this disparity by pushing out relational words (moving them to lexicalized edges) and "light" content words.

Some existing literature on this topic, discussed in Section 5.3, shows that differences in dependency representations can lead to non-negligible improvements in performance for various downstream tasks, ranging from heavily syntax-based tasks such as negation scope resolution (Elming et al., 2013) to the complex understanding task I target here, event extraction (Miwa et al., 2010a,b; Buyko and Hahn, 2010).

Most of this existing work compares the use of different syntactic representations, such as SD or CoNLL (Johansson and Nugues, 2007), for the particular task. The approach I propose here is different in an important way. Rather than comparing existing representations that have been proposed in the past, from the point of view of a passive consumer, I evaluate modifications that relate to specific design choices in one representation (namely UD), and offer a first investigation of the consequences that these choices may have downstream.

This work is not exclusively for the designers of UD, however. The modifications made to UD here are achieved by means of simple operations for relabeling and restructuring edges in UD dependency trees. They can be applied to existing UD annotation, which means that users can adopt these proposed strategies as a post-processing step for existing dependency trees, potentially adapting them for their particular purposes.

Four types of modification strategies are presented: **conflation strategies**, which introduce new label types that conflate existing distinctions; **headedness strategies**, already seen in Chapter 3, which promote functional heads in exocentric phrases; **enhancement strategies** (due to Schuster and Manning, 2016 and inspired by the semantically enriched versions of SD), which recover implicit syntactic relations; and **path enrichment strategies**, which add information about subcategorization frames to grammatical function labels. I present extensive exploratory experiments on one data set, investigating these strategies in isolation and in motivated combinations, and finally perform selected hypothesis testing experiments on two more data sets.

5.2 Background

The experiments in this chapter compare the performance of an event extraction model that extracts dependency features from a given set of parses to the performance of a model in which the same features were extracted from a different parse. The models are produced with a system for event extraction that took first place in the BioNLP'09 Shared Task (Kim et al., 2009), the Turku Event Extraction System

(TEES, Björne et al., 2011; Björne and Salakoski, 2011, 2013). Some of the existing literature on extrinsic evaluation of dependency representations has also used systems that participated in the same task. There are multiple reasons for the interest in this task: (1) it is a complex understanding task, and therefore a good example of an end task for an NLP pipeline; and (2) it is a task in which the use of dependencies was shown to add value to systems (Kim et al., 2009). The TEES system is an interesting candidate for the evaluation for practical reasons, such as the fact that (1) its implementation is completely representation-agnostic, making it suitable for comparisons; but also because (2) it uses dependencies in two ways in which many other relation extraction systems use dependencies: by extracting path n-gram features, and by using dependencies to establish a context for bag-of-words features; (3) it is modular, and makes use of dependencies in different ways for different subtasks, thereby representing more than one possible use case; (4) its use of dependency features has a solid basis, presented in a published comparison (Björne et al., 2009) between the features used and potential alternatives.

This section introduces both the shared task and the TEES system, which will be mentioned throughout the chapter.

5.2.1 The BioNLP'09 Shared Task

The BioNLP'09 Shared Task (Kim et al., 2009) was an event extraction competition in the domain of biomolecular events, broadly defined as changes of state to genes or proteins. The goal of the task is to identify all events in a corpus made up of abstracts of biomedical scientific literature, extracted from the PubMed platform. There were other editions of this task in 2011, 2013 and 2016.

Events These events are biological processes involving entities, and they are typed according to how many participants are involved, in which roles, and whether their participation is obligatory or optional.

Textually, events are realized by linguistic expressions called **triggers**. They are typed as one of nine categories. Five of those (Gene expression, Transcription, Protein catabolism, Phosporylation and Localization) take a single Protein-typed

Theme argument. A sixth type is a multiparticipant event (Binding) which takes an arbitrary number of Theme arguments, all typed Protein. These six types comprise the **Simple** events.

There are also three **Regulation** events (Regulation, Positive regulation and Negative regulation) that take two arguments, one Theme and one Cause. Each of these arguments can be a Protein or another event. Note that Regulation events are the only type of event in which determining the type of an argument is relevant; in the other event types, either an entity is a Theme argument, or it is not an argument. Events can have optional modifiers,[1] Location and Site (i.e., region of a protein involved in an event). These modifiers are associated with Theme arguments, and match them in number. An optional negative or speculative **Modification** may take scope over an event.

Data and task definition The data for this task is a portion of the GENIA corpus (Kim et al., 2003) which I will refer to as the GE09 corpus. The corpus includes named entity annotation of protein, gene and RNA types, so identifying Protein entities is not part of the task. Event triggers, Location and Site modifiers, on the other hand, all have to be identified. Negation and speculation do not need to be associated with specific linguistic triggers. Importantly, triggers may be shared by events.

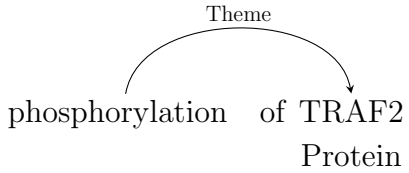
Task 1 The identification of different parts of events is divided into subtasks. Correctly identifying triggers and linking them to the given protein arguments consists of Task 1. A single-argument event typed Phosphorylation is shown in (180).

(180) **Input:**

phosphorylation	of	TRAF2
		Protein

[1] In the task definition, these are actually called event participants, and what I refer to as modifiers below are called arguments. I am adopting a different terminology, at the risk of some confusion, in order to be more consistent with linguistic theory.

Output:



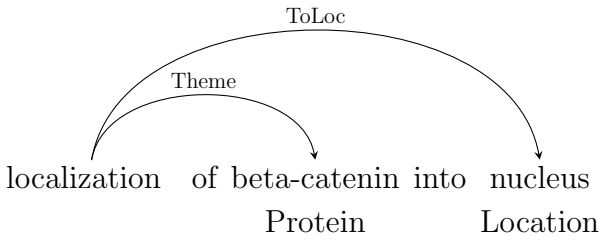
Phosphorylation

Task 2 Identifying the Localization or Site modifiers, attaching them to the correct event, and typing of the modifications consists of Task 2. A Localization event with a modifier typed ToLoc is shown in (181).

(181) **Input:**

localization of beta-catenin into nucleus
Protein

Output:

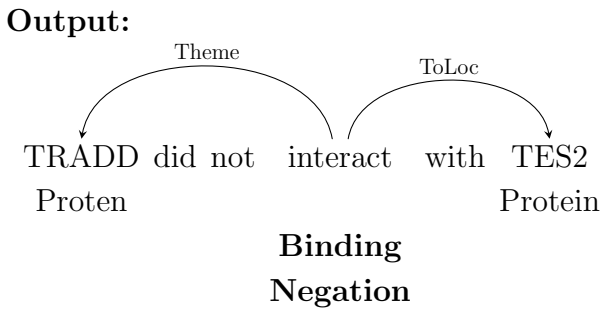


Localization

Task 3 Simply predicting Speculation or Negation of an event consists of Task 3. (Note that this implies that a negated event is still an event to be identified in Task 1.)

(182) **Input:**

TRADD did not interact with TES2
Protein Protein



There are multiple official evaluation metrics for the task. All require that event types be the same in predicted and gold events, but there are different criteria for trigger identification and argument identification, varying in strictness. The relaxed criterion for trigger identification is that the predicted trigger be contained anywhere within a window that extends one additional word beyond each side of the gold trigger, and it was adopted in the official metric of the shared task; in the case of (181), for example, *localization of* would be accepted as a correct trigger. The relaxed criterion for argument identification is that only Theme-typed arguments have to match the gold description; in the case of (181), only the identification of the Theme edge would count towards the score, not the ToLoc edge. There is also a **strict** evaluation mode, in which every aspect of the predicted event, including all arguments and modifiers, their types and text spans, is required to match the gold standard. This is the evaluation I report for all experiments in this chapter.

5.2.2 The Turku Event Extraction System

TEES was the best-performing system in the BioNLP'09 Shared Task. It consists of a pipeline of pre-processing followed by trigger recognition, argument detection, event post-processing and detection of speculation or negation. The pipeline for TEES 2.1, used in my experiments, is illustrated in Figure 5.1.

The trigger detection step identifies words that denote events. In edge detection, each pair formed by a trigger and a pre-identified Protein (given as part of the task input) is evaluated and classified as an edge or not an edge. The unmerging step ensures that constraints for well-formedness of events are met. Finally, negation and speculation modifiers are identified for remaining events. More detail about

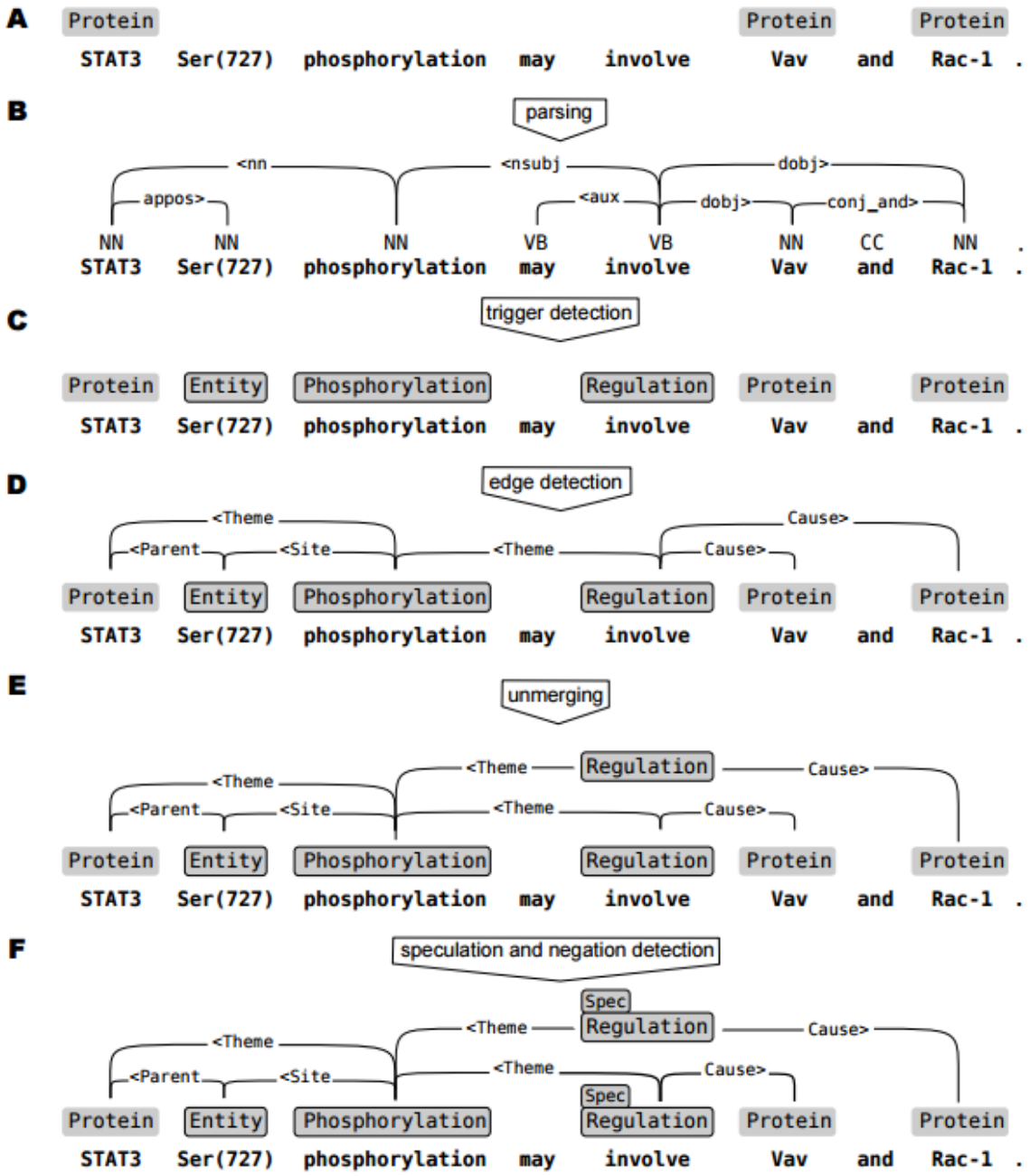


Figure 5.1: A schema of the TEES 2.1 pipeline (Björne and Salakoski, 2013).

these steps is given below, and a discussion of the features used in each classifier follows. In TEES 2.1, all steps are performed with SVM^{multiclass} with a linear kernel (Tsochantaridis et al., 2005).

Trigger detection TEES uses a NER-like classifier to identify trigger words. Each token is classified into a type of trigger or a negative class, based on features discussed below. The authors note that there is an important frequency-recall tradeoff to be made in this component: an event whose trigger is not identified will be missed entirely; but, because the edge detection component tends to find arguments for any trigger, false-positive triggers will usually propagate into wrong events, which is also undesirable. For that reason, they use a hyperparameter (tuned with a grid search) to optimize this tradeoff for the entire pipeline, rather than simply attempting to maximize performance for trigger detection itself.

Edge detection After triggers are identified, each trigger is paired with each named entity and each other trigger in the sentence, and the pair is classified into a type of argument or a negative. Pairs are classified independently, so there is no information about other arguments at classification time.

Unmerging and modality detection Finally, a post-processing step, which was rule-based in Björne et al. (2011) but evolved to learning-based in Björne and Salakoski (2011),"cleans up" these pairs in order to ensure that each event has the types of arguments that it needs to have by definition, with triggers duplicated when needed, and malformed events pruned out. This is non-trivial because a single linguistic trigger can correspond to more than one event in this task, and so a trigger associated with multiple Theme arguments, for example, can correspond to several well-formed events, or perhaps one malformed one. In the example in Figure 5.1, the trigger *involve* is initially associated with two Cause arguments; the unmerging step creates two events sharing this trigger, and each Cause argument is the assigned to one of them.

After these three steps, each event identified then goes through a classifier that

detects modifiers of negation and speculation, which is very similar to the trigger detection classifier.

The classification tasks are performed with a multi-class SVM with a linear kernel; the regularization parameter C is optimized with a grid search on the development set.

TEES processes sentences independently and cannot extract events expressed across multiple sentences, which make up about 5% of the annotated events in the task data.

Feature sets in TEES What makes TEES interesting for extrinsic evaluation of dependency representations is that it relies very heavily on dependency features, for both trigger and edge detection. The most important part of its implementation, for present purposes, is the definition of these features. They are particularly interesting because the system uses dependencies in varied ways: as a way to provide context for a word (complementing the linear context of preceding or succeeding words), as an annotation on a word (interpreting the dependency type as a word tag), and as a way of characterizing relations between words (by specifying the dependency path betweent them). In Björne et al. (2009), the authors show that this feature-based approach outperforms another setup based on an SVM with a graph kernel.

The features for **trigger detection** fall into three categories: token features, which comprise character n-grams, word shape, stems, etc.; frequency features, such as the number of named entities in a sentence or within some fixed proximity of the token in question; and dependency-based features. The dependency features are extracted for each token within three hops from the candidate trigger in the dependency parse. The features extracted for each of those tokens consist of token features (as defined above), the dependency type of the token, and the sequence of dependency types in the path to the token.

The feature set for **edge detection** makes extensive use of dependencies. The most important source of features is the shortest (undirected) path between the head word of the trigger and the head word of the candidate argument. (Head words are identified "with a simple heuristic," according to the authors.) When multiple

shortest paths exist, all paths are used. This is crucial for some of my experiments, as explained in Section 5.5.2.

Again we have frequency features for the number of entities and triggers of each type in the sentence, and an additional frequency feature for the length of the shortest path. There are features combining the attributes of the nodes (type of event/entity). Then, for each token and each edge in each shortest dependency path from the trigger to the candidate argument, features are built that combine the position of that token or edge in the path (that is, whether it is an interior or terminal node in the path) and attributes of the token or edge. Dependency n-grams of length 2–4 are extracted (directed from trigger to argument), both for tokens in the path and for edge types in the path. Finally, triples are built representing each edge and each token in each path. The token triples combine the token itself and the dependency types of the two edges that touch it in the path; the edge triples consist of the edge's type and the two tokens the edge connects. This means that reducing the length of paths between arguments and triggers may reduce the noise in classification.

In the shared task, the parses came from a domain-adapted parser (McClosky and Charniak, 2008) and were converted to CC-PROCESSED SD with the Stanford converter de Marneffe et al. (2006).

TEES performance Although 3 evaluation metrics are used in the task, as described in Section 5.2.1, the most interesting one is the strict evaluation, in which an event is considered correctly extracted if the trigger span is exactly correct and all the arguments are recursively correct. The system obtained 47.41% F-score on the task's official test set by this metric.

The authors note that errors are roughly evenly distributed between trigger detection and edge detection errors.

5.3 Related work

Elming et al. (2013) address the varied ways in which dependencies can be used downstream by offering a systematic extrinsic evaluation of four dependency representations in five NLP tasks.

The representations tested are Yamada-Matsumoto (Yamada and Matsumoto, 2003), CoNLL 2007 (Johansson and Nugues, 2007), Google Universal Dependencies (McDonald et al., 2013), and LTH (Johansson and Nugues, 2007). These schemes make different choices with respect to four parameters: choice of head between (1) auxiliary and main verb; (2) complementizer and verb; (3) conjunction and conjuncts; and (4) preposition and noun.[2] Additionally, when possible, baselines without syntactic features are included.

The same pre-processing pipeline is used in all the tasks, and MateParser (Bohnet, 2010) is used for producing dependencies. The fact that the same parser is used allows for the effects of the representation to be separated from the effects of particular parser models, but the authors note that it could potentially introduce a bias.

The tasks are: negation resolution, in which negation cues and their scope should be identified; semantic role labeling for verbal predicates; reordering by parsing for SMT, which involves predicting the word order of a translation based on features of the source; sentence compression, a form of summarization; and perspective classification, a type of authorship attribution task. Each of these utilizes syntactic information in a different way: in SMT and sentence compression, dependency labels are used as token-level annotations, akin to part-of-speech (POS) tags. In negation resolution, dependency paths are used. In perspective classification, dependency edges (i.e., triples including the words on each end and the type) are used as features. Finally, in SRL many different types of dependency-based features are used.

In the tasks of negation resolution, SRL, sentence compression and perspective classification, one scheme was significantly better—namely, Yamada-Matsumoto in negation resolution (by 2 out of 3 metrics, with LTH barely winning in the third),

[2]This paper focuses on the issue of lexical *vs.* functional heads, much as Chapter 3 of this dissertation; but instead of exploring an intrinsic evaluation, as that chapter does, the paper presents an extrinsic evaluation.

sentence compression and again in perspective classification; Google Universal Dependencies in SRL in 3 out of 4 conditions (the conditions being using gold or predicted POS tags in one of two test sets), with LTH outperforming it in the fourth. In reordering for SMT, no representation was a clear winner, and all offered a significant improvement over a no-syntax baseline.

Generally, the authors note that the number of labels in the representation does not seem to be a big predictor of its usefulness, and also that syntax-oriented annotation (with functional heads) and semantic-oriented annotation (with lexical ones) do better in different tasks.

Miyao et al. (2008) present an extrinsic evaluation of eight (dependency and constituency) parsers with different native representations. Two are native dependency parsers, four are constituency parsers, and two are HPSG parsers. The target task is Protein-Protein Interaction (PPI), a simple BioNLP information extraction task in which a pair of proteins is classified into interacting or not interacting, based on textual features. Automatic conversions are used to make the same parser output available in different representations.

The F-scores of the resulting systems lie between 52.5% and 58.4%, which shows that choosing the right parser and representation can make a big difference for a downstream task. Importantly, the Penn Treebank (PTB, Marcus et al. 1993) constituent trees yield systematically worse performance in the task, for all six parsers that produce it. The absolute difference in F-Score with respect to other representations (without changing the parser) is as high as 4.7% F-score. Their results also indicate that CoNLL allows better performance than SD.

Interestingly, an experiment where features from different representations of the same parse were used together showed that this method produced better results than obtaining single-representation features. It seems that different representations can make complementary contributions.

Miwa et al. (2010b) set up an extrinsic evaluation of different parsers and dependency representations by plugging them into a BioNLP event extraction system for the GENIA task in BioNLP'09. The authors propose a comparison of three dependency representations, and six parsers. In addition to comparing these dependencies in the

context of the shared task, the authors also provide an intrinsic evaluation of the parsers against the GENIA treebank, which allows them to discuss the relationship between intrinsic and extrinsic evaluation.

The system used for the evaluation is decribed in Miwa et al. (2010c), developed for Task 1 of the BioNLP'09 GENIA track. The system is based on TEES, but replaces the rule-based heuristic used in earlier versions of TEES for separating triggers and arguments into well-formed events with a learning-based classifier.[3] This system was shown to outperform TEES on the BioNLP'09 data.

In Miwa et al. (2010b), six parsers are evaluated. These parsers produce representations based on different linguistic frameworks and vary between dependency-native and constituency-native. The authors applied automatic conversions to the outputs of the different parsers in order to obtain SD and CoNLL-X dependencies. Additionally, they evaluated Enju's (Miyao et al., 2008) native predicate-argument structure (PAS) output, which is similar to dependencies in that it is also based on bilexical asymmetric relations; but this representation was evaluated only in conjunction with the Enju parser, and the output of other parsers was not converted to it. The authors also converted the gold-standard parses from the GENIA treebank to evaluate the different parsers.

The intrinsic evaluation shows that Enju is the best-performing parser, and that, unsurprisingly, the domain-adapted parsers perform better.

In the extrinsic evaluation, the authors make a few interesting observations. The best-performing representation overall is PAS, but it is also only produced by one parser, Enju, which is the best-performing parser; it is unclear whether this advantage is truly due to the representation, or if it is due to the parser. While the PAS-represented parse does fare better than its conversions to CoNLL and SD, this could be attributed to lossy conversions.

With respect to SD, the authors test system performance on the four conversion formats that were offered at the time—BASIC, CC-PROCESSED, COLLAPSED, and a combination of the last two. They found that BASIC dependencies performed best, which is quite unexpected: as mentioned in Section 5.1, the other conversion styles

[3]TEES 2.1, used for all the experiments that follow, adopts this learning-based approach.

were developed specifically for use in information extraction. This result held across parsers, so they use the BASIC dependencies to compare SD to other representations.

Overall, comparing parsers and the gold-standard annotation, the authors note that there is little room for gain from improvement in parser performance. The results with the parsed trees from Enju are only slightly worse than the results with the gold trees.

CoNLL and SD are roughly tied: the second best parse for Task 1 is represented in SD, and the second best parse for Task 2 is represented in CoNLL. The differences between these two representations are not large: only 0.08% F-Score for Task 1, and 0.24% in Task 2.

They also run experiments with untyped versions of the parses. Overall, the untyped CoNLL does better than the untyped SD for most parsers, with a few almost-ties; untyped SD shows a larger drop in performance with relation to the typed version, which may mean that SD's type system is more useful than CoNLL's.

Miwa et al. (2010a) overlaps with Miwa et al. (2010b). The contribution specific to this paper is an evaluation of the use of parser ensembles in the event extraction system, which the authors show to be effective. Their results indicate that parser ensembles work well only when the parses are not only produced by different parsers, but only represented in different dependency standards. When either of those factors is the same in the ensemble of parses, no gains are observed. This is a different conclusion on ensembles than that from Miyao et al. (2008); those authors found, as mentioned before, that even adding only a different representation brought gains.

Commenting on the comparison of the different versions of SD, the authors write that "dependencies are generalized by removing expressions after '-' of the dependencies (e.g. 'with' in prep-with)" (Miwa et al., 2010a, p. 783), which seems to indicate that they erased the lexical information that is built into the dependency edges in the non-basic versions of SD. If this is the case, it is not surprising that BASIC SD performed better. When prepositions are collapsed, they cease to be nodes in the dependency trees to become part of the lexicalized edges. If the edges were delexicalized, then all the information about prepositions would have disappeared from the dependency trees.

This paper also includes evaluations for untyped versions of the different SD-based representations, and the authors note that the scores for BASIC dependencies go down more than for the other versions; in fact, there is a small increase over the typed trees for the COLLAPSED dependencies. They suggest that typing errors or sparseness could be preventing the system from taking advantage of the collapsed edges.

Buyko and Hahn (2010) also provide an extrinsic evaluation of dependency representations in the context of the BioNLP'09 event extraction task, noting the convergence of the top-performing systems on the use of dependencies. The evaluation is based on the JulieLab system (Buyo et al. 2009), which placed second in the competition making use of CoNLL-based dependencies obtained with the GDep parser (Sagae and Tsujii, 2007). The authors compare the use of this representation with the use of SD, verifying that the CoNLL-X representation yields higher performance; they then propose modifications to CoNLL-X, in the spirit of SD's COLLAPSED representation, that lead to further improvements.

They test six parser-model combinations. CoNLL outperforms SD in all cases, by as much as 1.4% in F-Score. As in Miwa et al. (2010a,b), BASIC dependencies fare better than the other modes of SD, although in this case there is no comment on delexicalizing collapsed edges. In error analysis, the authors look at dependencies that are overrepresented in the SD-based systems' false positives as opposed to the best system, and observe that the collapsing of some prepositions and the presence of `nsubjpass` characterize those false positives.

In terms of modifying the CoNLL parse, the modifications proposed are four: demoting auxiliaries and modals (in the style of BASIC SD); creating lexicalized dependency types for prepositional complements (in the style of COLLAPSED); propagating dependencies of a word to its conjuncts (in the style of CC-PROCESSED) and adding structure to adjectival phrases such as "IL-10 induced", making the noun dependent on the adjective rather than being its sister, as in the original representation. The authors break down the events in the task by category and show that different combinations of modifications are helpful in each one.

Overall, it seems that auxiliary demotion and conjunct-dependency propagation are more consistently beneficial across the different types of events, while the other

two enhancements are only helpful in one event class each. Enhancements are not always beneficial.

In conclusion, the authors note that the differences between representations are wider than the differences between parsers. Their final approach is to treat representation modifications as a hyperparameter to be tuned per event class, leading to a performance that beats the then-state-of-the-art system of Miwa et al. (2010) by 2.6% points in F-Score.

5.4 Methodology

The experiments presented in this chapter are divided into two phases. The first phase consists of exploratory experiments performed on the GE09 corpus. In this phase, I test 50 different representations, each of which introduces a minimal change to the UD representation. I also establish 3 baselines, one with UD dependencies and two with impoverished versions, described in Section 5.5.1. These are versions which encode less structural information than the normal UD dependencies. These results are used to drive error analysis and motivate combinations of minimal changes. I then test 6 motivated combinations, and at the end select the 5 most promising representations for the second phase of experiments.

These 5 finalists are then tested in a generalization phase. In this second phase, the evaluation targets are two event extraction tasks from the BioNLP'11 challenge. Both draw from the same domain as GE09 and have the same type of event schemes; the data sets are new, although GE09 is added as training data for one of the tasks, as explained in Section 5.6.1. If the success of the selected transformations on GE09 is motivated by linguistic principles, gains should be expected to generalize for these new corpora and tasks—and hopefully beyond.

All experiments in both phases are performed with 10-fold cross-validation on the union of the training and development sets from the task. This is because, probably due to the high sparsity of the data, there is very high variance in the performance of the system from fold to fold, as shown in 5.5. Cross-validation was implemented differently in the two phases: in the exploratory experiments, each fold was partitioned

90/10, and the results reported are the development-set results after hyperparameter tuning. In the hypothesis testing experiments, each fold is partitioned 80/10/10 and both development- and test-set results are reported.

The metric used for comparisons in this chapter is the absolute difference in F-Score between a system trained on a transformation and the BASIC baseline, as measured by the *strict* evaluation from the shared task, explained in Section 5.2.1. I found that the different metrics appear to be strongly correlated, so the choice of metric probably does not change the results of the comparisons.

All p-values reported are estimated with paired bootstrap tests across all folds, run with 10,000 iterations. In the exploratory experiments, I did not apply a multiple comparison correction;[4] the discussion of the results refers to any result with $p < 0.05$ as **significant**. In the generalization phase, which is geared towards more rigorous hypothesis testing, I applied a Bonferroni correction for 5 experiments within each data set.

5.5 Exploring the data

This section reports on exploratory experiments with the GE09 corpus. The baseline parse was obtained by using v.3.6.0 of the Stanford Converter, which implements UD, on the constituency parses of the data, produced with the parser described in McClosky and Charniak (2008). This is the same method that TEES has always used in it submissions to the different editions of BioNLP.

The GE09 data set includes nine types of events, as explained in Section 5.2.1. These events are not distributed uniformly in the corpus; Table 5.1 shows the break-down of events by type.

Table 5.2 shows a break-down of event triggers in the training set of the GE09 corpus by coarse parts-of-speech. As is clear, the triggers are mostly nominal, almost 50% more often they are than verbal. A smaller proportion are adjectives.

Table 5.3 shows the frequency of dependency types in argument-trigger paths

[4]Because no correction was applied, I include the estimated p-values in the reporting of the results.

Event Class	freq
Localization	174
Binding	347
Gene expression	722
Transcription	137
Protein catabolism	14
Phosphorylation	135
Regulation	291
Positive regulation	983
Negative regulation	379
Total	3182

Table 5.1: Distribution of events by type in GE09.

Coarse POS	Freq.
Other	27
Preposition	57
Adjective	713
Verb	2301
Noun	3380

Table 5.2: Distribution of triggers by coarse POS tags in GE09.

and in the corpus as a whole. The prevalence of nominal triggers is reflected in the dependency types that link participants to events. It is clear that prepositional modifiers and adjectives, two of the three most common dependency types, are very important in these event representations. Predicate arguments occur less often overall, but have relatively high probability of appearing in a path. There is a noteworthy asymmetry between subjects and objects: the type `dobj` is over two times more likely to appear in a path than `nsubj`. There is also an interesting asymmetry between `nsubj` and `nsubjpass`: passive subjects are roughly two times more likely to occur in a path than their active counterparts.

Table 5.4 shows how the lengths of dependency paths from event triggers to event arguments are distributed for each argument (or modifier) type. Overall, the overwhelming majority of arguments have a single-edge path to their respective triggers, which indicates that dependency structures are very close to event structures.

5.5.1 Baselines

Representations

I establish three baselines for comparison: one using BASIC UD, which I take as a basis for my transformations and the primary reference for all comparisons; and another two with UNTYPED dependencies and TRIVIAL dependencies. The UNTYPED dependencies preserve the structure of the BASIC trees, but are stripped of all type distinctions; the TRIVIAL dependencies are restructured trees in which there are not only no type distinctions, but also no meaningful head-dependent relations: each word is simply a dependent on the previous word, which implies that dependency paths amount to windows between the target nodes, and the dependency context of a word reduces to its linear context.

Results

Results for the three baselines are given in Table 5.5. As expected, untyped dependencies perform worse than typed ones, and trivial ones worse still. It is interesting that the losses are relatively small: only 1.39% absolute difference between BASIC or

	All Types	Site	Theme	ToLoc	AtLoc	SiteParent	Cause	In Corpus
amod	8.73	0.20	7.26	0.11	0.09	0.58	0.50	25549
case	0.37	0.01	0.29				0.06	24649
nmod	16.42	0.95	13.22	0.05	0.08	0.48	1.63	23843
punct	0.01		0.01					20383
compound	7.36	0.35	5.59		0.04	1.07	0.31	17450
det	0.02		0.02					14522
nsubj	10.59	0.44	6.38		0.01	0.19	3.58	8439
conj	14.54	0.45	11.90		0.03	0.41	1.76	7151
dobj	24.03	1.23	21.29		0.04	0.33	1.14	6684
advmod	1.04	0.07	0.82		0.02	0.04	0.09	4499
mark	0.02		0.02					4066
nsubjpass	20.46	0.77	18.25		0.04	0.11	1.29	2712
appos	7.98	0.15	6.76			0.31	0.76	2619
nummod	10.06		8.58			1.06	0.42	2366
acl	15.97	0.71	12.82		0.04	0.40	2.00	2254
dep	9.27	0.39	6.90		0.05	0.79	1.13	2028
cop	0.12		0.06		0.06			1687
ccomp	3.55	0.18	2.95			0.12	0.30	1661
advcl	11.03	0.48	7.17				3.38	1659
acl:relcl	15.85	1.60	11.72				2.53	1186
xcomp	14.35	0.70	9.57				4.09	1150
neg	0.26		0.13				0.13	763
nmod:poss	0.28						0.28	716
mwe	0.26		0.26					391
parataxis	15.32	2.42	12.10				0.81	124
iobj	20.00	2.11	14.74			1.05	2.11	95
discourse	2.63		2.63					38
csubj	3.70						3.70	27

Table 5.3: Percentage of instances of a dependency type that occur inside a path from a trigger to an argument in the training set of GE09, by argument type. The last column shows the absolute frequency of that label in the corpus (whether or not it appears in an event-argument path).

l	Frequency
0	63
1	4083
2	2147
3	826
4	337
5+	129

Table 5.4: Number of event-argument paths per path length. (In the case of 0-length paths, the trigger word also expresses the argument.)

F Δ (dev)	p	*modification*
0.00	-	BASIC
-1.04^*	<0.0001	UNTYPED
-1.39^*	0.0001	TRIVIAL

Table 5.5: Results for baseline representations. In this and all other tables, **F** Δ refers to the absolute difference in the F-Score obtained by the model, across all folds, with respect to BASIC. A negative difference indicates that the model is worse than that baseline; a positive difference indicates that it is better. The p**-value** given for the difference is obtained with a paired bootstrap test. Results with $p < 0.05$ are marked with a *.

TRIVIAL. The gain obtained from adding dependency types is larger than the gain from adding structure.

This is probably due to a combination of factors. One factor is the language: English syntax relies heavily on word order, which means that the linear context of an event trigger is normally similar to the dependency context. This being the case, the dependency tree without labels may not add very much beyond the order of words. Another factor is the nature of the task definition: protein recognition is given as a precondition, which narrows the search space for event edges. This makes it unnecessary to discover, for example, that event arguments tend to be nominal, or that they appear in argumental positions with respect to even triggers—the only candidates being considered are pre-identified proteins, which will in general already display those characteristics.

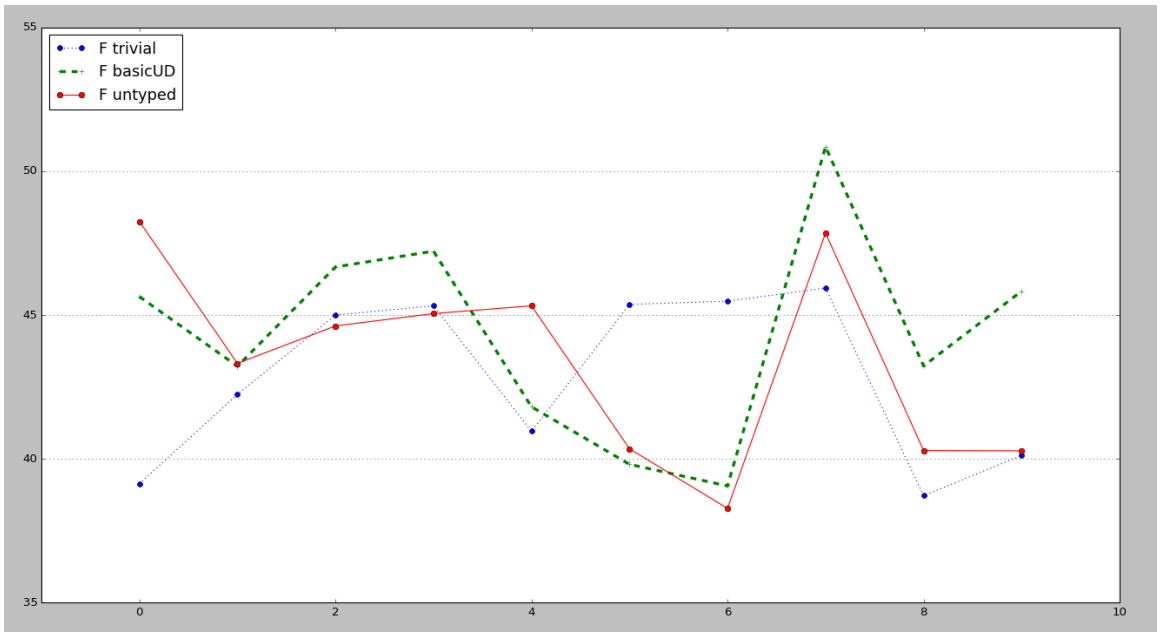


Figure 5.2: F-Score from fold to fold (numbered 0 through 9) for the baseline representations TRIVIAL, BASIC and UNTYPED.

These three baselines are a good example of the variation of the results across folds. Figure 5.2 shows a plot of the F-Score of each baseline across the 10 folds. Even though BASIC is a clear winner overall, there are folds in which it performs worse than UNTYPED and even TRIVIAL. It is possible that those folds have more parser errors, which would make the parsers less useful. So, for example, if we happened to be considering only folds 5 or 6, it would appear that having no dependencies at all is beter than having untyped or typed dependencies; on the other hand, if we looked only at fold 0 or fold 4 (or even at both together), then untyped dependencies would appear to be better. This volatility in the rankings is why all the experiments in this chapter were performed with cross-validation: varying the data partition makes the results more likely to generalize to other data sets.

5.5.2 Conflation strategies

While UD's rich label set can be an asset for writing precise syntactic patterns, it also introduces sparsity in statistical learning. Some label distinctions are far beyond

core clausal syntax and may not be relevant for relation extraction—the difference between `vocative` and `discourse` comes to mind, for example. Some distinctions probably serve only as a source of noise. The conflation strategies introduced in this section reorganize the UD type system, undoing distinctions that may not be useful for the task while preserving linguistic coherence.

The number of different partitions of a set of 40 labels is evidently quite large,[5] so there is no hope of attempting an exhaustive search over all possible label conflations and their combinations. Instead, these transformations are based on linguistic intuitions and make reference to the feature decomposition of UD types, introduced in Section 2.4.2. They target cohesive subsets of labels and collapse them into a single label.

For all these label-based strategies, whether they split or conflate labels, there are two options: we can either add edges with new labels, or relabel existing edges. TEES uses all shortest-distance paths between two nodes as sources of features, so we can expect this to make a difference.[6] Accordingly, each transformation was tested in two versions: one where dependency types are merely conflated; another where a conflated label is added to the existing labels. This means that for each conflation P, there is a corresponding **duplicated**-P.

Splitting `nmod` These tranformations were not simply applied on top of BASIC, but rather on a modified version. The label `nmod`, as discussed in Chapter 3, is one of few dependency types that are used in the nominal and predicate domains alike. Because `nmod` is particularly important in this task, since nominalizations are rampant in the corpus, I created a version of the BASIC dependencies in which `nmod` is split into two labels, one for each domain. This was implemented as a change in the constituency-to-dependency converter.[7] I used this version with the split `nmod` label to be able to make conflations in the nominal and predicate domains separately. With this, `nmod` is defined to be [+**adnominal**], and in the predicate domain, the

[5] In fact, this is the 40th Bell number, which is 157450588391204931289324344702531067.

[6] A system that was not prepared to extract features from more than one dependency path would of course have to be treated differently.

[7] Thanks to Sebastian Schuster for helping me implement this.

label `nmod:adv` is introduced as its [−**adnominal**] counterpart.

Grouping labels The dependency types which enter these conflation transformations can be divided into eight sets, by signature type. The transformations preserves the integrity of these sets.

1. [+**adnominal, functional**]: `det`, `case`, `number`, `det:predet`
2. [+**adnominal, clausal**]: `acl`, `acl:relcl`
3. [+**adnominal, phrasal**]: `amod`, `appos`, `nmod`, `nmod:poss`
4. [−**adnominal, functional**]: `aux`, `auxpass`, `mark`, `expl`
5. [−**adnominal, clausal, −core**]: `advcl`
6. [−**adnominal, clausal, +core**]: `csubj`, `csubjpass`, `ccomp`, `xcomp`
7. [−**adnominal, phrasal, −core**]: `advmod`, `nmod:adv`[8], `nmod:tmod`, `nmod:npmod` [9]
8. [−**adnominal, phrasal, +core**]: `nsubj`, `nsubjpass`, `iobj`, `dobj`
9. [+**external, clausal, +passive**]: `csubjpass`
10. [+**external, clausal, −passive**]: `csubj`
11. [+**external, phrasal, +passive**]: `nsubjpass`
12. [+**external, phrasal, −passive**]: `nsubj`
13. [−**external, clausal**]: `xcomp`, `ccomp`
14. [−**external, phrasal**]: `dobj`, `iobj`

[8] Obtained by splitting `nmod`.

[9] Actually both `nmod:tmod` and `nmod:npmod` are underspecified for the **adnominal** attribute and also occur outside nominals. But in the output of the Stanford converter, which was used in these experiments, these two labels are always [−**adnominal**], because no conversion rules produce these labels inside nominals.

While there are other dependencies that could in principle be conflated by feature signature, none are clear targets for doing so, for various reasons. The [−**structural**] types, although somewhat uniform, are rare in this data. The same holds of the [+**structural, +extraclausal**] types. On the other hand, the types that have the features [+**structural, −extraclausal, ±adnominal**] (`neg`, `compound`, `cc`, `conj`) form a very heterogeneous group and it seems inappropriate to conflate them.

The conflation strategies are split into two groups: the first deal with conflation of labels beyond the core argument types; the second, within those. The underlying assumption, linguistically motivated, is that core arguments are important to relation extraction, and conflating core argument labels with other types of labels is unlikely to bring any gains.

Representations (outside the core domain)

A set of 7 experiments combine the noncore types in different ways, summarized in Table 5.6.

ADNOMINAL All adnominal dependents are combined. Dependencies in groups 1, 2 and 3 are all relabeled `adnominal`.

FUNCTIONAL All functional dependents are combined, whether adnominal or not. Types from groups 1 and 4 are renamed `functional`.

MODIFIER All noncore modifiers are renamed `modifier`, adnominal or not, clausal or not. These are groups 2, 3, 5 and 7.

NONCORE Everything that is not a core argument is labeled `noncore`. This conflates dependencies from groups 1, 2, 3, 4, 5 and 7.

PREDNONCORE All predicate dependents that are not core dependents are conflated into `prednoncore`; this amounts to groups 4, 5 and 7.

new label	*relabeled*	*new label*	*relabeled*
adnominal	acl,acl:relcl,amod, appos,case,det, det:predet,nmod, nmod:poss,number	prednoncore	advcl,advmod, aux,auxpass,expl, mark,nmod:adv, nmod:npmod,nmod:tmod
functional	aux,auxpass, case,det,det:predet, expl,mark,number	adnommod	acl,acl:relcl,amod, appos,nmod,nmod:poss
modifier	acl,acl:relcl,advcl, advmod,amod, appos,nmod, nmod:adv,nmod:npmod, nmod:poss,nmod:tmod	predmod	advcl,advmod,nmod:adv, nmod:npmod,nmod:tmod
noncore	acl,acl:relcl,advcl, advmod,amod,appos, aux,auxpass,case, det,det:predet, expl,mark,nmod, nmod:adv,nmod:npmod, nmod:poss,nmod:tmod, number	subjpass	csubjpass,nsubjpass
		subjnom	nsubj,nsubjpass
		subjcl	csubj,csubjpass
		subj	csubj,nsubj
		subject	csubj,csubjpass, nsubj,nsubjpass
		internal	ccomp,dobj,iobj,xcomp
		comp	ccomp,xcomp
		obj	dobj,iobj
		core	ccomp,csubj,csubjpass, dobj,iobj, nsubj,nsubjpass,xcomp

Table 5.6: Set of labels targeted by each conflation strategy.

ADNOMMOD Adnominal modifiers are combined together, clausal or not. This merges groups 2 and 3 into `adnommod`.

PREDMOD Noncore predicate dependents—groups 5 and 7—are combined together into `predmod`.

Some of these transformations test the contribution of specific features. The transformations FUNCTIONAL and MODIFIER test the contribution of the **nominal** feature, because they conflate dependents differentiated by it; ADNOMINAL and PREDNONCORE test the contribution of **functional**; PREDMOD tests the contribution of **clausal**.

F Δ (dev)	*p*	*modification*	*added*
0.61*	0.0019	ADNOMINAL	85562
0.28	0.082	ADNOMMOD	46352
1.74*	<0.0001	DUPLICATED-ADNOMINAL	85562
0.62*	0.0041	DUPLICATED-ADNOMMOD	46352
0.25	0.0922	DUPLICATED-FUNCTIONAL	47748
0.73*	0.0002	DUPLICATED-MODIFIER	62420
−0.26	0.1068	DUPLICATED-NONCORE	110168
0.08	0.346	DUPLICATED-PREDMOD	16068
0.47*	0.0094	DUPLICATED-PREDNONCORE	24606
0.32	0.054	FUNCTIONAL	47748
0.42*	0.0243	MODIFIER	62420
0.35	0.0618	NONCORE	110168
0.27	0.1032	PREDMOD	16068
0.19	0.1875	PREDNONCORE	24606

Table 5.7: Results for modifications outside the core domain. F-Δ refers to the absolute difference in the F-Score obtained by the model, across all folds, with respect to *basic*; p is the p-value for the comparison; *modification* is the name of the modified representation. Results with $p < 0.05$ are marked with a *. In this and all following results tables, *added* is the number of dependency edges present in the corpus that do not exist in the *basic*-represented corpus. Because this number refers exclusively to new edges, the entry for any conflation P is identical to the entry for duplicated-P.

Results (outside the core domain)

Table 5.7 shows the results. Most p-values here are high, and most transformations do not make a large difference. There is a general trend of small positive effects, confirming the underlying intuition that modifiers and function words do not need to be finely distinguished. Even the most radical transformation, NONCORE, which destroys all the distinctions outside the core argument domain, yields a small, non-significant gain. The surprising aspect of this result, however, is that the duplicated version of the transformation, which preserves the original label but adds the conflated label (as explained above), performs much worse than the replacement version.

UD inherited from SD a rich set of distinctions in the nominal domain, and it is notable the most attractive transformations operate in that domain: DUPLICATED-ADNOMMOD (which collapses nominal modifiers) and DUPLICATED-ADNOMINAL (which collapses nominal modifiers and nominal function words). In both cases, the duplicated version of the transformation seems to do much better than the replacement version. More generally, five out of six of the positive results with $p < 0.05$, with the exception of DUPLICATED-PREDNONCORE, operate within the nominal domain. Although this does call into question the intuition that rich distinctions in the nominal domain are good for relation extraction, it should be noted that the best results come from duplicating edges, preserving the distinction while also adding a higher level of abstraction.

The result for DUPLICATED-ADNOMINAL is the best single-transformation result, not only among conflation strategies but among all the strategies tried.

Representations (in the core domain)

These additional conflation strategies, summarized in Table 5.6, target distinctions within the core domain.

CORE This is the converse of NONCORE. All core arguments—subjects and complements alike—simply get relabeled `core`. This tests the fundamental idea that grammatical functions are helpful for relation extraction.

SUBJ This conflation combines `nsubj` and `csubj`, which are subjects of active clauses.

SUBJPASS This conflation combines `nsubjpass` and `csubjpass`, which are subjects of passive clauses.

SUBJNOM This conflation combines `nsubj` and `nsubjpass`, which are nominal subjects.

SUBJCL This conflation combines `csubj` and `csubjpass`, which are clausal subjects.

SUBJECT This conflation all subject labels—`nsubj`, `csubj`, `nsubjpass` and `csubjpass`.

INTERNAL Conversely, this conflation combines all complement labels—`dobj`, `iobj`, `ccomp` and `xcomp`.

COMP This conflation combines clausal complements, labeled `ccomp` or `xcomp`.

OBJ This conflation combines nominal complements, labeled `dobj` or `iobj`.

Here we can see that transformations CORE, SUBJECT and INTERNAL target the **clausal** feature; SUBJNOM and SUBJCL test the **passive** feature.

Results (in the core domain)

The expectation about these transformations is that they should make performance worse, because core argument distinctions carry information about predicate-argument relations that we expect to see leveraged relation extraction. The results of the experiments are shown in Table 5.8.

F Δ (dev)	p	*modification*	*added*
0.25	0.1083	COMP	2811
−0.28	0.095	CORE	20780
0.01	0.4659	DUPLICATED-COMP	2811
−0.04	0.4099	DUPLICATED-CORE	20780
0.17	0.1768	DUPLICATED-INTERNAL	9590
0.07	0.3632	DUPLICATED-OBJ	6779
0.43*	0.0136	DUPLICATED-SUBJ	8466
−0.01	0.4862	DUPLICATED-SUBJCL	39
−0.59*	0.0006	DUPLICATED-SUBJECT	11190
0.26	0.0946	DUPLICATED-SUBJNOM	11151
−0.11	0.2823	DUPLICATED-SUBJPASS	2724
0.08	0.322	INTERNAL	9590
0.27	0.0623	OBJ	6779
−0.10	0.3047	SUBJ	8466
−0.21	0.1459	SUBJCL	39
−0.20	0.1765	SUBJECT	11190
−0.22	0.1445	SUBJNOM	11151
−0.27	0.0841	SUBJPASS	2724

Table 5.8: Results for modifications within the core domain. F-Δ refers to the absolute difference in the F-Score obtained by the model, across all folds, with respect to BASIC; p is the p-value for the comparison; *modification* is the name of the modified representation; *added* is the number of edges that are new in this representation, with respect to BASIC. Results with $p < 0.05$ are marked with a *.

As with the noncore conflations, most results are not significant at the $p < 0.05$ level, but overall the trend is that conflating core arguments has poor results. Curiously, the transformations that target internal arguments have small (non-significant) gains, suggesting that conflation in that domain might be useful.

It is interesting that the only two results with $p < 0.05$ are for similar transformations, but have opposite signs: DUPLICATED-SUBJECT conflates the four subject labels, `nsubj`, `nsubjpass`, `csubj` and `csubjpass` and brings a small improvement; DUPLICATED-SUBJ affects only `nsubj` and `csubj` and hurts performance.

5.5.3 Path enrichment strategies

Representations

Chapters 2 and 4 both discuss, in broad strokes, how UD indirectly encodes information about semantic role assignment: I argue that UD should be used in such a way that makes the semantic role of an argument with respect to a predicate largely determined by the following triple: (GP, GF, SF)—the conjunction of the identity of the governing predicate, the grammatical function of the argument and the subcategorization frame of the predicate.

Lexicalized dependency paths between a predicate and its arguments naturally encode the identity of the predicate and the grammatical function of the argument. But they do not encode subcategorization frames. To repeat example (114), the contrast between the subject's roles in pair (183) cannot be captured in terms of the lexicalized paths between the predicate and the subject.

(183) a. Jack broke the window. (nsubj)

b. The window broke. (nsubj)

The path enrichment strategies I propose here augment core argument types with subcategorization information. Specifically, they include in the higher argument's labels information about the presence of lower core dependents. The paths for the subjects in (183) would be distinguished as in (184). This completes the information necessary for determining semantic roles in this case, because we know that intransitive uses of *break* are associated with the selection of a Theme in argument structure. The information about the predicate being intransitive is not in the BASIC dependency paths, but it is introduced with this transformation.

(184) a. Jack broke the window. (nsubj:trans)

b. The window broke. (nsubj:intrans)

SplitSubjects This transformation applies only to nominal subjects labeled `nsubj`, and treats `ccomp` and `dobj` as types of *complement*; it then annotates the subject as `nsubj:intrans` if no complement is present, `nsubj:trans` if a complement is present, and `nsubj:ditrans` if a complement *and* an indirect object are both present. It does not, however, take `xcomp` into consideration.

SplitComplements This transformation applies to `ccomp` and `dobj`; it extends them with `:iobj` or `:xcomp` when such dependents are present, respectively. (In English, these labels are not expected to co-occur under the same governor.)

SplitXcomp This transformation splits the `xcomp` label itself: if a `ccomp` or `dobj` dependent is selected by the open complement's governor, then the `xcomp` label is augmented with `xcomp:obj`. This will in general distinguish object control from subject control.

Results

F Δ (dev)	p	*modification*	*added*
0.27*	0.0343	SplitComplements	169
−0.40*	0.0202	SplitSubjects	8439
0.58*	0.0001	SplitXcomp	76

Table 5.9: Results for path enrichment modifications. F-Δ refers to the absolute difference in the F-Score obtained by the model, across all folds, with respect to basic; p is the p-value for the comparison; *modification* is the name of the modified representation; *added* is the number of edges that are new in this representation, with respect to basic. Results with $p < 0.05$ are marked with a *.

The results in Table 5.9 show that encoding subcategorization frames in subject labels was, somewhat surprisingly, not helpful. This is puzzling because, the distribution of these modified labels inside the paths from gold arguments to gold triggers reveals that `nsubj:intransitive` occurs over four times more often than `nsubj:transitive` in paths to Theme arguments, while `nsubj:transitive` conversely occurs over four times more often in paths to Cause arguments. These

differences make the distinction seem intuitively useful. It is possible that the events where the distinction could be useful are harder for the system to learn about, and the sparsity introduced by splitting the labels creates problems for other predictions.

On the other hand, it seems that the information about object versus subject control encoded by SplitXcomp is helpful despite the small impact on the corpus; SplitComplements also brings a (smaller) gain.

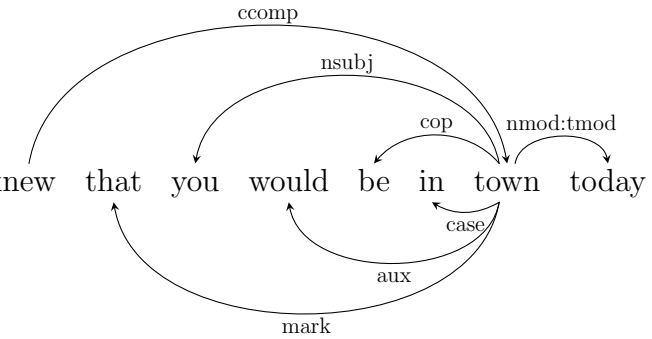
5.5.4 Headedness strategies

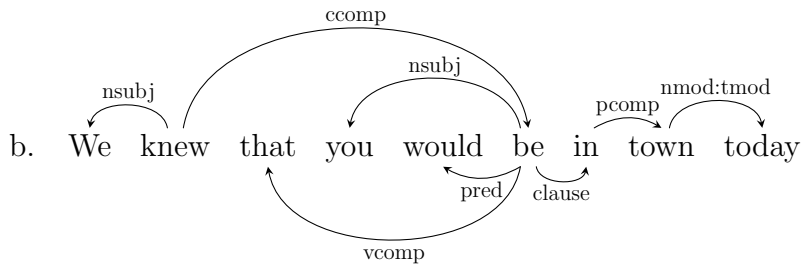
Representations

This type of strategy has already been discussed at length in Chapter 3 (see Section 3.3). Here I offer only a very brief reminder.

Many phrases, such as prepositional phrases (PPs), have distinct lexical and functional heads. In UD, the functional head depends on the lexical one, and it is labeled with a dedicated dependency type. The headedness transformations target four types of functional heads, each of which is labeled with a dedicated dependency type: prepositions (labeled `case`), auxiliary and modal verbs (`aux`), copulas (`cop`), and complementizers (`mark`). The sentence in (185) contains all four. The tree in (185a) is the UD representation of the sentence; each transformation changes the relation of each of these types of functional head to its lexical head, and one targets all four types. Applied individually, each one yields a different tree for the example in (185a); applied together, they yield the tree shown in (185b).

(185) a. We knew that you would be in town today





In Section 3.3.3, I discuss three strategies for handling the dependents of the lexical head when it is demoted; here I test exclusively the PARTIAL strategy, illustrated in (185b).

Results

F Δ (dev)	p	*modification*	*added*
0.14	0.2621	AUX$_{\text{P}}$	12636
0.47	0.006	COP$_{\text{P}}$	5266
0.76*	0.0008	ALL$_{\text{P}}$	71450
−0.32	0.0608	MARK$_{\text{P}}$	8152
0.08	0.3689	CASE$_{\text{P}}$	49349

Table 5.10: Results for headedness modifications. F-Δ refers to the absolute difference in the F-Score obtained by the model, across all folds, with respect to BASIC; p is the p-value for the comparison; *modification* is the name of the modified representation; *added* is the number of edges that are new in this representation, with respect to BASIC. Results with $p < 0.05$ are marked with a *.

As we can see from the numbers in Table 5.10, here the only improvement at the $p < 0.05$ level is obtained from the use of ALL$_{\text{P}}$. It is surprising that the CASE$_{\text{P}}$ representation does not fare better. One might expect that including prepositions in paths between predicates and arguments would lead to gains, since prepositions are key to marking arguments in nominalizations.

Overall, these results go against the claim in de Marneffe and Manning (2008) that representing relations between content words is more useful for applications; specifically, it seems that including words with relational meanings in dependency paths leads to better results than creating paths around those words. However, this

comparison is against the BASIC version of UD; it says nothing about the strategy of putting words with relational meanings on dependency edges, which is another way of making them part of dependency paths, while preserving dependency edges between content words. This alternative strategy is discussed in Section 5.5.5, along with others.

The gains here seem to be roughly additive; even though, applied individually, the modifications do not bring significant gains, the combination adds up to a 0.76% absolute improvement in F-Score.

5.5.5 Enhancement strategies

The final type of modification applied to the dependency trees is meant to capture the significance of certain dependency path regularities and distill them into shorter and hopefully more useful paths between words. These are the enhancements proposed in Schuster and Manning (2016), which are the most recent incarnation of the post-processed versions of SD. These enhanced dependencies include edges that mirror semantic relations between words more closely, moving away from surface syntax.

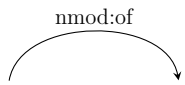
As we saw in Section 5.3, past extrinsic evaluations from Miwa et al. (2010a); Buyko and Hahn (2010) show that the post-processed versions of SD led to worse performance than the BASIC trees. The enhanced dependencies for UD are related to those previous efforts, but also have significant differences, justifying a separate evaluation.

Seven types of enhancements are proposed in Schuster and Manning (2016): four making up the ENHANCED representation, and an overlapping six making up the ENHANCED++ representation. The ENHANCED representation may have edges added, but it preserves the dependencies from BASIC; the ENHANCED++ one sometimes removes edges present in the BASIC trees. Neither is guaranteed to form a tree. I evaluate each enhancement in isolation, as well as the two packaged representations proposed in Schuster and Manning (2016).

All examples illustrating the transformations below come from Schuster and Manning (2016).

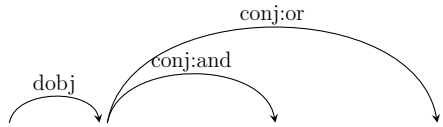
Representations

Augment modifiers: MARKER-EDGE This enhancement lexicalizes edges of types `nmod`, `advcl` and `acl`. The relation between such edges and their governors is characterized by a marker (typed `case` or `mark`). The transformation adds the identity of this marker to the edge type (186). This includes multiword prepositions, as defined by the English guidelines for the use of the `mwe` relation.

(186) the house on the hill


This is a strategy for handling words with relational meanings (such as *on* in (186)): rather than removing it entirely from the path between *house* and *hill*, as UD does, or making the relational word a node in the path, as ALL_P does, this enhancement pushes the preposition to a dependency edge. This keeps relational words in the dependency path between the words that they relate, without making the path longer for it.

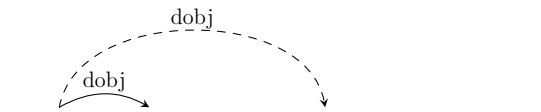
Augment conjuncts: CC-EDGE The same lexicalization strategy is applied to dependents labeled `conj`: the identity of the associated conjunction is appended to the label.

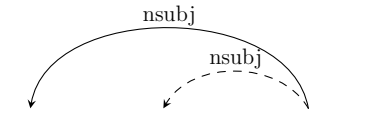
(187) eating apples and bananas or oranges


Propagate relations to conjuncts: TREAT-CC In BASIC, only the first conjunct in conjoined phrases enters dependencies with other words outside the conjoined phrase, as explained in Section 2.4.3 and exemplified by *apples* in (187). The other conjuncts are attached to the first one with the `conj` label, as is the case of *bananas* in (187): it is not represented as an object of *eating*.

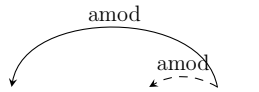
The TREAT-CC enhancement propagates some types of relation between the first conjunct and its governor to other conjuncts. In the case of conjoined nouns, the

dependency between the first conjunct and its governor is propagated to the other nouns, as exemplified in (188a). In the following examples, dashed edges are added edges that do not appear in the BASIC representation.

(188) a. 

b. 

In the case of adjectives, the dependency between the first adjective and the noun it modifies is propagated (189).

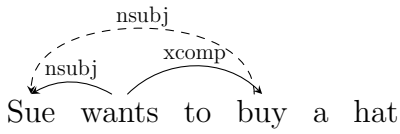
(189) 

Finally, in the case of verbs, a dependent is also propagated: the subject of the first verb also becomes subject of the other verbs (190). (This does not apply to internal arguments.)

(190) 

This transformation is implemented to work with lexicalized conjunction edges, so I tested it in combination with CC-EDGE.

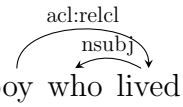
Identify controlled subjects: XSUBJ The label `xcomp` is used to identify an open clausal complement, which by definition includes a null (controlled) subject. There is no relation in the BASIC dependencies that links the embedded verb to the controller of that null subject in the matrix clause. This transformation adds a nominal subject edge, typed `nsubj` or `nsubjpass` according to the voice of the clause, between the embedded predicate and the direct object (if present) or subject of the matrix clause.

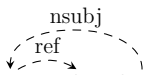
(191) 

This transformation together with the previous three comprise the ENHANCED dependencies representation for English UD, as proposed in Schuster and Manning (2016).

Identify referent of relative pronoun: REF Normally, in UD, a relative pronoun gets assigned a role inside the relative clause that it introduces; the only connection between that relative clause and the referent of the pronoun outside the clause is the `acl:relcl` label. This connection does not specify where the modified noun should be interpreted in the relative clause. That inference is made by identifying the referent with the role of the relative pronoun. This transformation makes that connection explicit, by moving the dependency edge that goes to the relative pronoun to the pronoun's referent. The relative pronoun becomes a dependent of its referent (192).

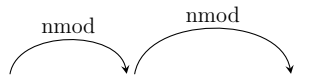
In the following examples, the (a) items correspond to the BASIC representation; they are depicted for reference only. The (b) items are the respective enhanced representations. The dashed edges continue to represent arcs that do not appear in BASIC.

(192) a. 

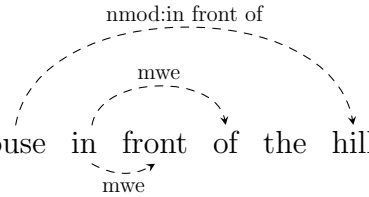
b. 

Extend multiword markers: MW-MARKER-EDGE This transformation is an extension of MARKER-EDGE, and replaces it in the ENHANCED++ representation. In addition to lexicalizing single prepositions and complex prepositions normally annotated with `mwe`, MW-MARKER-EDGE identifies other complex prepositions, given a nested PP analysis in BASIC, and lexicalizes edge labels with them.

(193) a. the house in front of the hill



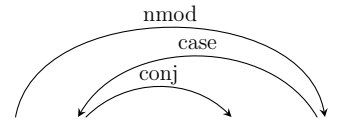
b. the house in front of the hill



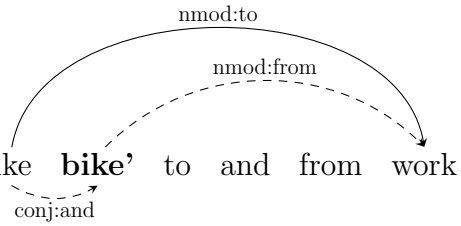
Expand prepositional phrases: EXPAND-PP When clauses contain conjoined prepositions inside prepositional phrases, only the first preposition gets annotated on the dependency edge by MARKER-EDGE. The ENHANCED++ representation takes this lexicalization one step further. It creates a copy of the PP's governor, which allows for the creation of another lexicalized `nmod` edge corresponding to the conjoined preposition. The conjunction is then moved to link the copies of the governor, instead of the prepositions.

Example (194) shows the enhancement. The BASIC tree is given in (194a). In (194b), the word *bike* is copied in the dependency representation[10] in order to accommodate a second `nmod` edge that represents the ellided PP *to work*. The edges that result from the enhancement are similar to the BASIC dependencies that would be used to represent the sentence *I bike to work and bike from work.*.

(194) a. I bike to and from work



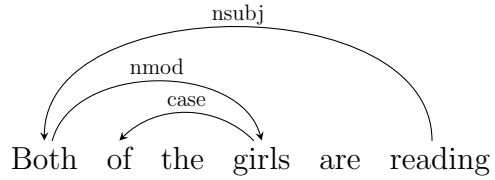
b. I bike **bike'** to and from work



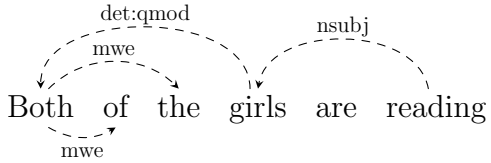
[10]In TEES, the system does not copy the node, so the edges labeled `nmod:from` and `nmod:to` would have the same governor.

Demote partitives and light nouns: QMOD A number of partitive noun phrases, such as *both of the girls* in (195), receive an analysis where the partitive word is the head in BASIC. However, clearly that is not the semantically salient head of the phrase. This syntax-semantics mismatch is addressed by this transformation, which demotes partitive phrases and light noun constructions such as *a group of* and makes them multiword determiners attached to the semantically salient noun. This enhancement addresses a fuzzy area of the border between content words and function words, and it represents another major strategy for creating edges directly between content words: preventing syntactically salient light nouns from getting in the way of the semantically salient word they associate with.

(195) a.



b.



Results

The results are given in Table 5.11. Overall, we see that there are gains from the use of both enhanced representations, which goes against the related findings of Miwa et al. (2010a,b); Buyko and Hahn (2010) for SD. But when tested in isolation, many of the enhancements seem problematic.

There are no significant changes in performance when applying CONJ-EDGE, TREAT-CC, MARKER-EDGE or MW-MARKER-EDGE in isolation. With $p < 0.05$ we actually see a drop in performance when applying QMOD and EXPANDPP. On the other hand, XSUBJ and REF bring significant gains.

The puzzle here is that the enhanced representations perform well, in spite of these problems. In particular, the more aggressive ENHANCED++, which combines

F Δ (dev)	*p*	*modification*	*added*
0.00	0.4941	CONJ-EDGE	7142
0.73*	0.0005	ENHANCED	40305
1.14*	<0.0001	ENHANCED++	43517
−0.75*	<0.0001	EXPANDPP	1059
−0.13	0.2883	MARKER-EDGE	25484
0.16	0.2659	MW-MARKER-EDGE	25679
−0.59*	0.0003	QMOD	745
0.58*	0.0003	REF	2208
−0.25	0.1499	TREAT-CC	14070
0.34*	0.0244	XSUBJ	658

Table 5.11: Results for enhancements. F-Δ refers to the absolute difference in the F-Score obtained by the model, across all folds, with respect to BASIC; p is the p-value for the comparison; *modification* is the name of the modified representation; *added* is the number of edges that are new in this representation, with respect to BASIC. Results with $p < 0.05$ are marked with a *.

QMOD, REF, XSUBJ, TREAT-CC, EXPANDPP and MW-MARKER-EDGE, outperforms both BASIC and ENHANCED.

Error analysis

Some of the enhancements presented in this section are the most aggressive transformations studied in this chapter, and one important question is how successful their application to an imperfect parse can be. To the extent that the inferences represented by the enhancements are wrong due to parser errors in BASIC, we may expect to find more serious propagated errors in the enhanced versions, harming their performance with respect to BASIC dependencies.

I examined a sample from the output of the enhancements in the development set, with the exception of the edge lexicalization strategies MARKER-EDGE, CONJ-EDGE and MW-MARKER-EDGE, which are particularly straightforward. The sample was not random—it consisted of the first 50 changes, with relation to BASIC (or, in the case of TREAT-CC, with relation to CONJ-EDGE), that the enhancement introduces. For QMOD and EXPANDPP I evaluated only the occurrences in the development set, which

were fewer than 50. Each change usually corresponds to a new edge introduced and possibly one or more edges restructured; I evaluated them based on whether the new edges corresponded to a valid relation, from an interpretation standpoint.

This was a loose evaluation, without strict criteria—the numbers should not be taken as a rigorous accuracy score. It should also be noted that the evaluation focuses on the interpretation of the added edges, more than the structure of the dependency tree. Parser errors are sometimes propagated, but even then, the resulting enhancement does not always receive a negative assessment. For example, with XSUBJ there are several instances of a purpose clause wrongly parsed as an `xcomp`, triggering the insertion of an extra subject. However, because these clauses are themselves often open, with subjects bound by the matrix subject, the added `nsubj` edges are often still correct, even if for the wrong reasons.

The results are summarized in Table 5.12 and commented below.

Enhancement	Correct	Incorrect
REF	45	5
XSUBJ	41	9
TREAT-CC	36	14
QMOD	29	3
EXPANDPP	28	5

Table 5.12: Correct and incorrect added edges per enhancement strategy.

REF **performs well** This transformation has 90% accuracy in this sample, and is generally unobjectionable. The errors in REF come from wrongly attached relative clauses. The success of the REF transformation, under both intrinsic and extrinsic evaluation, is evidence for the usefulness of the language-specific `acl:relcl` relation, which identifies the relative-clause structure targeted.

XSUBJ **operates on many false positives** The errors in XSUBJ are due to adverbial clauses, and in one case a clausal subject, wrongly identified as open complements.[11]

[11]Five times in this sample, the XSUBJ transformation targets the complements of implicative verbs such as *fail*, in which case the relation recovered concerns an event that is being negated. I did

TREAT-CC **suffers with PP attachment errors** The number of wrong new edges introduced by TREAT-CC is mostly due to the curse of PP attachment; this notoriously difficult problem appears frequently in the trees, and the transformation brings its effects to the added edges.

QMOD **demotes modifiers** The application of the QMOD transformation rarely interacts with parser errors, and is mostly correct. Nevertheless, this operation sometimes demotes modifiers that bring in additional information, in phrases such as *a malignant form of tumor*. While this is not necessarily a problem, the path between *malignant* and *tumor* in these cases becomes longer (depending on the exact phrase), which may have consequences for classification. This occured 6 times in my small sample.

EXPANDPP **is often lossy** The EXPANDPP often applies to conjunctions of the form *X but not Y*. This may be part of the issue: in these cases, the dependency path from the governor of the conjoined phrase to the negated conjunct is longer in the BASIC trees. This path is shortened by the transformation, and although this is not incorrect syntactically, it may be undesirable.

5.5.6 Combining strategies

Taking stock

All the transformations that showed a positive difference in development-set F-Score compared to the BASIC baseline, under the strict evaluation criterion, are shown in Table 5.13. The table also includes differences in precision and recall, as well as differences in F-Score broken down by type of event simple and regulation events, and for the modifier task (see Section 5.2 for descriptions).

Overall, we notice that the improvements in F-Score come almost uniformly from improvements in precision, with several transformations actually hurting recall. In this sense the most balanced gains come from the ENHANCED++ representation.

not consider these errors, since these extra subjects are no different than the subjects in sentences such as *He denied that he had an affair*.

	All			Simple	Regulation	Modifier
	F Δ	P Δ	R Δ	F Δ	F Δ	F Δ
ALL$_{\text{P}}$	0.76	2.14	−0.59	0.65	0.94	−0.55
ADNOMINAL	0.61	1.25	−0.05	0.82	0.32	0.43
D-ADNOMINAL	1.74	3.83	−0.27	1.82	1.74	1.01
D-ADNOMMOD	0.62	1.45	−0.21	0.23	0.91	−0.08
D-MODIFIER	0.73	1.59	−0.13	0.81	1.03	−0.48
D-PREDNONCORE	0.47	1.41	−0.48	0.38	0.72	−1.02
ENHANCED	0.73	1.36	0.09	0.76	0.63	0.51
ENHANCED++	1.14	1.75	0.51	1.07	1.05	1.53
MODIFIER	0.42	0.79	0.05	0.44	0.40	0.28
REF	0.58	1.67	−0.51	0.25	0.83	−0.16
SPLITCOMPLEMENTS	0.27	0.91	−0.38	0.05	0.38	0.03
SPLITXCOMP	0.58	1.13	0.02	0.28	0.75	0.73
XSUBJ	0.34	0.66	0.00	−0.02	0.54	0.38

Table 5.13: Results for each representation that brings a positive gain with $p < 0.05$. Metrics reported are differences in F-Score (F), Precision (P) and Recall (R) with respect to BASIC for all events in the task; and differences in F-Score for three types of prediction—Simple events, Regulation events and Modification. (See Section 5.2 for descriptions.) The letter D stands for DUPLICATED.

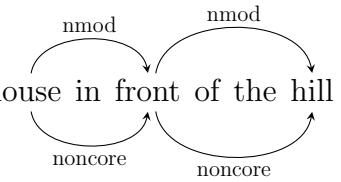
In terms of the different types of events, gains are often uneven between simple events (which have only a Theme argument) and regulation events (which have a Theme and a Cause argument), but all the successful transformations are beneficial for both types. This does not hold for modifiers: the identification of speculation and negation modifiers (which accounts for a small part of the overall score) is often worse than the baseline with these transformations.

By far the best single-transformation result comes from DUPLICATED-ADNOMINAL, with a 1.74% absolute improvement in F-Score in relation to BASIC. I used this to drive the search for beneficial transformations: I tested different representations together with this transformation.

Combinations

Some of the other conflation-based transformations have positive results, but I did not pursue combinations of those with DUPLICATED-ADNOMINAL. This was because each transformation with duplications can significantly increase the number of distinct dependency paths between any two words. Take the example in (196): because of the duplication, there are now 4 paths from *house* to *hill*.

(196) the house in front of the hill



To avoid an explosion of paths, I did not combine any duplicated conflations. This problem does not affect replacement-only conflations, so DUPLICATED-ADNOMINAL could have been combined with other (non-duplicated) conflation strategies; however, all the ones that created significant positive results with low p-values have some overlap with DUPLICATED-ADNOMINAL, so that combining them would require changing the way DUPLICATED-ADNOMINAL is applied.

In the domain of path enrichment I combined DUPLICATED-ADNOMINAL with SPLITCOMPLEMENTS and SPLITXCOMP. Among the headedness transformations, I combined it with the ALL-P representation; and among enhancements, I combined it with REF and XSUBJ (together), as well as the two enhanced representations.

Results

	All			Simple	Regulation	Modifier
	F Δ	P Δ	R Δ	F Δ	F Δ	F Δ
D-ADNOMINAL+ALL-P	−0.46	−0.10	−0.82	−0.78	-0.25	−1.20
D-ADNOMINAL+REF+XSUBJ	−0.32	−0.43	−0.21	−0.76	0.02	−0.43
D-ADNOMINAL+ENHANCED	0.94*	1.26	0.60	0.63	1.10	1.36
D-ADNOMINAL+ENHANCED++	0.61*	0.95	0.26	0.40	0.65	1.59
D-ADNOMINAL+SPLITCOMPLEMENTS	0.25	1.02	−0.52	0.20	0.24	0.71
D-ADNOMINAL+SPLITXCOMP	0.33	0.75	−0.11	−0.07	0.50	1.29

Table 5.14: Results for each combination of DUPLICATED-ADNOMINAL with another representation, as explained in Section 5.5.6. Metrics reported are differences in F-Score (F), Precision (P) and Recall (R) with respect to BASIC for all events in the task; and differences in F-Score for three types of prediction—Simple events, Regulation events and Modification. (See Section 5.2 for descriptions.) The letter D stands for DUPLICATED.

Results are shown in Table 5.14. All combinations perform worse than DUPLICATED-ADNOMINAL alone, and, except in the case of ENHANCED, all also perform worse than the other representation in the combination. The best-performing combinations are those merging DUPLICATED-ADNOMINAL with the enhanced transformations from Schuster and Manning (2016), discussed above in Section 5.5.5; but they still do not perform as well as the enhanced transformations by themselves.

This confirms that it is frustratingly difficult to predict the results of combining modifications. In Section 5.5.5, we saw enhancements that did not perform well individually have a positive effect when combined. Now we see the opposite: the impact of the most promising modifications is completely deflated when they get combined.

5.6 Generalizing results

Because I performed many experiments with a single data set, one might ask if the positive results found in the last section are the product of a higher-order overfitting of the representation to the particular data. In order to address this question, I turned

to two other event extraction tasks from the BioNLP'11 challenge (Pyysalo et al., 2012), with distinct event characterizations and distinct sets of data. These were the Epigenetics and Post-translational Modifications Task (EPI11) and the Infectious Diseases Task (ID11).

There are two interesting contrasts between these data sets. As mentioned below in Section 5.6.1, the training data for ID11 is a superset of GE09. Therefore, the results for ID11 correspond to extending the data and the task definitions from GE09; the results for EPI11 correspond to largely the same event types, but with new data. Additionally there is a data size contrast: the ID11 models are trained on more data than the EPI11 models.

5.6.1 The ID11 task

This task focuses on the biomolecular mechanisms of infectious diseases, and extends the event definitions from BioNLP'09 with new entity categories and a new event type. The data consists of full papers (as opposed to abstracts in GE09) selected by domain experts.

The TEES submission for this task was trained on the union of the tasks's training data with existing data for the GENIA task in BioNLP'11 (GE11). GE11 augments the existing GE09 corpus by 30% with the addition of full papers. I adopted the same procedure, and hence the ID11 task results are trained on a superset of the GE09 data. This additional data was not partitioned and was included in all folds.

5.6.2 The EPI11 task

This task focuses on events relating to epigenetic change. There are eight types of event; four types follow the definition for Phosporylation events in BioNLP'09, and thus have one Protein argument to be identified from pre-annotated entities, and one optional Site argument consisting of an entity not annotated in the test data. Another three event types define a third, optional, argument in addition to Theme and Site. Finally, the last event type, Catalysis, takes two obligatory arguments: one event-typed Theme argument, and one Protein-typed Cause argument.

The data consists of PubMed abstracts, with additional training data selected from domain databases.

5.6.3 Results

	ID11		EPI11	
	F Δ (test)	F Δ (dev)	F Δ (test)	F Δ (dev)
ENHANCED	1.14	−1.96	−0.14	−0.26
ENHANCED++	5.38*	2.64	0.34	0.87
D-ADNOMINAL+ENHANCED	0.70	−1.96	0.51	0.32
D-ADNOMINAL+ENHANCED++	4.99*	1.07	0.52	0.26
D-ADNOMINAL	0.26	1.16	−0.83	−0.92

Table 5.15: Results for the five best-performing representations from the exploratory experiments, in two new data sets: ID11 and EPI11. Metrics reported are differences in F-Score (F) with respect to BASIC for all events in the task, in the test set and in the development set. The letter D stands for DUPLICATED.

Table 5.15 shows results for the two datasets: ID11 and EPI11, and Figure 5.3 shows the performance of each of these representations in each of the 10 folds for all three datasets. At the $p < 0.01$ level,[12] only the differences for ENHANCED and ENHANCED++ in the ID11 data set are significant. In EPI11, all the differences are small, but it seems that adding the conflation to the enhanced representations is better than using them by themselves.

Overall, we can see in Figure 5.3 that ENHANCED++ does very well; despite not making a significant difference in the EPI11 experiments, it performs among the best representations in almost every fold of each data set, and brings a large absolute gain of over 5% in the F-Score metric.

Another noticeable trend is that, in almost every case, the four representations that use enhancements bring a larger improvement on the test set than on the development set. This is an indication that these enhancements help prevent overfitting.

[12]Obtained by applying a Bonferroni correction to a 0.05 threshold.

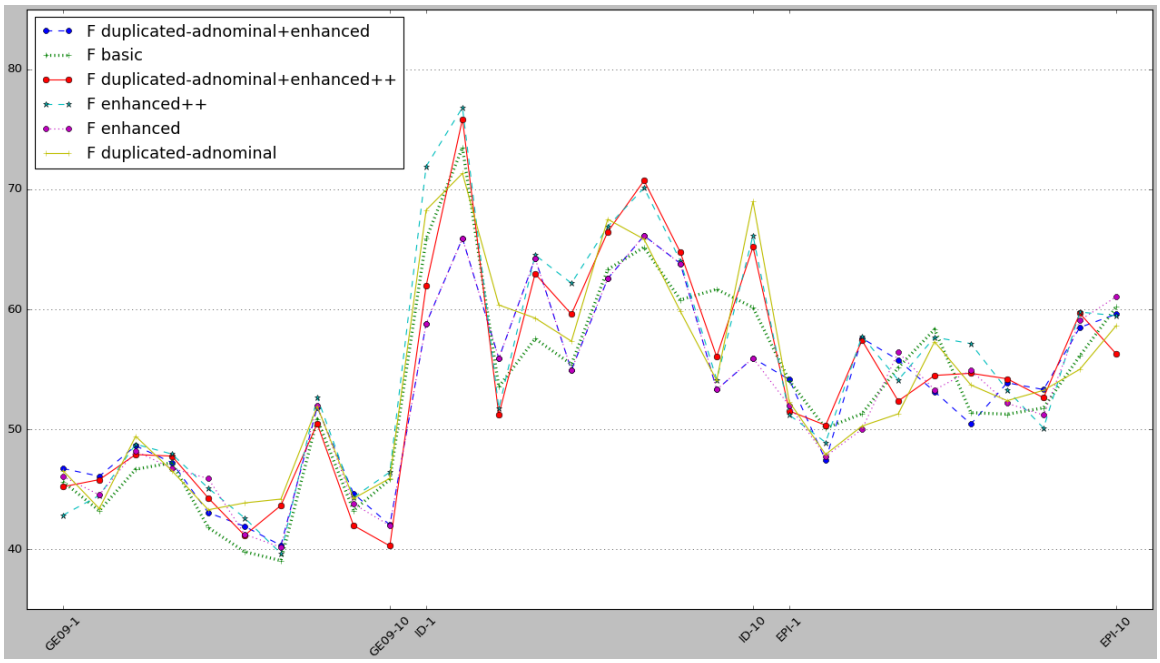


Figure 5.3: F-Score from fold to fold (1 through 10) for the baseline representation BASIC and the 5 best performing representations, for three datasets: GE09, ID11 and EPI11.

5.6.4 Discussion

Overall, it is clear that designing modifications that can robustly bring gains across a variety of data sets and tasks is very difficult. The only consistent result across the three tasks and data sets is the superiority of ENHANCED++; however, the large variation found for other representations suggests that even this finding may not hold in other conditions.

The conclusions here go in a different direction than the results reported in Miwa et al. (2010b,a); Buyko and Hahn (2010) for the BioNLP'09 task: while those authors found that the BASIC SD dependencies performed better than their corresponding "enhanced" versions, suggesting that modifying the dependency structure to reflext "deeper" structure was not a good strategy, here we find the opposite. The result is clearly that the ENHANCED++ dependencies of Schuster and Manning (2016) outperform the BASIC UD trees.

It is possible that the reason for this conflict is that the systems used for extrinsic evaluation in those papers benefit from different properties in the dependency representation. Additionally, as mentioned in Section 5.3, Miwa et al. (2010b,a) delexicalize enriched edges in the COLLAPSED and CC-PROCESSED representations, which is a problematic choice. However, since both papers report results for a only single data set and a single data partition, for the moment there is more evidence in favor of the enhanced representations. The results reported here examine ten partitions for each of three data sets, and ENHANCED++ dependencies perform significantly better than BASIC.

5.7 Conclusion

This chapter presented over 50 ways to modify UD for information extraction, and an extrinsic evaluation of these modifications with an event extraction system that relies heavily on dependency features. I provided an extrinsic evaluation of each modification, and offered evidence that they can bring as much as 5.38% absolute F-Score improvement in the performance of an event extraction system heavily reliant

on dependency features.

Each modification was evaluated on several data samples: the results come from three datasets for slightly different event extraction tasks, and each experiment was performed with 10-fold cross-validation. The extreme variablity of the results across data sets for the same type of applied task implies that a dependency representation cannot easily be optimized for downstream performance, even for a single system.

One important takeaway here is methodological: there is a high degree of variation even from fold to fold, as illustrated in Figure 5.2, which shows that single-experiment results cannot be expected to generalize well. This is likely why the results presented here are not consistent with the results reported in Buyko and Hahn (2010); Miwa et al. (2010a).

Another takeaway is that the ENHANCED++ representation of Schuster and Manning (2016) performed consistently better than the UD baseline in the three data sets, by as much as 5.38% in the test data on the ID11 task. This indicates that the endeavor to bring out semantic relations to the forefront, by restructuring and relabeling dependencies to demote light content words, turn relational words into edge labels, and make explicit inferences supported by the syntactic structure, can be highly successful. Unfortunately, it also becomes apparent from these results that it is very difficult to predict the impact of such an endeavor, because the relation between the effects of atomic transformations and of their combinations is not linear.

This exploration is situated between representation design and engineering. The transformations can be performed without external sources of information, and are therefore simple rearrangements of information already contained in the dependency trees. Accessing this "rearranged" information by means of feature engineering would certainly be a path for replicating the results I obtained. To give an example, conflating some dependency types can also be seen as creating disjunctions of label-based features. In this sense, the strategies I investigate here concern representation design as much as they concern feature engineering. However, regardless of implementation, these experiments address interesting high-level questions about what kind of structural information needs to be encoded, or can profitably be encoded, for recovering

information such as event-participant relations in event extraction. These are, fundamentally, questions about how to represent syntactic information for NLP, and thus questions about syntactic representation design.

Chapter 6

Conclusion

The goal of this dissertation has been to study the empirical consequences of the design of Universal Dependencies (UD, Nivre et al. 2016), a multilingual syntactic representation for use in Natural Language Processing (NLP). This work takes as a premise that there are design choices for syntactic representations that lie beyond the domain of linguistic argumentation alone, requiring abstract decisions about the shape of the representation; and that these decisions can be made with a view to the results obtained from their employment on natural linguistic data and in real NLP pipelines. I presented quantitative and qualitative analyses of how specific design choices have effects on different facets of UD usage. These results were, namely, the degree of parallelism in UD-represented structures under translation; the rules for mapping UD dependency types to semantic roles; the accuracy of parsing UD; and the performance of a syntax-informed relation extraction system using UD dependency features.

These analyses lay a foundation for a better understanding of representation design for NLP, and can inform high-level tradeoffs that need to be made for UD. The motivation for this work, reviewed in the introduction, was my awareness of these tradeoffs, which designers are inevitably faced with when planning an enterprise such as UD, as well as of the need for more thorough and comprehensive work addressing the empirical consequences of the different choices.

Chapter 2 discussed the theoretical foundations of UD and related it to other

frameworks for the study of syntax. This discussion serves two purposes: it shows how the principles underlying UD compare to different design decisions in other representations; it also shows the UD standard unfolds from those principles, which reflect particular goals for large-scale annotation and use in NLP pipelines. The principles governing decisions about headedness and type distinctions are introduced there.

Chapter 3 presented a thorough investigation of the issue of headedness in dependency representations, and the consequences of headedness choices for parsing. It includes a thorough discussion of the advantages and challenges of using a parsing representation, as well as a detailed presentation of what that representation should be. Experiments for data in six languages show that there is variation in whether a parsing representation is useful, but in some languages that are large gains to be obtained. The most important takeaway from Chapter 3 is that a functional-head design can be very beneficial for parsing; but, since these benefits can be obtained, largely, by means of parser-internal conversions, and since in some languages a lexical-head design can be better for parsing, this is a weak objection to UD's lexical-head design—a fact which went unnoticed in some previous work on these issues (Schwartz et al., 2012). The extent to which it can promote parallelism, on the other hand, is a strong argument in favor of that design.

Chapter 4 presented the first detailed linguistic analysis of UD-represented data. Its focus is on how UD can be used to account for linguistic phenomena at the right level of detail to capture important similarities and differences across languages. The analysis is a case study of four Romance languages, zooming in on very productive phenomena that do not have parallels in English. The chapter shows that keeping the UD representation lean and committed to grammatical functions leads to a simple and straightforward analysis of Romance SE constructions, which are very difficult to unify in generative syntax. On the other hand, UD's concise representational toolkit proves insufficient to satisfactorily represent complex predicates, which may require the introduction of a new dependency type in the future. The Romance data also presents an interesting challenge for the definition of function words, and I argue that, going forward, the best way to maximize the crosslinguistic uniformity of the distinction between function and content words is to rely on a small amount of

semantic information, in addition to syntactic cues.

Finally, Chapter 5 investigates the actual usage of UD in a pipeline, with an extrinsic evaluation of UD against many actual and potential alternatives. The main takeaway from Chapter 5 may actually be methodological: it is very difficult to obtain consistent improvements across data sets by manipulating the dependency representation, and this calls into question previous results in this line of investigation. But the most consistent result obtained was an improvement in performance when an enhanced representation was used, that is, a version of UD that is restructured and relabeled to have shorter predicate-argument paths. This goes against previous, less robust results indicating that restructuring operations do not improve performance (Buyko and Hahn, 2010; Miwa et al., 2010a).

In all of these areas, there is more work to be done. In parsing, the question remains open of which languages benefit from a parsing representation, and why. It would also be interesting to know how the learning curves of a functional-head representation and a lexical-head representation compare for the same language, and whether there are differences in learning rate between those two scenarios.

In terms of language parallelism, this dissertation shows that lexical-head representations can offer more parallelism across languages, but it does so in a very limited context, for English and Spanish. Going forward, it is important to do this at a larger scale for other language pairs, in order to obtain a better, more complete characterization of the potential for parallelism afforded by UD, of how that potential compares to the possibilities created by a functional-head design, and of how the differences translate from human-corrected annotations (as in my small experiment) to automatic annotations. The evaluation presented here was intrinsic, based on a metric defined for parallelism; but ideally, we want to see the same results come out of an extrinsic evaluation based on usage in multilingual applications. Setting up such an evaluation is important work for the future.

Chapter 4 raises the challenge of changing the type system, in order to better accommodate complex predicates, which (though scarce in Romance) are common in many languages. The analysis present there will benefit immensely from being complemented with diverse data on complex predicates in other languages. It also

brings out the fact that the distinction between content and function words, essential for UD, is very difficult to enforce in practice; a more precise specification of how this decision should be made across languages is needed.

Finally, Chapter 5 shows how difficult it is to optimize the representation for a specific application, as observed in the volatility of the performance of an event extraction system. While the use of TEES in the BioNLP Shared Task has a number of characteristics that make it an interesting case study for this question, it also has an important flaw: TEES relies on the use of a linear classifier in a very high-dimensional space for relatively a small amount of data. This leaves us wondering to what extent the variability in TEES' performance may be due to feature sparsity, and whether different results would obtain from another type of task in which the features were denser. This remains to be investigated.

More generally, this dissertation tackles three important facets of the usage of UD (crosslinguistic description, parsing and information extraction), but others remain unexplored. One more important issue is the extent to which UD is suitable—and can be made more suitable—for parser transfer, which has been one of the important motivations for the development of a multilingual representation. The success of parser transfer can also, in a way, be understood as a measure of structural parallelism, and it is a different measure than the one used in Chapter 3. An empirical investigation on the ways in which the design of UD can support parser transfer would be an important extension to the work developed here. Recent work reported in Rasooli and Collins (2015) is a valuable step in this direction.

Another crucial aspect of usage that is beyond the scope of this dissertation is suitability for annotation. Consistent and correct annotation is essential for developing treebanks that can then be used to trained parsers, and having parsers that can output UD is a *de facto* requirement for it to be useful in NLP research. The granularity of labels, especially, can have important consequences for the work of annotators; distinctions that cannot be made reliably in annotation need to be considered very carefully. The present work does not address these issues.

At a high level, this dissertation addresses some issues that have long been open questions in linguistic annotation—such as the advantages or disadvantages of a

lexical-head design—and puts forward thorough empirical answers where there had previously been only speculation, or partial experimentation.

Overall, the results and analyses presented here should change some of the conceptions behind the current design of UD, even when the design itself remains the same. If we take the case of English, this means that for monolingual scenarios the lexical-head design looks unconvincing. A functional-head representation is more expressive, as seen in Chapter 3 (most noticeably in the case of copulas). It leads to higher-accuracy parsing. Chapter 5 shows that it also yields better results for event extraction, contrary to previous suggestions (de Marneffe et al., 2006) that were historically argued to support the design of Stanford Dependencies (de Marneffe et al., 2006; de Marneffe and Manning, 2008).

But monolingual scenarios are not, of course, UD's main concern. The major reason to adopt a lexical-head design for UD is to support crosslinguistic parallelism. This claim is discussed in Chapters 2 and 3, and empirically supported by a small study reported in Chapter 3. Furthermore, while for English parsing accuracy improves with a functional-head design, across languages we find that lexical heads can be better (as was shown for German) or that there may be no difference (in the case of Italian). Even when functional heads are more favorable, this can be worked around by using a parser-internal transformation.

For UD, this means that the main (and perhaps the only) reason to use a lexical-head design is to support crosslinguistic parallelism. However, crosslinguistically the lexical-head design can only support this parallelism if function words are defined in a uniform manner across languages. This being the case, the analysis presented in Chapter 4 shows that it is crucial to create robust crosslinguistic guidelines for the definition of function words, and that the best way to create these guidelines is to admit some semantic criteria into the definition. This indicates (as is no surprise for linguists) that there is enormous tension between the goal of annotating surface syntax and the goal of creating crosslinguistically parallel annotations, but also that it may be possible to reconcile these goals by making only very limited excursions out of surface syntax.

It is important to acknowledge that not every experiment presented can be translated into a specific design decision. The complexity of the results highlights the importance of precise hypothesis formulation, rigorous methodology and diverse angles of investigation. It also shows that small changes to the design of a syntactic representation can have significant and sometimes unpredictable consequences for entire NLP pipelines, and a single design cannot necessarily serve every purpose equally well.

Knowing this, one of the most useful things that designers can do is provide a discussion of the properties of their representation for users, in such a way that the users feel empowered to make transformations such as the many examples that were illustrated in this dissertation. Adjusting the level of granularity of the dependency types, changing the headedness principles that underlies dependency trees, or enriching structures with semantically motivated arcs—all of these are legitimate ways to use UD that can be useful in different circumstances, but they may not be obvious (or clear) for users who do not completely understand the way UD trees are constructed.

A deep understanding of syntactic representations creates flexibility for users to exploit their properties in the way that is most suitable for a particular task and data set. One of the main goals of this dissertation has been to create this deep understanding about UD, thereby, hopefully, enabling users to utilize it in the way that is most suitable for them.

Bibliography

Abeillé, A. and Godard, D. (2003). Les prédicats complexes dans les langues romanes. In Godard, D., editor, *Problèmes de la phrase simple*, pages 125–185.

Afonso, S., Bick, E., Haber, R., and Santos, D. (2002). "Floresta sintá(c)tica": a treebank for Portuguese. In *Proceedings of the 3rd International Conference on Language Resources and Evaluation (LREC)*, pages 1698–1703, Las Palmas, Spain.

Aissen, J. and Perlmutter, D. (1983). Clause Reduction in Spanish. In Perlmutter, D., editor, *Studies in Relational Grammar*, pages 360—403.

Alsina, A. (1996a). *Predicate Composition: A Theory of Syntactic Function Alternations*. CSLI Publications, Stanford.

Alsina, A. (1996b). *The role of argument structure in grammar*. CSLI Publications, Stanford.

Andrews, A. (2007). The Major Functions of the Noun Phrase. In Shopen, T., editor, *Language typology and syntactic description*. Cambridge University Press, New York, 2nd ed. edition.

Baker, M. (1988). *Incorporation: A Theory of Grammatical Function Changing*. University of Chicago Press, Chicago.

Ballesteros, M. and Nivre, J. (2012). MaltOptimizer: An Optimization Tool for MaltParser. In *Proceedings of the Demonstrations at the 13th Conference of the European Chapter of the Association for Computational Linguistics*, EACL '12, pages 58–62, Stroudsburg, PA, USA. Association for Computational Linguistics.

Bangalore, S., Joshi, A. K., and Rambow, O. (2003). Dependency and Valency in Other Theories: Tree Adjoining Grammar. In Ágel, V., Eichinger, L. M., Eroms, H. W., Hellwig, P., Heringer, H. J., and Lobin, H., editors, *Dependency and valency: an international handbook of contemporary research*, number 25 in Handbooks of linguistics and communication sciences. Walter de Gruyter, Berlin; New York.

Belletti, A. (1982). Morphological Passive and Pro-Drop: The Impersonal Construction in Italian. *Journal of Linguistic Research*, 2:1–34.

Bengoetxea, K. and Gojenola, K. (2009). Exploring Treebank Transformations in Dependency Parsing. In *Proceedings of the International Conference RANLP-2009*, pages 33–38, Borovets, Bulgaria. Association for Computational Linguistics.

Böhmová, A., Hajič, J., Hajičová, E., and Hladká, B. (2003). The Prague Dependency Treebank. In Abeillé, A., editor, *Treebanks: Building and Using Parsed Corpora*, pages 103–127. Springer Netherlands, Dordrecht.

Bikel, D. M. (2004). Intricacies of Collins' Parsing Model. *Comput. Linguist.*, 30(4):479–511.

Björne, J., Ginter, F., Heimonen, J., Pyysalo, S., and Salakoski, T. (2009). Learning to extract biological event and relation graphs. In *Proceedings of the 17th Nordic Conference on Computational Linguistics (NODALIDA'09)*.

Björkelund, A., Hafdell, L., and Nugues, P. (2009). Multilingual Semantic Role Labeling. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning: Shared Task*, CoNLL '09, pages 43–48, Stroudsburg, PA, USA. Association for Computational Linguistics.

Björne, J., Heimonen, J., Ginter, F., Airola, A., Pahikkala, T., and Salakoski, T. (2011). Extracting Contextualized Complex Biological Events with Rich Graph-Based Feature Sets. *Computational Intelligence*, 27(4):541–557.

Björne, J. and Salakoski, T. (2011). Generalizing Biomedical Event Extraction. In *Proceedings of the BioNLP Shared Task 2011 Workshop*, BioNLP Shared Task '11, pages 183–191, Stroudsburg, PA, USA. Association for Computational Linguistics.

Björne, J. and Salakoski, T. (2013). TEES 2.1: Automated annotation scheme learning in the BioNLP 2013 Shared Task. In *Proceedings of the BioNLP Shared Task 2013 Workshop*, pages 16–25.

Bohnet, B. (2010). Very High Accuracy and Fast Dependency Parsing is Not a Contradiction. In *Proceedings of the 23rd International Conference on Computational Linguistics*, COLING '10, pages 89–97, Stroudsburg, PA, USA. Association for Computational Linguistics.

Bok-Bennema, R. (2006). Clitic Climbing. In Everaert, M. and van Riemsdijk, H., editors, *The Blackwell Companion to Syntax*.

Bosco, C., Montemagni, S., and Simi, M. (2013). Converting Italian Treebanks: Towards an Italian Stanford Dependency Treebank. In *Seventh Linguistic Annotation Workshop & Interoperability with Discourse*.

Bresnan, J. (1982). *The Mental Representation of Grammatical Relations*. MIT Press, Cambridge, MA.

Bresnan, J. (2015). *Lexical-functional syntax*. Wiley-Blackwell, second edition.

Bresnan, J. and Kanerva, L. M. (1989). Locative inversion in Chicheŵa: A case study of factorization in grammar. 20(1):1–50.

Burzio, L. (1986). *Italian Syntax: A Government-Binding Approach*. Reidel, Dordrecht.

Buyko, E. and Hahn, U. (2010). Evaluating the Impact of Alternative Dependency Graph Encodings on Solving Event Extraction Tasks. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 982–992, Cambridge, MA. Association for Computational Linguistics.

Camacho, J. (2012). Ser and Estar: The Individual/Stage-level Distinction and Aspectual Predication. In *The Handbook of Hispanic Linguistics*, pages 477–502. John Wiley & Sons, Ltd.

Candan, K. S., Liu, H., and Suvarna, R. (2001). Resource Description Framework: Metadata and Its Applications. *SIGKDD Explor. Newsl.*, 3(1):6–19.

Carroll, J. and Minnen, G. (1999). Corpus annotation for parser evaluation. In *In Proceedings of the EACL-99 Post-Conference Workshop on Linguistically Interpreted Corpora*.

Chang, P.-C., Tseng, H., Jurafsky, D., and Manning, C. D. (2009). Discriminative Reordering with Chinese Grammatical Relations Features. In *Third Workshop on Syntax and Structure in Statistical Translation*, pages 51–59.

Chierchia, G. (1984). Anaphoric Properties of Infinitives and Gerunds. In Mark Cobler, S. M. and T, M., editors, *Proceedings of the Third West Coast Conference on Formal Linguistics*.

Choi, J. D. and Nicolov, N. (2009). K-best, locally pruned, transition-based dependency parsing using robust risk minimization. In *Recent Advances in Natural Language Processing V: Selected papers from RANLP 2007*, Current Issues in Linguistic Theory, pages 205–216.

Chomsky, N. (1981). *Lectures on government and binding*. Foris, Dordrecht, Holland.

Cinque, G. (1988). On Si-Constructions and the Theory of Arb. *Linguistic Inquiry*, 19:521.

Cinque, G. (2004). 'Restructuring' and Functional Structure. In Cinque, G. and Rizzi, L., editors, *Structures and Beyond: The Cartography of Syntactic Structures*, volume 3.

Cinque, G. and Rizzi, L. (2008). The cartography of syntactic structures. *Studies in linguistics*, 2:42–58.

Comrie, B. (1989). *Language Universals and Linguistic Typology: Syntax and Morphology*. University Of Chicago Press, Chicago, 2nd edition.

Cortis, K., Bhowan, U., Mac an tSaoir, R., McCloskey, D., Sogrin, M., and Cadogan, R. (2014). What or Who is Multilingual Watson? In *COLING (Demos)*, pages 95–99.

de Marneffe, M.-C., Connor, M., Silveira, N., Bowman, S. R., Dozat, T., and Manning, C. D. (2013). More Constructions, More Genres: Extending Stanford Dependencies. *Proceedings of the Second International Conference on Dependency Linguistics (DepLing 2013)*, pages 187–196.

de Marneffe, M.-C., Dozat, T., Silveira, N., Haverinen, K., Ginter, F., Nivre, J., and Manning, C. D. (2014). Universal Stanford dependencies: A cross-linguistic typology. *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC-2014)*.

de Marneffe, M.-C., MacCartney, B., and Manning, C. (2006). Generating Typed Dependency Parses from Phrase Structure Parses. *Proceedings of the Fifth International Conference on Language Resources and Evaluation (LREC'06)*.

de Marneffe, M.-C. and Manning, C. D. (2008). The Stanford Typed Dependencies Representation. *Coling 2008: Proceedings of the workshop on Cross-Framework and Cross-Domain Parser Evaluation*, pages 1–8.

Dixon, R. M. W. (1994). *Ergativity*. Cambridge University Press, Cambridge; New York.

Dobrovie-Sorin, C. (1998). Impersonal Se Constructions in Romance and the Passivization of Unergatives. *Linguistic Inquiry*, 29(3):399–438.

Dobrovie-Sorin, C. (2006). The SE-anaphor and its role in argument realization. In Everaert, M. and Riemsdijk, H. v., editors, *The Blackwell Companion to Syntax*.

Elming, J., Johannsen, A., Klerke, S., Martinez, H., and Søgaard, A. (2013). Downstream effects of tree-to-dependency conversions.

Futrell, R., Mahowald, K., and Gibson, E. (2015). Large-scale evidence of dependency length minimization in 37 languages. *Proceedings of the National Academy of Sciences*, 112(33):10336–10341.

Goldberg, Y. and Elhadad, M. (2010). An Efficient Algorithm for Easy-first Non-directional Dependency Parsing. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, HLT '10, pages 742–750, Stroudsburg, PA, USA. Association for Computational Linguistics.

Gonçalves, A. (1998). On Restructuring Constructions in European Portuguese. In Tina Cambier-Langeveld, A. L. and Redford, M., editors, *Proceedings of ConSole 6*, pages 75–88.

González Vergara, C. (2006). *Las Construcciones no Reflexivas con "se": Una Propuesta desde la Gramática del Papel y la Referencia*. PhD thesis, Universidad Complutense de Madrid e Instituto Universitario de Investigación Ortega y Gasset.

Grimshaw, J. (1982). *On the Lexical Representation of Romance Reflexive Critics*. MIT Press, Cambridge.

Guasti, M. T. (1992). Incorporation, Excorporation and Lexical Properties of Causative Heads. 8:209–232.

Guasti, M. T. (1993). *Causative and Perception Verbs: A Comparative Study*. Rosenberg and Sellier, Turin.

Guasti, M. T. (2006). Analytic Causatives. In Everaert, M. and Riemsdijk, H. v., editors, *The Blackwell Companion to Syntax*.

Haverinen, K. (2013). Syntax Annotation Guidelines for the Turku Dependency Treebank. Technical Report 1034, Turku Centre for Computer Science.

Hellwig, P. (2003). Dependency Unification Grammar. In Ágel, V., Eichinger, L. M., Eroms, H. W., Hellwig, P., Heringer, H. J., and Lobin, H., editors, *Dependency*

and valency: an international handbook of contemporary research, number 25 in Handbooks of linguistics and communication sciences. Walter de Gruyter, Berlin; New York.

Hopper, P. and Traugott, E. (2003). *Grammaticalization*. Cambridge University Press, Cambridge, UK; New York, NY, 2nd edition.

Huddleston, R. and Pullum, G. K. (2002). *The Cambridge Grammar of the English Language*. Cambridge University Press.

Hudson, R. A. (1984). *Word grammar*. B. Blackwell, Oxford.

Ivanova, A., Oepen, S., and Øvrelid, L. (2013). Survey on parsing three dependency representations for English. *51st Annual Meeting of the Association for Computational Linguistics Proceedings of the Student Research Workshop*, pages 31–37.

Ivanova, A., Oepen, S., Øvrelid, L., and Flickinger, D. (2012). Who Did What to Whom?: A Contrastive Study of Syntacto-semantic Dependencies. In *Proceedings of the Sixth Linguistic Annotation Workshop*, LAW VI '12, pages 2–11, Stroudsburg, PA, USA. Association for Computational Linguistics.

Jackendoff, R. (1990). *Semantic Structures*. MIT Press, Cambridge.

Johannsen, A., Hovy, D., and Søgaard, A. (2015). Cross-lingual syntactic variation over age and gender. In *Proceedings of CoNLL*.

Johansson, R. and Nugues, P. (2007). Extended constituent-to-dependency conversion for English. In Nivre, J., Kalep, H.-J., Muischnek, K., and Koit, M., editors, *NODALIDA 2007 Proceedings*, pages 105–112, Tartu, Estonia. University of Tartu.

Jurafsky, D. and Martin, J. H. (2009). *Speech and language processing: an introduction to natural language processing, computational linguistics, and speech recognition*. Pearson Prentice Hall, Upper Saddle River, N.J., 2nd edition.

Kayne, R. (1975). *French Syntax: The Transformational Cycle*. MIT Press, Cambridge.

Kayne, R. (1988). Romance Se/Si. volume 20 of *GLOW Newsletter*.

Kayne, R. (1989). Null Subjects and Clitic Climbing. In Jaeggli, O. and Safir, K., editors, *The Null Subject Parameter*, pages 239—-261.

Kayne, R. S. (1991). Romance clitics, verb movement and PRO. *Linguistic Inquiry*, 22:647–686.

Kübler, S., McDonald, R., and Nivre, J. (2009). Dependency Parsing. *Synthesis Lectures on Human Language Technologies*, 2(1):1–127.

Kim, J.-D., Ohta, T., Pyysalo, S., Kano, Y., and Tsujii, J. (2009). Overview of BioNLP'09 Shared Task on Event Extraction. In *Proceedings of the Workshop on Current Trends in Biomedical Natural Language Processing: Shared Task*, BioNLP '09, pages 1–9, Stroudsburg, PA, USA. Association for Computational Linguistics.

Kim, J.-D., Ohta, T., Tateisi, Y., and Tsujii, J. (2003). GENIA corpus–semantically annotated corpus for bio-textmining. *Bioinformatics (Oxford, England)*, 19 Suppl 1:i180–182.

King, T. H., Crouch, R., Riezler, S., Dalrymple, M., and Kaplan, R. M. (2003). The PARC 700 Dependency Bank. In *In Proceedings of the 4th International Workshop on Linguistically Interpreted Corpora*, pages 1–8.

Kirilina, A. and Versley, Y. (2015). What is hard in Universal Dependency Parsing? In *Proceedings of the Sixth Workshop on Statistical Parsing of Morphologically-Rich Languages*, SPMRL '15, pages 31–38, Stroudsburg, PA, USA. Association for Computational Linguistics.

Klein, D. and Manning, C. D. (2003). Accurate Unlexicalized Parsing. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics*, volume 1 of *ACL '03*, pages 423–430, Stroudsburg, PA, USA. Association for Computational Linguistics.

Klein, D. and Manning, C. D. (2004). Corpus-based Induction of Syntactic Structure: Models of Dependency and Constituency. In *Proceedings of the 42nd Annual*

Meeting on Association for Computational Linguistics, ACL '04, Stroudsburg, PA, USA. Association for Computational Linguistics.

Koehn, P. (2005). Europarl: A Parallel Corpus for Statistical Machine Translation. In *Conference Proceedings: the tenth Machine Translation Summit*, pages 79–86, Phuket, Thailand. AAMT.

Kromann, M. T., Mikkelsen, L., and Lynge, S. K. (2004).

Kruijff, G. J. M. (2006). Dependency Grammar. In *Encyclopedia of Language & Linguistics (Second Edition)*, pages 444–450. Elsevier, Oxford.

Labelle, M. (2008). The French Reflexive and Reciprocal SE. *Natural Language and Linguistic Theory*, 26(4):833–876.

Lin, D. (2003). Dependency-Based Evaluation of Minipar. In Abeillé, A., editor, *Treebanks*, number 20 in Text, Speech and Language Technology, pages 317–329. Springer Netherlands.

Manning, C. D. (1992). Romance is so complex. Technical report 168, Center for the Study of Language and Information.

Manzini, M. R. (1986). On Italian Si. In Borer, H., editor, *The Syntax of Pronominal Clitics*.

Manzini, R. M. (1983). *Restructuring and Reanalysis*. PhD thesis, Cambridge.

Marcus, M. P., Marcinkiewicz, M. A., and Santorini, B. (1993). Building a large annotated corpus of English: The Penn Treebank. *Computational linguistics*, 19(2):313–330.

Masullo, P. (1999). La Interfaz Léxico-Sintaxis: Presencia y Ausencia del Clítico Se en Construcciones Inacusativas del Español.

McClosky, D. and Charniak, E. (2008). Self-Training for Biomedical Parsing. In *Proceedings of ACL-08: HLT, Short Papers*, pages 101–104, Columbus, Ohio. Association for Computational Linguistics.

McDonald, R., Lerman, K., and Pereira, F. (2006). Multilingual Dependency Analysis with a Two-stage Discriminative Parser. In *Proceedings of the Tenth Conference on Computational Natural Language Learning*, CoNLL-X '06, pages 216–220, Stroudsburg, PA, USA. Association for Computational Linguistics.

McDonald, R., Nivre, J., Quirmbach-Brundage, Y., Goldberg, Y., Das, D., Ganchev, K., Hall, K., Petrov, S., Zhang, H., Täckström, O., Bedini, C., Bertomeu Castelló, N., and Lee, J. (2013). Universal Dependency Annotation for Multilingual Parsing. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 92–97, Sofia, Bulgaria. Association for Computational Linguistics.

McDonald, R., Petrov, S., and Hall, K. (2011). Multi-Source Transfer of Delexicalized Dependency Parsers. *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 62–72.

Medová, L. (2009). *Reflexive Clitics in the Slavic and Romance Languages*. PhD thesis, Princeton University.

Mel'čuk, I. A. (1988). *Dependency Syntax: Theory and Practice*. State University of New York, Albany.

Mel'čuk, I. A. (2009). Dependency in Natural Language. In Polguère, A. and Mel'čuk, I. A., editors, *Dependency in linguistic description*, volume 111 of *Studies in language companion series*. John Benjamins Pub. Co, Amsterdam; Philadelphia.

Mendikoetxea, A. (1997). Clitics as Agr, and PRO in finite clauses. In Catalá, N. and Bargalló, M., editors, *Proceedings of the IV Colloquium of Generative Grammar*.

Mendikoetxea, A. (1999). Construcciones con SE: Medias, pasivas e impersonales. In Bosque, I. and Demonte, V., editors, *Gramática Descriptiva de la Lengua Española*.

Mendikoetxea, A. (2012). Passives and se Constructions. In *The Handbook of Hispanic Linguistics*, pages 477–502. John Wiley & Sons, Ltd.

Miwa, M., Pyysalo, S., and Hara, T. (2010a). Evaluating dependency representation for event extraction. In *In Proceedings of the 23rd International Conference on Computational Linguistics*, pages 779–787.

Miwa, M., Pyysalo, S., Hara, T., and Tsujii, J. (2010b). A Comparative Study of Syntactic Parsers for Event Extraction. In *Proceedings of the 2010 Workshop on Biomedical Natural Language Processing*, BioNLP '10, pages 37–45, Stroudsburg, PA, USA. Association for Computational Linguistics.

Miwa, M., Saetre, R., Kim, J.-D., and Tsujii, J. (2010c). Event extraction with complex event classification using rich features. *Journal of Bioinformatics and Computational Biology*, 8(1):131–146.

Miyao, Y., Sætre, R., Sagae, K., Matsuzaki, T., and Tsujii, J. (2008). Task-oriented Evaluation of Syntactic Parsers and Their Representations. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*.

Mohanan, T. (2006). Grammatical Verbs (with special reference to light verbs). In Everaert, M. and van Riemsdijk, H., editors, *The Blackwell Companion to Syntax*.

Montemagni, S., Barsotti, F., Battista, M., Calzolari, N., Corazzari, O., Lenci, A., Zampolli, A., Fanciulli, F., Massetani, M., Raffaelli, R., Basili, R., Pazienza, M. T., Saracino, D., Zanzotto, F., Mana, N., Pianesi, F., and Delmonte, R. (2003). Building the Italian syntactic-semantic treebank. In Abeillé, A., editor, *Building and using Parsed Corpora*, Language and Speech series, pages 189–210, Dordrecht. Kluwer.

Moore, J. (1991). *Reduced Constructions in Spanish*. Routledge, New York.

Moore, J. (1998). Object-controlled Restructuring in Spanish.

Mott, J., Bies, A., Laury, J., and Warner, C. (2012). Bracketing Webtext: An Addendum to Penn Treebank II Guidelines.

Nilsson, J., Nivre, J., and Hall, J. (2006). Graph Transformations in Data-driven Dependency Parsing. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th Annual Meeting of the Association for Computational Linguistics*, ACL-44, pages 257–264, Stroudsburg, PA, USA. Association for Computational Linguistics.

Nilsson, J., Nivre, J., and Hall, J. (2007). Generalizing Tree Transformations for Inductive Dependency Parsing. *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 968–975.

Nishida, C. (2013). Restructuring and Clitic Climbing in Romance: A Categorial Grammar Analyis. In Artemis Alexiadou, T. K. and Müller, G., editors, *Local Modelling of Non-Local Dependencies in Syntax*, pages 371–400.

Nivre, J. (2009). Non-projective Dependency Parsing in Expected Linear Time. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP:*, volume 1 of *ACL '09*, pages 351–359, Stroudsburg, PA, USA. Association for Computational Linguistics.

Nivre, J., Agić, e., Ahrenberg, L., Aranzabe, M. J., Asahara, M., Atutxa, A., Ballesteros, M., Bauer, J., Bengoetxea, K., Berzak, Y., Bhat, R. A., Bosco, C., Bouma, G., Bowman, S., Cebiroğlu Eryiğit, G., Celano, G. G. A., Çöltekin, a., Connor, M., de Marneffe, M.-C., Diaz de Ilarraza, A., Dobrovoljc, K., Dozat, T., Droganova, K., Erjavec, T., Farkas, R., Foster, J., Galbraith, D., Garza, S., Ginter, F., Goenaga, I., Gojenola, K., Gokirmak, M., Goldberg, Y., Gómez Guinovart, X., Gonzáles Saavedra, B., Grūzītis, N., Guillaume, B., Hajič, J., Haug, D., Hladká, B., Ion, R., Irimia, E., Johannsen, A., Kaşıkara, H., Kanayama, H., Kanerva, J., Katz, B., Kenney, J., Krek, S., Laippala, V., Lam, L., Lenci, A., Ljubešić, N., Lyashevskaya, O., Lynn, T., Makazhanov, A., Manning, C., Mărănduc, C., Mareček, D., Martínez Alonso, H., Mašek, J., Matsumoto, Y., McDonald, R., Missilä, A., Mititelu, V., Miyao, Y., Montemagni, S., Mori, K. S., Mori, S., Muischnek, K., Mustafina, N., Müürisep, K., Nikolaev, V., Nurmi, H., Osenova, P., Øvrelid, L., Pascual, E., Passarotti, M.,

Perez, C.-A., Petrov, S., Piitulainen, J., Plank, B., Popel, M., Pretkalniņa, L., Prokopidis, P., Puolakainen, T., Pyysalo, S., Ramasamy, L., Rituma, L., Rosa, R., Saleh, S., Saulīte, B., Schuster, S., Seeker, W., Seraji, M., Shakurova, L., Shen, M., Silveira, N., Simi, M., Simionescu, R., Simkó, K., Simov, K., Smith, A., Spadine, C., Suhr, A., Sulubacak, U., Szántó, Z., Tanaka, T., Tsarfaty, R., Tyers, F., Uematsu, S., Uria, L., van Noord, G., Varga, V., Vincze, V., Wang, J. X., Washington, J. N., Žabokrtský, Z., Zeman, D., and Zhu, H. (2016). Universal Dependencies 1.3. *http://universaldependencies.org/*.

Nivre, J., Agić, e., Aranzabe, M. J., Asahara, M., Atutxa, A., Ballesteros, M., Bauer, J., Bengoetxea, K., Bhat, R. A., Bosco, C., Bowman, S., Celano, G. G. A., Connor, M., de Marneffe, M.-C., Diaz de Ilarraza, A., Dobrovoljc, K., Dozat, T., Erjavec, T., Farkas, R., Foster, J., Galbraith, D., Ginter, F., Goenaga, I., Gojenola, K., Goldberg, Y., Gonzales, B., Guillaume, B., Hajič, J., Haug, D., Ion, R., Irimia, E., Johannsen, A., Kanayama, H., Kanerva, J., Krek, S., Laippala, V., Lenci, A., Ljubešić, N., Lynn, T., Manning, C., Mărănduc, C., Mareček, D., Martínez Alonso, H., Mašek, J., Matsumoto, Y., McDonald, R., Missilä, A., Mititelu, V., Miyao, Y., Montemagni, S., Mori, S., Nurmi, H., Osenova, P., Øvrelid, L., Pascual, E., Passarotti, M., Perez, C.-A., Petrov, S., Piitulainen, J., Plank, B., Popel, M., Prokopidis, P., Pyysalo, S., Ramasamy, L., Rosa, R., Saleh, S., Schuster, S., Seeker, W., Seraji, M., Silveira, N., Simi, M., Simionescu, R., Simkó, K., Simov, K., Smith, A., Štěpánek, J., Suhr, A., Szántó, Z., Tanaka, T., Tsarfaty, R., Uematsu, S., Uria, L., Varga, V., Vincze, V., Žabokrtský, Z., Zeman, D., and Zhu, H. (2015a). Universal Dependencies 1.2. *http://universaldependencies.github.io/docs/*.

Nivre, J., Bosco, C., Choi, J., de Marneffe, M.-C., Dozat, T., Farkas, R., Foster, J., Ginter, F., Goldberg, Y., Hajič, J., Kanerva, J., Laippala, V., Lenci, A., Lynn, T., Manning, C., McDonald, R., Missilä, A., Montemagni, S., Petrov, S., Pyysalo, S., Silveira, N., Simi, M., Smith, A., Tsarfaty, R., Vincze, V., and Zeman, D. (2015b). Universal Dependencies 1.0. *http://universaldependencies.github.io/docs/*.

Nivre, J., Hall, J., Nilsson, J., Chanev, A., Eryigit, G., Kübler, S., Marinov, S., and Marsi, E. (2007). MaltParser: A language-independent system for data-driven

dependency parsing. *Natural Language Engineering*, 13(2):95–135.

Nivre, J., Nilsson, J., and Hall, J. (2006). Talbanken05: A Swedish treebank with phrase structure and dependency annotation. In *Proceedings of the Fifth International Conference on Language Resources and Evaluation (LREC 2006)*, Genova, Italy. European Language Resources Association (ELRA).

Oliva, K. (2003). Dependency, Valency and Head-Driven Phrase-Structure Grammar. In Ágel, V., Eichinger, L. M., Eroms, H. W., Hellwig, P., Heringer, H. J., and Lobin, H., editors, *Dependency and valency: an international handbook of contemporary research*, number 25 in Handbooks of linguistics and communication sciences. Walter de Gruyter, Berlin; New York.

Osborne, T. (2015). Diagnostics for Constituents: Dependency, Constituency, and the Status of Function Words. In *Proceedings of the Third International Conference on Dependency Linguistics (Depling 2015)*, pages 251–260, Uppsala, Sweden.

Otero, C. P. (1986). Arbitrary subjects in finite clauses. In I. Bordelois, K. Zagona, and H. Contreras, editors, *Generative Studies in Spanish Syntax*.

Otero, C. P. (1999). Pronombres reflexivos y recìprocos. In Bosque, I. and Demonte, V., editors, *Gramática de la lengua española*, pages 1428–1517.

Perlmutter, D. M., editor (1983). *Studies in Relational Grammar 1*.

Petrov, S., Barrett, L., Thibaux, R., and Klein, D. (2006). Learning Accurate, Compact, and Interpretable Tree Annotation. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th Annual Meeting of the Association for Computational Linguistics*, ACL-44, pages 433–440, Stroudsburg, PA, USA. Association for Computational Linguistics.

Petrov, S., Das, D., and McDonald, R. (2012). A Universal Part-of-Speech Tagset. *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC-2012)*.

Pollard, C. J. and Sag, I. (1994). *Head-driven phrase structure grammar*. Center for the Study of Language and Information;, Stanford.

Pustet, R. (2003). *Copulas: Universals in the Categorization of the Lexicon*. Oxford University Publications, Oxford.

Pyysalo, S., Ginter, F., Laippala, V., Haverinen, K., Heimonen, J., and Salakoski, T. (2007). On the unification of syntactic annotations under the stanford dependency scheme: A case study on bioinfer and genia. In *Biological, translational, and clinical language processing*, pages 25–32, Prague, Czech Republic. Association for Computational Linguistics.

Pyysalo, S., Ohta, T., Rak, R., Sullivan, D., Mao, C., Wang, C., Sobral, B., Tsujii, J., and Ananiadou, S. (2012). Overview of the ID, EPI and REL tasks of BioNLP Shared Task 2011. *BMC bioinformatics*, 13(11):1.

Raposo, E. and Uriagereka, J. (1996). Indefinite Se. *Natural Language and Linguistic Theory*, 14:749–810.

Rasooli, M. S. and Collins, M. (2015). Density-driven cross-lingual transfer of dependency parsers. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 328–338, Lisbon, Portugal. Association for Computational Linguistics.

Reinhart, T. and Siloni, T. (2004). Against the Unaccusative Analysis of Reflexives. In Artemis Alexiadou, E. A. and Everaert, M., editors, *The Unaccusativity Puzzle*, pages 159–180.

Rivero, M. L. (2001). On impersonal reflexives in Romance and Slavic and semantic variation. In J. Camps and C.R. Wiltshire, editors, *Romance Syntax, Semantics and L2 Acquisition*.

Rizzi, L. (1978). A Restructuring Rule in Italian Syntax. In J, S., editor, *Recent Transformational Studies in European Languages*.

Rizzi, L., editor (1982). *Issues in Italian Syntax*. Foris, Dordrecht.

Rizzi, L. and Cinque, G. (2016). Functional Categories and Syntactic Theory. *Annual Review of Linguistics*, 2(1):139–163.

Roberts, I. (1997). Restructuring, Head Movement and Locality. *Linguistic Inquiry*, 28:423–460.

Roby, D. B. (2009). *Aspect and the Categorization of States*. Number 114 in Studies in language companion series. John Benjamins Publishing Company.

Rosa, R., Mašek, J., Mareček, D., Popel, M., Zeman, D., and Žabokrtský, Z. (2014). HamleDT 2.0: Thirty Dependency Treebanks Stanfordized. *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC-2014)*.

Rouveret, A. and Vergnaud, J.-R. (1980). Specifying Reference to the Subject: French Causatives and Conditions on Representations. *Linguistic Inquiry*, 11:97–202.

Sag, I. A., Baldwin, T., Bond, F., Copestake, A. A., and Flickinger, D. (2002). Multiword Expressions: A Pain in the Neck for NLP. In *Proceedings of the Third International Conference on Computational Linguistics and Intelligent Text Processing*, CICLing '02, pages 1–15, London, UK, UK. Springer-Verlag.

Sagae, K. and Tsujii, J. (2007). Dependency Parsing and Domain Adaptation with LR Models and Parser Ensembles. In *Proceedings of the CoNLL Shared Task Session of EMNLP-CoNLL 2007*, pages 1044–1050, Prague, Czech Republic. Association for Computational Linguistics.

Schmid, H. (1995). Improvements in Part-of-Speech Tagging with an Application to German. In Ireland, D., editor, *Proceedings of the ACL SIGDAT-Workshop*.

Schuster, S. and Manning, C. D. (2016). Enhanced English Universal Dependencies: An Improved Representation for Natural Language Understanding Tasks. In *Language Resources and Evaluation (LREC)*.

Schwartz, R., Abend, O., and Rappoport, A. (2012). Learnability-based Syntactic Annotation Design. In *Proceedings of COLING 2012: Technical Papers*, pages 2405–2422, Mumbai.

Sgall, P., Hajicová, E., Panenová, J., and Mey, J. (1986). *The Meaning of the Sentence in its Semantic and Pragmatic Aspects*. Springer, Dordrecht; Boston: Hingham, MA, U.S.A.

Shriberg, E. E. (1994). *Preliminaries to a theory of speech disfluencies*. PhD thesis, University of California, Berkeley.

Silveira, N., Dozat, T., de Marneffe, M.-C., Bowman, S., Connor, M., Bauer, J., and Manning, C. D. (2014). A Gold Standard Dependency Corpus for English. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC-2014)*.

Silveira, N. and Manning, C. (2015). Does universal dependencies need a parsing representation? an investigation of english. In *Proceedings of the Third International Conference on Dependency Linguistics (Depling 2015)*, pages 310–319, Uppsala, Sweden. Uppsala University, Uppsala, Sweden.

Simpson, J. (1983). Resultatives. In Beth Levin, M. R. and Zaenen, A., editors, *Papers in Lexical-Functional Grammar*.

Sleator, D. D. and Temperley, D. (1991). *Parsing English with a Link Grammar*.

Starosta, S. (1988). *The case for lexicase: an outline of lexicase grammatical theory*. Pinter Publishers, London;.

Stassen, L. (2013). Consonant Inventories. In Dryer, M. S., editor, *The World Atlas of Language Structures*.

Surdeanu, M., Johansson, R., Meyers, A., Màrquez, L., and Nivre, J. (2008). The CoNLL-2008 Shared Task on Joint Parsing of Syntactic and Semantic Dependencies. In *Proceedings of the Twelfth Conference on Computational Natural Language*

Learning, CoNLL '08, pages 159–177, Stroudsburg, PA, USA. Association for Computational Linguistics.

Swanson, B. and Charniak, E. (2014). Data Driven Language Transfer Hypotheses. In *EACL*, pages 169–173.

Teomiro García, I. (2010). *Anaphors at the Interfaces: A Comparative Study of the Variation of the Anaphoric Systems of English, Dutch and Spanish.* PhD thesis, Universidad Autónoma de Madrid.

Tesnière, L. (2015). *Elements of structural syntax.* John Benjamins.

Trautner-Kromann, H. (2003). The Danish Dependency Treebank and the DTAG Treebank Tool.

Tsarfaty, R. (2013). A unified morpho-syntactic scheme of stanford dependencies. In *Proceedings of ACL.*

Tsarfaty, R., Nivre, J., and Andersson, E. (2012). Cross-framework Evaluation for Statistical Parsing. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, EACL '12, pages 44–54, Stroudsburg, PA, USA. Association for Computational Linguistics.

Tsarfaty, R., Nivre, J., and Ndersson, E. (2011). Evaluating dependency parsing: robust and heuristics-free cross-nnotation evaluation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 385–396. Association for Computational Linguistics.

Tsochantaridis, I., Joachims, T., Hofmann, T., and Altun, Y. (2005). Large margin methods for structured and interdependent output variables. *J. Mach. Learn. Res.*, 6:1453–1484.

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

Van Valin Jr., R. (1992). A synopsis of Role and Reference Grammar. In Van Valin Jr., R., editor, *Advances in Role and Reference Grammar*. John Benjamins Publishing Company, Amsterdam; Philadelphia.

Wehrli, E. (1986). On Some Properties of French Clitic Se. In Borer, H., editor, *The Syntax of Pronominal Clitics*, pages 263–283.

Wurmbrand, S. (2001). *Infinitives: Restructuring and Clause Structure*. Mouton de Gruyter, Berlin/New York.

Wurmbrand, S. (2006). Verb Clusters, Verb Raising, and Restructuring. In Everaert, M. and Riemsdijk, H. v., editors, *The Blackwell Companion to Syntax*.

Yamada, H. and Matsumoto, Y. (2003). Statistical dependency analysis with support vector machines. In *Proceedings of IWPT*, volume 3.

Yan, X. and Han, J. (2002). gSpan: Graph-Based Substructure Pattern Mining. In *Proceedings of the 2002 IEEE International Conference on Data Mining*, ICDM '02, pages 721–, Washington, DC, USA. IEEE Computer Society.

Zeman, D. (2008). Reusable Tagset Conversion Using Tagset Drivers. In *Proceedings of the Sixth International Conference on Language Resources and Evaluation*.

Zeman, D., Mareček, D., Popel, M., Ramasamy, L., Štěpánek, J., Žabokrtský, Z., and Hajič, J. (2012). HamleDT: To Parse or Not to Parse? *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC-2012)*.

Zeman, D. and Resnik, P. (2008). Cross-Language Parser Adaptation between Related Languages. In *IJCNLP 2008 Workshop on NLP for Less Privileged Languages*, pages 35–42, Hyderabad, India. International Institute of Information Technology.

Zucchi, A. (1993). *The Language of Propositions and Events (Studies in Linguistics and Philosophy)*. Kluwer, Dordrecht.