University of Warsaw
Faculty of Philosophy

Magdalena Borysiak
Record book number: 446267

Dependency structure of English coordination: a surface-syntactic approach

Bachelor's thesis
in the field of Cognitive Science

The thesis was written under the supervision of
prof. dr hab. Adam Przepiórkowski
Faculty of Philosophy, University of Warsaw
Institute of Computer Science, Polish Academy of Sciences

Warsaw, June 2024

Summary

This thesis describes a corpus study on coordinate structures in English. Previous studies have shown some tendencies in how coordinations are formed and the theoretical consequences of those tendencies. The Dependency Length Minimisation effect was used to argue for the symmetric approaches to the dependency structure of coordination in one of the previous studies and another argued specifially in favour of the multi-headed approach. This current study continues the research, but using the Surface-syntactic Universal Dependencies instead of Universal Dependencies to create trees for analysis of coordinations. This choice is motivated by the strive to improve the quality of the data by using an annotation scheme that was found to possibly improve parser performance and by taking advantage of the structural qualities of the trees created according to this scheme. The chosen annotation scheme does not necessarily improve the data quality, but still allows for an analysis of coordinations. Evaluation of the discussed parsing strategies is presented, as well as the results of the coordination analysis, which provide some evidence for one of the symmetric interpretations of coordinate structures. Those results are discussed along with some unexpected findings.

Keywords

coordination, dependency grammar, corpus linguistics, Dependency Length Minimization, neural network training, Universal Dependencies, Surface-syntactic Universal Dependencies

Title of the thesis in Polish language

Struktura zależnościowa koordynacji w języku angielskim: podejście powierzchniowo-składniowe

Contents

Acknowledgements

I would like to sincerely thank my thesis supervisor, prof. Adam Przepiórkowski, for all of the support he provided and for having more patience for me than anyone should ever have.

I would also like to thank my family and friends, especially Alicja Caban and Wojciech Stempniak, for their emotional and linguistic support.

Chapter 1

Introduction

The aim of this work is to expand on the research on the coordinate structures in the English language. According to (Huddleston and Pullum, 2002, p. 66), "coordination is a relation between two or more elements of syntactically equal status", which are here called conjuncts. The joining of those conjuncts can be marked using a word called conjunction – this could for instance be *and*, *but* or *or*. An example of a coordination is given in (1) – the word *and* serves as the conjunction, *some apples* and *the oranges your mother gave you* as the conjuncts.

(1) *Bring* [[some apples] and [the oranges your mother gave you]].

This coordination is consistent with the observation that in English there is a tendency for the conjuncts on the left of a coordination to be shorter than the ones on the right. According to Przepiórkowski and Woźniak (2023), however, the placement of the coordination's governor (the word *Bring* in the sentence (1)) might have an influence on this tendency: if the governor is on the left (as in (1)) or absent from the coordination altogether (as in (2)), then the tendency for the shorter conjunct to be placed on the left grows, but not if the governor is on the right (as in (3)).

(2) [[Buy some apples] or [steal as many oranges as you can hold]].

(3) [[Three long orange peels] and [an apple]] *fell* out of the bag.

Authors of the study found that this tendency changes with the length difference between the conjuncts of a coordination. For instance, the pressure to order the conjuncts a certain way was stronger in sentence (1) than in the sentence *Bring apples and oranges*. The Dependency Length Minimization effect was proposed as an explanation for this. The DLM effect is a tendency observed in some languages to form sentences in a way that minimises dependency lengths, which is achieved by placing related phrases and words close to each other.

Przepiórkowski and Woźniak (2023) conducted a corpus study and investigated the way in which coordinations are formed. Their observations had theoretical consequences for the possible ways of syntactic annotation of coordinations. However, the corpus they used was relatively small, therefore Przepiórkowski et al. (2024) tried to replicate the results on a bigger corpus. Their results sharpened the conclusions from previous research, but the quality of data was low. The aim of the current study is to investigate the coordinations in English again, but using a somewhat different approach to finding the coordinations in text to improve data quality.

The structure of this thesis is the following: Chapter 2 describes in more detail the theoretical aspects mentioned in the current chapter – this includes the Dependency Length Minimization effect, different aproaches to syntactic annotation and specifically to annotation of coordinate structures. Chapter 3 describes how the corpus data was prepared for analysis, i.e., how the raw text was processed to find coordinations. Chapter 4 presents the statistical analysis of the data, which is discussed in Chapter 5.

Chapter 2

Theoretical background

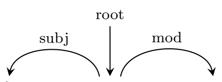
2.1 Dependency grammars

The first full-fledged theory of dependency grammar was proposed by Tesnière (1959, 2015). One of the key ideas included in his work was that a sentence is not comprised solely of its words, but also of the connections between them – dependencies. The connections he proposed were directed, therefore one of the two connected words is always a governor (head) and the other is a dependent. Another crucial element of Tesnière's approach was verb centrality – the verb is the root of every sentence structure in dependency grammar.

(4)
$$\begin{tikzcd}\text{Alfred} & \text{speaks} \arrow[l, bend right] \arrow[r, bend left] & \text{slowly.}\end{tikzcd}$$

Tesnière as an example used the sentence *Alfred speaks slowly*, for which a dependency tree is shown in (4). It visualises the rules described above – all of the words in a sentence are connected, those connections are directed and the verb is central to the whole structure.

Dependency grammars have changed significantly since Tesnière's ideas were published. One example of such changes is the widespread usage of dependency labels, which describe the grammatical function that a word serves – Tesnière differentiated only between actants and circumstants, which can be understood as obligatory dependencies of a verb, which complete its meaning, and optional dependencies, which are not necessary to complete the meaning of the verb. Different corpora have their own ideas for sets of dependency labels, for instance full annotation for (4) could look similarly to (5), with the label `root` marking the central element of the sentence, `subj` marking the subject of the sentence and `mod` marking a modifier of the verb.

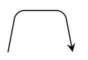


(5) Alfred speaks slowly.

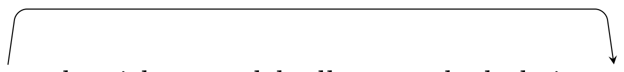
Section 2.2 describes some specific phenomena that can be explained using dependency grammars. Section 2.3 presents some of the theoretical approaches to coordinate structures and Section 2.4 describes the studies that were the basis for the present one. The last two sections of this chapter delve into more detail about two projects concerned with creating consistent dependency annotation schemes – Universal Dependencies in Section 2.5 and Surface-syntactic Universal Dependencies in Section 2.6.

2.2 Dependency length minimization

Familiarity with dependency grammars helps understand the principle of Dependency Length Minimization (henceforth DLM). It states that natural languages prefer shorter dependencies in their sentences. An example from Hunter and Prideaux (1983), shown in (6), illustrates this.



(6) a. The janitor threw out the rickety and badly scratched chair.



b. The janitor threw the rickety and badly scratched chair out.

The study has shown that speakers deem sentences similar to (6a) more acceptable than the ones similar to (6b).[1] Proposed explanations for this preference are based on language-processing constraints, which are usually said to be caused by working memory limitations. With longer dependencies, while reading or hearing a sentence, a person has to keep certain words in their working memory for a longer time. The longer the dependency, the harder the retrieval of the needed word from the working memory. Similar effects have been found in other studies, both psycholinguistic ones (King and Just, 1991; Gibson, 1998) and those based on corpus research (Gildea and Temperley, 2007, 2010; Futrell et al., 2020; Dyer, 2023).

The DLM effect has been observed both at the level of usage and at the level of grammar. The level of usage is visible in (6) – when there are multiple

[1]In the study, it is actually found that it is not the distance between the verb and the particle that affects the acceptability of sentences like those, but the syntactic complexity of the phrases within that distance. However, according to Wasow (2002), syntactic complexity as a measure of dependency length correlates with many others proposed, including the number of intervening words.

grammatical word orders available, people tend to choose those with shortest dependencies, because it makes the sentence easier to understand. As for DLM in grammar, an example taken from (Hawkins, 1994, p. 20) is in (7).

(7) a. * Did $_{S}$[that John failed his exam] surprise Mary?

b. Did $_{NP}$[that fact] surprise Mary?

Both of the sentences in (7) have a constituent embedded inside of them. In (7a) that constituent is a subordinate clause *that John failed his exam*, whereas in (7b) the constituent is a noun phrase *that fact*. Subordinate clauses are usually longer than noun phrases, therefore dependencies in sentences with embedded clauses can be much longer. According to the DLM principle, this makes the sentence more difficult to process and Hawkins argues that due to this processing difficulty the effect was grammaticalised in English and therefore sentences with clauses embedded this way have become ungrammatical. Since noun phrases are usually shorter, sentences similar to (7b) are allowed.

2.3 Possible dependency structures of a coordination

Different dependency structures of coordination are assumed in different dependency grammars and dependency corpora. Popel et al. (2013) proposed a taxonomy of those, which consists of three families of structures: Prague, Stanford and Moscow. Przepiórkowski and Woźniak (2023) add to those three a London family. All four of those families are described in more detail in the following subsections. Diagrams are used to better illustrate them, where:

- $\odot$ is the governor of the coordination;
- each $\diamond$ symbolises a token, grouped together with a few others in a rectangle, forming a conjunct;
- $\square$ is the conjunction of the coordination.

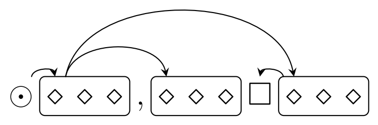
Therefore, using this set of symbols, the sentence *Mary and her younger sister laughed* would look like in (8).

(8)	Mary	and	her younger sister	laughed
	$\boxed{\diamond}$	$\square$	$\boxed{\diamond \diamond \diamond}$	$\odot$

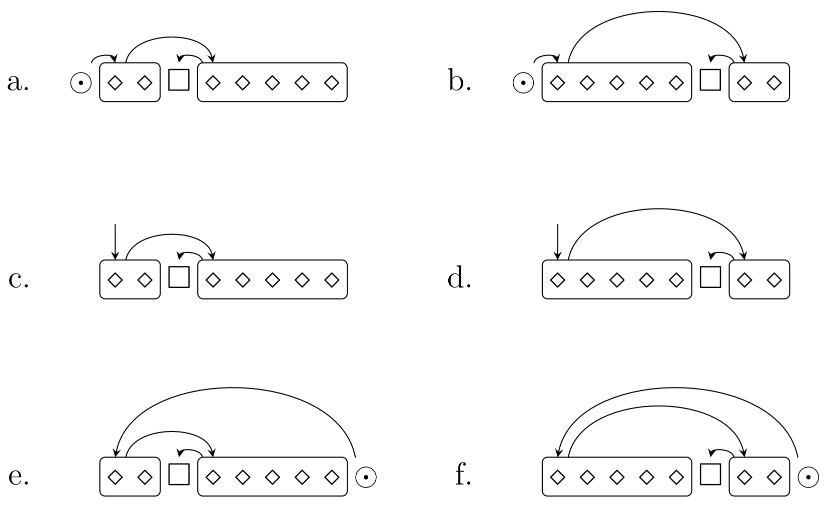
This section describes the predictions about ordering conjuncts in a coordination that can be made assuming different approaches to coordination and the DLM effect at usage. The predictions are simplified for clarity, but more thorough analysis is provided in Section 2.4. In all of the diagrams in the current section, it is assumed that the head of the conjunct is its first word. This assumption is justified, because the work presented here is based solely on the English language, which is mostly head-initial, therefore the diagrams presented here are more likely to be shaped as they are here than in any other way.

2.3.1 Bouquet/Stanford

The bouquet structure comes from the Stanford parser (de Marneffe et al., 2006). Coordination with three conjuncts and a governor on the left would be annotated in the bouquet approach as shown in (9).

(9) 

In this approach there is a dependency connecting the governor of the coordination to the first conjunct, which is then connected to the heads of each conjunct, thus forming a bouquet. The conjunction is attached to the last conjunct in the structure. This structure is one of the asymmetrical ones, since it does not treat all conjuncts of the coordination equally – it places emphasis on the first conjunct of a coordination by making it the head of every other conjunct.

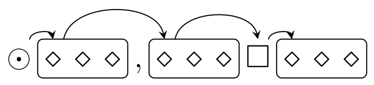
(10) 

The diagrams in (10) show the Stanford structure with different governor positions and conjunct placements. In (10a–b) the governor is on the left with the shorter conjunct on the left in (10a) and on the right in (10b). In those diagrams, two of the dependencies drawn have the same lenght in both cases, but the third one is visibly longer in (10b). This means that, according to the DLM principle, the structure in (10a) should be preferred over (10b), so coordinations with conjuncts of significantly different lengths should have the shorter conjunct on the left.

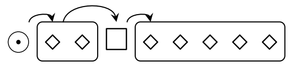
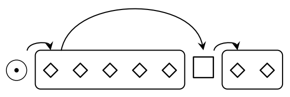
Within the Stanford approach this is the case for every governor position. Diagrams (10c–d) show the structure of coordination when the governor is absent, with dependencies shorter in (10c) than in (10d), i.e., when the shorter conjunct is on the left. Diagrams (10e–f) show the structure when the governor is on the right, with dependencies shorter in (10e) than in (10f), also when the shorter conjunct is on the left. Therefore within this approach the position of the governor does not influence the preferred order of the conjuncts and the shorter conjunct should always be placed on the left.

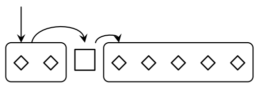
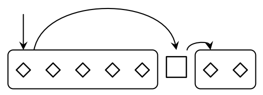
2.3.2 Chain/Moscow

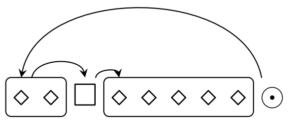
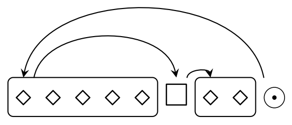
Another asymmetrical approach is the chain, or Moscow one, shown in (11). It is postulated in the Meaning-Text Theory (Mel'čuk, 1988).

(11) 

The structure is created by connecting the governor to the first conjunct of the coordination, then every element of the coordination (including the conjunction) to the next one. As the dependency between the governor and the coordination is always between the governor and the first conjunct, the chain approach is also asymmetrical.

(12) a.  b. 

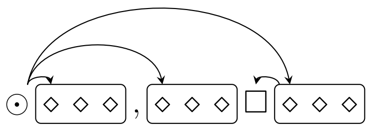
c.  d. 

e.  f. 

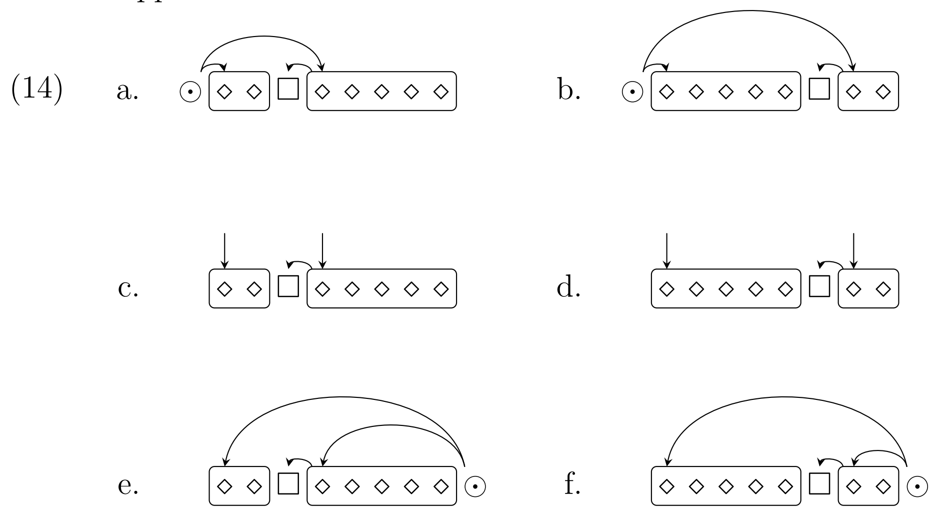
The diagrams in (12) show that, in this case, similarly to the Stanford approach, the placement of the shorter conjunct on the left should always be preferred, assuming the DLM principle. The placement of the governor, again, does not have an effect on the ordering.

2.3.3 Multi-headed/London

The symmetrical approaches, as the name suggests, treat every conjunct in the coordination the same way. One of them is the multi-headed, or London approach, for which Przepiórkowski and Woźniak (2023) propose the name based on its appearance in Word Grammar, developed by Hudson (1984, 2010) at University College London.

(13) 

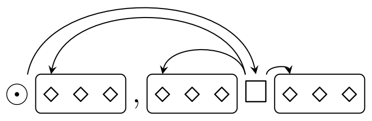
The symmetry of the approach comes from the fact that no conjunct is distinguished by being the only direct dependent of coordinations governor. Instead, all of the conjuncts have dependencies connecting them to the governor, and the conjunction is dependent on the last conjunct, similarly to the Stanford approach.

(14) 

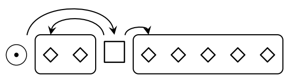
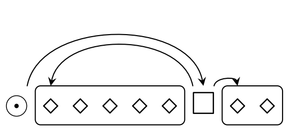
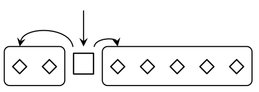
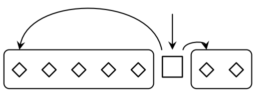
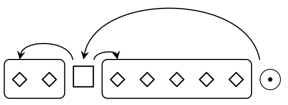
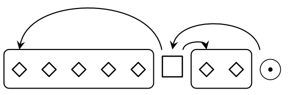
Here, the predictions for the preferred ordering of conjuncts are different from those in asymmetrical approaches. According to this approach the placement of the governor influences the conjunct ordering, specifically the shorter conjunct will tend to be placed near the governor. In (13a) and (13b) the governor is on the left and the dependencies are shorter when the shorter conjunct is also on the left. The opposite is seen in (13e–f): the governor is on the right and putting the shorter conjunct on the right results in shorter dependencies in total. When the coordination has no governor, there is no preference for either of the options, as both have the same sum of dependency lengths.

2.3.4 Conjunction-headed/Prague

The last approach discussed here is the one associated with the Prague Dependency Treebank, called the Prague approach by Popel et al. (2013) or the conjunction-headed approach by Przepiórkowski and Woźniak (2023).

(15)


This is another example of symmetrical approaches, as here again the governor treats all of the conjuncts the same way – in this case, it does not connect to any of them. Instead, there is a dependency connecting the governor and the conjunction, which then has the conjuncts of the coordination as its dependents.

(16)
a. 
b. 
c. 
d. 
e. 
f. 

This annotation style again generates new predictions about the preferred ordering of conjuncts. When the governor is on the left, as in (15a–b), putting the shorter conjunct on the left is preferred, the same happens in (15c–d), when there is no governor. When the governor is on the right, however, both (15e–f) have the same dependency lengths, therefore neither of these is preferred, because neither will shorten the dependencies.

2.4 Previous studies

The current study is a replication of Przepiórkowski and Woźniak (2023), which researched coordinate structures to find out what affected the ordering of conjuncts in an English coordination. The goal was to see whether it is as simple as placing shorter conjuncts on the left or whether the placement of the governor of the coordination has some influence on the ordering. They used the

Penn Treebank, which is an annotated corpus of texts from the Wall Street Journal. This relatively small, but high quality dataset allowed them to make an argument for the symmetric styles of annotating coordination.

The authors first compared the total proportions of coordinations with shorter left conjuncts depending only on the position of the governor, so the probability of finding the shorter conjunct of a coordination on the left depending on whether the governor of the coordination is on the left, on the right or absent. This comparison was not enough to show the influence of the governor, because regardless of the governor position the proportion of shorter left conjuncts was higher, than of shorter right conjuncts. The influence was visible, however, when they took into account how this proportion changes with growing differences in conjunct lengths. The effect of length difference is illustrated in the sentence (17). In (17a) the difference in conjunct lenghts is equal to 1 character, which is not enough to affect the working memory. The difference is bigger in sentence (17b) – 21 characters, 4 words – therefore it is more likely to affect the working memory.

(17) a. Bring $\Big[$[apples] and [oranges]$\Big]$.
b. Bring $\Big[$[some apples] and [the oranges your mother gave you]$\Big]$.

Figure 2.1 presents how modelled proportions of coordinations with the shorter conjunct placed on the left changed with growing differences in conjunct lengths, here measured in words. When the governor is on the left or when it is absent altogether, the proportions grow with length differences. This means that it is more likely that the shorter conjunct will be on the left in (17b) than in (17a). No such tendency was found when the governor is on the right.

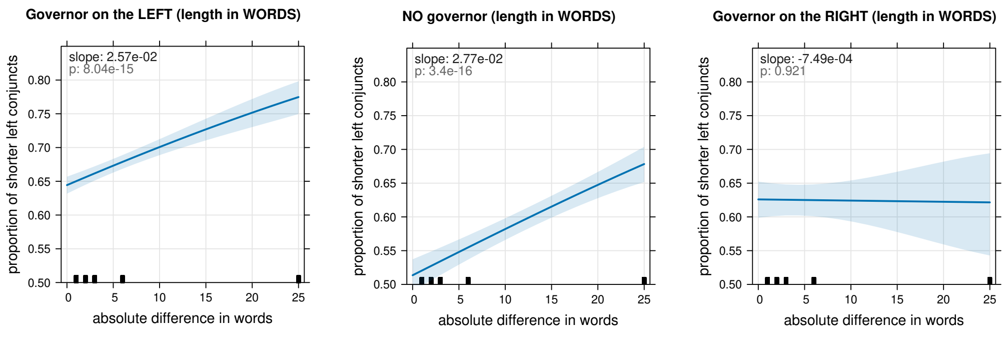


Figure 2.1: Modelled proportions of coordinations with left conjuncts shorter depending on difference in conjunct lengths from Przepiórkowski and Woźniak (2023)

The results obtained by Przepiórkowski and Woźniak (2023) are compatible with the predictions of the symmetric annotation styles: the Prague one, assuming DLM working at the level of usage and the London one, assuming also DLM at the level of grammar. An example of DLM in grammar was given in Section 2.2, but it was not described how it could present itself in coordinations. As was mentioned previously, English is a mostly head-initial language, which in coordinations means that the governor is usually on the left. When the governor is in fact on the left, the dependencies are shortest when the shorter conjunct is also on the left. There is therefore a grammatical pressure to always put shorter conjuncts on the left, because it should usually lead to shorter dependencies in total. This means that when the coordination has no governor and there is no immediate pressure to order the conjuncts in any way, the shorter conjunct still may be placed on the left, because there is a grammatical pressure to do so. However as Przepiórkowski and Woźniak (2023) point out, this pressure may be reduced when the length differences between conjuncts are noticably bigger, because then the DLM effect at the level of usage is stronger.

Przepiórkowski et al. (2024) have already attempted replicating the results of this research. The aim was to see whether the conclusions drawn in the original study hold up when the data come from a bigger and more diverse corpus. The results are presented in Figure 2.2 – slightly different from what was found in the original study, but they sharpened the original conlusions. In the bigger dataset the coordinations with the governor on the left and without a governor behave the same – with growing length differences between conjuncts grows also the proportion of shorter left conjuncts. The difference is that, with the governor on the right, proportion of coordinations with the shorter conjunct on the left decreases with the growing length difference. In the study by Przepiórkowski and Woźniak (2023) it was not clear whether any tendency can be observed when the governor is on the right, therefore this difference was the novel finding of Przepiórkowski et al. (2024).

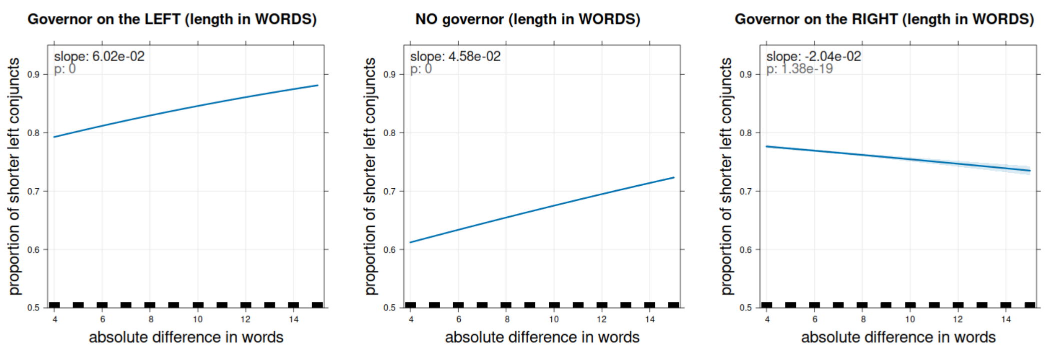


Figure 2.2: Modelled proportions of coordinations with left conjuncts shorter depending on difference in conjunct lengths from Przepiórkowski et al. (2024)

The results shown in Figure 2.2 point to only one of the annotation styles, namely the London one. Within this style, dependencies are minimised when the shorter conjunct is placed closer to the governor, assuming there is one. Therefore with the governor on the left, the shorter conjunct should also be placed on the left (and the chances of that happening grow with the growing length differences, as shown in the left plot in Figure 2.2) and with the governor on the right, the shorter conjunct should also be placed on the right (and the chances of that happening grow with the growing length differences, as shown in the right plot in Figure 2.2). If the coordination has no governor, neither placement should be preferred, unless DLM at the level of grammar is taken into account – then the shorter conjunct should be preferred on the left, since that usually helps minimise the dependencies. This is again compatible with the data, as shown in the middle plot in Figure 2.2.

Przepiórkowski and Woźniak (2023) conducted their study on a relatively small, but manually annotated corpus. Przepiórkowski et al. (2024) replicated that study on a larger, but automatically annotated corpus. This automatic annotation resulted in a poor quality of data – after evaluating the coordinations extracted for analysis, they found only 50.1% of their sample to be correctly extracted. This study attempts the replication again, with the same larger corpus but aiming to improve the quality of the automatic annotation by using a different dependency annotation scheme – Surface-syntactic Universal Dependencies instead of Universal Dependencies. The following sections describe those two projects and present the reasoning behind choosing one of the schemes over the other.

2.5 Universal Dependencies

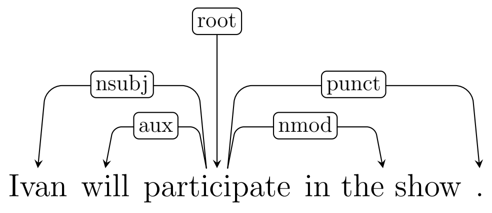
As seen in Section 2.3, there are many ideas on the dependency structure of coordination. Not only coordinations, but whole sentences can have different structures depending on the chosen dependency approach. One of those approaches is presented in this section.

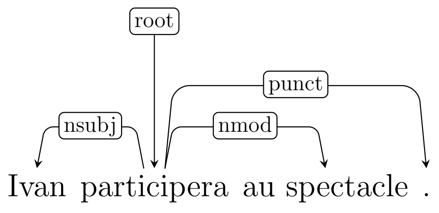
Universal Dependencies (UD, henceforth) is a project focused on formulating guidelines for creating dependency annotation, that would suit as many languages as possible, while maintaining the possibility to represent phenomena specific to any given language. The guidelines give instructions about word segmentation, part-of-speech tagging, assigning morphological features and creating an appropriate dependency tree for a sentence.

Here, the most relevant part of the project are the rules for creating a dependency tree. In the first version of UD (Nivre et al., 2016), three of them were specified:

1. dependency relations appear between content words,
2. function words are attached to the content words which they describe,
3. punctuation marks are attached to the head of the phrase or clause in which they appear.

Content words can otherwise be called “lexical” or “semantic” centres, which means their main purpose is to carry meaning in the sentence, whereas function words serve mostly a syntactic purpose. The difference between the two is visible in sentence (18), where the word *participate* carries the meaning, therefore it is the content word and the root of the sentence. The word *will* is a function word and is attached to the content word.

(18)


(19)


The reasoning behind setting those criteria is that it increases the chance of finding similar tree structures in different languages, for example when comparing sentences between English and French, which is morphologically richer. (19) is a tree for the French translation of the sentence in (18). Even though the French sentence does not have an auxiliary word to mark the future tense, the structures of those sentences are almost identical.

2.6 Surface-syntactic Universal Dependencies

Surface-syntactic Universal Dependencies (SUD, henceforth) is another example of a project aiming to create a set of universal guidelines for dependency annotation. Gerdes et al. (2018) describe it as “near-isomorphic to UD” and propose a set of conversion rules between the schemes. Most of the SUD annotation guidelines are the same as in UD, the most prominent difference is the change in choosing dependency heads, from content words to function words. This section covers the relevant differences between the schemes and how those are beneficial for research described in this work.

2.6.1 Criteria for choosing heads of dependencies

The UD dependency trees have heads of their dependencies chosen based on the distinction between content words and function words, whereas in SUD, heads of dependencies are chosen based on the distributional criteria. Gerdes et al. (2018) say that "the surface syntactic head determines the distribution of the unit", so the head of a dependency behaves in sentences similarly to the way the whole unit does, the unit meaning the head and the dependant with all of its dependencies, if there are any. The authors also describe how to test which word is the head of the dependency: in different sentences that word can be replaced by the unit and *vice versa*. The example sentence the authors use to explain this is *The little boy talked to Mary*. There is a dependency between words *little* and *boy*, and sentences in (20) and (21) show why the head of this dependency is the word *boy*. In (20a) the word *boy* can be replaced by the unit *little boy*, as shown in (20b), and the sentence is still grammatical.

(20) a. I saw a **boy**.
b. I saw a **little boy**.

Similar replacement cannot be done with the word *little*. Trying to replace that word in (21a) with the whole unit results in an ungrammatical sentence in (21b). Sentences (21c–d) show that replacement in the other way is not possible either, therefore the word *little* cannot be the head of this dependency.

(21) a. The boy was **little**.
b. * The boy was **little boy**.
c. I found the **little boy**.
d. * I found the **little**.

It is not always possible to test both of the words within a dependency, but in such cases showing that one of the words does not commute with the whole unit is enough to decide it is not the head, therefore the other one must be. As shown in Gerdes et al. (2018), that is exactly the case with the words *to Mary* – it is impossible to see how the word *to* behaves on its own, since it needs a noun or a verb. However, the sentences in (22) show that *Mary* does not have the same distribution as those two words together – *Mary* in (22a) cannot be replaced by *to Mary*, as (22b) shows, and *to Mary* in (22c) cannot be replaced by *Mary*, as (22d) shows. This is enough to choose the word *to* as the head of this dependency.

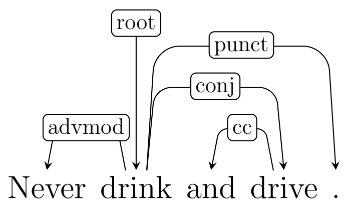
(22) a. I saw **Mary**.
b. * I saw **to Mary**.

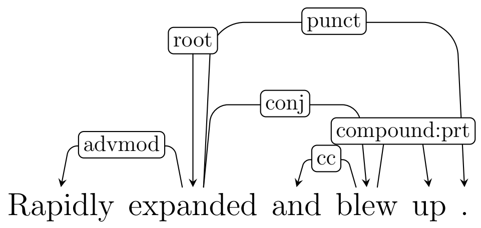
c. I talked **to Mary**.

d. * I talked **Mary**.

Because of this difference between UD and SUD in how dependency heads are chosen, SUD trees reflect more of the syntax of a sentence than the UD trees. This affects which words are found to be inluded in a coordination and consquently what are the length differences between conjuncts, which is an integral part of this work.

As Przepiórkowski and Woźniak (2023) mention, the UD scheme is not ideal for coordination analysis, as it is not clear which dependencies are shared by the conjuncts and which are private. This is illustrated by the sentences in (23) and (24).

(23)


(24)


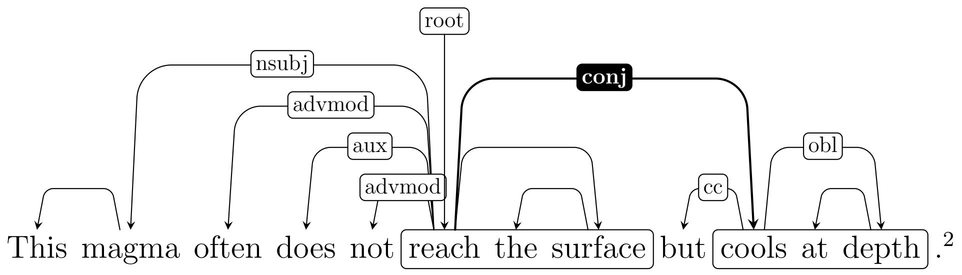
In (23) the coordinated words are *drink* and *drive*, the word *never* is attached to the first conjunct and even though English speakers reading this sentence know that it applies to the whole coordination, it is not obvious from the structure of the sentence. The structure in (24) is similar, but this time the word *rapidly* applies only to the first conjunct. In both of those sentences knowledge of the real world is required to accurately determine whether the dependency is shared by the whole coordination (as in (23)) or private to the first conjunct (as in (24)). Because of this ambiguity, it is difficult to construct accurate heuristics for determining the extent of each conjunct in a coordination. This issue appears in both UD and SUD, however the following examples show that some of the ambiguities present in UD can be resolved in SUD.

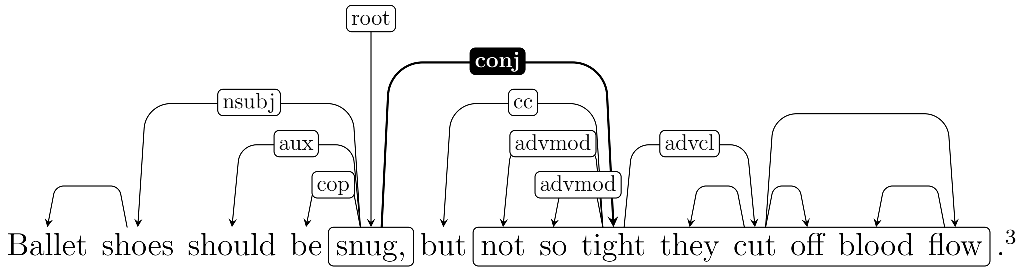
Przepiórkowski et al. (2024) describe the heuristics they used for finding the extent of the left conjunct in a coordination in the following way:

1. All dependents directly to the right of the conjunct head were considered private to that conjunct.
2. Similarly for compound dependents to the left.
3. If a dependency to the left of the head had any other label, it was checked whether any other conjuncts had a dependency with the same label.
   a. If so, these dependencies were considered private to the particular conjuncts.

b. If only the first conjunct had a given dependency type, it was treated as shared by all conjuncts.

Let us use an example to consider how those heuristics can be applied and where they fail – sentence *This magma often does not reach the surface but cools at depth* (with a tree shown in (25)) has a coordination with conjuncts *does not reach the surface* and *cools at depth*. The `conj` dependency connects the words *reach* and *cools*, so the word *reach* is the head of the left conjunct. It has a dependency directly on the right, which is always included in the conjunct (based on the Heuristic 1), therefore so far the text of the left conjunct is *reach the surface*. The head also has some dependencies on the left: one labelled `aux`, one labelled `nsubj` and two labelled `advmod`. According to the Heuristic 3b, all of those have to be shared by the whole coordination, meaning that they cannot be included in the left conjunct. Since those are all of the dependencies the head has, the text of the left conjunct is *reach the surface*, which is incorrect.

(25)

root
nsubj
advmod
aux
advmod
conj
cc
obl
This magma often does not reach the surface but cools at depth .[2]

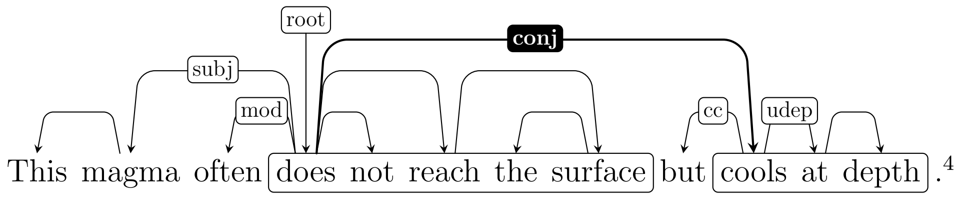
(26)

root
conj
nsubj
aux
cop
cc
advmod
advmod
advcl
Ballet shoes should be snug, but not so tight they cut off blood flow .[3]

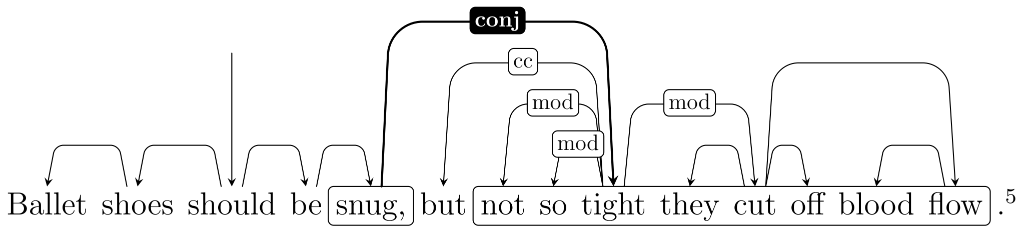
Modifying the heuristics, so that they fit this example, for instance by saying that `aux` dependencies should always be included in the left conjunct, would on the other hand mean that sentences such as in (26) would have incorrect extracted coordinations. In this example the correct coordinate structure has conjuncts *snug* and *not so tight they cut off blood flow*, but if the algorithm included the `aux` dependency in the first conjunct (as would be required in (25)), the result would be conjuncts *should be snug* and *not so tight they cut off blood flow*.

This however is not an issue when using the SUD scheme. As shown in (27), the words *does not* cannot be dependencies of the whole coordination, because the word *does* is the head of the left conjunct and the word *not* is one

[2]Sentence `w01031015` from the `UD_English-PUD` corpus (Zeman et al., 2017).
[3]Sentence `GUM_whow_ballet-14` from the `UD_English-GUM` corpus (Zeldes, 2017).

of its dependencies on the right and thus is always included in the conjunct. Changing the annotation scheme to SUD does not affect the sentence in (26) – the word *snug* has to be the whole left conjunct, because it also does not have any dependencies in this annotation scheme.

(27)

root
conj
subj
mod
cc
udep
This magma often does not reach the surface but cools at depth .[4]

(28)

conj
cc
mod
mod
mod
Ballet shoes should be snug, but not so tight they cut off blood flow .[5]

Therefore, the focus on syntax in SUD makes it a better fit for coordination analysis.

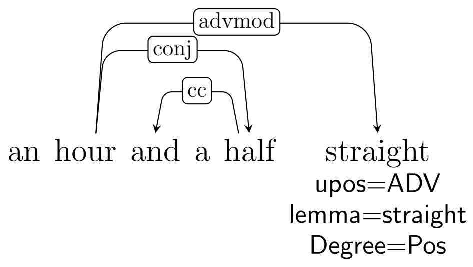
2.6.2 Explicit information about shared dependencies

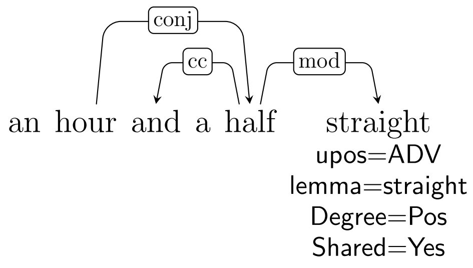
Besides the structural advantages that SUD has over UD when it comes to analysing coordination, there is one additional feature that is added in SUD treebanks that helps find the extent of conjuncts.

While UD corpora are often created specifically for the purpose of participating in the UD project or converted with manual corrections from different dependency annotations, the SUD corpora are mostly converted automatically from UD. There are a few French treebanks, as well as treebanks for Beja, Zaar, Chinese and Naija, that are natively made for SUD, but all others are converted from UD using rule-based graph transformation grammars.

UD uses the Bouquet approach to annotating coordination, while SUD uses the Chain one. This means that in the conversion process, some information about the privacy status of a dependency of the coordination can be lost. This is visible in the coordination presented in the sentence *I just sat in there for like an hour and a half straight and studied.*[6] As the UD annotation in (29) shows, the word *straight* is shared by the whole structure. This is not structurally visible in the SUD version in (30), where the `mod` dependency for the word *straight* is attached to the last conjunct.

[4]Sentence `w01031015` from the `SUD_English-PUD`.
[5]Sentence `GUM_whow_ballet-14` from the `SUD_English-GUM` corpus.
[6]Sentence `GUM_vlog_studying-27` from the GUM corpus (Zeldes, 2017).

(29) 

(30) 

So as not to lose this information while converting the annotation scheme, feature Shared=Yes is added. Similarly, in coordinations where a dependent is attached to the right conjunct in the UD scheme (therefore private to the right conjuct), during the conversion to SUD the feature Shared=No is added.

2.6.3 Learnability of dependency schemes

The current study is another replication of Przepiórkowski and Woźniak (2023). As was pointed out in Chapter 1, Przepiórkowski et al. (2024) have conducted a similar analysis on the COCA corpus annotated automatically in the UD scheme. After evaluating the automatically annotated data they found only 50.1% of the coordinations in the evaluation sample to be correctly extracted from the corpus. Reasons for such an outcome can be twofold: the issues lie either within the parsing accuracy or within the script for extracting coordinations from dependency trees.

Issues within the script have been addressed to some extent in Section 2.6.1 – heuristics for finding conjunct extents are easier to develop within a function-word focused annotation scheme. As for the parsing performance, there are studies showing that the UD scheme is harder to parse. Rehbein et al. (2017) show that choosing content words rather than function words for dependency heads increases arc direction entropy (a measure describing how consistent dependency directions in a given treebank are), which then lowers parsing accuracy. In another study, Kohita et al. (2017) converted UD trees into ones with function heads, rather than content heads. They then used the converted trees for training parsers and parsed 19 treebanks using both UD and converted models. After parsing, the results from the converted models were converted back to UD and for most of the languages (11 out of 19) those results had better scores.

The criterion for choosing dependency heads may not be the only structural advantage that SUD has over UD in terms of parsing. As Gerdes et al. (2018) demonstrate, the Chain approach to annotating coordination, that is used in SUD, minimises the dependency lengths compared to the Bouquet approach used in UD. This may be beneficial for parsing accuracy, as parsers tend to perform better when working with shorter dependencies (Nilsson et al., 2006; Eisner and Smith, 2005).

In the studies cited above the comparison was between UD and a scheme that differed from UD only in some particular aspect, not a new, comprehensive scheme. Tuora et al. (2021), however, compared UD to SUD, which matters because, as they say, "any realistic annotation schema which employs a more 'syntactic' approach to headedness than UD will also differ from UD in the repertoire and distribution of dependency labels, and will also take into account the intrinsic linguistic interaction between various constructions". They trained five parsers, two of which were transition-based and three graph-based, using 21 corpora representing 18 languages. While transition-based parsers seemed to perform similarly on both annotation schemes, the graph-based ones preferred SUD. As for attachment scores for the English corpus tested in this experiment (GUM), all of the parsers scored higher with the SUD annotation. The parser utilised in the current study, Stanza, is graph-based and the language of the texts it annotates is English, therefore SUD might be the better choice for the annotation scheme for this data.

Chapter 3

Data processing

The data used in this work is based on the Corpus of Contemporary American English. The corpus consists of raw texts collected in a span of 30 years (1990 – 2019) representing 8 styles: academic, fiction, newspapers, magazines, TV/movies, websites, blogs and spoken data. For the analysis of coordinations to be possible, first the texts have to be annotated syntactically – here the Stanza parser (Qi et al., 2020) was chosen for this task. The first subsection of this chapter describes how the parser works and how it was trained for annotation. The second subsection describes the procedure of finding coordinate structures in parsed sentences and creating tables with data ready for analysis.

3.1 Parser training

Stanza is a Python package intended for natural language analysis, which contains multiple processors responsible for different steps of said analysis, e.g. tokenisation, lemmatisation, part-of-speech tagging, dependency parsing, sentiment analysis. All of the processors are neural networks, which together are put into a pipeline that takes raw text as input and returns documents with parsed sentences as output. The default parsing model provided by Stanza for English annotates according to the UD scheme, therefore two processors – part-of-speech tagger and dependency parser – had to be trained to use the SUD scheme.

The dependency parser creates the dependency trees that can later be searched for coordinations. The part-of-speech tagger assigns the part of speech as well as the features appropriate for each word in the input. The parts of speech used in SUD are the same as in UD, but the features may include additional information about shared dependencies, as was explained in Section 2.6.2. If the neural network is trained on data containing this information, it can than be able to determine which of the dependencies are shared and which are private to specific conjuncts.

Dependency trees as shown in previous chapters, though possibly comprehensible for people, are not written in a way that is easily understandable by computer programs, including parsers. Buchholz and Marsi (2006) created the CoNLL-X format, which was first intended for the comparison of parser outputs in a dependency parsing shared task. Today this format is widely used for representing dependency trees in a plain-text form. The UD project adapted the format to their needs by replacing some of the information included in CoNLL-X and thus creating the CoNLL-U format, now also used for SUD data. Appendix A shows an example of an SUD dependency tree presented as a tree and in the CoNLL-U format.

Training was conducted using the scripts made available by the Stanza developers.[1] The models were trained on the English SUD corpora, which were created by converting the UD corpora into SUD using a set of graph conversion rules developed by the authors of SUD (Gerdes et al., 2018).[2] If a corpus is large enough, it is split into three parts: training, development and testing. The training set is used to expose the parser to the correct dependency trees and based on that a prediction model is created. The model is tuned using the development set – the model tries to predict what is the correct dependency tree for a sentence and the prediction is then confronted with the data in the set. Adjustments are made until there is no gain in the scores acheived by the model. The testing set is used later to evaluate the performance of the model – model creates dependency trees for the sentences in the testing set and those trees are compared to the original, manually annotated ones. This way the evaluation is more accurate, because the model has not been trained on this set of trees therefore it could not learn what the dependency trees for those specific sentences should look like.

Training a model requires word vector data (which is provided with the default model for English) and a prepared treebank – this means that all of the possible annotations from a treebank have to be listed for the prediction model to choose from. After all the needed files were provided, the training script was run. The batch size was set to 1000 and the dropout rate was 0.33. This means that the whole training dataset was split into batches, each with 1000 elements, which were then given to the model to assign weigths to different possible parses of a sentence. Dropout rate is the proportion of nodes in the neural network that are dropped during training. Without any dropout, a model might become overfitted for the training data. This means that it will

[1]https://github.com/stanfordnlp/stanza-train

[2]https://github.com/surfacesyntacticud/tools/blob/v2.12/converter/grs/UD$_t$o$_S$UD.grs

be very good at predicting annotations for the sentences it has already seen, but not so much with any other data. The dropout rate chosen for training here was recommended in the training documentation, the batch size was dictated by the hardware limitations.

The following subsections describe the corpora used to train the models for this study.

3.1.1 Combined model

The combined model was used to annotate most of the data analysed in this study. It was trained on the combined training sets from the EWT, GUM and ParTUT corpora, all available converted to the SUD annotation scheme.[3] A corresponding model was also trained for the UD scheme to compare the performance on those two schemes.

EWT is the English Web Treebank. The data was collected between the years 1999 and 2011 and comes from 5 primary sources: weblogs, newsgroups, emails, reviews and question-answers. In total, the corpus contains 254,820 words, which makes it the biggest available SUD corpus for English. The texts originally had constituency annotation, which was then automatically converted into Stanford Dependencies and then manually corrected to UD.

The second corpus used for this model was GUM – the Georgetown University Multilayer corpus. The early versions of GUM were annotated according to the Stanford Dependencies scheme, later they were manually converted into UD and the subsequent additions to the corpus have been annotated natively using UD. The corpus contains 228,399 tokens and is made up of a variety of styles, for instance academic, interviews, travel guides, letters, how-to guides and forum discussions.[4]

The third corpus used for training the combined model was one based on the English part of ParTUT, the multilingual parallel treebank from the University of Turin. It consists of legal texts, Wikipedia articles and transcriptions of TED Talks. It was originally manually annotated in a style specific to the treebanks developed at the University of Turin, then converted to UD. The corpus has 49,602 tokens, which is a lot less compared to the corpora described earlier, but is still a significant contribution to the model.

Corpora listed above were chosen because of their size and the consistency of annotation. Some corpora, despite the style diversity they could provide for

[3]https://surfacesyntacticud.github.io/data/

[4]The GUMReddit corpus, which contains the forum discussions, was here included in the whole GUM corpus. Before training any models it is required to run a script that recovers the textual data that is by default not included. The script is available in the GUM corpus repository: https://github.com/amir-zeldes/gum/blob/master/get$_t$ext.py.

the parsing model, had to be excluded from training, because some information was missing or annotated inconsistently with the other corpora used here.

3.1.2 Spoken model

Spoken and written language differ significantly, therefore a model trained mainly on one type of data can perform poorly when presented with the other type. Most of the corpus data comes from written text, as it is easier to obtain. This experiment involves training a model specialising in spoken data to avoid the poor quality resulting from an ill-fitted model.

name	no. of tokens	source texts	annotation style
combined model			
EWT	251 492	weblogs, newsgroups, emails, reviews and question-answers	constituency, then converted to Stanford Dependencies, then to UD
GUM	228 399	academic, Wikipedia articles, vlogs, conversations, courtroom transcripts, essays, fiction, forum, how-to guides, interviews, letters, news stories, podcasts, political speeches, textbooks, travel guides	Stanford Dependencies, then converted to UD
ParTUT	49 602	legal texts, Wikipedia articles, public talk transcripts	own annotation style, then converted to UD
total	529 493		
spoken model			
Atis	61 879	transcriptions of questions about flight information	natively UD
GUM (parts)	51 451	interviews, conversations, vlogs	Stanford Dependencies, then converted to UD
total	113 330		

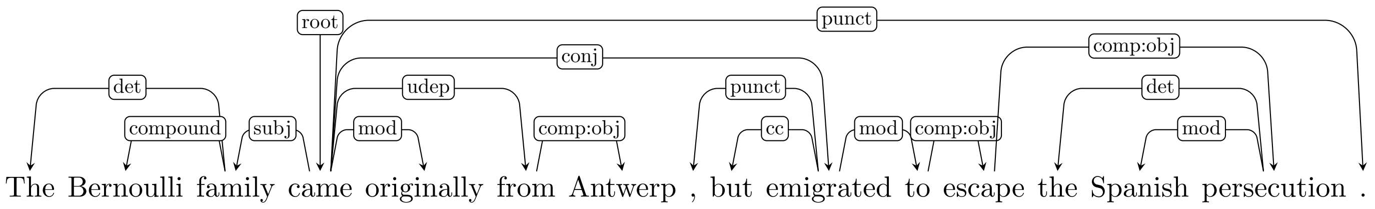
Table 3.1: Summary of the information about corpora used to train models

The corpora used for this model were parts of the GUM corpus, specifically those with interviews, conversations and vlogs (51,451 tokens) and the Atis corpus. Atis comprises sentences from the Airline Travel Informations dataset, which come from transcriptions of people asking automated inquiry systems for flight information. The corpus has 61,879 tokens and was natively annotated in UD.

3.2 Data extraction

The scripts used for parsing and extracting data are available in a github repository.[5] The extraction process will be illustrated by the sentence *The Bernoulli family came originally from Antwerp, but emigrated to escape the Spanish persecution.*[6]

Texts from the COCA corpus are first split into sentences using the Trankit parser (Nguyen et al., 2021), which deals with the task more accurately than Stanza. Those sentences are then put into the Stanza's parsing pipeline: first the sentences are tokenised, then lemmas of all of the words in the sentence are found, parts of speech and morphological features are assigned and finally the dependency trees for all of the sentences are created. After running through the pipeline, the example sentence has a dependency tree shown in (31).

(31)

root
punct
comp:obj
conj
det
udep
punct
det
compound
subj
mod
comp:obj
cc
mod
comp:obj
mod
The Bernoulli family came originally from Antwerp , but emigrated to escape the Spanish persecution .

Every dependency tree is then searched for coordinations, which are marked by the dependency label `conj`. In (31) there is one `conj` dependency that connects the words *came* and *emigrated*. If such a dependency is found, the algorithm looks for every conjunct within that coordinatsion and checks whether there are any other coordinations embedded inside of the one already found – if there are any, they are separated and analysed later. After all coordinations in a document are found, the algorithm searches for all information necessary for later analysis: the conjunction, heads of conjuncts, the exact text and length of the left and right conjunct, the governor position and additional information about parts of speech and morphological features of all of the elements of the coordination. In the example (31) the heads of the left and right conjuncts are the words *came* and *emigrated* respectively. The word *do* has no head in this sentence, therefore there is no governor of the coordination. If the head of the right conjunct has a `cc` dependency, that dependency is the conjunction of the coordination – in (31) this is the word *but*.

Then the algorithm looks for the text of the right conjunct – it does not add to the conjunct the dependency labelled `cc`, because this is a seperate element of the coordination. It does not add the `punct` dependency either, if

[5]Scripts used here (available at https://github.com/bmagdab/sud-coords) were based on those used by Przepiórkowski et al. (2024) (available at https://github.com/bmagdab/LGPB23-24).

[6]Modified version of the sentence `GUM_bio_bernoulli-9` from the GUM corpus (Zeldes, 2017)

it appears at the beginning of the conjunct, because a conjunct has to start with a word. The only other dependency that the word *emigrated* has is `mod`. Since it appears after the head of the right conjunct, it can be either private to the conjunct or shared by the whole coordination. The heuristic applied here after Przepiórkowski et al. (2024) is that if any of the other conjuncts have a dependency like this, this dependency is private. Otherwise it is shared by the whole coordination. In this case the only other conjunct is headed by the word *came* and it does have a `mod` dependency, therefore each head has a private `mod` dependency that is included in the appropriate conjunct. After all direct dependencies of the head are covered, the whole branches starting with those direct dependencies are added to the conjunct. This means that since the word *to* from the `mod` dependency has been added to the conjunct, this words dependencies are also added – here this is the word *escape*. This continues until there are no more nodes in the branch of the tree to add. The text of the right conjunct is found this way and it is *emigrated to escape the Spanish persecution*. The process is then repeated for the left conjunct. In (31) the word *came* has three dependencies: `subj`, `mod` and `udep`. The rule here is mirroring the one from the right conjunct – dependencies appearing on the left side of the left conjunct are private to the conjunct if any of the other conjuncts have the same dependency, otherwise they are shared by the whole coordination. The `mod` and `udep` dependencies appear after the head of the conjunct and are therefore automatically added to the conjunct. The `subj` dependency has to be checked – this time the algorithm finds that no other conjunct has the same dependency, therefore this dependency has to be shared by the whole coordination and is not included in the left conjunct. The text of the left conjunct is found to be *came originally from Antwerp*.

The last step is measuring the lengths of both conjuncts in characters, syllables and words. All of the information found during this process is put in a table similar to the Table 3.2.

governor.position	**governor.word**	**conjunction.word**	**no.conjuncts**
0		but	2
L.conjunct			
came originally from Antwerp			
L.dep.label	**L.words**	**L.syllables**	**L.chars**
root	4	9	28
R.conjunct			
emigrated to escape the Spanish persecution			
R.dep.label	**R.words**	**R.syllables**	**R.chars**
conj	6	14	43
sentence			
The Bernoulli family came originally from Antwerp, but emigrated to escape the Spanish persecution.			

Table 3.2: An example of a table with the extracted information about coordinations. Some columns are excluded for simplicity.

Chapter 4

Statistical analysis

4.1 UD and SUD model comparison

The first hypothesis this work aims to verify is that the SUD annotation scheme is better for the analysis of coordination presented in Przepiórkowski et al. (2024), because of better accuracy scores achieved by parsers trained on this scheme (according to Tuora et al. (2021)) and because the structures produced according to this scheme allow for more precise extraction of coordinations. Two evaluations were conducted to verify this.

The first evaluation concerned only the parser performance and was conducted automatically using the same Python script as the one used by Tuora et al. (2021).[1] The script compared a set of manually annotated trees to those produced by the parsing model. The manually annotated trees were taken from the testing set of each of the corpora that the parser was trained on (listed in Table 3.1). Then, each sentence in the testing set was parsed by the model and the result was compared to the original dependency tree from the corpus. Two metrics are commonly used to judge the performance in dependency parsing: unlabelled and labelled attachment score. Unlabelled attachment score, or UAS, is the percentage of words in a sentence that have a correctly predicted governor. Labelled attachment score, or LAS, is the percentage of words that have a correctly predicted governor as well as the dependency label that connects that word to its governor. Those metrics were calculated for all of the trees created by all four of the models: the combined and spoken models trained on UD and on SUD corpora. Scores for all four models are presented in Table 4.1 along with the differences between those accuracies – the significant differences (according to the McNemar's test) are in bold.

The combined UD model had significantly better scores than the combined

[1] The script used here was `conll18_ud_eval.py` and can be found in the repository https://github.com/ryszardtuora/ud_vs_sud.

combined model						spoken model					
UAS			LAS			UAS			LAS		
UD	SUD	Δ	UD	SUD	Δ	UD	SUD	Δ	UD	SUD	Δ
89.75	88.95	**0.8**	87.29	86.80	**0.49**	82.56	83.44	-0.88	78.78	80.76	**-1.98**

Table 4.1: UAS and LAS for the models trained on combined UD and SUD corpora and the models trained for the spoken data on UD and SUD corpora. Significant differences between the scores are in bold.

SUD model, both UAS and LAS. As for the spoken models, the SUD model had better both UAS and LAS scores, but only the difference between the LAS scores was significant.

The second evaluation was conducted manually and concerned both the parsing process and the process of extracting coordinations, which makes it more similar to the evaluation conducted by Przepiórkowski et al. (2024). This means that the results reflected the accuracy of the whole UD-based approach (a parsing model trained on the UD copora and a script for extracting coordinations with heuristics fitted to the UD dependency trees) and the whole SUD-based approach (a parsing model trained on the SUD copora and a script for extracting coordinations with heuristics fitted to the SUD dependency trees).[2] For this evaluation coordinations were extracted from the trees in the training sets made by both the UD-trained model and the SUD-trained model. Those coordinations were then comapred between the two approaches: if the coordination had the same conjuncts according to both approaches, it was counted as extracted correctly by both. If there was any difference between the texts of conjuncts of a coordination between the two approaches, it was then manually marked which of the extracted coordinations was correct. One coordination from the manual evaluation is presented in Table 4.2 as an example.

governor.position	**L.conjunct**	**R.conjunct**	**scheme**	**correct**
L	further symphonies	other works	ud	0
L	two further symphonies	other works	sud	1

In 1874 he made a submission to the Austrian State Prize for Composition, including scores of two further symphonies and other works.

Table 4.2: Fragment of the evaluation table for a coordination found in the sentence `GUM_bio_dvorak-10` from the GUM corpus (Zeldes, 2017).

[2]Scripts for extracting coordinations are available in the following repositories: https://github.com/bmagdab/LGPB23-24 (for the UD-based approach), https://github.com/bmagdab/sud-coords (for the SUD-based approach).

In total there were 1526 coordinations found in the testing sets of which 276 required manual evaluation. Table 4.3 presents percentages of correctly extracted coordinations using both approaches. The SUD-based approach has better results, although not significantly, according to the McNemar's test.

UD	SUD	Δ
86.96%	87.48%	-0.52

Table 4.3: Percentages of coordinations extracted correctly from the testing sets, using the UD-based approach and SUD-based approach

Thus the first hypothesis was not confirmed. The evaluation of the parsing models alone is inconsistent, but shows general better performance of the UD-trained model. The manual evaluation does not show significant difference between the two approaches. However, both the SUD-trained model and the whole SUD-based approach had good enough scores to use them to analise the extracted coordinations.

4.2 Governor's impact on the coordination

The second hypothesis tested here is that placement of the shorter conjunct in a coordination is affected by the governor position and the length difference between the conjuncts. Figure 4.1 shows the observed proportions of coordinations with shorter left conjuncts depending on the length difference between the conjuncts, grouped by the possible positions of the governor. Regardless of the measure, the proportion of coordinations with shorter left conjuncts increases steadily when the governor is on the left. When there is no governor the proportion decrases at first and starts to grow around length differences equal to 30 characters or 7 words. Similarly with the governor on the right, the proportions decrease initially and rise slightly with bigger length differences, but the changes happen at different rates between the two presentes measures. The Appendix B shows the same plots, but grouped by all eight of the genres available in the COCA corpus.

Figure 4.2 shows the results of fitting logistic regression models to the observations presented in Figure 4.1. Here the slopes are more consistent between the utilised measures. When the governor is on the left, the slopes are positive, when there is no governor they are slightly negative and when the governor is on the right they are also negative, but this time much steeper.

The corresponding plots based on the data gathered using the UD-based approach are shown in Figures 4.3 and 4.4. They are similar to the ones obtained by Przepiórkowski et al. (2024), but different from Figures 4.1 and 4.2. Those differences are discussed in Chapter 5.

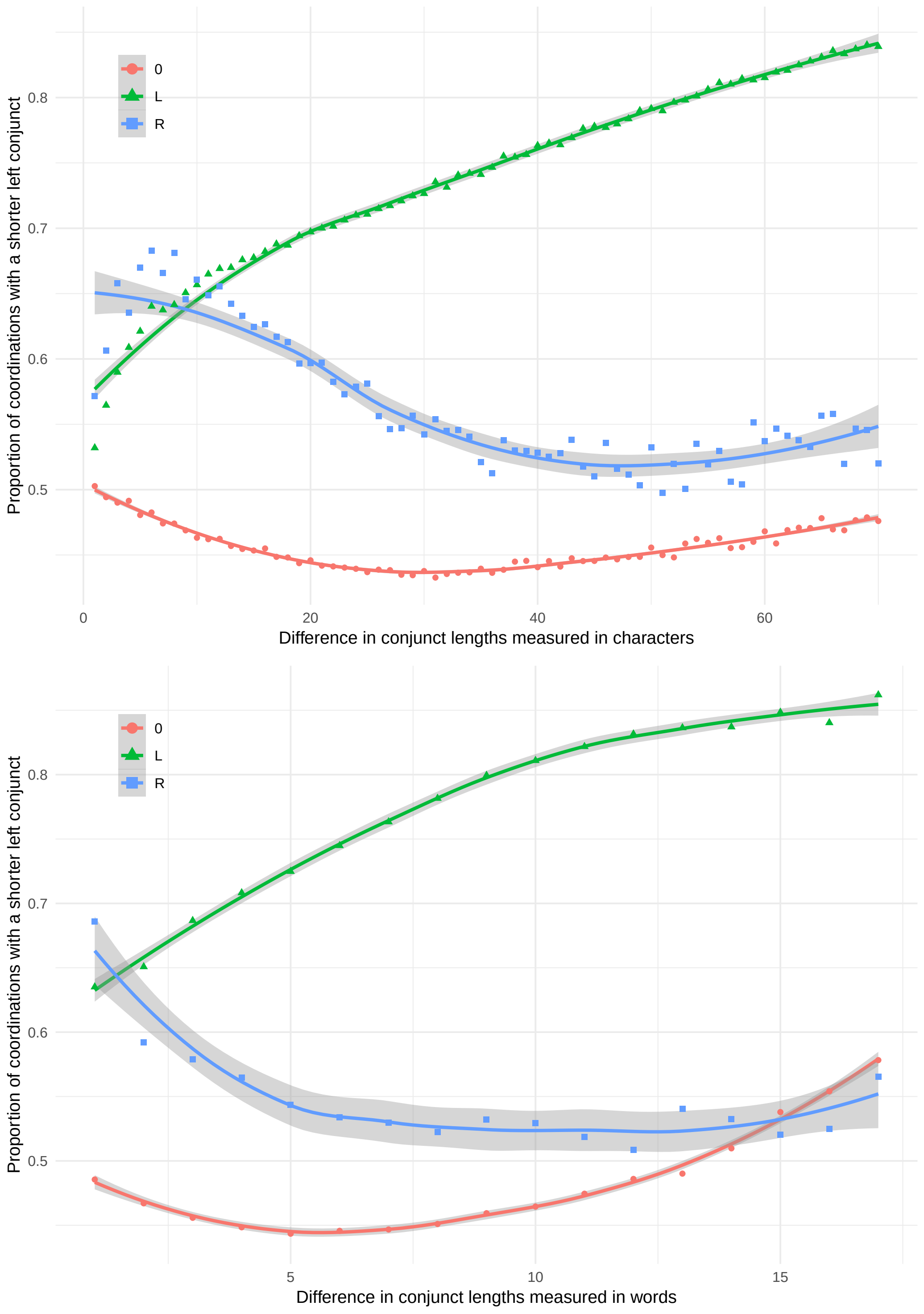


Figure 4.1: Observed and loess-smoothed proportions of coordinations with shorter left conjuncts depending on the length difference between the conjuncts and on the position of the governor, data according to the SUD-trained model

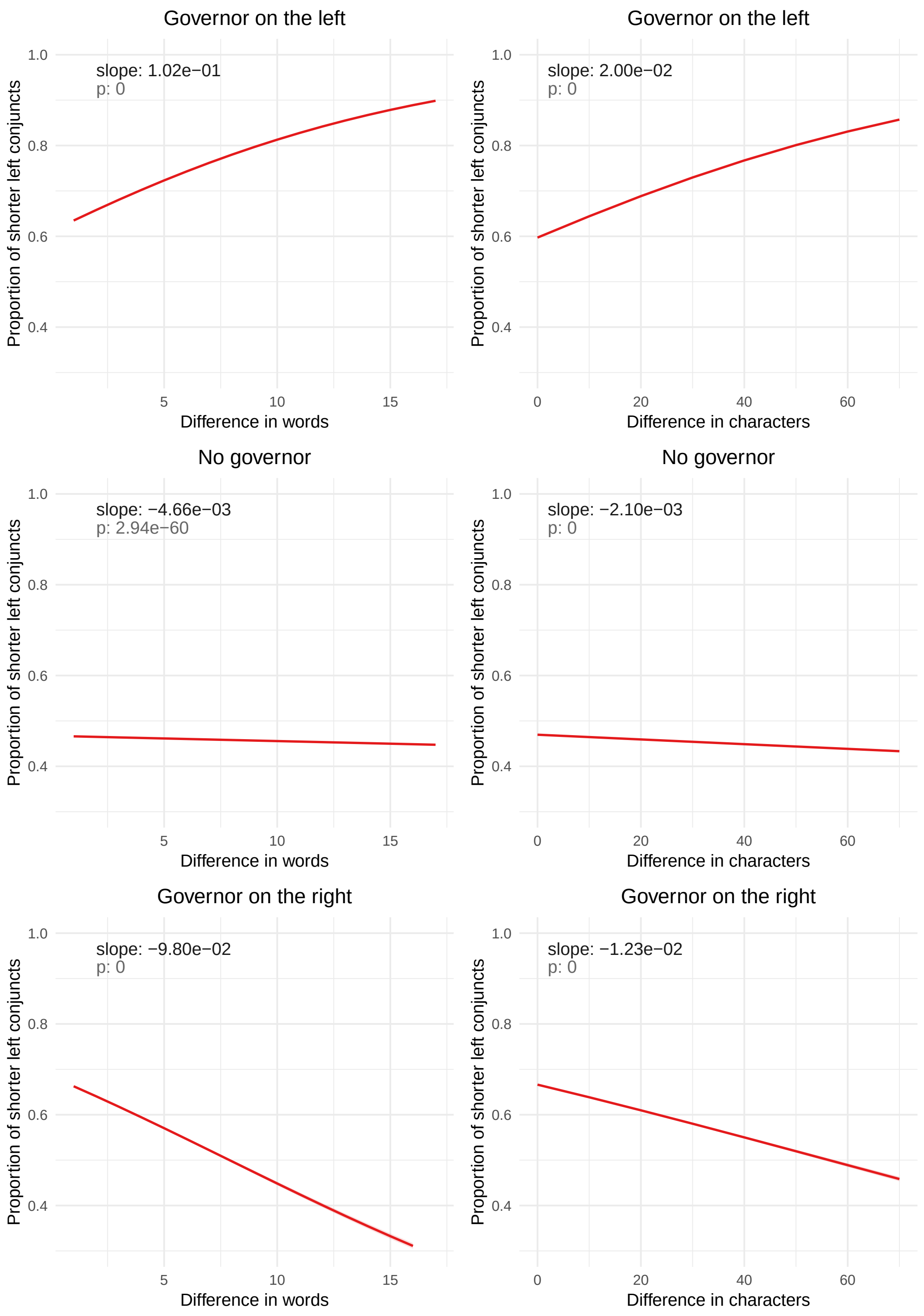


Figure 4.2: Modelled proportions of coordinations with shorter left conjunct depending on the length difference between the conjuncts, data according to the SUD-trained model

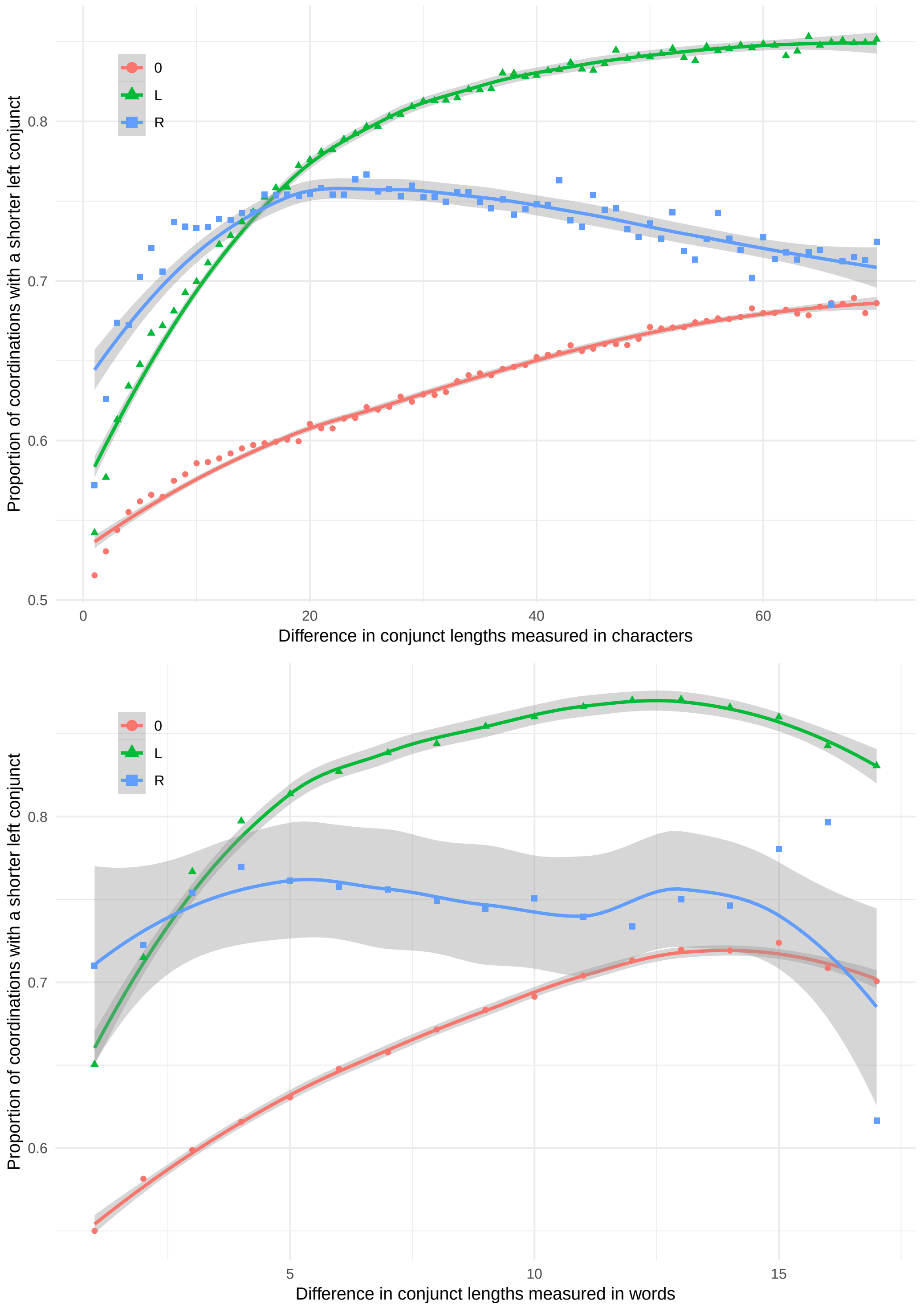


Figure 4.3: Observed and loess-smoothed proportions of coordinations with shorter left conjuncts depending on the length difference between the conjuncts and on the position of the governor, data according to the UD-trained model

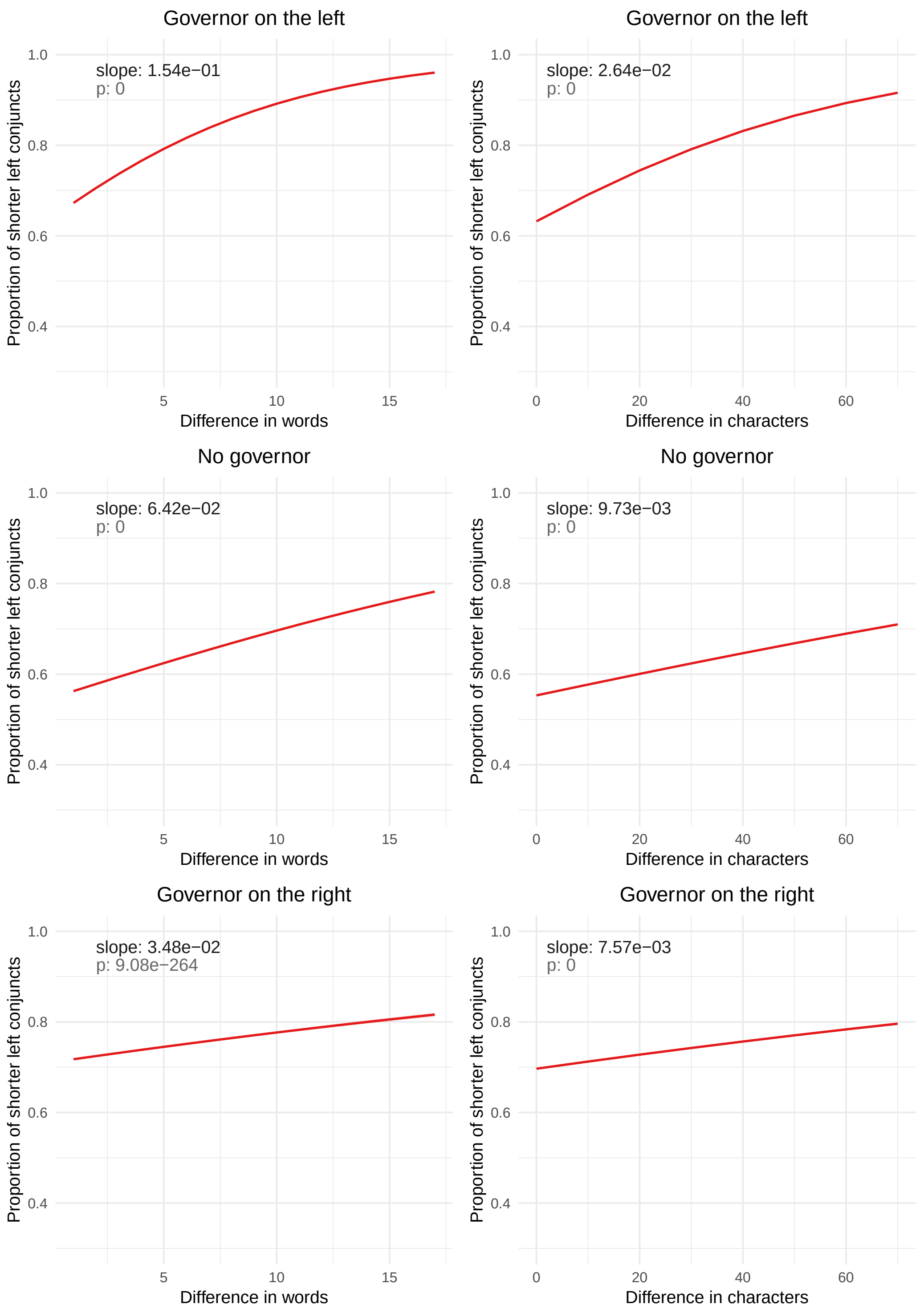


Figure 4.4: Modelled proportions of coordinations with shorter left conjunct depending on the length difference between the conjuncts, data according to the UD-trained model

Chapter 5

Discussion and limitations

5.1 Parsing models performance

The performance of the models trained here is lower than of those used by default by the parsers. This can be explained by the training parameters, specifically by the batch size, which was set to a 1000 for every training. A smaller batch size could possibly improve the performance, but this could not be tested here due to limited resources.

The combined model trained on UD data perfomed significantly better than the model trained on SUD according to both measures calculated here. This is contrary to what was expected based on the previous studies on learnability of dependency schemes. One possible explanation for this outcome is that SUD-trained models do not perform well when training on mixed domain data. Despite the poor performance of the combined SUD model, the spoken SUD model performed better (according to the LAS score) than the UD-trained one. Additionally, in the study conducted by Tuora et al. (2021) the model for parsing English was trained only on the English Web Treebank. Both of the data sources for those models are more homogenous than the data for the combined model trained for this study, which included for instance academic texts, travel guides, public talk transcripts and emails. To find if any of the annotation schemes is better for parsing diverse or more specific data, further research should be conducted.

5.2 Manual evaluation results

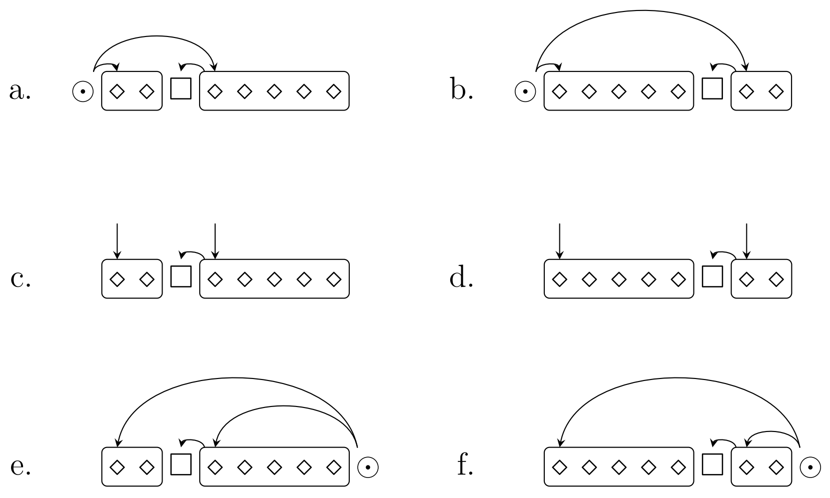
The manual evaluation showed that the SUD-based extraction of coordinations was more accurate, but not significantly. One reason for weaker performance of the SUD-based approach than expected could be the reliance on the `Shared` feature. As was presented in Section 2.6.2, some dependencies in SUD trees are marked as shared by the whole coordination by adding a `Shared=Yes` feature to their morphological features. To also add this information to the parsed trees for the COCA corpus, a part-of-speech tagger from the Stanza pipeline was trained additionally to the dependency parser. The number of words with this feature in the training datasets could be too small for the model to properly learn when this feature should be added. In the manual evaluation 13.6% of the coordinations extracted incorrectly using the SUD-based approach would be extracted correctly if the `Shared` feature was not given priority when deciding which dependents are part of the conjunct. It is however unclear how many of the coordinations would not have been extracted correctly without that feature, therefore this issue requires further investigation.

The evaluation itself has some limitations. A bigger and better balanced sample size should be evaluated – here the evaluated coordinations were those found in the testing sets of the corpora used for training. The coordinations found there did not have sufficient representation of coordinations of different length differences and different governor positions. While this evaluation gives some insight into the accuracy of the approaches, a more thorough evaluation should be conducted to investigate the cons of each approach.

5.3 Fitting dependency annotation for coordinations

Figure 5.1 shows how the proportion of coordinations with shorter left conjuncts changes with the length difference between the leftmost and rightmost

(32)



conjuncts and depending on the position of the governor.

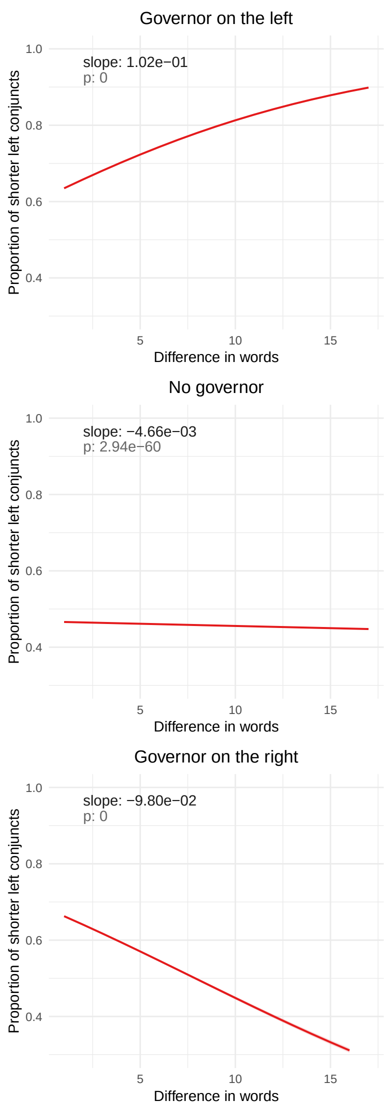


Figure 5.1: Modelled proportions of coordinations with shorter left conjunct depending on the length difference between the conjuncts, data according to the SUD-trained model with lengths measured in words.

At first glance, the results support the London or multi-headed approach to annotating coordinations. To remind what this approach looks like, diagrams (14) from Section 2.3.3 are repeated here as (32). (32a–b) demonstrate that when the governor is on the left, the depenencies are minimised when the shorter conjunct is also on the left. This is compatible with the top plot in Figure 5.1, where the bigger the difference between conjunct lenghts (and therefore the bigger the pressure of the DLM effect for ordering conjuncts in a way that minimises dependency lengths), the more coordinations have their shorter conjunct on the left. Similarly when the governor is on the right – (32e–f) show that the shorter conjunct should also be placed on the right to minimise the dependency lengths and the bottom plot in Figure 5.1 confirms that there is a tendency to do that.

The issue becomes more complex when looking at the middle plot in Figure 5.1. The slope is significantly negative, therefore the chance of no tendency being present is very low, whish is not compatible with the predictions of the London approach. According to (32c–d) there should be no tendency in any direction, as neither of the conjunct placements minimises the dependency lengths.

A look at the plots in the Appendix B, which shows the tendencies in conjunct placement in different styles, could explain why the middle plot in Figure 5.1 looks differently than expected. In most of the plots, the fluctuations in proportions of shorter left conjuncts when the coordination has no governor are small and those proportions are usually close to 0.5. The only plots that stand out in this matter are the ones for spoken data and data from TV and movies. According to Davies (2021), the data in the TV and movies section of the COCA corpus is a good representation of the spoken, informal English, because the source of this data are mostly subtitles from TV shows and movies. It has also been shown that spoken English has a smaller tendency towards minimising dependency lengths than written (Liu, 2019). If the spoken English language (represented here by the spoken and TV/movies subsets of COCA) does not minimise dependency lengths, then the results from this data cannot point to any of the annotation schemes based on the DLM effect. Appendix ref shows the modelled plots after excluding the spoken and TV/movies data – the slopes are less negative and the p-value is lower.

The annotation scheme most compatible with the results shown in Figure 5.1 is therefore the London approach. This is not however fully compatible with the results from the spoken data, which should be studied with more detail in the future. An important point here is that the study was conducted solely on the English language. Conducting similar studies on different languages could give a better insight into how coordinations are interpreted.

5.4 Difference between the data resulting from different annotation schemes

Figures 4.1 and 4.3 show the proportions of coordinations with shorter left conjuncts extracted from the same corpus, but the plots look different. The difference between the data used for those plots is the approach in which the coordinations were analised – 4.1 shows the data from the SUD-based approach and 4.3 from the UD-based approach.

Plots from Figure 4.3 resemble those from Przepiórkowski et al. (2024), because the same corpus and the same (UD-based) approach was used there.[1] Authors of the provious study suspected that the DLM effect in coordinations was grammaticalised and because of that all of the tendencies in the plots were more positive, than if the DLM effect was working only at the level of usage. If that were the case, then the grammaticalisation would be also visible in the data from the SUD-based approach. Since that is not the case, it can be argued that it was the UD annotation scheme that influenced the results. Specifically,

[1]There are some differences between those plots and three main reasons can explain them: the parsing model used here may be less accurate and therefore some of the coordinations could be extracted differently, the spoken parts of the COCA corpus were included in the current study, but not in the previous one and the current study is based on the whole COCA corpus, whereas the previous study used only parts of the corpus. As for the difference in model performance between the UD-trained model here and the default Stanza model for English used by Przepiórkowski et al. (2024), it is not possible to say which of the model is more accurate, because the accuracy scores of the combined Stanza model for English were not published.

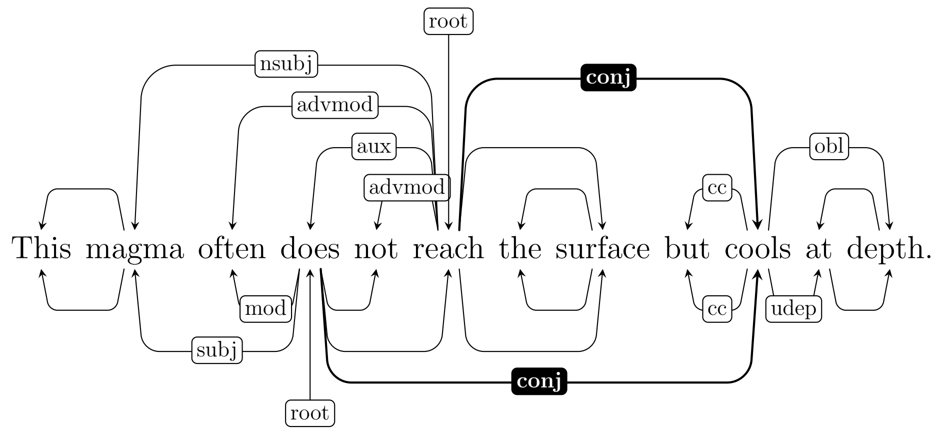
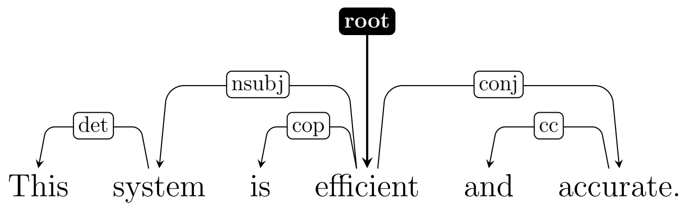
(33) 
root
nsubj
advmod
aux
advmod
conj
cc
obl
This magma often does not reach the surface but cools at depth.
mod
subj
cc
udep
conj
root

Figure 5.2: Sentence `w01031015` from the PUD corpus (Zeman et al., 2017) with a UD dependency tree above the text and a SUD dependency tree below the text.

the difference between annotation schemes that can explain the differences observed here is the one that was mentioned in Section 2.6.1. Because of the choice to have the content words as dependency heads, in some coordinations finding the extent of the left conjunct is impossible. In the troubling example (25) (repeated here as (33)) the left conjunt is cut according to the UD-based approach. If this mistake happens often enough in the corpus, it could cause the proportion of coordinations with shorter left conjuncts to be bigger than it actually is. In the plots the difference appears mainly in coordinations with smaller length differences between the conjuncts and the coordination in (33) is an example of those. Making sure that this is the cause for differences in results requires further investigation.

It is also worth noting is that the same coordination can have different governor positions in each annotation scheme. This is illustrated with a UD tree in (34) and a corresponding SUD tree in (35), where the coordination *efficient and accurate* does not have a governor according to UD, but has a governor on the left according to SUD. It requires further research how many of the coordinations in the whole dataset have ambiguous governor positions like this. The number could affect the results, especially considering that those coordinations constituted 15% of the coordinations in the manual evaulation.

(34)



(35)

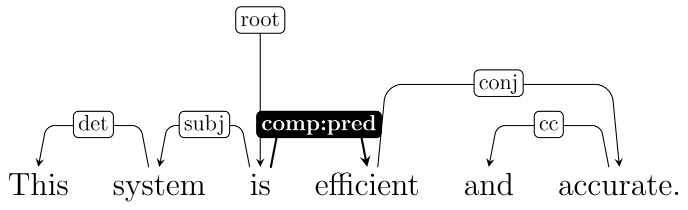


Figure 5.3: Fragment of the sentence `GUM_academic_census-25` with a UD tree in (34) and a SUD tree in (35). The governor of the coordination is marked with a dark label – in the UD tree that label is `root`, therefore the coordination does not have a governor.

This difference cannot explain why the results from the UD-based and SUD-based approach were different, but should it be mentioned because it affects one of the main explanatory variables used in the analysis of coordinations presented here and in Przepiórkowski et al. (2024). One way to avoid this issue is to analise constituency trees instead of dependency trees, as was done by Przepiórkowski and Woźniak (2023). The information about the governor position was not explicitly available in their treebank, but the rules they applied for finding the position of the governor were evaluated to be 97% accurate. This way the data is not skewed by any decisions made by the annotation scheme creators.

Chapter 6

Summary

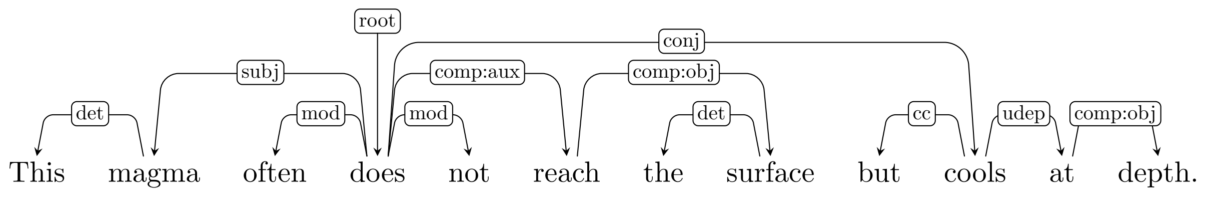
The aim of this work was to investigate the coordinate structures in English in an approach not utilised of this task before – using the Surface-syntactic Universal Dependencies annotation scheme. The choice of this approach was motivated by the goal of improving the quality of the analised data.

Performance of the parser trained on the SUD treebanks was worse than than the performance of the parser trained on the UD treebanks for most of the data. The evaluation of the whole SUD-based approach (including the process of extracting coordinations from the SUD trees) showed insignificantly better results than of the UD-based approach. The results presented here, considering the Dependency Length Minimisation effect, point towards the London style of annotating coordinations. It requires, however, taking into account that dependency lengths in the spoken language are not minimised to the same extent as in written language. It should also be considered that the dependency annotation used for syntactic analysis of the corpus influences the results obtained in the study.

Further studies can focus on possible improvements of the parsing models based on different annotation schemes and the specific effects the annotation scheme has on the analysis of coordination. As for further studies on coordinate structures, while this and previous studies on the English coordinations have been fairly consistent in arguing for the symmetric approaches, specifically the London one, the argument requires proof from other languages. Researching coordinate structures in more languages can give far more insight into how language is processed.

Appendix A

Example of a CoNLL-U representation


root
conj
subj
comp:aux
comp:obj
det
mod
mod
det
cc
udep
comp:obj
This magma often does not reach the surface but cools at depth.

```
# text = This magma often does not reach the surface but cools at depth.
 1    This     this     DET    DT   Number=Sing|PronType=Dem                                 2    det       _  _
 2    magma    magma    NOUN   NN   Number=Sing|Shared=No                                    4    subj      _  _
 3    often    often    ADV    RB   _                                                        4    mod       _  _
 4    does     do       AUX    VBZ  Mood=Ind|Number=Sing|Person=3|Tense=Pres|VerbForm=Fin    0    root      _  _
 5    not      not      PART   RB   _                                                        4    mod       _  _
 6    reach    reach    VERB   VB   VerbForm=Inf                                             4    comp:aux  _  _
 7    the      the      DET    DT   Definite=Def|PronType=Art                                8    det       _  _
 8    surface  surface  NOUN   NN   Number=Sing                                              6    comp:obj  _  _
 9    but      but      CCONJ  CC   _                                                        10   cc        _  _
 10   cools    cool     VERB   VBZ  Mood=Ind|Number=Sing|Person=3|Tense=Pres|VerbForm=Fin    4    conj      _  _
 11   at       at       ADP    IN   Shared=No                                                10   udep      _  _
 12   depth    depth    NOUN   NN   Number=Sing                                              11   comp:obj  _  _
 13   .        .        PUNCT  .    _                                                        4    punct     _  _
```

Figure A.1: A SUD dependency tree with a corresponding CoNLL-U table. Columns in the table contain the following information in this order: word index, word form or punctuation symbol, lemma of the word, UD part-of-speech, optional part-of-speech tag, morphological features, ID of the dependency head of the word, SUD dependency label, Enhanced UD dependencies (not included in SUD trees), miscellaneous information.

Appendix B

Shorter left conjuncts in coordinations grouped by genres

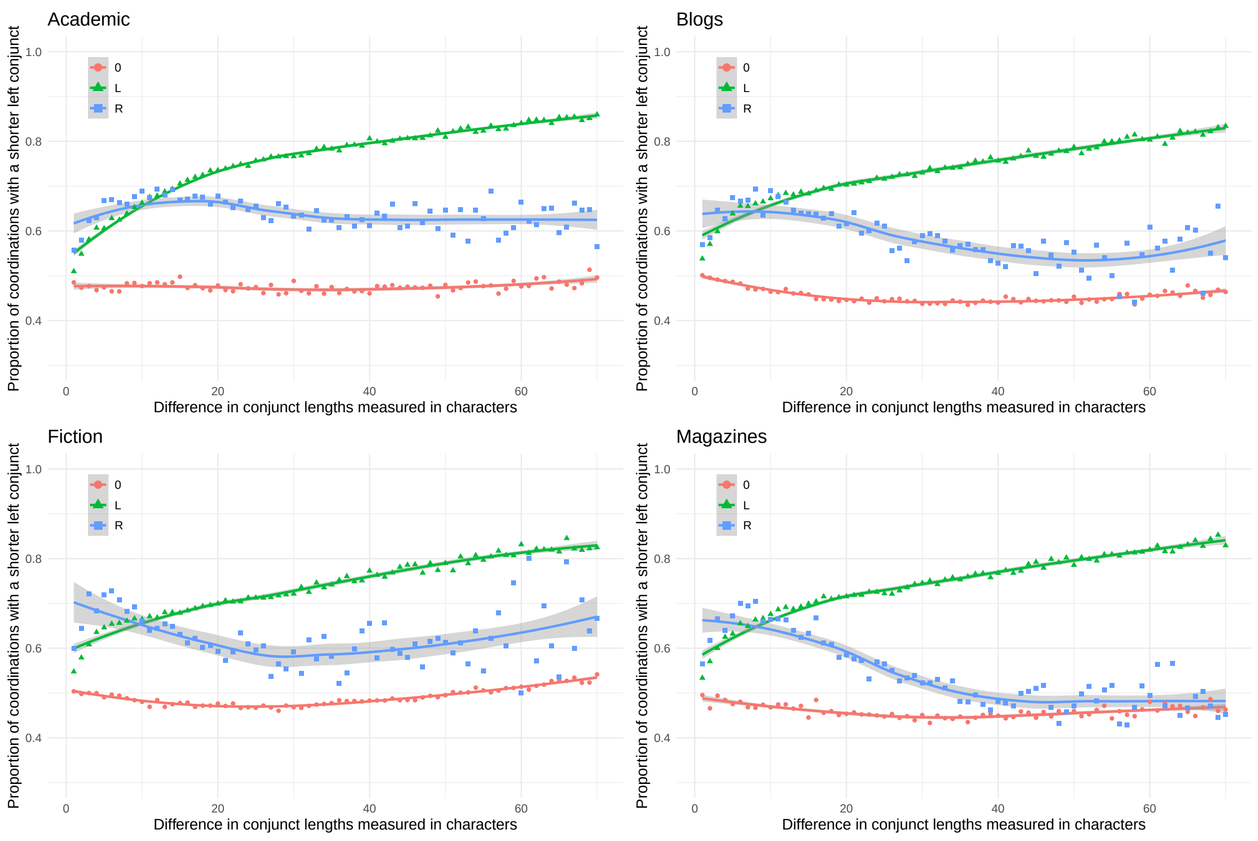


Figure B.1: Observed and loess-smoothed proportions of coordinations with shorter left conjuncts depending on the length difference between the conjuncts and on the position of the governor in the four of the styles in COCA: academic, blogs, fiction and magazine.

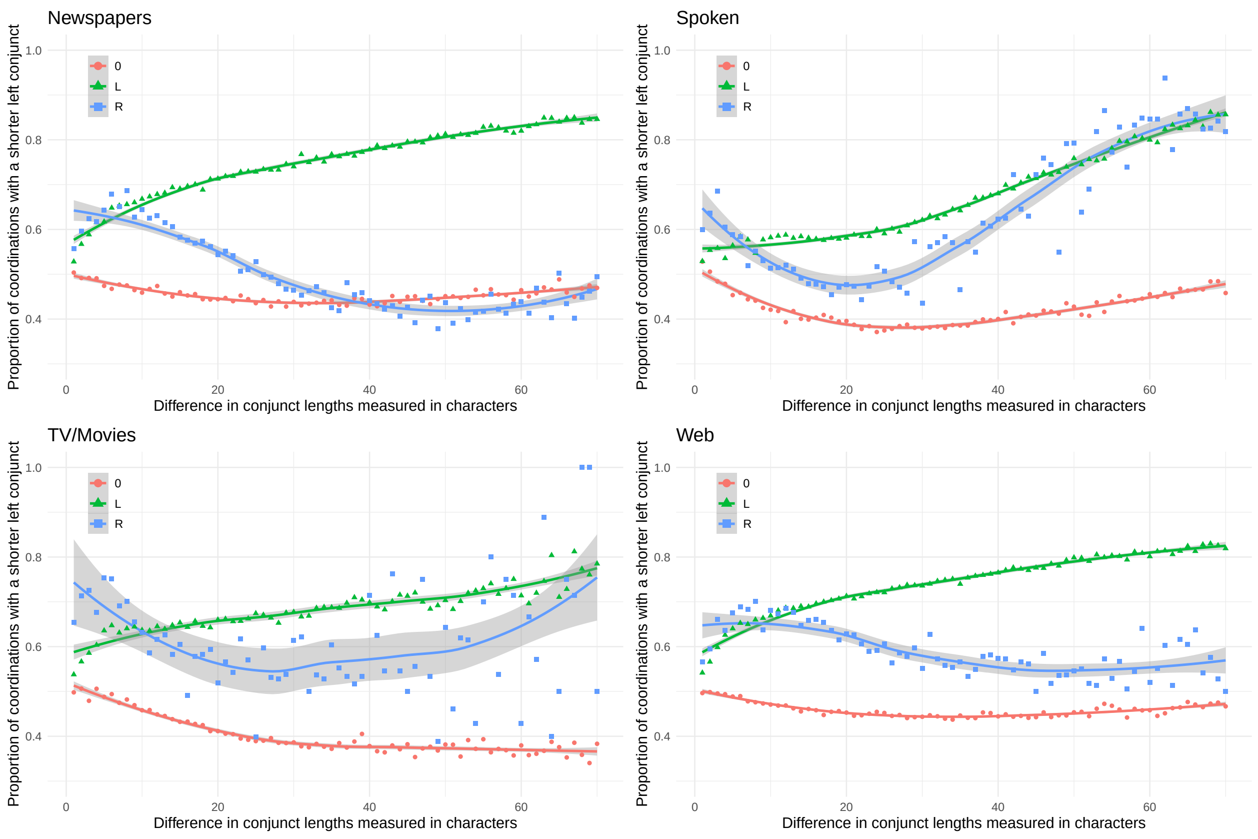


Figure B.2: Observed and loess-smoothed proportions of coordinations with shorter left conjuncts depending on the length difference between the conjuncts and on the position of the governor in the four of the styles in COCA: newspapers, spoken, TV/movies, web.

Appendix C

Logistic regression results after excluding the spoken data

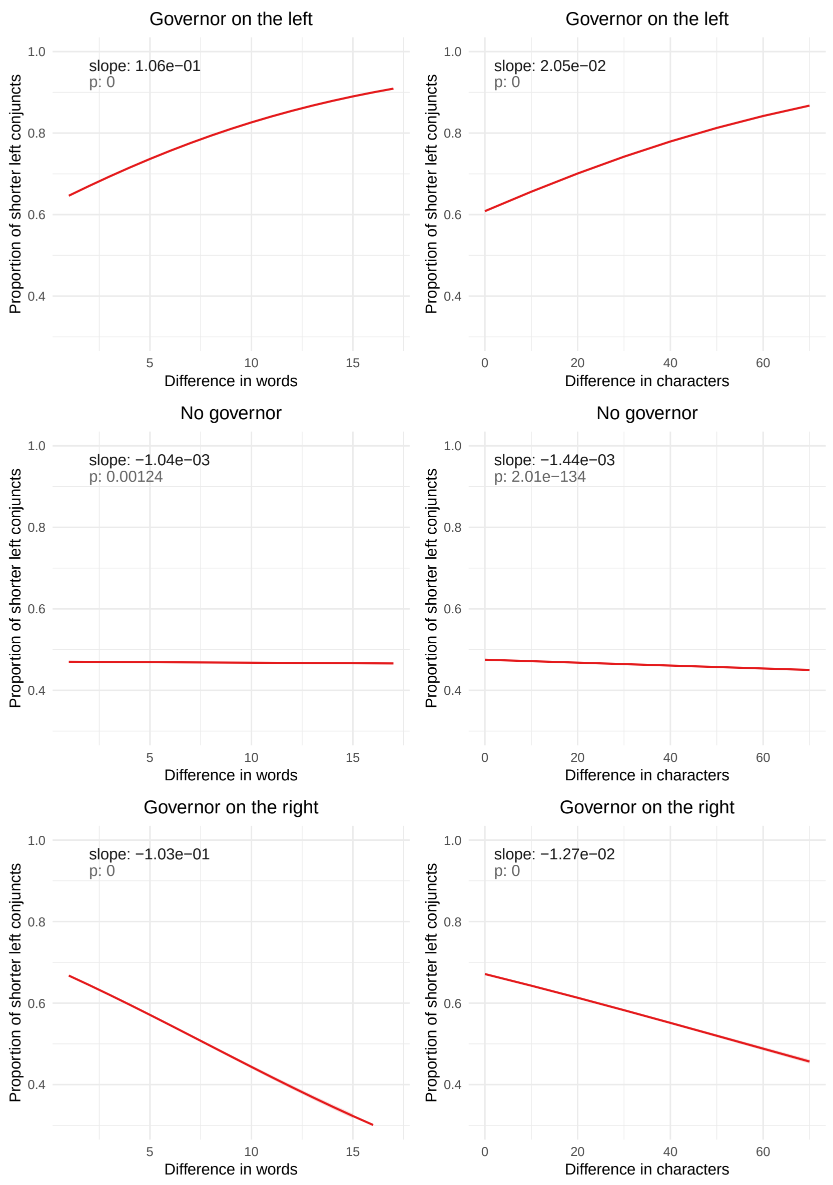


Figure C.1: Modelled proportions of coordinations with shorter left conjunct depending on the length difference between the conjuncts, data according to the SUD-trained model. Spoken data (spoken and TV/movies subsets of the COCA corpus) is excluded.

Bibliography

Buchholz, S. and Marsi, E. (2006). CoNLL-X shared task on multilingual dependency parsing. In Màrquez, L. and Klein, D., editors, *Proceedings of the Tenth Conference on Computational Natural Language Learning (CoNLL-X)*, pages 149–164, New York City. Association for Computational Linguistics.

Davies, M. (2021). The tv and movies corpora: Design, construction, and use. *International Journal of Corpus Linguistics*, 26(1):10–37.

de Marneffe, M.-C., MacCartney, B., and Manning, C. D. (2006). Generating typed dependency parses from phrase structure parses. In Calzolari, N., Choukri, K., Gangemi, A., Maegaard, B., Mariani, J., Odijk, J., and Tapias, D., editors, *Proceedings of the Fifth International Conference on Language Resources and Evaluation (LREC'06)*, Genoa, Italy. European Language Resources Association (ELRA).

Dyer, A. (2023). Revisiting dependency length and intervener complexity minimisation on a parallel corpus in 35 languages. In *Proceedings of the 5th Workshop on Research in Computational Linguistic Typology and Multilingual NLP*, pages 110–119.

Eisner, J. and Smith, N. A. (2005). Parsing with soft and hard constraints on dependency length. In Bunt, H. and Malouf, R., editors, *Proceedings of the Ninth International Workshop on Parsing Technology*, pages 30–41, Vancouver, British Columbia. Association for Computational Linguistics.

Futrell, R., Levy, R., and Gibson, E. (2020). Dependency locality as an explanatory principle for word order. *Language*, 96:371–412.

Gerdes, K., Guillaume, B., Kahane, S., and Perrier, G. (2018). SUD or Surface-Syntactic Universal Dependencies: An annotation scheme near-isomorphic to UD. In de Marneffe, M.-C., Lynn, T., and Schuster, S., editors, *Proceedings of the Second Workshop on Universal Dependencies (UDW 2018)*, pages 66–74, Brussels, Belgium. Association for Computational Linguistics.

Gibson, E. (1998). Linguistic complexity: locality of syntactic dependencies. *Cognition*, 68(1):1–76.

Gildea, D. and Temperley, D. (2007). Optimizing grammars for minimum dependency length. In Zaenen, A. and van den Bosch, A., editors, *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 184–191, Prague, Czech Republic. Association for Computational Linguistics.

Gildea, D. and Temperley, D. (2010). Do grammars minimize dependency length? *Cognitive Science*, 34(2):286–310.

Hawkins, J. A. (1994). *A Performance Theory of Order and Constituency*, volume 73 of *Cambridge Studies in Linguistics*. Cambridge University Press, Cambridge.

Huddleston, R. and Pullum, G. K. (2002). *The Cambridge Grammar of the English Language*. Cambridge University Press.

Hudson, R. (1984). *Word Grammar*. Blackwell, Oxford.

Hudson, R. (2010). *An Introduction to Word Grammar*. Cambridge Textbooks in Linguistics. Cambridge University Press.

Hunter, P. J. and Prideaux, G. D. (1983). Empirical constraints on the verb-particle construction in English. *Journal of the Atlantic Provinces Linguistic Association*.

King, J. and Just, M. A. (1991). Individual differences in syntactic processing: The role of working memory. *Journal of Memory and Language*, 30(5):580–602.

Kohita, R., Noji, H., and Matsumoto, Y. (2017). Multilingual back-and-forth conversion between content and function head for easy dependency parsing. In Lapata, M., Blunsom, P., and Koller, A., editors, *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 1–7, Valencia, Spain. Association for Computational Linguistics.

Liu, Z. (2019). A comparative corpus analysis of PP ordering in English and Chinese. In Chen, X. and Ferrer-i Cancho, R., editors, *Proceedings of the First Workshop on Quantitative Syntax (Quasy, SyntaxFest 2019)*, pages 33–45, Paris, France. Association for Computational Linguistics.

Mel'čuk, I. (1988). *Dependency Syntax: Theory and Practice.* The SUNY Press, Albany, NY.

Nguyen, M. V., Lai, V., Veyseh, A. P. B., and Nguyen, T. H. (2021). Trankit: A light-weight transformer-based toolkit for multilingual natural language processing. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations.*

Nilsson, J., Nivre, J., and Hall, J. (2006). Graph transformations in data-driven dependency parsing. In Calzolari, N., Cardie, C., and Isabelle, P., editors, *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 257–264, Sydney, Australia. Association for Computational Linguistics.

Nivre, J., de Marneffe, M.-C., Ginter, F., Goldberg, Y., Hajič, J., Manning, C. D., McDonald, R., Petrov, S., Pyysalo, S., Silveira, N., Tsarfaty, R., and Zeman, D. (2016). Universal Dependencies v1: A multilingual treebank collection. In Calzolari, N., Choukri, K., Declerck, T., Goggi, S., Grobelnik, M., Maegaard, B., Mariani, J., Mazo, H., Moreno, A., Odijk, J., and Piperidis, S., editors, *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 1659–1666, Portorož, Slovenia. European Language Resources Association (ELRA).

Popel, M., Mareček, D., Štěpánek, J., Zeman, D., and Žabokrtskỳ, Z. (2013). Coordination structures in dependency treebanks. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 517–527.

Przepiórkowski, A., Borysiak, M., and Głowacki, A. (2024). An argument for symmetric coordination from Dependency Length Minimization: A replication study. In Calzolari, N., Kan, M.-Y., Hoste, V., Lenci, A., Sakti, S., and Xue, N., editors, *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 1021–1033, Torino, Italy. ELRA and ICCL.

Przepiórkowski, A. and Woźniak, M. (2023). Conjunct lengths in English, Dependency Length Minimization, and dependency structure of coordination. In Rogers, A., Boyd-Graber, J., and Okazaki, N., editors, *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15494–15512, Toronto, Canada. Association for Computational Linguistics.

Qi, P., Zhang, Y., Zhang, Y., Bolton, J., and Manning, C. D. (2020). Stanza: A Python natural language processing toolkit for many human languages. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*.

Rehbein, I., Steen, J., Do, B.-N., and Frank, A. (2017). Universal Dependencies are hard to parse – or are they? In Montemagni, S. and Nivre, J., editors, *Proceedings of the Fourth International Conference on Dependency Linguistics (Depling 2017)*, pages 218–228, Pisa, Italy. Linköping University Electronic Press.

Tesnière, L. (1959). *Elements of Structural Syntax*. Klincksieck, Paris.

Tesnière, L. (2015). *Elements of Structural Syntax*. John Benjamins, Amsterdam.

Tuora, R., Przepiórkowski, A., and Leczkowski, A. (2021). Comparing learnability of two dependency schemes: 'semantic' (UD) and 'syntactic' (SUD). In *Findings of the Association for Computational Linguistics: EMNLP 2021*. Association for Computational Linguistics.

Wasow, T. (2002). *Postverbal Behavior*. CSLI Publications, Stanford.

Zeldes, A. (2017). The GUM corpus: Creating multilayer resources in the classroom. *Language Resources and Evaluation*, 51(3):581–612.

Zeman, D., Popel, M., Straka, M., Hajic, J., Nivre, J., Ginter, F., Luotolahti, J., Pyysalo, S., Petrov, S., Potthast, M., Tyers, F., Badmaeva, E., Gokirmak, M., Nedoluzhko, A., Cinkova, S., Hajic jr., J., Hlavacova, J., Kettnerová, V., Uresova, Z., Kanerva, J., Ojala, S., Missilä, A., Manning, C. D., Schuster, S., Reddy, S., Taji, D., Habash, N., Leung, H., de Marneffe, M.-C., Sanguinetti, M., Simi, M., Kanayama, H., dePaiva, V., Droganova, K., Martı́nez Alonso, H., Çöltekin, c., Sulubacak, U., Uszkoreit, H., Macketanz, V., Burchardt, A., Harris, K., Marheinecke, K., Rehm, G., Kayadelen, T., Attia, M., Elkahky, A., Yu, Z., Pitler, E., Lertpradit, S., Mandl, M., Kirchner, J., Alcalde, H. F., Strnadová, J., Banerjee, E., Manurung, R., Stella, A., Shimada, A., Kwak, S., Mendonca, G., Lando, T., Nitisaroj, R., and Li, J. (2017). Conll 2017 shared task: Multilingual parsing from raw text to universal dependencies. In *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 1–19, Vancouver, Canada. Association for Computational Linguistics.