Doctoral Dissertation

Neural Network Approaches to Coordination Disambiguation

Hiroki Teranishi

Program of Information Science and Engineering
Graduate School of Science and Technology
Nara Institute of Science and Technology

Supervisor: Prof. Taro Watanabe
Computational Linguistics Laboratory
(Division of Information Science)

Submitted on September 16, 2020

A Doctoral Dissertation
submitted to Graduate School of Science and Technology,
Nara Institute of Science and Technology
in partial fulfillment of the requirements for the degree of
Doctor of ENGINEERING

Hiroki Teranishi

Neural Network Approaches to Coordination Disambiguation[*]

Hiroki Teranishi

Abstract

Coordination is a syntactic phenomenon in which two or more elements, known as *conjuncts*, are linked together typically by a coordinating conjunction. Coordinate structures frequently occur in natural language and a major source of ambiguities that cannot be resolved easily even by humans. Coordination has challenged linguists for decades because it imposes numerous exceptions on theories of syntax, such as phrase structure grammars. Due to its intractability, many natural language processing (NLP) applications, such as syntactic parsing, named entity recognition, and machine translation, suffer from the presence of coordination. Recent advances in deep artificial neural networks have greatly improved the performance for many NLP tasks. However, coordination is a still difficult aspect to take into account.

In this dissertation, I focus on identifying the conjuncts of coordinate structures, especially in English text. I mainly propose two methods for this task: (1) a top-down approach and (2) a bottom-up approach, both of which utilize deep neural networks.

The top-down approach first identifies a coordinate structure and then retrieves the individual conjuncts from it. In this approach, the *similarity* and *replaceability* properties of conjuncts are exploited. For instance, *"The company provides [language instruction] and [translation services] in 25 countries"* has symmetry between the two conjuncts, and it is synonymous to say *"The company provides [translation services] and [language instruction] in 25 countries."* The

[*]Doctoral Dissertation, Division of Information Science, Graduate School of Science and Technology, Nara Institute of Science and Technology, September 16, 2020.

proposed neural networks incorporate these characteristics of conjuncts as features without external thesauri, language models, or syntactic parsers, which have been used in most previous studies. Because the proposed model is lightweight, the system can examine all possible coordination spans. Although this approach outperforms existing methods, analyses reveal that the system is poor at finding conjuncts in coordination.

In contrast, the bottom-up approach first finds individual conjuncts and then constructs a coordinate structure from them. In this approach, coordinate structures for a given sentence are identified in the form of a syntactic tree, which is produced by a context-free grammar designed for recognizing coordination. This ensures that any two coordinate structures are disjoint or nested, and therefore they never conflict with each other. The proposed neural network model consists of submodels, each of which is specialized in capturing different parts of coordinate structures. Using the submodels in the Cocke–Kasami–Younger algorithm, the system efficiently produces coordinate structures as a tree with great accuracy.

The main contribution of this dissertation is the demonstration of effective frameworks that combine neural networks with algorithms for coordination disambiguation. Experimental results show that the proposed methods achieve state-of-the-art performances with no dependence on external resources, ensuring that the global structure of coordination is consistent.

Keywords:

coordinate structure analysis, conjuncts, neural networks, syntactic parsing, CKY algorithm

Acknowledgements

This dissertation would not have been possible without the help of my advisers, other collaborators, and the members of the NAIST Natural Language Processing lab. First, I am grateful to my thesis supervisor, Prof. Taro Watanabe. He always gave me his best support even though he had recently been appointed professorship at the graduate school. The work in this dissertation was profoundly shaped by his numerous insightful comments. I wish I could have more time to work with him and learn more from him. Secondly but foremost, I have great gratitude to Dr. Yuji Matsumoto, who formerly led the computational linguistics lab at NAIST and had been my supervisor for four years. His continuous support helped me make up my mind to pursue a Ph.D. under his supervision, and my career as a researcher has developed since. I was fortunate to start my project with him, which has now become the theme of this dissertation. I greatly enjoyed our countless discussions, which later encouraged me to continue writing my dissertation. I am thankful to him for being a member of my dissertation committee. I also deeply thank Prof. Satoshi Nakamura and Assistant Prof. Hiroyuki Shindo as co-supervisors for their valuable comments and encouragement throughout my master's and doctoral courses in the graduate school at NAIST.

I would like to express my gratitude to Prof. Yoav Goldberg for accepting me as an intern and mentoring me at the BIU NLP lab. During our discussions, he patiently listened to my ideas and always gave me insightful and valuable comments. It was a great privilege to work closely with him on syntactic parsing projects. I also thank Prof. Ido Dagan and all 40+ members of the lab who I cannot list fully here but who I spent an unforgettable time with. My seven months in Israel were tough for me, but they always encouraged and helped me, which now motivates me to continue research in NLP to meet them again at a conference. Unfortunately, I had to leave Israel earlier than expected because of the COVID-19 pandemic, and I regret that I could not fully express my gratitude to Prof. Goldberg, Prof. Dagan, and every member of the lab. I look forward to seeing and collaborating with them in the future.

Being a member of the NAIST NLP lab for four and a half years has been stimulating and fun. I appreciate the uninterrupted discussions, valuable comments, and memorable activities that I shared with colleagues, alumni, faculties, and staff in the group. They taught me many things, which now form the foundation of my expertise and attitude towards research. I am very proud to have worked with them and to have been a part of the group. I am also very thankful to secretary Ms. Yuko Kitagawa for her support and encouragement of our work in the lab.

Last but not the least, I would like to thank my wife and parents for always respecting my decisions and supporting my endeavours.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Natural language is the primary means of communication between humans. Whether it is conveyed in text, speech, or even sign language, ambiguities of syntax or semantics can cause misunderstanding. People are capable of interpreting language by leveraging the knowledge, context, and background they share, even if it is ambiguous. On the contrary, computers have difficulty understanding it due to their lack of "common sense" possessed by humans. Recent advances in artificial intelligence have greatly enhanced the understanding of human languages by computers. Feeding masses of text data to artificial neural networks enables computational systems to superficially behave as if they understand the meaning and structures of languages as we do. Arbitrary measures in natural language processing (NLP) tasks reveal that computers can now achieve relatively good performance on tasks that are trivially solved by humans. On the basis of this fact, could we say that "*AI systems can now outperform humans*"?

There are some notorious linguistic phenomena that may cause confusion or ambiguity in languages. One of these is *coordination*—a syntactic phenomenon in which two or more elements, known as *conjuncts*, are linked by a coordinating conjunction, such as *and* or *or*. Coordination is observed across all languages and ages; for example, the bible says (originally in Hebrew):

> *And God said, "Let the earth bring forth the living creature after his kind, cattle, and creeping thing, and beast of the earth after his kind." And it was so. And God made the beast of the earth after his kind and cattle after their kind, and everything that creeps upon the earth*

> *to its kind. And God saw that it was good.* (Genesis 1:24-25)

In contemporary language, complex sentences involved in coordination frequently appear in corpora in any domain, e.g., the Penn Treebank [67] in the news domain contains the following:

> Toshiba's line of portables, for example, features the T-1000, which is in the same weight class <u>but</u> is much slower <u>and</u> has less memory, <u>and</u> the T-1600, which also uses a 286 microprocessor, <u>but</u> which weighs almost twice as much <u>and</u> is three times the size. (*The Wall Street Journal*)

Coordination usually cause difficulty in understanding even for native speakers of the language, which sometimes causes different interpretations among individuals. Nevertheless, we could take the time to understand the entire structure of the example by resolving each coordination one by one because we already know some characteristics of coordination. A coordinate structure, an instance of coordination, typically consists of phrases of the same syntactic category. For example,

> The company rewarded [general managers] <u>and</u> [designers].

In this sentence, the two phrases *general managers* and *designers* are conjoined by *and*. We have little difficulty identifying the conjuncts because we can find that both of them are noun phrases that denote a position or occupation. Nevertheless, why can we not simply say that *managers* and *designers* are in parallel and exclude *general* from the coordination? This is because the word *general* is unlikely to modify *designers* even though *general designers* is grammatical. Here, we exploit our knowledge of word distribution; the occurrence of *general designers* is much less frequent than that of *general managers*, and thus *general* associates only with *managers*. However, such knowledge is not always helpful to disambiguate coordination, as the following example indicates:

> The company rewarded product managers <u>and</u> designers.

We can say that *managers* and *designers* are coordinated because *product* can modify both nouns and *product designers* is likely to occur, but a coordinate structure formed by *product managers* and *designers* is also acceptable. The correct coordinated phrases depend on the context or convention, and this results in a major source of ambiguity for humans as well as NLP systems. In fact, most state-of-the-art syntactic parsers perform poorly in coordination construction because, unlike us, they do not make use of syntactic and semantic relatedness between conjuncts. AI systems seem to be far from outperforming humans in understanding complex and context-dependent structures such as coordination. Problems of treating coordination still remain in NLP research and development.

1.1 Objectives and themes

This dissertation focuses on disambiguating coordination. The research presented here is not just aimed at integrating a coordination disambiguation technique with NLP systems to improve performance. Coordination itself is interesting in computational linguistics because it induces many exceptional behaviors in natural language. Through research on coordination disambiguation, I attempt to reveal how coordination affects the structure and meaning of language and to find a way to treat irregular movements caused by coordination. The primary goal of this dissertation is the development of automatic systems capable of identifying elements conjoined by coordinating conjunctions with good accuracy. To purse this goal, I particularly focus on the unique properties and structural constraints of coordination.

Properties of coordination

Coordination has two properties with respect to conjuncts—*similiarity* and *replaceability*. Some prior methods rely only on the similarity between conjuncts when focusing on noun phrase coordination. These methods perform well on noun phrases but poorly on other types of coordination, such as verb phrases and clauses. Other methods utilize the replaceability property to tackle this issue, but their performances are very limited. The biggest drawback of these approaches is that they strongly depend on external resources, such as thesauri, language models, syntactic parsers, and co-occurrence information taken from large-scale corpora. The first attempt in this dissertation is to remove the dependence on external resources as much as possible. I propose a mechanism to incorporate the two properties as continuous feature representations computed by neural networks with minimal use of part-of-speech tags for words. I then demonstrate which information and features are crucial for the task of coordination disambiguation.

Constraints of coordination

Coordination imposes structural constraints when two of more coordinate structures appear in a sentence at the same time: they must be nested or disjoint. Many prior methods focus on resolving each instance of coordination individually, and consequently two detected neighboring coordinate structures could overlap with each other, leading to inconsistency. The production rules for coordination developed by Hara et al. [39] make it possible to identify the optimal combination of coordinate structures that are consistent in their intersection. The second attempt in this dissertation is to extend the rules of Hara et al. [39] to deal with a wider variety of coordination constructions without degrading the performance of coordination disambiguation. I propose a novel division of the task into subtasks and integrate them into a parsing algorithm with the production rules. I finally demonstrate the performance improvement achieved by the integration.

1.2 Contributions

The main contribution of this dissertation is the development of two automatic systems capable of identifying coordinate structures; one of the systems is based on a top-down algorithm and the other on a bottom-up algorithm. Specifically, their contributions can be summarized as follows:

- **Effective feature representations with minimal use of external resources**: Both proposed methods produce continuous vector representations that encode the similarity and replaceability properties between conjuncts. These representations enable the models to identify both similar pairs of conjuncts, such as noun phrases, and dissimilar pairs of conjuncts, such as verb phrases and clauses. As a result, the proposed methods outperform existing methods with or without the use of part-of-speech tags and/or pretrained word embeddings; thus, the proposed methods require no prerequisite external resources. On the contrary, many previous studies have relied heavily on external resources, such as thesauri, language models, and syntactic parsers.

- **Training method for boundary identification**: The proposed methods enumerate all possible spans of coordination or conjuncts efficiently. With no restriction on candidate spans during training, the models learn much from negative examples and successfully predict conjuncts robustly.
- **Consistent production with context-free grammar rules**: A context-free grammar can effectively be used to produce coordinate structures without conflict; it is able to ensure that any two coordinate structures are always either disjoint or nested. The use of a context-free grammar was originally proposed by Hara et al. [39], but their context-free grammar rules restrict the forms of coordinate structures. The proposed bottom-up method extends their rules to produce more various forms of coordination and makes it possible to produce 99.5% of coordination constructions signaled by four typical coordination conjunctions in the Penn Treebank: *and*, *or*, *but*, and *nor*.
- **Framework of efficient parsing**: The context-free grammar rules for coordination are used in the Cocke–Kasami–Younger (CKY) algorithm to produce consistent coordinate structures. Inspecting a variety of coordinate structures incurs huge computational cost. The proposed bottom-up method divides the model into submodels so that each of them learns different parts of coordinate structures. This division successfully keeps the time complexity of the parsing at $\mathcal{O}(N^3)$, where N is the number of words in a given sentence, while the proposed top-down method, which does not use production rules, has a time complexity of $\mathcal{O}(N^2)$.
- **Performance gain**: Both proposed methods outperform previous methods by a great margin, and the best proposed model achieved 76.64% F1 and 74.15% recall scores for the task of coordination disambiguation on the Penn Treebank [28] and the GENIA Treebank [54] corpora, respectively. It is also demonstrated that the use of contextualized word embeddings enhances

their performance further. At the same time, both methods ensure that no conflicts exist between coordinate structures.

1.3 Structure of the dissertation

The remainder of this dissertation is organized as follows. Chapter 2 introduces coordination from many points of view: the structure of coordination construction, discussions of coordination in the linguistics literature, and its intractability in syntactic parsing. This chapter also illustrates the task definition and the resources used in the experiments. Chapter 3 describes prior research related to coordination in the field of computational linguistics and NLP. Chapter 4 presents the proposed top-down method for disambiguating coordination using artificial neural networks. This chapter demonstrates effective feature representations encoded by the networks with no dependence on external resources. Chapter 5 presents the proposed bottom-up method for identifying coordinated elements with simple generative rules. This chapter shows that the algorithm is capable of producing coordinate structures without conflicts. In Chapter 6, quantitative experiments comparing the two methods with prior systems are reported, and a qualitative analysis is presented. Finally, Chapter 7 concludes the dissertation by summarizing its achievements and discussing future research directions for the field of NLP.

Chapter 2

Coordination

This chapter introduces the syntactic phenomenon of *coordination* from various viewpoints, including the structure of coordination construction, discussions of coordination in the linguistics literature, and its intractability in syntactic parsing. The discussions here mainly focus on English, but many of the properties of coordination in English are shared with other languages. This chapter also defines the task that is considered in this dissertation and presents the resources used in the experiments.

2.1 Coordination

Coordination is the juxtaposition of two or more elements, known as *conjuncts*, typically linked by a coordinating conjunction (*coordinator*), e.g., *and*, *or*, and *but* in English. Each conjunct may consist of a single word or a much longer phrase of any syntactic category. An instance of coordination formed by a coordinator and conjuncts is called a *coordinate structure*. While conjuncts characterize coordination, the presence of coordination is often signaled by a coordinator. This section gives numerous examples to demonstrate the unique behavior of coordination.

2.1.1 Structure of coordination

The most frequent form of coordination is “A and B”, where two conjuncts are conjoined by a single coordinator:

(1) [John] and [Mary] went out for dinner.

(2) He bought [an old book] and [three pencils] yesterday.

In all examples in this dissertation, conjuncts of coordinate structures are marked using square brackets. In (1), the conjuncts *John* and *Mary* are joined by the coordinator *and*. In general, conjuncts are either individual words, as in (1), or phrases, as in (2). Conjunctions are typically either *and* or *or*. Accompanied by punctuation marks, additional conjuncts can be attached to a coordinate structure:

(3) [Jane], [John] and [Mary] went out for dinner.

(4) He bought [two textbooks], [a new magazine], [an old book], and [three pencils] yesterday.

An unlimited number of elements can be concatenated in parallel to form a single instance of coordination.

More complicated coordination can occur when two or more coordinate structures appear in one sentence because they can potentially be nested:

(5) [Jane visited [John] and [Mary]], but [they had gone out for dinner].

(6) [He bought [an old book] and [three pencils]] and [his brother bought [two textbooks] and [a new magazine]] yesterday.

These nested structures make it more difficult for humans to grasp which elements are conjoined.

Conjuncts typically have a symmetric relationship with each other, as opposed to the asymmetric relationship of subordination. This symmetry could be observed in a number of ways. First, each conjunct is equally able to expand conjoined structures. Given (2), the two sentences in (7) can be retrieved.

(7) He bought [an old book] yesterday. He bought [three pencils] yesterday.

Secondly, each conjunct generally has the same case that the coordinate structure would have. In (1), the coordinate structure is in the nominative case, and each conjunct is nominative as well. Thus, they could be replaced with pronouns, as in (8).

(8) [He] and [she] went out for dinner. They went out for dinner.

Thirdly, conjuncts are often alike or identical in syntactic category, as in the examples above. However, there are many instances where the conjuncts do not match in category:

(9) Mike is [a fire fighter]$_{\mathrm{NP}}$ and [proud of his job]$_{\mathrm{ADJP}}$.

(10) Bill is [in trouble]$_{\mathrm{PP}}$ and [trying to come up with an excuse]$_{\mathrm{VP}}$.

(11) They are now [married]$_{\mathrm{ADJP}}$ and [thinking of having children]$_{\mathrm{VP}}$.

We have thus far exemplified coordinate structures formed by one coordinator. Such coordinate structures are called *syndetic* coordination. In syndetic coordination with more than two conjuncts, the conjunction is placed between the last two conjuncts, as shown in (3) and (4). In contrast, all conjuncts can be linked by coordinating conjunctions. This type of coordination is called *polysyndetic*. A passage from *The Book of Genesis* is emphasized by the multiple uses of the conjunction *and*:

> *And God said, "Let the earth bring forth the living creature after his kind, cattle, and creeping thing, and beast of the earth after his kind." And it was so. And God made the beast of the earth after his kind and cattle after their kind, and everything that creeps upon the earth to its kind. And God saw that it was good.* (1:24-25)

Polysyndetic coordination is used to slow down the rhythm and draw attention to each phrase. Conversely, no coordinating conjunction could be placed between conjuncts to form a coordinate structure, which is called *asyndetic* coordination. Examples include *"veni, vidi, vici,"* a famous Latin phrase said to be spoken by Julius Caesar, and its English translation "I came, I saw, I conquered." Asyndetic coordination is used to speed up the rhythm and make a single idea more memorable. The remainder of this dissertation focuses only on syndetic coordination because this is the most common form in broad domains of English text. In addition, coordinating conjunctions in this work are limited to single words including *and* and *or*, not multi-word expressions such as *as well as*.

2.1.2 Coordination in linguistics

Coordination has been studied extensively in linguistics, but there remain challenges to explain many exceptions in coordination. This section briefly introduces some concepts of coordination in the linguistics literature focusing on which elements can constitute a coordinate structure and why such a structure is acceptable.

Coordinate Structure Constraint

In generative syntax, the *Coordinate Structure Constraint* (CSC) is a constraint on movement that was proposed by Ross [95]; he states that "in a coordinate structure, no conjunct may be moved, nor may any element contained in a conjunct be moved out of that conjunct." The CSC prevents extraction of a single conjunct or an element from a single conjunct. This accounts for the ungrammaticality of the following sentences:

(12) *What did he buy [an old book] and [__] yesterday?

(13) *What is Mike [a fire fighter] and [proud of __]?

These movements at coordination fail because extraction cannot affect just one conjunct of a coordinate structure.

Across-the-Board Extraction

It is trivial to find exceptions that violate the CSC, such as the following:

(14) What did [Mary cook __] and [John eat __]?

The CSC prohibits movement out of a conjunct, but extraction out of all conjuncts in parallel, known as *Across-the-Board* (ATB) extraction [95], is allowed. Other than ATB extraction, non-ATB extraction from a single conjunct is possible when the order of the conjuncts does matter, such as when there is an implicit temporal or causal relation between them, as shown in the following example:

(15) Here's the whiskey which [I went to the store] and [bought __].

While ATB extraction shows a noticeable behavior, Williams [112] discovered an important asymmetry in *wh*-movement out of conjuncts:

(16) a. Who do you think [John saw __] and [Mary hugged __]?
b. Who do you think [__ saw John] and [__ hugged Mary]?
c. *Who do you think [__ saw John] and [Mary hugged __]?
d. *Who do you think [John saw __] and [__ hugged Mary]?

When movement is out of structurally parallel positions, as in (16a) with object gaps and (16b) with subject gaps, the result is acceptable, but when movement is out of different positions, the resulting sentence is no longer grammatical, as seen in (16c) and (16d).

Law of Coordination of Likes

As illustrated above, conjuncts typically belong to the same syntactic category. This assumption is known as the *Law of Coordination of Likes* (LCL) [112]. According to the LCL, ill-formedness of coordination often comes from disagreement among the syntactic categories of conjuncts, even though each conjunct would be possible within the sentence on its own. An example of violation of the LCL is as follows:

(17) *John made Mary [an omelet]$_{\mathrm{NP}}$ and [happy]$_{\mathrm{ADJP}}$.

However, there are many apparent counterexamples, such as (9)–(11), where conjoined elements are of syntactically different categories. Coordinated *wh*-phrases can also be straightforwardly deviated from the LCL, as in (18):

(18) [What] and [when] did you sing?

Many linguists have attempted to incorporate coordination of unlike categories [97, 74, 4, 87]. They argue that coordination should be acceptable when the conjuncts are alike in syntactic functions or semantic types, which allows coordination as in (9)–(11).

Non-Constituent Coordination

Coordination is one of the most commonly used tests to examine whether a string is a constituent. This is known as a *constituency test.* The test using coordination assumes that only constituents can be coordinated. The following examples demonstrate which coordinated elements are admitted to be constituents:

(19) The boy from Osaka bought apples.

- a. [The boy from Osaka] and [the girl from Tokyo] bought apples.
- b. The boy from Osaka [bought] and [ate] apples.
- c. The boy from Osaka bought [apples] and [bananas].
- d. The boy from Osaka [bought apples] and [ate bananas].
- e. *The boys [from Osaka bought] and [from Tokyo ate] apples.
- f. *The [boy from] and [girl in] Osaka bought apples.

The coordinate structures shown in (19a)–(19d) are possible, but those in (19e) and (19f) are not. Thus, these tests suggest that *the boy from Osaka*, *bought*, *apples*, and *bought apples* are constituents, but *from Osaka bought* and *boy from* are not.

The constituency test with coordination seems plausible, but coordination is not always a reliable indicator:

(20) The boy from Osaka bought [apples on Monday] and [bananas on Friday].

(21) The boy from Osaka [bought these] and [ate those] apples.

(22) He gave [the book to Mary] and [the magazine to John].

(23) [John likes swimming] and [Mary singing].

Although the above examples of coordination are grammatical, the conjuncts in (20)–(22) and the conjunct immediately following the coordinator in (23) are not viewed as constituents in most theories of syntax. Coordinate structures can be formed by coordinated strings that do not qualify as constituents, and this indicates that a diagnostic test based on coordination might not be suitable for identifying constituents.

Right Node Raising

The term *Right Node Raising* (RNR) refers to a construction in which a shared argument appears at the right periphery of parallel structures [90]. The parallel structures of RNR are typically the conjuncts of a coordinate structure, although the phenomenon is not limited to coordination. In fact, we have already seen RNR in (21), where *apples* is the material shared by the conjuncts. A key observation is that the right-peripheral constituent must be missing from all of the conjuncts; that is, RNR obeys ATB extraction and results in the non-constituency of the parallel structures.

Gapping

Gapping denotes a type of ellipsis that occurs in the non-initial conjuncts of coordinate structures. Gapping usually elides a finite verb, as in (23), but allows exclusion of further materials as well. The gapped material *likes* in (23) leaves the two remnants, *Mary* and *singing*, which compose the non-constituent conjunct. Further examples are shown below, where the elided material is indicated with a smaller font.

(24) [Mary will donate the money to the school] and [John $_{\text{will donate the money}}$ to the hospital].

(25) *[Mary will donate the books to the school] and [John $_{\text{will donate the}}$ money to the hospital].

(26) [Mary expected John to help], but [John $_{\text{expect}}$ Mary $_{\text{to help}}$].

(27) *[Mary said that they spoke Portuguese], but [John $_{\text{said that they spoke}}$ Spanish].

As shown in (24), the gapped material is not necessarily a constituent, but the two remnants appear to be constituents. Example (25) is not acceptable because the gap unsuitably breaks the constituent *the money* and the remnant *money to the hospital* does not qualify as a constituent. The gapping in (25) leaves three remnants, *John*, *money*, and *to the hospital*, but there appears to be a strong preference for exactly two remnants, placed on each side of the gapped material. Nevertheless, the gap can be discontinuous, as in (26). Gapping can span a verbal material and non-finite clause boundaries, as shown by the examples. However, it cannot be applied across a finite clause boundary, as shown in (27), even though the two remnants are constituents. Note that the requirements for gapping mentioned thus far are not exhaustive. Further discussions are presented in the literature [38, 47]. Gapping poses a challenge to phrase structure grammars because it is not evident how one might give a satisfactory analysis of material that can be gapped.

Although various solutions have been proposed, there is not yet a satisfactory explanation for all of the problems discussed here. Coordination occurs both in phrase structure and sentence structure, but the relationship between the two remains unclear. The role of coordination in sentence construction is also largely undetermined. Coordination therefore remains difficult to explain by any formal linguistic theory.

2.1.3 Punctuation in English language

Punctuating around quotation marks

The placement of punctuation and quotation marks differs between American English and British English. In American style, commas and periods are always inside the quotation marks, even if they are not in the original material. In British style, on the other hand, unquoted commas and periods are placed outside the quotation marks. For all other punctuation, the American and British styles are in agreement—unless the punctuation is part of the quoted material, it goes outside the quotation marks.

(28) “I like all of his songs,” Mary said, “because his lyrics are touching.” (American style)

(29) ‘I like all of his songs’, Mary said, ‘because his lyrics are touching’. (British style)

In this dissertation, the placement matters when it involves coordination:

(30) It filled its daily schedule with newscasts called “Daybreak,” “Daywatch,” “Newsday,” and “Newsnight,” but the shows varied little in content, personality or look. (*The Wall Street Journal*)

In (30), the commas between the conjuncts no longer work as delimiters and therefore the punctuation and quotation marks must be treated properly to retrieve the conjuncts.

Serial comma

In English language, a serial comma, also known as *Oxford comma*, is a comma placed immediately before the coordinating conjunction in a series of three or more terms. The serial comma is usually used to avoid ambiguity:

(31) I went to the zoo with my parents, John and Mary.

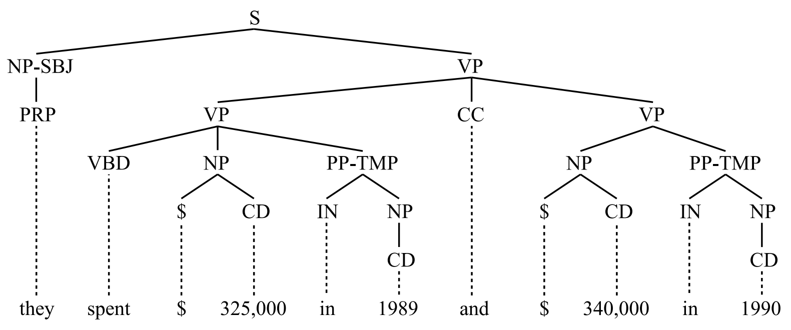
(32) I went to the zoo with my parents, John, and Mary.

In (31), there is ambiguity about the parentage because “John and Mary” can be read as being in apposition to *my parents*. In contrast, (32) indicates that John and Mary are not the writer’s parents because *my parents* is in parallel with *John* and *Mary*. However, the necessity and usage of the serial comma are still controversial. Opinions on whether to use the serial comma differ among style guides.

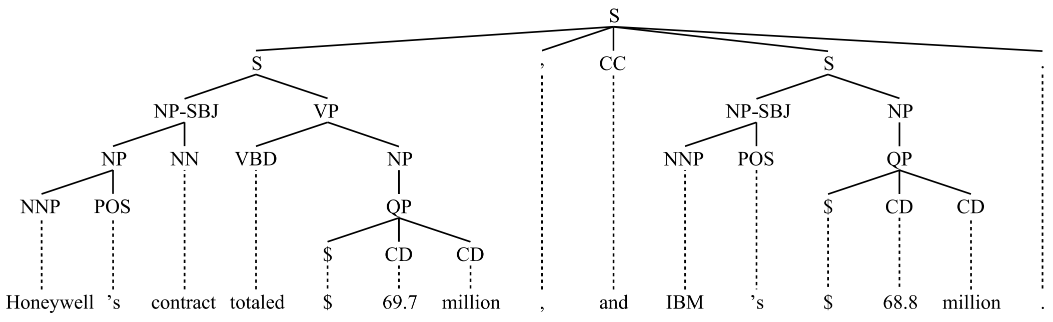
2.2 Limitations of syntactic parsing

2.2.1 Coordination in constituency parsing

In phrase structure grammars, there are a number of coordination instances that are not clearly qualified as constituents. For example, in the clause “. . . they spent [\$325,000 in 1989] and [\$340,000 in 1990] . . . ”, we can easily discover what are coordinated. Nonetheless, the two conjuncts *\$325,000 in 1989* and *\$340,000 in 1990* do not qualify as constituents because the noun phrase in each conjunct is the object of the finite verb *spend*, while the prepositional phrase modifies the verb phrase. Phrase structure grammars are not capable of expressing such



(a) Non-constituent coordination.

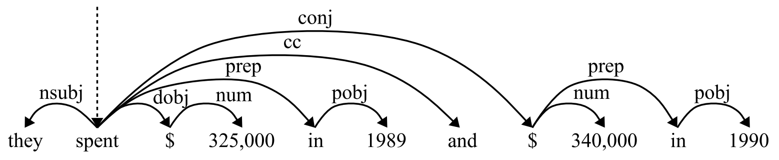

S
S
,
CC
S
.
NP-SBJ
VP
NP-SBJ
NP
NP
NN
VBD
NP
NNP
POS
QP
NNP
POS
QP
$
CD
CD
$
CD
CD
Honeywell 's contract totaled $ 69.7 million , and IBM 's $ 68.8 million .

(b) Gapping in coordination.

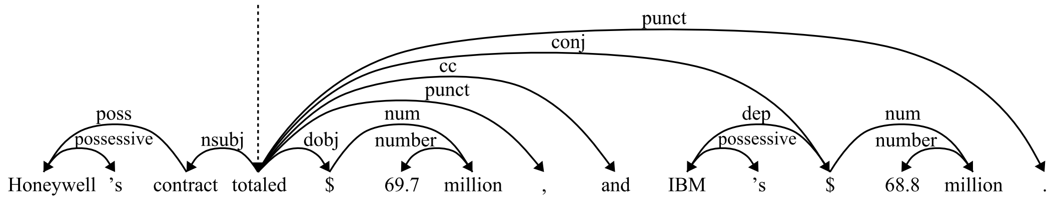
Figure 2.1: Malformed constituents in the Penn Treebank.

relations in non-constituent coordination. Figure 2.1(a) shows the annotation of a clause in the Penn Treebank [67]. In spite of the absence of a verb, the NP and PP form a VP and two VPs form coordination, but the verb in the second conjunct drops as an ellipsis. This can be problematic not only because it makes parsing more difficult but also because coordinated materials cannot be retrieved from the parse tree. Another issue in grammars is gapping. Figure 2.1(b) shows a constituency tree example. In the example, the two clauses *Honeywell's contract totaled $69.7 million* and *IBM's $68.8 million* are coordinated, but the latter does not have the head word *contract* of the NP-SBJ or the finite verb *total*. These malformed constituents make it difficult for parsers to produce trees correctly and collapse the symmetry of conjuncts.

Besides problems with the annotation scheme, constituency parsers in general



(a) Non-constituent coordination.



(b) Gapping in coordination.

Figure 2.2: Malformed dependencies in the Penn Treebank.

suffer from coordination [39, 29]. This is because parsers do not have the ability to capture syntactic and semantic similarities between conjuncts. Several researches have tried to incorporate features of coordination [91, 14, 26, 42].

2.2.2 Coordination in dependency parsing

Coordination deforms dependency trees as well as constituency trees. Figure 2.2 shows examples of this. In Figure 2.2(a), the prepositional phrase in the latter conjunct modifies *$340,000* as if the NP and PP form a larger NP. In Figure 2.2(b), the subject *IBM's* is attached to the object *$68.8 million* with the *dep* relation, which can easily confuse a parser. Because the possessive modifier *contract* is omitted in the conjunct, a parser may recognize *$68.8 million* as the possessive modifier of *IBM's*. In both cases, the head and modifier of *conj* relation do not share any structural similarity, and thus, it becomes more difficult for a parser to capture the symmetry between conjuncts. Incorporating conjunction features into dependency parsers has been explored [60, 77, 78, 51, 114, 30].

Dependency grammars are not sufficiently descriptive to disambiguate coordination, even if a parser succeeds in identifying all dependencies in a sentence. For instance, the two sentences “We saw an old [man] and [woman].” and “We saw a [beautiful woman] and [man]." have different scopes of coordination but identical parse trees. Thus, we cannot retrieve conjuncts from dependency trees expressed in Stanford/Universal Dependencies [21, 79]. This motivates researchers to develop methods that focus on identifying coordination.[1]

2.3 Coordinate structure analysis

2.3.1 Task definition

Coordination disambiguation, also called *coordination boundary identification* or *coordinate structure analysis*, refers to resolving ambiguity in the scope of coordination. The task of coordination disambiguation is to find conjuncts for a given word that can potentially play the role of a coordinator. In this dissertation, we call such words *coordinator keys*. More specifically, when a coordinator key actually forms a coordinate structure, all of the conjuncts must be identified in the form of $[begin, end]$, which represents the span of the conjunct. In case a coordinator key does not act as a coordinator, a system must return None, denoting the absence of a coordinate structure. Figure 2.3 shows the expected input and output of the task. We assume that it is not always possible to distinguish whether a coordinator key is an actual coordinator or not by its appearance, as in the case where *but* is not a coordinating conjunction but a preposition. In addition to coordinating conjunctions, some punctuation marks secondarily function to connect two conjuncts. We refer to such punctuation marks as *sub-coordinator keys* and as *sub-coordinators* when they actually glue two conjuncts together. Sub-coordinators cannot independently conjoin elements to form a coordinate structure without a coordinator. Throughout this dissertation, we define *and*, *or*, *but*, *nor*, and *and/or* as coordinator keys and comma (,), colon (:), and semicolon (;) as sub-coordinator keys. Thus, multiword expressions, such as "*as well as*", are not regarded as coordinator keys.

[1]Some grammars, such as combinatory categorial grammar, can account for various coordination constructions, but integrating methods for coordination disambiguation into parsers for those grammars is not explored in this dissertation and remains as future work.

Input	*But*$_1$ it said Charles Johnston, ISI chairman *and*$_9$ president, agreed to sell his 60% stake in ISI to Memotec upon completion of the tender offer for a combination of cash, Memotec stock *and*$_{37}$ debentures.
Output	*but*$_1$: None *and*$_9$: [8, 8] chairman ; [10, 10] president *and*$_{37}$: [33, 33] cash ; [35, 36] Memotec stock ; [38, 38] debentures

Figure 2.3: Expected input and output of the task.

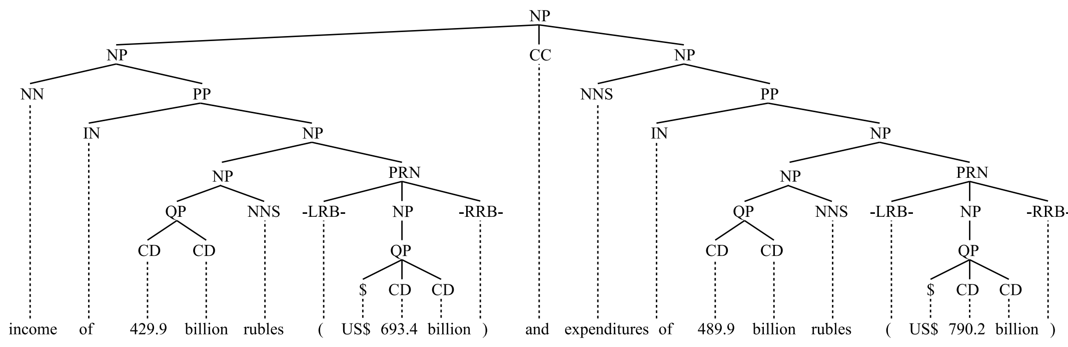
2.3.2 Difficulties

The difficulties in this task arise when there are multiple coordinate structures in a sentence or more than two conjuncts in a single coordinate structure. If there is more than one coordinate structure in a sentence, each coordinate structure must be isolated from or embedded in the other(s). In other words, coordinate structures cannot be partially overlapped. When there are more than two conjuncts in a coordinate structure, we must ascertain whether the punctuation marks are sub-coordinators that glue one more conjunct, and if so, which coordinate structure they belong to. Thus, we must identify how many conjuncts a coordinate structure contains and the locations of those conjuncts in the coordinate structure, i.e., whether it is nested in or isolated from other coordinate structures.

2.3.3 Important properties

Coordination has many unique traits besides its structure. Here, we focus on the two key properties between conjuncts that can be helpful to disambiguate a coordination boundary [29, 106].

- **Similarity**: Conjuncts in coordination have similarities in their structures and meanings.
- **Replaceability**: Each conjunct in coordination can be replaced with another conjunct.


NP
NP
CC
NP
NN
PP
NNS
PP
IN
NP
IN
NP
NP
PRN
NP
PRN
QP
NNS
-LRB-
NP
-RRB-
QP
NNS
-LRB-
NP
-RRB-
CD
CD
QP
CD
CD
QP
$
CD
CD
$
CD
CD
income of 429.9 billion rubles (US$ 693.4 billion) and expenditures of 489.9 billion rubles (US$ 790.2 billion)

(a) Similar structure and meaning between conjuncts.

The Tass news agency said the 1990 budget anticipates
[*expenditures of 489.9 billion rubles (US$ 790.2 billion)*]$_{\mathrm{NP}}$ and [*income of 429.9 billion rubles (US$ 693.4 billion)*]$_{\mathrm{NP}}$.

(b) Coherent sentence where conjuncts are exchanged.

Figure 2.4: Properties of conjuncts.

These properties are exemplified by the sentence *"The Tass news agency said the 1990 budget anticipates income of 429.9 billion rubles (US$ 693.4 billion) and expenditures of 489.9 billion rubles (US$ 790.2 billion)."* The two conjuncts *income of 429.9 billion rubles (US$ 693.4 billion)* and *expenditures of 489.9 billion rubles (US$ 790.2 billion)* conjoined by the coordinating conjunction *and* are both noun phrases and have symmetry in their syntactic structures, as shown in Figure 2.4(a). Moreover, the two conjuncts can replace each other while maintaining grammaticality and fluency in the newly-generated sentence, as in Figure 2.4(b).

2.4 Resources

2.4.1 Penn Treebank

The Penn Treebank [67] is the de facto standard English corpus for training and evaluating syntax-based NLP systems. Despite being in widespread use, the Penn Treebank has an annotation deficiency. Vadas and Curran [109] noted the lack of internal structures in base noun phrases. Hogan [42] also showed inconsistencies in annotation of flat structures, coordinated noun phrases, and part-of-speech tags. Dickinson and Meurers [23] proposed a method to detect and clean annotation noises. Maier et al. [66] introduced a distinction between coordinating and non-coordinating punctuation to add explicit boundaries between conjuncts.

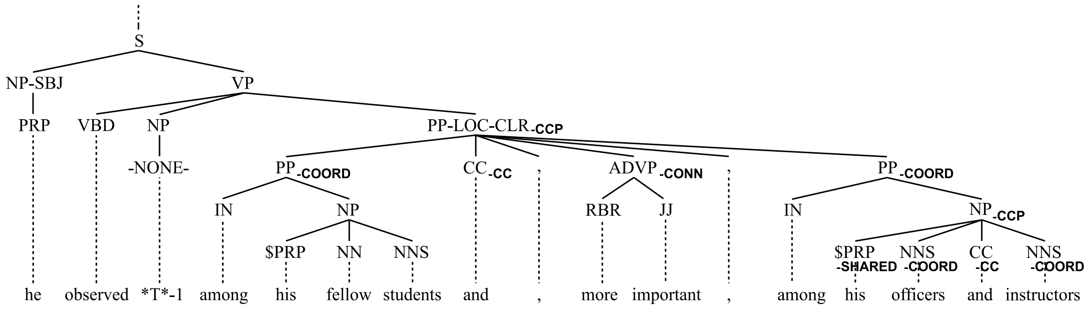
The work of Ficler and Goldberg [28] aimed for annotation in which coordinate structures are explicitly marked and the existing errors involving coordination are fixed. They added six different function labels to non-terminal symbols of nodes participating in coordinate structures while minimizing deviation from the original trees. Figure 2.5 shows an example of annotation and its corresponding tree representation. The corpus specifically uses the following function labels:

- **CC**: Coordinators.
- **CCP**: Phrases containing coordination.
- **COORD**: Conjuncts.
- **SHARED**: Modifiers or arguments shared by conjuncts (e.g., "*Bob cleaned and refueled the spaceship*").
- **CONN**: Connectives and parentheticals (e.g., "*The vacation packages include hotel accommodations and, in some cases, tours*").
- **MARK**: Markers (e.g., "*Both Alice and Bob are Aliens*").

Table 2.1 summarizes the statistics of the corpus. Sentences in sections 02–21, 22, and 23 of the Wall Street Journal were collected as the training, development, and test sets, respectively. In the Penn Treebank extended by Ficler and Goldberg, there are 24,450 coordinate structures within 19,095 of a total 49,208 sentences. Thus, coordination construction occurs in 38.8% of the sentences in the corpus. It is also worth noting that coordinated elements marked by an annotator agreed with 92.8% of 1,000 coordination phrases annotated by an additional linguist, which indicates how difficult it is even for humans to identify coordinate structures consistently.

```
...
(S
  (NP-SBJ (PRP he) )
  (VP (VBD observed)
    (NP (-NONE- *T*-1) )
    (PP-LOC-CLR-CCP
      (PP-COORD (IN among)
        (NP (PRP$ his) (NN fellow) (NNS students) ))
      (CC-CC and)
      (, ,)
      (ADVP-CONN (RBR more) (JJ important) )
      (, ,)
      (PP-COORD (IN among)
        (NP-CCP (PRP$-SHARED his) (NNS-COORD officers) (CC-CC and)
(NNS-COORD instructors) )))))
...
```

(a) Original annotation in the corpus.



(b) Corresponding constituency tree.

Figure 2.5: Coordination annotation in the Penn Treebank.

	#sent.	#tokens/sent.	#coord.	#sent. w/ coord.
Penn Treebank	49,208	23.85	24,450	19,095
Training	39,832	23.85	19,890	15,481
Development	1,700	23.59	848	673
Test	2,416	23.46	1,099	873
GENIA Treebank	18,541	26.23	17,167	11,259

Table 2.1: Statistics of the Penn Treebank and the GENIA Treebank.

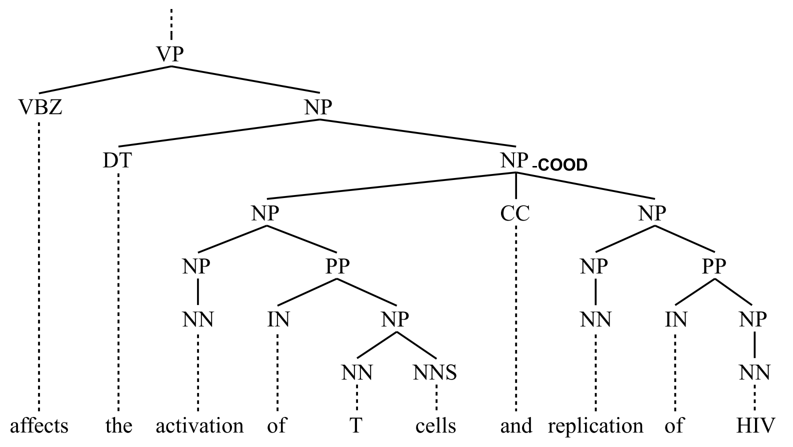
2.4.2 GENIA Treebank

The GENIA Treebank [54, 104] is a syntax-annotated corpus constructed from a collection of abstracts of papers in the biomedical domain. The annotation scheme of the GENIA Treebank essentially follows the Penn Treebank II scheme but includes some modifications. One of the modifications is explicitly marking a coordinate structure with the attribute "COOD", which is not the case in the original Penn Treebank scheme. Figure 2.6 shows an example and its corresponding tree representation. Note that a major source of difficulty in annotation is coordination, especially that involving ellipses. When annotating the phrase "CD2 and CD25 receptors", for example, we want to know whether the term "CD2 receptors" is valid or not. One way of knowing that is to examine the term in the collected texts. In the case where the term does not appear in the texts, it is more difficult to determine whether "receptors" is shared by "CD2" and "CD25" or not without deep domain knowledge. The authors reported that the annotation disagreements between annotators actually were in the cases involving coordination.

The GENIA Treebank is challenging in some respects for the task of coordination disambiguation. First of all, the underlying domain itself is difficult to understand without domain knowledge because a number of technical terms appear in the corpus. Second, sentences tend to be long and complicated because of coordination and subordination. Third, coordinate structures in the biomedical-domain corpus more frequently emerge than in general texts, and they often involve ellipses. Table 2.1 summarizes the statistics of the corpus. In the GENIA Treebank, there are 17,167 coordinate structures within 11,259 of a total 18,541 sentences, which means coordination appears in 60.7% of the sentences in the corpus; therefore, the frequency of coordination in the GENIA Treebank is much larger than that in the Penn Treebank. However, Tateisi et al. [104] argued that a major source of difficulty and disagreement for annotations was coordination, especially when coordinated phrases had modifiers or involved ellipses. In such cases, the annotator often left a comment saying "unsure." The experiments reported in Chapter 6 used the beta version of the GENIA Treebank for evaluating the performance of the proposed approaches in practical domains.

```
...
<cons cat="VP">
  <tok cat="VBZ">affects</tok>
  <cons cat="NP">
    <tok cat="DT">the</tok>
    <cons cat="NP" syn="COOD">
      <cons cat="NP">
        <cons cat="NP"><tok cat="NN">activation</tok></cons>
        <cons cat="PP">
          <tok cat="IN">of</tok>
          <cons cat="NP"><tok cat="NN">T</tok><tok cat="NNS">cells</tok></cons>
        </cons>
      </cons>
      <tok cat="CC">and</tok>
      <cons cat="NP">
        <cons cat="NP"><tok cat="NN">replication</tok></cons>
        <cons cat="PP">
          <tok cat="IN">of</tok>
          <cons cat="NP"><tok cat="NN">HIV</tok></cons>
        </cons>
      </cons>
    </cons>
  </cons>
</cons>
...
```

(a) Original annotation in the corpus.



(b) Corresponding constituency tree.

Figure 2.6: Coordination annotation in the GENIA Treebank.

Chapter 3

Related Work

Many efforts have been made to integrate grammars into parsers to deal with coordination [113, 92, 20, 32, 62, 56, 49, 73, 19, 99]. Defining grammars for coordination is difficult because they must account for complicated phenomena, such as gapping, non-constituent conjuncts, and mismatched syntactic category. Given the advancement of statistical methods for NLP, attention has turned from dealing with coordination in grammars to how to disambiguate boundaries of coordinate structures.

3.1 Coordination disambiguation

To identify boundaries of coordinate structures, many researches have exploited the similarity property of coordination. Agarwal and Boggess [1] proposed a simple deterministic method to find conjuncts. They regarded the phrase immediately following a coordinator as the post-conjunct and then found the pre-conjunct among similar phrases preceding the coordinator by using syntactic and semantic labels. Kurohashi and Nagao [59, 60, 58] introduced a chart-based method to examine similarities between two arbitrary series of words surrounding a coordinator. They assigned scores between two *bunsetsu*, which is comprised of a content word followed by function words in Japanese, using heuristic rules and manually-designed weights. Okumura and Muraki [83] took advantage of symmetric patterns between conjuncts. Their model calculates the best-balanced pair of word sequences on the basis of feature representations of the patterns. All methods described above are heuristic, and their similarity scores were manually designed.

In contrast with deterministic methods, several studies have exploited statistics from corpora and other resources. Resnik [94, 93] resolved ambiguities for the form of coordination "noun1 and noun2 noun3" using taxonomic relations taken from WordNet [72] and the co-occurrence relations of words. Goldberg [34] applied a statistical model to determine the attachment of ambiguous coordinated phrases whose forms are "noun1 preposition noun2 and noun3". Chantree et al. [13] used the distributional similarity between head words of coordinated phrases. Nakov and Hearst [75] disambiguated coordinated nouns using web-based statistical features. Hogan [43] investigated which similarity measures are helpful for identifying conjuncts. Most of these works outperformed rule-based methods, but they mostly only deal with conjunction between nouns.

Advances in machine learning methods have led to the development of coordination disambiguation. Shimbo and Hara [100] proposed a sequence alignment model with dynamic programming to capture locally similar structures in two conjuncts on the basis of a set of features, including word surfaces, part-of-speech tags, and morphological characteristics. The similarity score is computed by a weighted linear combination (perceptron) of manually designed features assigned to edges and nodes in graphs, which is very similar to the chart-based method of Kurohashi and Nagao [60], but it differs in that the weight parameters for the handcrafted features are automatically tuned through training. Okuma et al. [82] improved the method of Shimbo and Hara for coordination in Japanese. Buyko et al. [11], Buyko and Hahn [10] attempted to resolve coordination ambiguities involving noun compounds as a sequential labeling problem. They used a linear-chain conditional random field (CRF), which also incorporates morpho-syntactic features of conjuncts and their shared elements. While the methods described above focus on non-nested coordination, Hara et al. [39] extended the work of Shimbo and Hara to accommodate nested coordination using context-free grammar rules. A consistent global structure of coordination is produced using discriminative functions based on the similarity of conjuncts with dynamic programming. Hanamoto et al. [37] used dual decomposition to combine a head-driven phrase structure grammar (HPSG) parser with the discriminative model developed by Hara et al.

The machine learning methods described above rely heavily on the observation that conjuncts tend to have similar structures and meaning. However, that observation is not always true because conjuncts sometimes belong to different syntactic categories, e.g., PP and ADJP. Some researchers have attempted to identify conjuncts using characteristics of conjuncts other than similarity. Kawahara and Kurohashi [52, 53] focused on resolving the ambiguities of coordinate structures without the use of any similarities. Their method relies on the dependency relations surrounding the conjuncts, which implicitly provide selectional preferences, and the generative probabilities of phrases. As the resources of selectional preferences to support coordinate structures, they used automatically constructed case frames and co-occurrences of noun–noun modification from a large-scale web corpus. Ogren [80, 81] exploited a language model to test the sentences expanded from coordinate structures. Ficler and Goldberg [29] proposed a neural network that incorporates the similarity and replaceability between conjuncts in the computation of scores for candidate pairs of conjuncts produced by an external constituent parser.

3.2 Coordination in other tasks

Special treatments of coordination have been explored in syntactic parsing tasks. Collins [18] and Bikel [7] altered the generative process for coordination in their lexicalized probabilistic context-free grammar (PCFG) parsers. Charniak and Johnson [14] directly incorporated features to capture syntactic parallelism in coordination for their maximum entropy-based discriminative reranking parser. Dubey et al. [25, 26] proposed a method for incorporating syntactic priming of parallel structures into an unlexicalized PCFG parser. Nilsson et al. [77, 78] attempted to transform dependency structures to facilitate parsing of coordinate structures and verb groups for data-driven dependency parsers. Hogan [42] proposed a method for improving the disambiguation of noun phrase coordination within the framework of a lexicalized history-based parsing model, in which bi-lexical head–head co-occurrences are employed to identify nominal heads of conjuncts. Kawahara and Kurohashi [51] integrated coordination disambiguation into a fully-lexicalized generative dependency parser. Their model measures similarities between conjuncts in the same way as Kurohashi and Nagao [60] and calculates generative probabilities of coordination using these similarities by taking advantage of large-scale case frames. Kübler et al. [57] used a PCFG reranking parser with automatically detected scopes of conjuncts to improve parsing in German. Ficler and Goldberg [30] introduced features specifically designed to capture the symmetry of conjunct heads to improve dependency parsing with neural networks.

Coordination resolution is also a major problem in named entity recognition (NER), in which named entities with conjunctions are often recognized as single entity mentions or two distinct entity mentions. For example, the string *Australia and New Zealand Banking Group Limited* is not the conjunction of two entities *Australia* and *New Zealand Banking Group Limited* but a single entity as a whole. In contrast, the string *alpha- and beta- globin* is actually composed of two entities, but the preceding conjunct *alpha-* is elliptical. Thus, two distinct entities, *alpha- globin* and *beta- globin* should be recognized. Ellipsis resolution of NER is the process of identifying the non-elliptical entity mentions. Finkel et al. [31] discussed the difficulty of the task without additional treatment of ellipses in coordination. Nenadic et al. [76] proposed patterns for automatic biomedical term recognition, which are similar to the ones presented by Jacquemin et al. [45] but differ in covering terminology variations, including coordinated terms. However, they concluded that the morpho-syntactic patterns are not sufficient and reliable and that semantic information and domain-specific knowledge are additionally required to handle structural variants. Mazur and Dale [68] distinguished four categories of conjunctions related to named entities and presented machine-learned classifiers to disambiguate the different uses of a conjunction. However, their work did not explicitly resolve ellipses in coordination. Buyko et al. [11] focused on elliptical NP coordination in the biomedical domain. Their pipeline approach consists of two stages: recognition of named entities, including elliptical entity expressions, followed by resolution of elliptical coordination. Their CRF-based method performed well in the task of conjunct identification but failed to resolve complex expressions, such as nested coordinate structures and expressions where forward and backward ellipses occur simultaneously. While most of the methods above can only deal with relatively simple elliptical patterns in coordinated NPs, Chae et al. [12] proposed an NER method for identifying non-elliptical entity mentions as well as simple or complex ellipses using linguistic rules and an entity mention dictionary.

Huang [44] dealt with conjunctions in a rule-based machine translation system. Popović and Castilho [89] conducted a quantitative analysis regarding the ability of different machine translation systems to disambiguate conjunctions in source languages. Cohen et al. [17] worked on event extraction from coordinate structures using a constituent parser. For the task of open information extraction (Open IE), Saha and Mausam [98] proposed a coordination analyzer that modifies parse trees produced by dependency parsers. Their system splits a sentence by conjuncts into coherent simple sentences using a language model and linguistic constraints and then feeds those sentences to their Open IE system.

Chapter 4

Top-Down Approach

4.1 Introduction

This chapter proposes a top-down algorithm for coordination disambiguation. Previous studies have been very limited in that they only dealt with two coordinated phrases for a coordinator, though more than two conjuncts frequently appear in practice. In contrast, the proposed approach first identifies the entire span of coordination and then splits it into individual conjuncts, enabling more than two conjuncts to be produced. A score for each candidate coordination span is computed by neural networks that aim to capture the similarity and replaceability between a pair of conjuncts. This enables more robust detection of non-similar conjuncts, such as VPs and coordinated clauses, while most previous studies relied on the similarity and distributional information for NP conjunctions. Although machine learning approaches [39, 37] can successfully identify NP conjunctions by extending the method of Shimbo and Hara [100], which is similar to the work of Kurohashi and Nagao [60], they require carefully hand-crafted features to incorporate morpho-syntactic similarities of conjuncts. On the other hand, neural network-based approaches do not require so-called *feature engineering* and can utilize richer continuous representations of words and other syntactic features. A neural network approach for coordination identification was first proposed by Ficler and Goldberg [29]. Although their method achieved state-of-the-art performance by modeling the similarity and replaceability of conjuncts, it still heavily relies on an external constituency parser that is trained on a treebank with special labels for coordination. The accuracy of coordination identification solely depends on the performance of the underlying syntactic parser, which is used to enumerate the candidates for conjuncts pairs and to compute scores for those candidates as additional features. The proposed end-to-end method does not require such parsing results and thus it enables more robust inference and easier use for coordination disambiguation.

4.2 Preliminaries

4.2.1 Neural networks

An *artificial neural network*, also simply called a *neural network*, is a computing system composed of interconnected artificial neurons that uses a mathematical model inspired by the biological brain for information processing. This section briefly introduces types of neural networks and training methods for them in the NLP literature. For a comprehensive introduction to neural networks and deep learning, I refer the reader to the textbook by Goodfellow et al. [35].

Feedforward Neural Networks

A *feedforward neural network* (FNN), also called a *fully-connected neural network* or *multilayer perceptron* (MLP), is a simple neural network wherein nodes are fully connected and the connections are acyclic. Each layer ℓ with $d^{(\ell)}$ nodes projects a given vector $\mathbf{x}$ onto another vector space:

$$f^{(\ell)}(\mathbf{x}; \theta) = g(\boldsymbol{W}^{(\ell)\top}\mathbf{x} + \mathbf{b}^{(\ell)}) \tag{4.1}$$

where $\boldsymbol{W}^{(\ell)}$ is a matrix $\boldsymbol{W}^{(\ell)} \in \mathbb{R}^{d^{(\ell-1)} \times d^{(\ell)}}$, $\mathbf{b}^{(\ell)}$ is a bias vector $\mathbf{b}^{(\ell)} \in \mathbb{R}^{d^{\ell}}$, and g is an activation function. Both $\boldsymbol{W}^{(\ell)}$ and $\mathbf{b}^{(\ell)}$ are learnable parameters for the network and included in a set of parameters θ. FNNs can be composed by stacking layers:

$$f(\mathbf{x}) = \left(f^{(L)} \circ \ldots \circ f^{(1)}\right)(\mathbf{x}) \tag{4.2}$$

where $\circ$ indicates function composition and L is the number of layers in the FNNs.

Training Deep Neural Networks

Typically, the weights of a neural network are tuned by minimizing the observed errors. The errors are measured by a *loss function* (*objective function* or *cost function*), which is evaluated periodically during training. The output (cost) of the function is minimized by moving weights in small steps proportional to the negative derivative. This technique is called *gradient descent*. *Backpropagation* is an algorithm to compute the gradient of the loss function efficiently with respect to the weights of the network. This efficient algorithm makes it feasible to use gradient methods for training neural networks. *Computational graphs* are especially helpful to express and understand the process of backpropagation, where the gradient is typically computed recursively from the last to the first layer of the network. A computational graph is a directed graph where the nodes correspond to mathematical operations or input values. A computational graph grows as an operation is applied in the network, and the derivatives for each operation can be computed by tracing the graph backwards.

Practically, mini-batch stochastic gradient descent (SGD) is the most used training method for neural networks. SGD is regarded as a stochastic approximation of gradient descent optimization because it uses an estimate of the gradient calculated from a randomly selected subset of the data instead of the actual gradient calculated from the entire data. Mini-batch SGD takes a small batch of examples from the data for estimating the gradient and updating weights, while regular (online) SGD uses one example per update. Mini-batch SGD is suitable for training neural networks for two reasons: (1) it can take advantage of parallel computation, while online SGD cannot, and (2) it can avoid local minima by moving stochastically, while batch gradient descent moves in one direction. For SGD training, the increment of movement towards a minimum, which is called the *learning rate*, is particularly important because if it is too low, it will either take too long to converge or get stuck in a local minimum, and if it is too high, the learning will jump over minima. Various algorithms to control the learning rate have been developed for SGD optimization.

Recurrent Neural Networks

A recurrent neural network [96, 27] (RNN) is a class of neural networks in which connections between nodes form a directed graph along a temporal sequence. RNNs apply a function repeatedly, consuming input vectors $\mathbf{x}_t$ one by one:

$$\mathbf{h}_t = f(\mathbf{x}_t, \mathbf{h}_{t-1}) \tag{4.3}$$

where $\mathbf{h}_t$ is a vector that represents the internal state (memory) of an RNN and t indicates a time step (i.e., position in a sequence). In the simplest case, the function f is defined as the following non-linear transformation, which is derived from FNNs:

$$f(\mathbf{x}_t, \mathbf{h}_{t-1}; \theta) = g(\boldsymbol{W}^\top \mathbf{x}_t + \boldsymbol{U}^\top \mathbf{h}_{t-1} + \mathbf{b}) \tag{4.4}$$

where $\boldsymbol{W}$ and $\boldsymbol{U}$ are weight matrices, $\mathbf{b}$ is a bias vector, and g is an arbitrary non-linear function, such as the hyperbolic tangent (tanh) or rectified linear unit (ReLU). The gradients for the parameters $\boldsymbol{W}$, $\boldsymbol{U}$, and $\mathbf{b}$ are computed by *back-propagation through time* (BPTT), which simply runs backpropagation on unfolded computational graphs. In practice, however, long short-term memory [41] (LSTM) is commonly used as an instance of RNNs because simple RNNs are likely to suffer from the exploding and vanishing gradient problems [5, 84] when trained with gradient methods and backpropagation. Unlike simple RNNs, LSTM has gated mechanisms, where hidden activations restrict input and output flows and allow the model to forget some internal states. Gated mechanisms enable LSTM to learn much more effectively than a simple RNN on long sequences. Other extensions, such as bidirectional RNNs and gated recurrent unit [15] (GRU), are also widely used.

Convolutional Neural Networks

A convolutional neural network [33, 61] (CNN) is a class of neural networks used in various applications in computer vision, speech processing, and NLP. This network applies a mathematical operation called *convolution* to all grids of a given matrix (e.g., 1D grid for a sequence of vectors) or tensor (e.g., 2D grid for images pixels with RGB channels). Especially in NLP, the operation is shifted with a fixed-sized filter on a sequence of vectors:

$$\mathbf{h}_i = g(\sum_{j=1}^{k} \boldsymbol{W}_j^\top \mathbf{x}_{i+j-1-\lfloor k/2 \rfloor} + \mathbf{b}) \tag{4.5}$$

where each $\boldsymbol{W}_j$ is a weight matrix $\boldsymbol{W}_j \in \mathbb{R}^{d \times d'}$, $\mathbf{b}$ is a bias vector $\mathbf{b} \in \mathbb{R}^{d'}$, g is an activation function, and $\lfloor x \rfloor$ represents the floor function, which returns the greatest integer less than or equal to x. d is the dimension of an input vector $\mathbf{x}_i \in \mathbb{R}^d$, and d' is the predefined dimension of the output vector $\mathbf{h}_i \in \mathbb{R}^{d'}$. The odd number k is called *window size*, which controls how far previous and following inputs are convolved together. For example, when $k = 5$, the preceding two vectors and following two vectors are weighted and summed with the current vector. Note that $\mathbf{x}_i$ is a zero vector or trainable parameter vector when $i \leq 0$ or $n < i$, where n is the length of a sequence. This operation can be applied to variable length sequences of vectors, which is beneficial for NLP.

4.2.2 Word representations

In recent NLP research, nearly all neural network models have employed a continuous-valued vector $\mathbf{w}$, called an *embedding*, associated with each w in a vocabulary $\mathcal{V}$.

$$f_{\mathbb{R}} : w \in \mathcal{V} \rightarrow \mathbf{w} \in \mathbb{R}^d \tag{4.6}$$

Word embeddings are typically denoted by a matrix $\boldsymbol{E} \in \mathbb{R}^{d \times |\mathcal{V}|}$, where each column is a word vector. The word vector for a word w is looked up through *one-hot* representation $\mathbf{w}' \in \mathbb{R}^{|\mathcal{V}|}$, where all values are 0 except the index of the word, which is 1 (i.e., $\mathbf{w} = \boldsymbol{E}\mathbf{w}'$). One of the simplest ways to obtain embeddings $\boldsymbol{E}$ is to use randomly initialized values and update them through backpropagation while training networks against specific tasks. In many cases, however, these embeddings are obtained by special algorithms to encode distributional information of words, such as *continuous bag-of-words* and *skip-gram* [70, 71]. The advantage of using word embeddings compared to other representations, such as one-hot ones, is that it is possible to encode rich semantic information into a vector. Embeddings can be associated with other information on words, such as part-of-speech (POS) tags.

Another way to represent a word as a vector is encoding morpho-syntax information. In this method, each character composing a word is mapped to a vector in the same manner as each word is mapped to an embedding. Then, a sequence of character vectors is reinterpreted as a single vector using either RNNs or CNNs.

Vector representations of words are crucial for coordination disambiguation. To measure the similarity between words, it can be computed as a cosine similarity using word vectors. Such similarity information is helpful to model symmetry between conjuncts. Other than the similarity measure, continuous representations can be exploited in various ways to capture the relationship between coordinated elements.

4.3 Related work

This section introduces the work of Ficler and Goldberg [29], which is the most relevant to the method presented in this chapter. Their method has three parts arranged in a pipeline manner: a binary classifier for detecting coordinators, a candidate conjuncts extractor, and a neural network-based scorer for candidates. In the first stage, a sentence composed of words is transformed to a sequence of word embeddings. To classify a coordinating word in the sentence, the corresponding vector obtained from the output sequence of a bidirectional LSTM is fed to a logistic classifier. In the second stage, they use the Berkeley parser [88] trained on the Penn Treebank with the coordination annotation extension [28], where coordinated phrases are explicitly marked with special function labels. They collect spans that are labeled as COORD, are adjacent to the coordinator, and have non-zero inside or outside probabilities. Candidates are constructed from all possible pairs of the collected spans. They reported that their method produced 6.25 candidates on average for each coordinating word and included the correct candidates for 94% of coordination instances on the development set of the corpus. In the last stage, they assign a score to each candidate pair of conjuncts and choose the highest scoring pair as the system output using the similarity feature, replaceability feature, and parser-based features. For the similarity feature, constituency labels are encoded by LSTMs using the paths towards the root starting from the words in a span. The similarity is then computed as the Euclidean distance between the pair of encoded representations. For the replaceability feature, two sentences simplified by replacing the whole coordinate structure with each of the two conjuncts are passed to LSTMs and transformed into feature representations. The parser-based features are assigned on the basis of the ranking measured by the product of inside–outside probabilities of each conjunct pair. These three types of features are finally fed into an MLP to produce scores for candidates.

The shortcoming of their method is clearly its dependence on the performance of the Berkeley parser. In their method, the parser is used three times—for candidate extraction, path encoding for the similarity features, and ranking features based of the parser results. In fact, they showed that much of the performance gain came from the parser-based features. Besides, the parser must be trained on a treebank with annotations specialized for coordination. Such requirements and dependence are not realistic, especially when applying their method to other domains, languages, or tasks. This motivated me to develop a method without the use of a syntactic parser or treebank.

4.4 Proposed method

4.4.1 Overview

This section describes the proposed method in detail. The proposed method identifies the whole span of the coordinate structure instead of the conjunct spans for a given *coordinator key* (introduced in Section 2.3.1). In other words, the model in this method learns and predicts only the beginning and end of a coordinate structure, assuming that the preceding conjunct ends immediately before the coordinator and the following conjunct appears immediately after the coordinator. These two conjuncts are hereafter referred to as the *pre-conjunct* and *post-conjunct*, respectively. When more than one conjunct accompanied with a comma(s) precedes a coordinator, the model regards those conjuncts as one large span that ranges from the beginning of the first conjunct to the end of the conjunct next to the coordinator (e.g., *"[cash], [Memotec stock] and [debentures]"*

$\rightarrow$ *"[cash, Memotec stock] and [debentures]"*). Note that we assume that conjuncts are always adjacent to a coordinator to reduce the time complexity and to enumerate all possible spans efficiently.

Specifically, given a sentence composed of N words $w_{1:N} = w_1, \ldots, w_N$ and a coordinator key w_k, the pre-conjunct and post-conjunct can be represented as $s_1 = w_i, \ldots, w_{k-1}$ $(1 \leq i \leq k-1)$ and $s_2 = w_{k+1}, \ldots, w_j$ $(k+1 \leq j \leq N)$, respectively. The proposed model predicts the most probable pair of s_1 and s_2 for the coordinator key w_k. When the coordinator key w_k does not connect any elements, the model returns NONE, which denotes the absence of a coordinate structure. The proposed method gives scores to all possible combinations of the beginning i and end j of a coordinate structure, including the case of NONE, and then retrieves the conjuncts $s_1 = w_{i:k-1}$ and $s_2 = w_{k+1:j}$ using the highest scoring pair of boundaries i and j. If the adjacent words w_{k-1} or w_{k+1} are punctuation marks, the boundaries are shifted not to include those marks. In addition, the pre-conjunct s_1 is split into smaller conjuncts by punctuation marks (sub-coordinators) to find more than two conjuncts for the coordinator. In the case where the score for NONE is the best among others, no coordinate structure is found for the coordinator key w_k.

4.4.2 Neural network architecture

Figure 4.1 shows an overview of the proposed neural network architecture, consisting of the following four components:

Input Layer: Maps a sequence of one-hot words and POS tags onto their representations from embeddings.

RNN Layer: Produces a sequence of sentence-level representations based on contexts using a bidirectional RNN.

Feature Extractor: Generates feature representations of possible pairs of coordinated spans.

Output Layer: Calculates the scores for given candidate pairs on the basis of their feature representations.

The following subsections describe these components in detail.

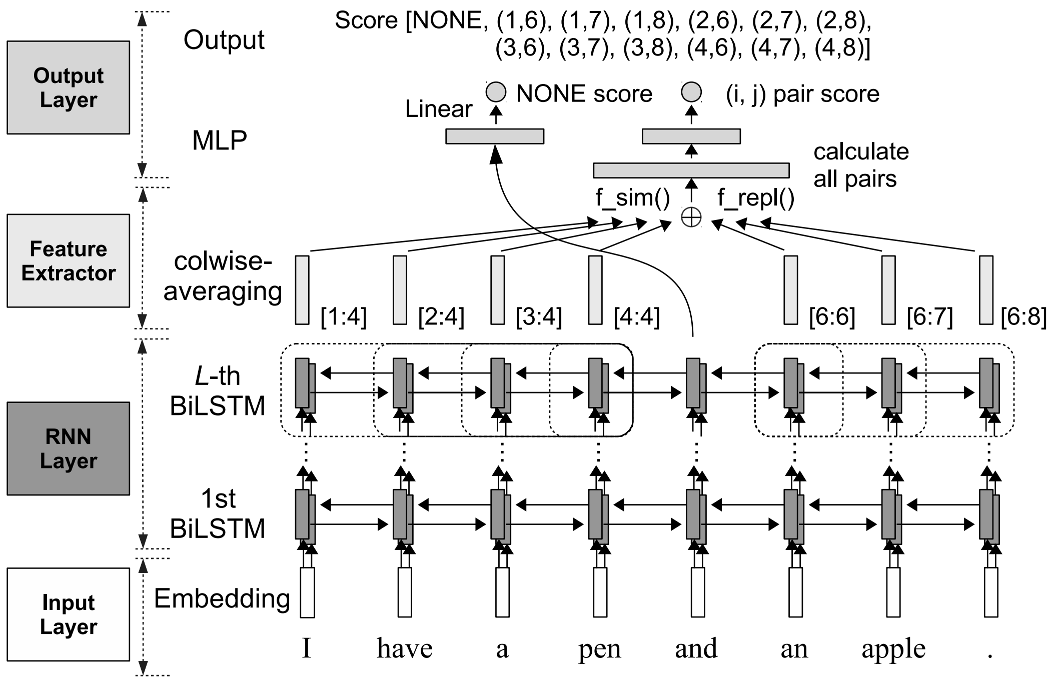


Figure 4.1: Overview of the neural network architecture in the proposed top-down approach.

Input layer

The first step of the neural network is to represent a sequence of words and POS tags in embeddings. The model receives sequences of one-hot encoded words $\mathbf{x}^{word}_{1:N} = \mathbf{x}^{word}_1, \ldots, \mathbf{x}^{word}_N$ and POS tags $\mathbf{x}^{pos}_{1:N} = \mathbf{x}^{pos}_1, \ldots, \mathbf{x}^{pos}_N$ and then looks them up in the matrices $\boldsymbol{E}^{word} \in \mathbb{R}^{d^{word} \times |v^{word}|}$ and $\boldsymbol{E}^{pos} \in \mathbb{R}^{d^{pos} \times |v^{pos}|}$, resulting in sequences of real-valued vectors $\mathbf{r}^{word}_n \in \mathbb{R}^{d^{word}}$ and $\mathbf{r}^{pos}_n \in \mathbb{R}^{d^{pos}}$, respectively. These real-valued vectors are concatenated as the input of the next layer:

$$\begin{aligned} \mathbf{r}^{word}_t &= \boldsymbol{E}^{word}\mathbf{x}^{word}_t \\ \mathbf{r}^{pos}_t &= \boldsymbol{E}^{pos}\mathbf{x}^{pos}_t \\ \mathbf{r}_t &= [\mathbf{r}^{word}_t; \mathbf{r}^{pos}_t] \\ \mathbf{r}_{1:N} &= \mathbf{r}_1, \ldots, \mathbf{r}_N \end{aligned} \tag{4.7}$$

where $[\mathbf{a}; \mathbf{b}]$ is a concatenation operation of vectors $\mathbf{a}$ and $\mathbf{b}$.

RNN layer

A sequence of distributed vectors is transformed into a sequence of hidden state vectors using stacked bidirectional RNNs. A bidirectional RNN processes a time series of inputs from the past to a future direction and from the future to a past direction. In the NLP literature, this network enables us to exploit left-to-right (forward) and right-to-left (backward) contexts of an input sequence, which is typically a sentence. The output of the ℓ-th layer of the stacked bidirectional RNNs at a time step t in the forward direction, which is denoted as $\mathbf{h}^F_{\ell,t}$, is computed as

$$\mathbf{h}^F_{\ell,t} = f(\mathbf{h}_{\ell-1,t}, \mathbf{h}^F_{\ell,t-1}) \tag{4.8}$$

where f is a non-linear function (defined as in Eq. 4.4), $\mathbf{h}_{\ell-1,t}$ is the output vector of the previous bidirectional layer at the same time step t, and $\mathbf{h}^F_{\ell,t-1}$ is the hidden state vector of the same layer at the previous time step $t-1$ in the forward direction. The hidden vector of the ℓ-th layer of the stacked bidirectional RNNs at a time step t in the backward direction ($\mathbf{h}^B_{\ell,t}$) is computed in the same way. The stacked bidirectional RNNs in this method produce the vector $\mathbf{h}_{\ell,t}$ by concatenating the vectors $\mathbf{h}^F_{\ell,t}$ from the forward direction and $\mathbf{h}^B_{\ell,t}$ from the backward direction at each time step t in every layer ℓ (i.e., $\mathbf{h}_{\ell,t} = \left[\mathbf{h}^F_{\ell,t}; \mathbf{h}^B_{\ell,t}\right]$).

In this method, the vector $\mathbf{r}_t$ from the input layer is used as an input for the first layer of the stacked bidirectional RNNs (i.e., $\mathbf{h}_{0,t} = \mathbf{r}_t$), and the vector $\mathbf{h}_{L,t}$ is simply referred to as $\mathbf{h}_t$, where L is the number of layers in the networks. In addition, the proposed method uses an LSTM unit as an instance of RNNs for each direction and every layer.

Feature extractor

This component produces two feature vectors based on the similarity and replaceability of pre- and post-conjuncts. The pre-conjunct representation $\mathbf{v}^{pre}_i$ and post-conjunct representation $\mathbf{v}^{post}_j$ are computed from a series of vectors $\mathbf{h}_{1:N}$ produced by the RNN layer using the function $f_{pooling}$. This work defines element-wise averaging as the function $f_{pooling}$:

$$f_{pooling}(\mathbf{h}_{l:m}) = \frac{1}{m-l+1}\sum_{t=l}^{m}\mathbf{h}_t \tag{4.9}$$

Thus, $\mathbf{v}_i^{pre}$ and $\mathbf{v}_j^{post}$ are expressed as follows:

$$\begin{aligned} \mathbf{v}_i^{pre} &= f_{pooling}(\mathbf{h}_{i:k-1}) \quad (1 \leq i \leq k-1) \\ \mathbf{v}_j^{post} &= f_{pooling}(\mathbf{h}_{k+1:j}) \quad (k+1 \leq j \leq N) \end{aligned} \tag{4.10}$$

Then, $\mathbf{v}_i^{pre}$ and $\mathbf{v}_j^{post}$ are fed into the following feature extraction function.

Similarity feature vector

To capture the similarity between the pre- and post-conjunct, the feature vector is computed as follows:

$$f_{sim}(\mathbf{v}_i^{pre}, \mathbf{v}_j^{post}) = \left[|\mathbf{v}_i^{pre} - \mathbf{v}_j^{post}|; \mathbf{v}_i^{pre} \odot \mathbf{v}_j^{post}\right] \tag{4.11}$$

where $|\mathbf{v}_i^{pre} - \mathbf{v}_j^{post}|$ is the absolute value of element-wise subtraction, and $\mathbf{v}_i^{pre} \odot \mathbf{v}_j^{post}$ is element-wise multiplication. These subtraction and multiplication operations are intended to model the semantic distance and relatedness [46, 103, 40].

Replaceability feature vector

The feature vector based on the replaceability of conjuncts is defined as follows:

$$\begin{aligned} &f_{repl}(\mathbf{h}_{1:N}, i, j, k) = \\ &\qquad \big[|\mathbf{h}_{i-1} \odot \mathbf{h}_i - \mathbf{h}_{i-1} \odot \mathbf{h}_{k+1}|; \\ &\qquad |\mathbf{h}_j \odot \mathbf{h}_{j+1} - \mathbf{h}_{k-1} \odot \mathbf{h}_{j+1}|\big] \end{aligned} \tag{4.12}$$

where $\mathbf{h}_{i-1}$ is the context vector linked to the beginnings of conjuncts and $\mathbf{h}_{j+1}$ is the context vector linked to the ends of conjuncts. The first subtraction $|\mathbf{h}_{i-1} \odot \mathbf{h}_i - \mathbf{h}_{i-1} \odot \mathbf{h}_{k+1}|$ is the difference between two context–conjunct connections at the beginning of coordination. The second subtraction $|\mathbf{h}_j \odot \mathbf{h}_{j+1} - \mathbf{h}_{k-1} \odot \mathbf{h}_{j+1}|$ is the difference between two context–conjunct connections at the end of coordination. These distance measures can be interpreted as the difficulty of replacing conjuncts. Intuitively, in the example *"The closely watched rate on [federal funds], or [overnight loans] between banks ... "*, the relation between *on* and *federal* should be close to that between *on* and *overnight*, and therefore the difference between the two relations should be small. This can be said for the relations *funds–between* and *loans–between*. Note that the first subtraction in the function f_{repl} is a zero vector when $i = 1$, and the second subtraction is a zero vector when $j = N$.

Output layer

This layer consists of MLPs to compute the scores of pairs of conjuncts based on the similarity and replaceability feature vectors. The score of a candidate pair of a pre-conjunct $w_{i:k-1}$ and post-conjunct $w_{k+1:j}$ is calculated as follows:

$$\textsc{Score}_\theta\big((i,j),k\big) = \mathrm{MLP}\big(\big[f_{sim}(\mathbf{v}_i^{pre}, \mathbf{v}_j^{post}); f_{repl}(\mathbf{h}_{1:N}, i, j, k)\big]\big) \tag{4.13}$$

To deal with the absence of coordination against a coordinator key w_k, a score is also assigned to a candidate of None. The score for None is simply computed as the product of a weight vector and sentence-level representation of the coordinator key produced by the RNN layer:

$$\textsc{Score}_\theta(\textsc{None}, k) = \mathbf{w} \cdot \mathbf{h}_k + \mathbf{b} \tag{4.14}$$

Using these scoring functions, the model assigns scores to all possible pre- and post-conjunct pairs. When the length of a sentence is N and a coordinator key appears as the k-th word, $(k-1) \times (N-k) + 1$ candidates are obtained, and the best scoring candidate is chosen among them:

$$S_k = \big\{\textsc{None}\big\} \cup \big\{(i,j) \,|\, 1 \leq i \leq k-1,\ k+1 \leq j \leq N\big\} \tag{4.15}$$

$$\hat{y}_k = \mathop{\arg\max}_{s \in S_k} \textsc{Score}_\theta(s, k) \tag{4.16}$$

where θ is the set of parameters used in all layers.

4.4.3 Learning

The loss function is defined as the negative log-likelihood of the true coordination span for a coordinator key:

$$P_\theta(y_k = s|x,k) = \frac{\exp\left(\textsc{Score}_\theta(s,k)\right)}{\sum\limits_{s' \in S_k} \exp\left(\textsc{Score}_\theta(s',k)\right)} \tag{4.17}$$

$$\ell_\theta(x,k,y_k) = -\log P_\theta(y_k|x,k) \tag{4.18}$$

where x is a sentence consisting of words and corresponding POS tags, k is the position of a coordinator key, and y_k is either the coordination span or label None.

The set of parameters θ in the model is trained by minimizing the following loss function:

$$L(\theta) = \sum_{(x,k,y_k) \in D} \ell_\theta(x,k,y_k) \tag{4.19}$$

where D is a set of triples of a sentence, coordinator key, and corresponding coordination span (or None if it is absent) in a training dataset.

4.4.4 Decoding

When a sentence contains more than one coordinate structure, any pair of them must be nested or disjoint, i.e., partial overlapping is prohibited. Individually determining the best span of each coordinate structure may cause such conflicts. Thus, the model is extended to find the best combination of coordinate structures, greedily choosing the most probable boundaries without conflicts.

Note that the proposed model learns and predicts the coordinate structure boundaries and not each conjunct; thus, the system simply divides the pre-conjuncts into sub-conjuncts using the character "," as a divider.

4.5 Preliminary experiments

4.5.1 Settings

This section presents preliminary experiments on the Penn Treebank (PTB) with the coordination annotation [28], comparing the performance of the proposed method with that of Ficler and Goldberg [29]. The corpus was split as in the previous work [29]: sections 02–21, 22, and 23 of the Wall Street Journal (WSJ) as the training, development, and test sets, respectively. Coordinating conjunctions *and*, *or*, *but*, *nor*, and *and/or* were defined as coordinator keys. The proposed model employed the pretrained 100-dimensional word embeddings of GloVe [85], which were not fixed but fine-tuned during training. The POS tags were obtained using the Stanford POS Tagger [107] with 10-way jackknifing, and the 100-dimensional POS tag embeddings were initialized with the normal distribution $\mathcal{N}(0, 1)$. The RNN layer was instantiated with two-layer bidirectional LSTMs. The dimensionality of the LSTM hidden vectors d^{hidden} in each direction was selected from {300, 400, 512}. The MLP in the model consisted of one hidden layer with ReLU activation and an output linear layer. The number of hidden layer units in the MLP was selected from $\{d^{hidden} \times 2,\ d^{hidden} \times 4\}$. The model parameters were optimized by mini-batch SGD with a batch size of 128. The learning rate was automatically tuned by Adam [55] with an initial learning rate 0.001 and without L2 regularization for the parameter weights. To prevent gradients from exploding, gradient clipping [84] was adopted with a threshold 5.0. In training, dropout [101] was applied to the embeddings, input vectors of each LSTM in the bidirectional LSTMs (except the first layer), and the hidden layer of the MLP. DropConnect [111, 69] was also applied to the hidden-to-hidden weight matrices within the LSTMs. The rates of dropout and DropConnect were selected from {0.00, 0.30, 0.50}. The model was trained for 50 epochs, and its snapshot was preserved for the evaluation with the test set when achieving the best F1 score based on the prediction of correct coordination spans on the development set. Table 4.1 presents the final hyperparameter selections.

Hyperparameter	Value
Dimension of LSTM hidden vector	300
MLP units in hidden layer	600
Dropout rate (all)	0.50
DropConnect rate	0.50

Table 4.1: Final hyperparameter settings for the preliminary experiments on the PTB.

Evaluation metrics

The proposed model was evaluated on the basis of the ability to predict the pre- and post-conjuncts for each coordinator using precision, recall, and F1 measures. In another setup, the evaluation focused on NP coordination, in which phrases of types NP and NX were considered as NP, as in the work of Ficler and Goldberg [29]. However, the proposed model does not predict the type of coordination. Thus, the model in the experiments additionally predicted whether a coordination candidate is NP coordination or not by feeding the hidden vector $\mathbf{h}_k$ to a logistic classifier, where k is the index of the coordinator key.

4.5.2 Results

Table 4.2 presents a comparison with existing methods. For all coordination, the proposed method outperformed existing systems with an F1 score of 73.90 on the test set, which is 1.20 better than the previously reported best result. The proposed top-down approach performed better than the method of Ficler and Goldberg. Note that their method makes use of syntactic trees produced by the Berkeley parser for extracting and scoring candidates, whereas the proposed method scans all possible coordination spans without such information. In addition, evaluating models by pre- and post-conjuncts is disadvantageous for the proposed model because others directly produce each conjunct span; the proposed model, on the other hand, learns not conjunct spans but coordination spans as a whole and splits them into conjuncts. The proposed method is likely to divide pre-conjuncts mistakenly by the appearance of false positive commas as

	Development			Test		
	Pre.	Rec.	F1	Pre.	Rec.	F1
	All Coordination					
Berkeley	70.14	70.72	70.42	68.52	69.33	68.92
Zpar	72.21	72.72	72.46	68.24	69.42	68.82
Ficler+16	72.34	72.25	72.29	72.81	72.61	72.7
top-down	75.08	76.76	**75.91**	72.85	74.97	**73.90**
	NP Coordination					
Berkeley	67.53	70.93	69.18	69.51	72.61	71.02
Zpar	69.14	72.31	70.68	69.81	72.92	71.33
Ficler+16	75.17	74.82	74.99	76.91	75.31	76.1
top-down	79.08	78.71	**78.89**	79.36	78.98	**79.16**

Table 4.2: Performance of predicting pre- and post-conjuncts on all coordination and on NP coordination in the PTB. The results for the three other methods are reported in the work of Ficler and Goldberg [29].

separators. Nevertheless, it performed the best even for NP coordination, which frequently contains more than two conjuncts.

To investigate the effectiveness of the proposed features, an additional experiment was performed with different uses of feature representations. As the baseline, a simple model was employed, using two averaged vectors as features (Eq. 4.9) and feeding them into the MLP instead of the similarity and replaceability features (Eq. 4.13). Note that the performance in this setting was measured on the basis of predicting coordination spans, not pairs of conjuncts adjacent to coordinators, as shown in Table 4.2, to ignore the drop caused by splitting. Table 4.3 summarizes the performances. The similarity and replaceability features worked better than the baseline independently. The former worked especially well for NP coordination, in which conjuncts are likely to be similar, whereas the latter feature may not capture the similarity between conjuncts. The joint model performed best by exploiting both features, which suggests the effectiveness of using two features together. When POS tags were not given to the model, the performance significantly dropped. I deduce that POS tags are crucial informa-

	All Coordination			NP Coordination		
	Pre.	Rec.	F1	Pre.	Rec.	F1
baseline	75.17	76.76	75.96	80.45	80.09	80.27
f_{sim}	77.36	79.00	78.17	80.68	80.32	80.50
f_{repl}	76.95	78.77	77.85	79.08	78.71	78.89
both	77.97	79.71	78.83	80.68	80.32	80.50
-POS tags	73.38	75.11	74.24	77.70	77.34	77.52
-GloVe	74.19	75.94	75.05	77.47	77.11	77.29
-decoding	77.50	79.24	78.36	80.22	79.86	80.04

Table 4.3: Performances with different settings for the PTB development set evaluated by coordination span prediction. "f_{sim}," "f_{repl}," and "both" indicate the use of similarity feature vectors, replaceability feature vectors, and both feature vectors, respectively.

tion to compare two spans in computing the similarity and replaceability features. The proposed model might benefit from the encoder being jointly trained for POS tagging. The model also performed poorly when it used randomly initialized embeddings instead of pretrained ones. It is worth noting that predicting more than one coordination span at once in the same sentence achieved a gain in performance. Without the joint decoding, the model identifies each coordination span individually, leading to possible conflicts.

Comparing the performances by different evaluation criteria, the proposed model apparently achieved better results when measured by performance on coordination span prediction. The drop in performance caused by splitting is shown as the difference between the best F1 score of 78.83 in Table 4.3 and that of 75.91 in Table 4.2 for all coordination. I believe that the performance could be improved further by more accurately retrieving individual conjuncts from coordination.

I further investigate the ability and performance of the proposed top-down approach in Chapter 6.

4.6 Summary

This chapter presented a novel top-down approach to coordination disambiguation. Unlike previous work, the proposed method first identifies the entire span of coordination instead of the conjuncts and then splits it into individual conjuncts. To determine the most promising span, the neural network model employed in this approach computes a score for a candidate span by exploiting two properties of conjuncts: (i) conjuncts tend to have a similar structure in syntax or semantics and (ii) conjuncts can be replaced with each other, maintaining the consistency of a sentence. On the basis of these observations, the model encodes two feature vectors from a sequence of vectors produced by bidirectional RNNs. These feature representations enable the model to identify both similar pairs of conjuncts (such as noun phrases) and dissimilar ones (such as verb phrases and clauses), which is experimentally validated, as reported in Chapter 6. As a result, the proposed method outperformed the existing best method and achieved state-of-the-art performance. The biggest contribution of the proposed top-down approach is resolving the dependence on information from syntactic parsers. Nevertheless, the model has room for improvement with a more sophisticated way to retrieve individual conjuncts from coordination.

Chapter 5

Bottom-Up Approach

5.1 Introduction

The previous chapter proposed a top-down approach for coordination disambiguation. The method can identify the coordination span instead of the conjunct spans for a given coordinating word by utilizing neural networks to incorporate the similarity and replaceability properties between coordinated elements. This overcomes one of the weaknesses of the neural network approach developed by Ficler and Goldberg [29]—its dependence on an external syntactic parser. The proposed top-down model outperformed existing methods despite its lack of the ability to delineate the boundaries of each conjunct. However, both neural network-based methods have difficulty producing complex coordinate structures, e.g., a coordinate structure with more than two conjuncts, and nested coordinate structures. In contrast, the method of Hara et al. [39] can handle such structures consistently by making use of production rules specialized for coordination.

This chapter presents an alternative bottom-up approach, which builds a coordinate structure by composing individual conjuncts proximate to a coordinator while keeping the end-to-end neural network model independent of external resources. Inspired by the work of Hara et al., the proposed method also defines context-free grammar rules to produce coordinate structures without conflicts and applies the rules with the CKY parsing algorithm. When scoring candidate conjunct pairs during CKY parsing, the proposed method employs neural network models similar to the ones used in the top-down approach in the previous chapter. This framework can successfully produce the optimal coordinate structures for a given sentence with good accuracy, where any two of them are not in conflict with each other.

5.2 Preliminaries

5.2.1 Context-free grammar

In formal language theory, a context-free grammar (CFG) is a formal grammar consisting of a set of production rules, each of which is expressed in the form

$$A \rightarrow \beta$$

where A is a single non-terminal symbol and β is a string of terminal and/or non-terminal symbols. CFGs are considered "context free" because any production rules in the grammar can be applied regardless of the context; no matter which symbols surround it, the single non-terminal on the left-hand side of the rule can be replaced by symbols on the right-hand side.

Formally, a CFG G is defined by the 4-tuple $G = (N, \Sigma, R, S)$, where

- N is a finite set of non-terminal symbols,
- Σ is a finite set (disjoint from N) of terminal symbols,
- R is a set of production rules, each of which is represented in the form $A \rightarrow \beta$, where A is a non-terminal symbol and β is a string of symbols from the infinite set of strings $(\Sigma \cup N)*$, and
- S (which is in N) is a start symbol.

For example, we define a CFG $G_0 = (N, \Sigma, R, S)$ as follows:

- $N = \{S, NP, VP, PP, PRP, VBD, DT, NN, IN\}$
- $\Sigma = \{I, saw, a, girl, with, telescope\}$
- $R = \{$
  S $\rightarrow$ $NP\ VP$

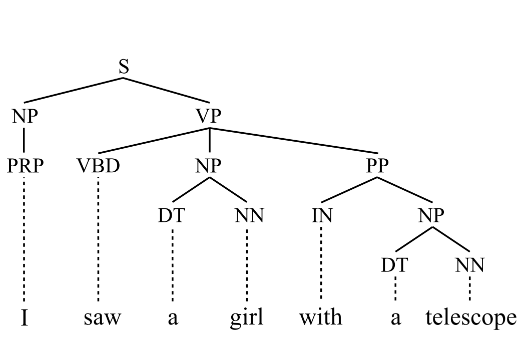
$$\begin{array}{rcl}
\text{NP} & \rightarrow & NP\ PP \ \mid\ PRP \ \mid\ DT\ NN \\
\text{VP} & \rightarrow & VBD\ NP \ \mid\ VBD\ NP\ PP \\
\text{PP} & \rightarrow & IN\ NP \\
\text{PRP} & \rightarrow & I \\
\text{VBD} & \rightarrow & saw \\
\text{DT} & \rightarrow & a \\
\text{NN} & \rightarrow & girl \ \mid\ telescope \\
\text{IN} & \rightarrow & with
\end{array}$$

}

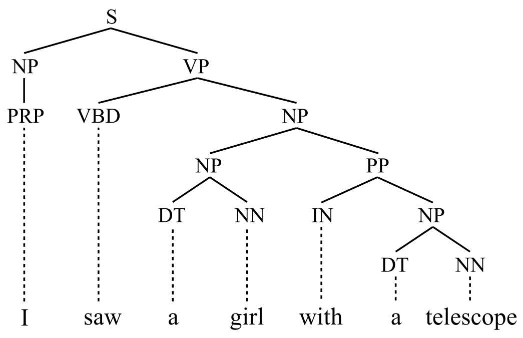
Note that we use the symbol | to indicate alternate possible expansions, which means we can apply either $A \rightarrow \beta_1$ or $A \rightarrow \beta_2$ from the rule $A \rightarrow \beta_1 \mid \beta_2$. A non-terminal symbol that immediately dominates a terminal, e.g., PRP, VBD, DT, NN and IN (POS tags) in the example, is called a pre-terminal. From G_0, we can generate the sentence *I saw a girl with a telescope* (in which punctuation marks are omitted for brevity) by applying production rules in the following derivation:

0. S
1. $\rightarrow NP\ VP$
2. $\rightarrow PRP\ VP$
3. $\rightarrow I\ VP$
4. $\rightarrow I\ VBD\ NP\ PP$
5. $\rightarrow I\ saw\ NP\ PP$
6. $\rightarrow I\ saw\ DT\ NN\ PP$
7. $\rightarrow I\ saw\ a\ NN\ PP$
8. $\rightarrow I\ saw\ a\ girl\ PP$
9. $\rightarrow I\ saw\ a\ girl\ IN\ NP$
10. $\rightarrow I\ saw\ a\ girl\ with\ NP$
11. $\rightarrow I\ saw\ a\ girl\ with\ DT\ NN$
12. $\rightarrow I\ saw\ a\ girl\ with\ a\ NN$
13. $\rightarrow I\ saw\ a\ girl\ with\ a\ telescope$

The structure resulting from the derivation can be represented by the parse tree shown in Figure 5.1(a). The set of strings derived from a CFG is called a *context-free language*. Thus, the example sentence can be said to be a string in the context-free language of the grammar G_0.



(a) I had the telescope.



(b) The girl had the telescope.

Figure 5.1: Different parse trees for a given sentence.

The example sentence can be generated by a different derivation as follows:

0. S
1. $\rightarrow$ $NP\ VP$
2. $\rightarrow$ $PRP\ VP$
3. $\rightarrow$ $I\ VP$
4. $\rightarrow$ $I\ VBD\ NP$
5. $\rightarrow$ $I\ saw\ NP$
6. $\rightarrow$ $I\ saw\ NP\ PP$
7. $\rightarrow$ $I\ saw\ DT\ NN\ PP$
8. $\rightarrow$ $I\ saw\ a\ NN\ PP$
9. $\rightarrow$ $I\ saw\ a\ girl\ PP$
10. $\rightarrow$ $I\ saw\ a\ girl\ IN\ NP$
11. $\rightarrow$ $I\ saw\ a\ girl\ with\ NP$
12. $\rightarrow$ $I\ saw\ a\ girl\ with\ DT\ NN$
13. $\rightarrow$ $I\ saw\ a\ girl\ with\ a\ NN$
14. $\rightarrow$ $I\ saw\ a\ girl\ with\ a\ telescope$

The corresponding parse tree is shown in Figure 5.1(b). This derivation produces a different parse tree. A grammar is said to be *ambiguous* if a string in the language of the grammar has more than one parse tree. Thus, the grammar G_0 is considered ambiguous.

```
Input: string $s$, grammar $G$
Output: $true$ if $s \in L(G)$; otherwise, $false$
 1: for $i \leftarrow 1$ to LENGTH($s$) do
 2:     $table[i,i] \leftarrow \{A \mid A \rightarrow s[i] \in G\}$
 3: end for
 4: for $j \leftarrow 2$ to LENGTH($s$) do
 5:     for $i \leftarrow j-1$ to $1$ do
 6:         for $k \leftarrow i$ to $j-1$ do
 7:             $table[i,j] \leftarrow table[i,j] \cup \{A \mid A \rightarrow BC \in G, B \in table[i,k], C \in table[k+1,j]\}$
 8:         end for
 9:     end for
10: end for
11: return $S \in table[1, \text{LENGTH}(s)]$
```

Figure 5.2: CKY algorithm. $L(G)$ denotes the context-free language of the grammar G, and S is the start symbol in G.

A CFG can be thought of in two ways: as a device for generating sentences, as explained above, and as a device for assigning a structure to a given sentence, which is known as *parsing*. In other words, parsing is a process of uncovering structures, such as parse trees, for languages. CFGs are powerful yet simple enough that efficient parsing algorithms exist for determining whether and how a given string can be generated from the grammar.

5.2.2 CKY algorithm

This section presents the Cocke-Kasami-Younger (CKY) algorithm [50, 115, 16], which is the most widely used dynamic programming-based approach to parsing. The CKY algorithm operates on CFGs given in Chomsky normal form (CNF), in which all production rules are of the form $A \rightarrow BC$ or $A \rightarrow s$, meaning that the right-hand side of each rule must expand either to two non-terminals or a single terminal. Every CFG can be transformed into an equivalent one in CNF. The algorithm is described in pseudocode in Figure 5.2. Because each non-terminal

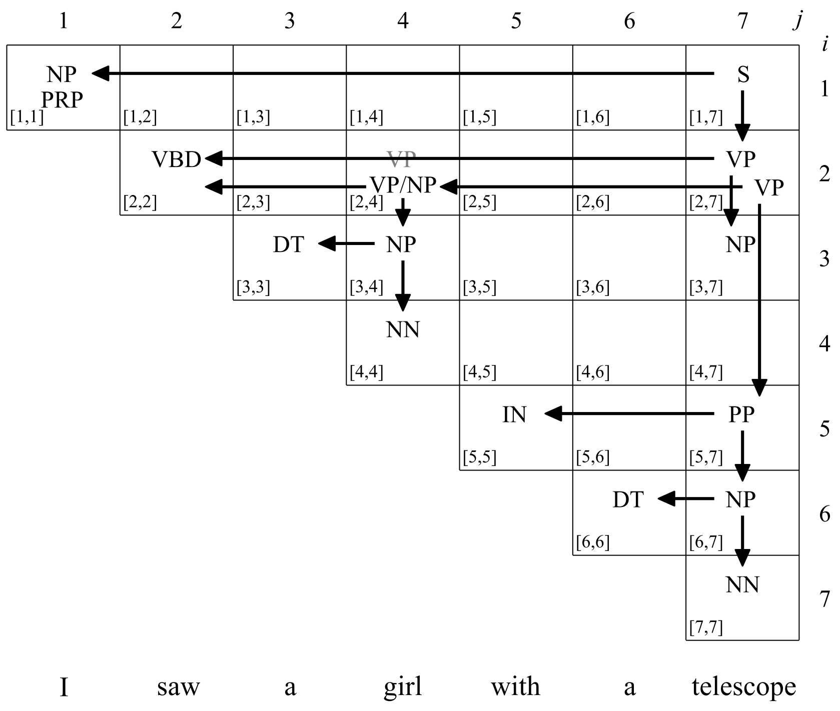


Figure 5.3: CKY chart. Rows and columns indicate start and end positions of constituents, respectively.

entry in the table has two child nodes in the parse, there must be a position in the input where a constituent $[i, j]$ can be split into two parts $[i, k]$ and $[k + 1, j]$ $(i \leq k < j)$. Such a position k lies somewhere along the row i and column j in the table. A concrete example of the CKY table is shown in Figure 5.3. This illustrates that the entry $[2, 7]_{\text{VP}}$ is derived from either $[2, 4]_{\text{VP/NP}}$ and $[5, 7]_{\text{PP}}$ or $[2, 2]_{\text{VBD}}$ and $[3, 7]_{\text{NP}}$. The importance of the CKY algorithm stems from its efficiency; the worst-case time complexity is $\mathcal{O}(n^3 \cdot |G|)$, where n is the length of the parsed string and $|G|$ is the size of the CNF grammar G.

The algorithm presented in Figure 5.2 can be used for recognizing a string as being in the context-free language of a grammar. To extract a parse tree for a given sentence, the algorithm must be extended so that it preserves a *backpointer* for each non-terminal symbol to track the instantiated rules in the table. However, if a string can be derived from more than one parse tree, how can we choose one from others? We can associate a score with each application of a rule, and the best parse tree is obtained by choosing the best scoring derivation, where each score is calculated as the sum of scores assigned to the rules used in the derivation:

$$\begin{aligned}\hat{T} &= \operatorname*{arg\,max}_{T \in \mathcal{T}_G(s)} score(T) \\ score(T) &= \sum_{A \to \beta \in T} s(A \to \beta)\end{aligned} \tag{5.1}$$

where s is the score for each rule. The time complexity of parsing is still cubic against the length of a string. The most famous application of Eq. 5.1 is assigning a (logarithm of) probability to each rule. A CFG that contains probabilistic rules is called a *probabilistic context-free grammar* [8, 9] (PCFG). In many cases, probabilities of rules in PCFGs are usually computed by maximum likelihood estimation on annotated corpora. I refer the reader to the textbook by Jurafsky and Martin [48] for further details.

5.3 Related work

This section introduces the work of Hara et al. [39], which is the most relevant to the method presented in this chapter. They proposed a framework that produces consistent coordinate structures derived from CFG rules specialized for coordination and made use of the rules in the CKY algorithm with a chart-based scoring system similar to the one by Kurohashi and Nagao [60]. Their CFG rules ensure that coordinate structures in a sentence are either disjoint or nested. The CFG is ambiguous and thus it can produce more than one parse tree, each of which contains different coordinated elements. To determine one parse tree, their system assigns a score to each candidate tree on the basis of similarities between conjuncts and returns the best scoring tree as the predicted coordinate structures.

The score in their work is computed as a weighted linear combination of manually designed features assigned to edges and nodes in an edit graph for the sequence alignment, which originates from the work of Shimbo and Hara [100]. Sequence alignment is a way of arranging sequences, such as DNA in bioinformatics, to identify similar (homological) regions, where the cost of transforming a sequence to another is calculated as a minimum-weight series of *insertion*, *deletion*, and *substitution* edit operations [36]. The cost, known as the *edit distance* or *Levenshtein distance* [63], is usually computed using a dynamic programming algorithm for the shortest path problem on a grid (edit graph). In the work of Shimbo and Hara, each edge on the edit graph has its own features, which consist of attributes, including the surface word, POS, and other morphological information. The weights of features are learned by a discriminative model (perceptron) so that one of the paths producing the correct conjuncts achieves the best score among all possible paths because word-to-word correspondences are not obvious. The score between conjuncts in the method of Hara et al. is computed in almost the same way as above, except that the averaged path score of all paths is taken in the associated edit graph instead of the best scoring path to reduce computation in the CKY algorithm, in which different subsequences are compared repeatedly.

Their contributions include consistency in coordinate structures, a machine-learnable method to capture the similarity between conjuncts, and not being limited to NP coordination. The drawbacks of their method are laborious feature engineering and lack of ability to identify dissimilar pairs of conjuncts, such as VPs and clauses. The idea for the bottom-up approach proposed in this chapter is based on their framework but aims to alleviate its drawbacks. Neural networks are employed to capture dissimilar conjuncts and to relieve feature engineering. The proposed method involves the integration of neural network-based scoring models with CKY parsing and the expansion of CFG rules to cover a broader variety of coordinate structures.

5.4 Proposed method

5.4.1 Background

Coordinate structures as a tree

Hierarchical relations of coordination and an unlimited number of coordinated elements in a sentence can be represented as a tree, which is referred to as a *coordinate tree* hereafter. Figure 5.4 shows an example of a coordinate tree. Tree structures are particularly suitable because the ranges of coordinate structures are always consistent (partial overlapping is not permitted), and conjuncts are shown as nodes with unlimited possible occurrences. In other words, for a given

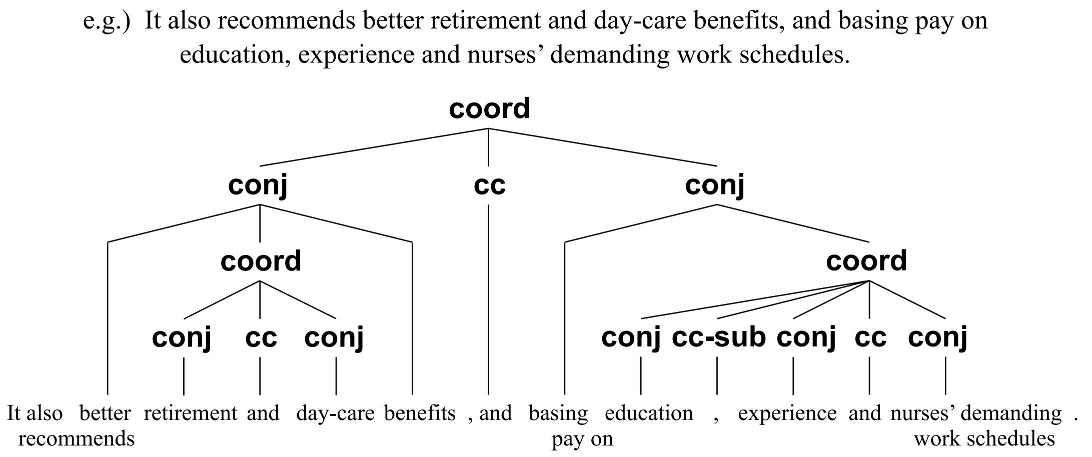

e.g.) It also recommends better retirement and day-care benefits, and basing pay on education, experience and nurses' demanding work schedules.
coord
conj
cc
conj
coord
coord
conj
cc
conj
conj
cc-sub
conj
cc
conj
It also recommends
better
retirement
and
day-care
benefits
, and
basing pay on
education
,
experience
and
nurses' demanding work schedules
.

Figure 5.4: Example of a coordinate tree. *coord* is a coordinate structure, *conj* is a conjunct, *cc* is a coordinator, and *cc-sub* is a sub-coordinator.

sentence, coordinate structures retrieved from a coordinate tree are always well-formed. Thus, the proposed method aims to produce a coordinate tree.

Combinatorial explosion of candidate coordination

One problem is that considering all possible combinations of conjuncts composing a coordinate structure for a given coordinator causes rapid growth of the problem complexity. Suppose the k-th word acts as a coordinator in a sentence consisting of N words. Assuming the coordinate structure has only two conjuncts adjacent to the coordinator, the number of possible pairs of conjuncts is $(k-1) \times (N-k)$. When the coordinator appears near the middle point in the sentence, the number is at most $(N/2)^2$. In fact, conjuncts are often far apart from a coordinator, and a non-coordinated element can be injected between the coordinator and one or both of the conjuncts, such as in "[A] and, on the other hand, [B]". If we take such cases into account, the number of candidates is $k(k-1)/2 \times (N-k)(N-k+1)/2$, and it is approximated to $(N/2)^4/4$ at most for a coordinator at the center position of the sentence.

For more complex sentences, such as those containing multiple coordinators and/or coordinate structures formed by more than two conjuncts, the number of possible candidates increases exponentially and becomes intractable. Thus, inspecting all possible candidates at once is not realistic. The proposed method focuses on recursively applying local evaluation and comparison to arbitrary conjunct pairs to suppress combinatorial explosion and make the inspection tractable.

5.4.2 Parsing with a context-free grammar for coordination

This section describes how to produce coordinate structures as a parse tree. The parse tree, which is in one-to-one correspondence with the coordinate tree, is derived from special CFG rules described in Table 5.1. Using CFG rules has two advantages: (1) exploration of only valid coordination, where any two of coordinate structures are always nested or disjoint and (2) efficient production of a coordinate tree with the CKY algorithm. The CFG rules in Table 5.1, which are extended from the ones by Hara et al. [39], can produce a coordinate structure whose conjuncts are apart from the coordinator, whereas the rules of Hara et al. can produce a coordinate structure whose conjuncts always adjoin the coordinator. As a result, the proposed CFG rules can produce coordinate trees for 99.5% of the sentences in the coordination annotated PTB [28].[1]

The CFG in Table 5.1 is ambiguous and thus it can produce different parse trees for a given sentence. Hence, the proposed system assigns a score to each tree and returns the best scoring tree:

$$\hat{T} = \operatorname*{arg\,max}_{T \in \mathcal{T}_G(s)} score(T) \tag{5.2}$$

where $\mathcal{T}_G(s)$ is the set of parse trees that the CFG G can derive from the sentence s.

Scoring for the CKY algorithm

For a given parse tree, the system gives scores only to coordination nodes, denoted as COORD, and pre-terminals in the tree. Scores for COORD nodes are assigned by the function $score_{coord}$. Scores for pre-terminals are assigned by the function

[1]Most of the non-derivable coordinate structures are in a form such as "A and B and C", where a coordinating word is regarded as a sub-coordinator. Even so, this expression can be parsed as a nested coordinate structure by the rules.

Non-terminals			
COORD			Coordination
CONJ			Conjunct
CC			Coordinating conjunction
CC-SUB			Sub-coordinator
W			Word
N			Non-coordination
S			Sentence
Rules for coordination			
(1)	COORD	→	CONJ N? CC N? CONJ
(2)	COORD	→	CONJ CC-SUB COORD
(3)	CONJ	→	COORD
(4)	CONJ	→	N
Rule for non-coordination			
(5)	S	→	COORD
(6)	S	→	N
(7)	N	→	COORD N
(8)	N	→	W COORD
(9)	N	→	W N
(10)	N	→	W
Rules for pre-terminals			
(11)	CC	→	(and|or|but|nor|and/or)
(12)	CC-SUB	→	(,|;|:)
(13)	W	→	*

Table 5.1: Production rules for coordinate trees. (...|...) represents one of the elements, and "*" represents any word. "?" indicates zero or one occurrence of the preceding element.

$score_{ckey}$ when a terminal belongs to the set of coordinator keys $\mathbb{S}_{cc}$ or the set of sub-coordinator keys $\mathbb{S}_{sub\text{-}cc}$; when a terminal does not belong to either $\mathbb{S}_{cc}$ or $\mathbb{S}_{sub\text{-}cc}$, the system assigns 0 to the pre-terminal because it is a node W and is not relevant to coordination. To summarize, the scoring function $score(T)$ is formulated as follows:

$$score(T) = \sum_{\langle [i,j],v \rangle \in T} score_{node}([i,j],v) \tag{5.3}$$

$$score_{node}([i,j],v) = \begin{cases} score_{coord}(i,j) & (v = \text{COORD}) \\ score_{ckey}(i,v) & (v \in \{\text{CC}, \text{CC-SUB}, \text{W}\}, \\ & \quad i = j, w_i \in \mathbb{S}_{cc} \cup \mathbb{S}_{sub\text{-}cc}) \\ 0 & (otherwise) \end{cases} \tag{5.4}$$

where $\langle [i,j],v \rangle$ is a node labeled as v in a parse tree T and spans from the i-th to the j-th word in a sentence, and w_i indicates the i-th word in the input sentence. The two scoring functions $score_{coord}$ for coordination nodes and $score_{ckey}$ for pre-terminals of coordinator keys and sub-coordinator keys are defined later.

The proposed method uses the CKY algorithm to produce parse trees for coordination from the CFG. To apply the algorithm, the CFG rules are transformed into the CNF.[2] The best scoring parse tree $\hat{T}$ resulting from Eq. 5.2 can be found efficiently using dynamic programming, as described in Section 5.2.2.

5.4.3 Pre- and post-processing

Commas and periods are moved inside the quotation marks in American English. This causes irregular coordinated phrases, such as ⟨... *"associations," "societies" and "councils"* ...⟩, which cannot be produced by rule (2) in Table 5.1 correctly. When applying rule (2) to such sentences, the conjuncts following sub-coordinators begin with closing quotations. This section proposes two pre- and post-processing methods to treat the irregular movement of punctuation marks.

[2]When a rule has more than two non-terminals, symbols on the right-hand side are transformed into binary rules from right to left.

Removing and recovering quotation

One way of pre- and post-processing quotation marks is to remove them before feeding a sentence to the system. At the same time, we keep a record of their positions and then insert them at their original positions after the system identifies coordinate structures. For example, when the system predicts the conjuncts as ⟨*... [A], [B] and [C], ...*⟩ for the given sentence ⟨*... "A," "B" and "C," ...*⟩, it recovers them as ⟨*... "[A]," "[B]" and "[C]," ...*⟩.

Swapping quotation and punctuation

Alternatively, it is possible to swap the positions of commas and the immediately following closing quotation marks before feeding a sentence to the system. We retain their original positions so that we can recover them when the system completes identification. For example, when the system returns the conjuncts as ⟨*... "[A]", "[B]" and "[C]", ...*⟩ for the given sentence ⟨*... "A," "B" and "C," ...*⟩, it recovers them as ⟨*... "[A]," "[B]" and "[C]," ...*⟩.

The proposed bottom-up approach adopts the swapping method for pre- and post-processing of quotation marks. This is because this method preserves quotation marks, and the system can make use of them as promising clues to identify conjuncts. In addition, this pre- and post-processing method does not affect British English text, which does not have irregular movement around quotations, whereas the removing method eliminates quotation marks even if no irregular movement occurs.

5.4.4 Parser models

This section defines the scoring functions of $score_{coord}$ and $score_{ckey}$ for the parsing algorithm described in the previous section; it also presents methods to train parameters associated with the scoring functions. The scoring functions consist of three submodels: *coordinator classification model*, *inner-boundary scoring model*, and *outer-boundary scoring model*. Figure 5.5 illustrates an overview of the proposed approach.

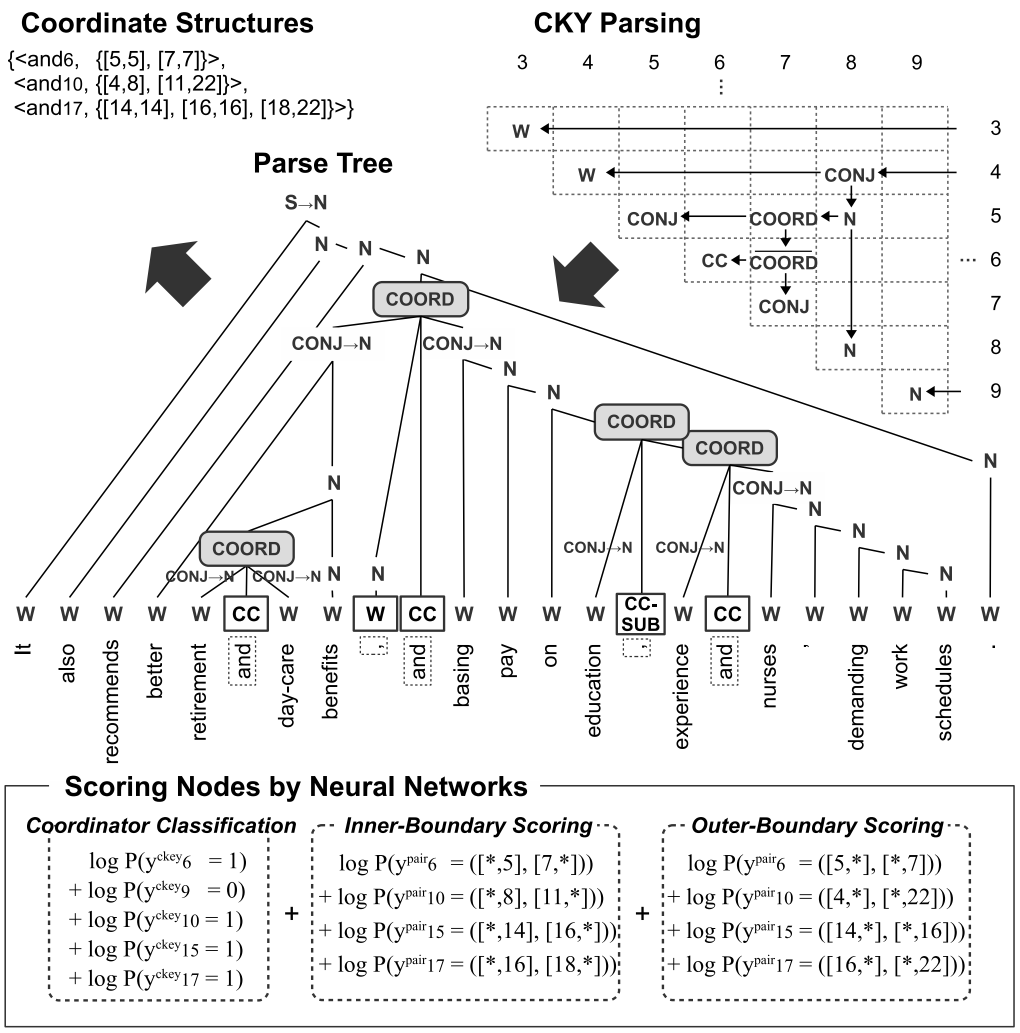

Coordinate Structures
{<and6, {[5,5], [7,7]}>,
<and10, {[4,8], [11,22]}>,
<and17, {[14,14], [16,16], [18,22]}>}
CKY Parsing
Parse Tree
S→N
COORD
CONJ→N
CC
CC-SUB
It also recommends better retirement and day-care benefits , and basing pay on education , experience and nurses , demanding work schedules .
Scoring Nodes by Neural Networks
Coordinator Classification
log P(y^ckey6 = 1) + log P(y^ckey9 = 0) + log P(y^ckey10 = 1) + log P(y^ckey15 = 1) + log P(y^ckey17 = 1)
Inner-Boundary Scoring
log P(y^pair6 = ([*,5], [7,*])) + log P(y^pair10 = ([*,8], [11,*])) + log P(y^pair15 = ([*,14], [16,*])) + log P(y^pair17 = ([*,16], [18,*]))
Outer-Boundary Scoring
log P(y^pair6 = ([5,*], [*,7])) + log P(y^pair10 = ([4,*], [*,22])) + log P(y^pair15 = ([14,*], [*,16])) + log P(y^pair17 = ([16,*], [*,22]))

Figure 5.5: Overview of the framework of the proposed bottom-up approach. The scores of rectangular nodes for pre-terminals are assigned by the coordinator classification model, and the scores of COORD nodes are assigned by the inner- and outer-boundary scoring models.

For the task of coordination disambiguation, a system must return a set of coordinate structures for a given sentence consisting of N words $w_{1:N} = w_1, \dots, w_N$:

$$\begin{aligned} X &= w_{1:N} \\ Y &= \big\{ \langle t, \{[b_t^{(k)}, e_t^{(k)}] \,|\, 1 \leq k \leq K_t\} \rangle \,|\, w_t \text{ is a coordinator} \big\} \\ &\quad (b_t^{(k)} \leq e_t^{(k)},\ K_t \geq 2) \end{aligned} \tag{5.5}$$

where t indicates the position of a coordinator, K_t is the number of conjuncts in the coordinate structure formed by the coordinator w_t, and $[b_t^{(k)}, e_t^{(k)}]$ is the k-th conjunct in the coordinate structure, which spans from the $b_t^{(k)}$-th to the $e_t^{(k)}$-th word in the sentence. K_t is greater than or equal to 2 because we cannot know in advance how many conjuncts belong to the coordinate structure formed by a coordinator w_k.

Alternatively, we could find pairs of conjuncts in a sentence by using coordinator and sub-coordinator keys:

$$\begin{aligned} X' &= \{w_{1:N}, C\} \\ C\ &= \{t \,|\, w_t \in \mathbb{S}_{cc} \cup \mathbb{S}_{\textit{sub-cc}}\} \\ Y' &= \{\langle y_t^{ckey}, y_t^{pair} \rangle \,|\, t \in C\} \end{aligned} \tag{5.6}$$

where y_t^{ckey} is a binary label indicating whether w_t is the actual (sub-)coordinator ($y_t^{ckey} = 1$) or not ($y_t^{ckey} = 0$), and y_t^{pair} is a pair of conjunct spans. $y_t^{pair} = \varnothing$ when $y_t^{ckey} = 0$ because there is no pair of conjuncts for w_t. When $t = 1$ or $t = N$, $y_t^{ckey} = 0$ because it does not form a coordinate structure within the sentence. The set of coordinator keys $\mathbb{S}_{cc}$ and the set of sub-coordinator keys $\mathbb{S}_{\textit{sub-cc}}$ are {“and”, “or”, “but”, “nor”, “and/or”} and {“,”, “;”, “:”}, respectively, as defined in Section 2.3.1.

To identify the four boundaries of y_t^{pair}—the beginnings and ends of the left and right conjuncts—the proposed approach uses two different models for the *inner boundary* (end of the left conjunct and beginning of the right conjunct) and the *outer boundary* (beginning of the left conjunct and end of the right conjunct). Dividing the four boundaries into two groups enables feasible enumeration because enumerating all possible inner and outer boundaries of y_t^{pair} has time complexity $\mathcal{O}(N^2) + \mathcal{O}(N^2) = \mathcal{O}(N^2)$, whereas enumerating all possible y_t^{pair} has time complexity $\mathcal{O}(N^4)$.

It should be noted that for division of the four boundaries, "two beginnings and two ends" or "left span and right span" can be chosen instead. However, in preliminary experiments, models for the division "left span and right span" performed poorly because they could not make use of the interaction of two spans and thus they could not capture the similarity and replaceability between two conjuncts. On the other hand, models for the division "two beginnings and two ends" performed relatively well because they could compare the beginnings/ends of conjuncts to measure the similarity and replaceability. In most cases, however, the end of the left conjunct and beginning of the right conjunct are next to the coordinator. Thus, the models tended to learn only the beginning of the left conjunct and end of the right conjunct independently, which might be one reason why they performed worse than those trained on inner and outer boundaries.

Coordinator classification model

The coordinator classification model is a binary classifier that predicts the label of a (sub-)coordinator key:

$$\begin{aligned} P_\theta(y_t^{ckey} = 1 \mid t) &= \frac{1}{1 + \exp\left(-\textsc{Score}_\theta^{ckey}(t)\right)} \\ P_\theta(y_t^{ckey} = 0 \mid t) &= 1 - P_\theta(y_t^{ckey} = 1 \mid t) \end{aligned} \tag{5.7}$$

where $\textsc{Score}_\theta^{ckey}$ is a scoring function for (sub-)coordinator keys, and θ is a set of model parameters. The training loss of the binary classification is computed by the following equation:

$$\ell_\theta^{ckey}(X', Y') = -\sum_{\langle y_t^{ckey}, y_t^{pair}\rangle \in Y'} \log P_\theta(y_t^{ckey} | t) \tag{5.8}$$

Inner-boundary scoring model

The inner-boundary scoring model assigns a score to a pair of conjunct spans on the basis of inner boundaries. b^l, e^l, b^r, and e^r denote the beginning of a left conjunct, end of a left conjunct, beginning of a right conjunct, and end of a right conjunct, respectively. The score of the inner-boundary pair (e^l, b^r) for a coordinator key w_t is assigned by a scoring function $\textsc{Score}_\theta^{inner}(e^l, b^r, t)$. The probabilities of the inner boundaries are normalized distributions over all possible inner-boundary pairs:

$$I_t = \big\{(1, t+1), (1, t+2), \ldots, (1, N), (2, t+1), \ldots, (t-1, N)\big\} \tag{5.9}$$

$$P_\theta(y_t^{pair} = \langle [*, e^l], [b^r, *] \rangle \mid t) = \frac{\exp\left(\textsc{Score}_\theta^{inner}(e^l, b^r, t)\right)}{\sum\limits_{(e'^l, b'^r) \in I_t} \exp\left(\textsc{Score}_\theta^{inner}(e'^l, b'^r, t)\right)} \tag{5.10}$$

$$\ell_\theta^{inner}(X', Y') = - \sum_{\langle y_t^{ckey}, y_t^{pair} \rangle \in Y'} y_t^{ckey} \log P_\theta(y_t^{pair} | t) \tag{5.11}$$

The term $y_t^{ckey} \log P_\theta(y_t^{pair}|t)$ means that the cross-entropy loss is activated only for positive coordinator keys ($y_t^{ckey} = 1$) and is deactivated otherwise ($y_t^{ckey} = 0$).

Outer-boundary scoring model

Similarly to the inner-boundary scoring model, for the outer-boundary pair (b^l, e^r), the probability $P_\theta(y_t^{pair} = \langle [b^l, *], [*, e^r] \rangle \mid t)$ is computed from the set of all outer-boundary pairs O_t; the loss is defined as ℓ_θ^{outer} using the scoring function $\textsc{Score}_\theta^{outer}(b^l, e^r, t)$ in the same way as ℓ_θ^{inner}. Note that I_t and O_t are identical because their possible pairs are the same. Using the inner pair probability $P_\theta(y_t^{pair} = \langle [*, e^l], [b^r, *] \rangle \mid t)$ and outer pair probability $P_\theta(y_t^{pair} = \langle [b^l, *], [*, e^r] \rangle \mid t)$, the most probable pair is produced by

$$\begin{aligned} y_t^{pair} = &\mathop{\arg\max}_{(\hat{e}^l, \hat{b}^r) \in I_t} P_\theta(\langle [*, \hat{e}^l], [\hat{b}^r, *] \rangle \mid t) \\ \cup &\mathop{\arg\max}_{(\hat{b}^l, \hat{e}^r) \in O_t} P_\theta(\langle [\hat{b}^l, *], [*, \hat{e}^r] \rangle \mid t) \end{aligned} \tag{5.12}$$

In practice, conjunct pairs are determined through the CKY algorithm because choosing the best pairs for an individual coordinator may cause a conflict with other coordinate structures.

5.4.5 Scoring

This section concretely defines the scoring functions $score_{coord}$ and $score_{ckey}$ presented in Section 5.4.2. When scoring pre-terminals for (sub-)coordinator keys

$w_t \in \mathbb{S}_{cc} \cup \mathbb{S}_{sub\text{-}cc}$, the log probability of each binary label y_t^{ckey} is assigned:

$$score_{ckey}(t, v) = \begin{cases} \log P_\theta(y_t^{ckey} = 1 \,|\, t) & (v \in \{\text{CC}, \text{CC-SUB}\}) \\ \log P_\theta(y_t^{ckey} = 0 \,|\, t) & (v = \text{W}) \end{cases} \tag{5.13}$$

The score of COORD is computed using the left and right conjuncts, which are linked by the CC. Assuming that a coordinator appears at the t-th position and the left and right spans are $[i, l]$ and $[m, j]$, respectively, the score of the coordinate structure $[i, j]$ formed by the two conjuncts linked by w_t is the sum of the log probabilities of the inner boundary (l, m) and outer boundary (i, j):

$$\begin{aligned} score_{coord}(i, j) &= \log P(y_t^{pair} = \langle [i, l], [m, j] \rangle \,|\, t) \\ &= \log P_\theta(\langle [*, l], [m, *] \rangle \,|\, t) + \log P_\theta(\langle [i, *], [*, j] \rangle \,|\, t) \end{aligned} \tag{5.14}$$

Rule (1) is represented as follows when we correspond the calculation of Eq. 5.14 with it:

$$\text{COORD}_{i,j} \rightarrow \ \text{CONJ}_{i,l} \ \ \text{N}_{l+1,t-1}? \ \ \text{CC}_{t,t} \ \ \text{N}_{t+1,m-1}? \ \ \text{CONJ}_{m,j}$$

When two conjuncts are linked by a sub-coordinator w_t, the score of COORD is calculated in the same way as Eq. 5.14 using the left and right spans $[i, l]$ and $[m, j]$, respectively. By applying Eq. 5.14 to rule (2), its right-hand side is expanded as follows:

$$\begin{aligned} \text{COORD}_{i,j} \rightarrow \ & \text{CONJ}_{i,l} \ \ \text{CC-SUB}_{t,t} \ \ \text{CONJ}_{m,j} \ \ \ldots \ \ \text{N? CC N? CONJ} \\ & (l = t - 1, m = t + 1) \end{aligned}$$

5.4.6 Model instantiations with neural networks

This section instantiates the three scoring functions $\textsc{Score}_\theta^{ckey}$, $\textsc{Score}_\theta^{inner}$, and $\textsc{Score}_\theta^{outer}$ with neural networks.

Encoder

Assume that the POS tags $p_{1:N} = p_1, \ldots, p_N$ for words $w_{1:N}$ are available. Sentence-level representations for a sequence of words and POS tags are obtained through bidirectional LSTMs (BiLSTMs):

$$\mathbf{h}_{1:N} = \text{BiLSTMs}(f_{input}(w_{1:N}, p_{1:N})) \tag{5.15}$$

The dimensionality of each resulting vector $\mathbf{h}_t$ is $2d^{hidden}$ when the dimensionality of a hidden vector from each LSTM is d^{hidden} because a vector $\mathbf{h}_t$ is computed as a concatenation of the two vectors encoded from the forward and backward directions by the LSTMs. For the BiLSTMs inputs, the function f_{input} maps words and POS tags onto their representations. We can use different word representations, including a pretrained word model, contextualized word embeddings (e.g., ELMo [86] or BERT [22]), and character-level LSTMs/CNNs. The use of different word representations is considered in Section 5.5. The entire network consisting of f_{input} and BiLSTMs is referred to as the encoder; it is shared by the three neural networks in the higher layer.

Coordinator classification model

The scoring function $\textsc{Score}_\theta^{ckey}$ consists of a linear transformation of the sentence-level representation of a coordinator key:

$$\textsc{Score}_\theta^{ckey}(t) = \mathbf{w}^{ckey}\mathbf{h}_t + \mathrm{b}^{ckey} \tag{5.16}$$

where $\mathbf{w}^{ckey} \in \mathbb{R}^{2d^{hidden}}$ and $\mathrm{b}^{ckey} \in \mathbb{R}$ are the model parameters of the classifier.

Inner-boundary scoring model

From the sentence-level representations produced by the encoder, the inner-boundary scoring model concatenates two representations of inner boundaries and then feeds the produced vector to an MLP:

$$\textsc{Score}_\theta^{inner}(e^l, b^r, t) = \mathbf{w}_2^{in}\,\mathrm{ReLU}(\mathbf{W}_1^{in}[\mathbf{h}_{e^l};\mathbf{h}_{b^r}] + \mathbf{b}_1^{in}) + \mathrm{b}_2^{in} \tag{5.17}$$

where $\mathbf{W}_1^{in} \in \mathbb{R}^{d^{in} \times 4d^{hidden}}$, $\mathbf{b}_1^{in} \in \mathbb{R}^{d^{in}}$, $\mathbf{w}_2^{in} \in \mathbb{R}^{d^{in}}$, and $\mathrm{b}_2^{in} \in \mathbb{R}$ are the parameters of the inner-boundary scoring model.

Outer-boundary scoring model

Using sentence-level representations, the outer-boundary scoring model takes two vectors calculated by subtracting the vectors adjacent to the coordinator from the boundary vectors. These subtraction operations are intended to capture the semantic distance and relatedness between two spans, as inspired by the operations for the similarity and replaceability features proposed in the top-down approach (Section 4.4). The model then passes the concatenated vector to an MLP:

$$\begin{aligned}&\textsc{Score}_{\theta}^{outer}(b^l, e^r, t) \\ &\quad= \mathbf{w}_2^{out}\,\mathrm{ReLU}(\mathbf{W}_1^{out}[\mathbf{h}_{b^l} - \mathbf{h}_{t+1}; \mathbf{h}_{e^r} - \mathbf{h}_{t-1}] + \mathbf{b}_1^{out}) + \mathrm{b}_2^{out}\end{aligned} \tag{5.18}$$

where $\mathbf{W}_1^{out} \in \mathbb{R}^{d^{out} \times 4d^{hidden}}$, $\mathbf{b}_1^{out} \in \mathbb{R}^{d^{out}}$, $\mathbf{w}_2^{out} \in \mathbb{R}^{d^{out}}$, and $\mathrm{b}_2^{out} \in \mathbb{R}$ are the parameters of the outer-boundary scoring model.

5.4.7 Learning

Training of the set of parameters θ in the neural networks is performed by minimizing the following loss function:

$$L(\theta) = \sum_{(X', Y') \in D} \left(\ell_{\theta}^{ckey}(X', Y') + \ell_{\theta}^{inner}(X', Y') + \ell_{\theta}^{outer}(X', Y')\right) \tag{5.19}$$

where D is a set of pairs of a sentence and the conjunct pairs within it in a training dataset. Thus, the three submodels are trained jointly.

Why local training?

Instead of learning the scoring functions on the basis of local decisions, we could directly train the models using a structured max-margin objective between the scores of the best predicted and gold trees, similarly to the work of Stern et al. [102]. In preliminary experiments, however, such global training requires careful hyperparameter tuning and is difficult to optimize stably, resulting in worse performance than those of models trained locally and the proposed top-down approach.

The success of local training for inner and outer boundaries is seemingly owed to the use of cross-entropy loss. Global training with a max-margin objective adjusts parameters against conjunct boundaries in the best predicted and gold trees. On the other hand, local training with cross-entropy loss performs well with not only the best and gold conjunct boundaries but all possible boundaries, and it lowers the scores for incorrect boundaries. To combine both training methods, suppose that we model a conditional probability $P(T|s)$ of a parse tree $T \in \mathcal{T}_G(s)$ for a given sentence s as follows:

$$P_\theta(T|s) = \frac{\exp\left(score_\theta(T)\right)}{Z_\theta(s)}, \quad Z_\theta(s) = \sum_{T' \in \mathcal{T}_G(s)} \exp\left(score_\theta(T')\right) \tag{5.20}$$

Updating the set of parameters θ by maximum likelihood estimation of the conditional probability $P(T|s)$ might achieve better results than that by local training; such training, however, is not feasible because the partition function Z requires scores to be computed for all possible parse trees that can be derived from the CFG G for the sentence s. Alternatively, partial global training can be done by approximating the partition function Z, as proposed in Andor et al. [2].

Recently, locally trained models for structured inference problems, such as constituency parsing [105] and dependency parsing [24], have achieved competitive performances without globally optimized training but with globally optimized inference. The proposed bottom-up model is similarly optimized by local training but employed with the CKY algorithm to produce the best parse tree. Thus, the proposed approach adopts the local training method partly because the performance difference between local and global training would be small if the proposed models learn the subtasks introduced in this chapter with relatively good accuracy.

5.5 Preliminary experiments

5.5.1 Settings

This section presents preliminary experiments on the coordination-annotated PTB [28]. Almost all settings were the same as those for the proposed top-down approach in Section 4.5. The MLPs in the inner- and outer-boundary scoring models have one hidden layer each with 600 units. The proposed method adopted the "swap" pre- and post-processing described in Section 5.4.3. The models were evaluated with precision, recall, and F1 measures for predicting the pre- and post- conjuncts for each coordinator. In addition, coordinate structures labeled as NP or NX were specifically evaluated, as in Section 4.5, because they are the most frequent types of coordination and likely to be similar in conjuncts.

	Development			Test		
	Pre.	Rec.	F1	Pre.	Rec.	F1
	All Coordination					
Ficler+16	72.34	72.25	72.29	72.81	72.61	72.7
top-down	75.08	76.76	75.91	72.85	74.97	73.90
bottom-up	79.29	79.48	**79.38**	78.17	78.52	**78.34**
	NP Coordination					
Ficler+16	75.17	74.82	74.99	76.91	75.31	76.1
top-down	79.08	78.71	78.89	79.36	78.98	79.16
bottom-up	81.42	81.23	**81.32**	80.86	80.73	**80.79**

Table 5.2: Performance of predicting pre- and post-conjuncts on all coordination and on NP coordination in the PTB.

5.5.2 Results

Table 5.2 presents the results. For all and just NP coordination, the proposed bottom-up method achieved better results than the method of Ficler and Goldberg [29] and the proposed top-down method. The bottom-up method was more accurate, especially for pre- and post-conjuncts, because it learns both the inner and outer boundaries of conjunct pairs, while the top-down method learns only the coordination boundaries and thus it is likely to retrieve wrong conjuncts due to the presence of false positive commas as dividers, as discussed in Section 4.5. For fair comparison, I further investigate the differences between the two methods in Chapter 6 from various perspectives.

An ablation study was conducted for the proposed model. As the baseline for the feature representation employed in the outer-boundary scoring model, the same representation as in the inner-boundary scoring model was used. Table 5.3 shows the results of the study. Without POS tags, the performance significantly dropped, as indicated for the top-down method in Section 4.5. The results also demonstrate that the model benefits from the pretrained word embedding for the task. I deduce that POS tags and morphological information are crucial for shorter and similar coordinated elements, such as in NP coordination, as also discussed in Section 4.5. To investigate the influence of accuracy of POS tagging,

	All Coordination			NP Coordination		
	Pre.	Rec.	F1	Pre.	Rec.	F1
baseline	77.85	77.94	77.90	80.45	80.09	80.27
default	79.29	79.48	79.38	81.42	81.23	81.32
-POS tags	76.17	76.17	76.17	77.06	76.88	76.97
-GloVe	77.23	77.59	77.41	79.12	78.94	79.03

Table 5.3: Performances with different settings for the PTB development set evaluated by pre- and post-conjuncts prediction.

the proposed model was newly trained with gold POS tags and then evaluated on the PTB development set. The model achieved 79.32 F1 score for all coordination, and the use of gold POS tags did not make a much difference. This might be because the Stanford POS Tagger used in these experiments achieved 96.7% accuracy (higher within NPs) on the PTB development set, and this performance is sufficient for coordination disambiguation. The default feature representation described in Eq. 5.18 performed better than the baseline feature because it is specifically designed to capture the similarity and replaceability of two spans, while the baseline representation has only the contextual information of the outer boundaries of a pair.

5.6 Summary

This chapter presented a simple and accurate bottom-up approach to coordination disambiguation. The proposed method decomposes the task into three subtasks and employs three different neural networks to tackle them. For inference, the CKY algorithm is applied with the CFG rules to produce globally consistent coordinate structures in a given sentence. Experimental results demonstrated that the locally trained models interoperate to obtain the optimal combination of coordinate structures and outperformed existing systems and the proposed top-down approach. The ability of the proposed bottom-up approach is explored in more detail in Chapter 6 through quantitative and qualitative analyses.

Chapter 6

Experiments and Analysis

This chapter reports experiments to evaluate the proposed top-down and bottom-up approaches for coordination disambiguation in a unified manner.

6.1 Experiments on the Penn Treebank

6.1.1 Settings

Dataset

Experiments reported in this section were conducted on the Penn Treebank (PTB) with the coordination annotation extension [28]. For the proposed bottom-up model, the "swap" method was applied for pre- and post-processing, as in Section 4.5. The dataset was split following the prior work of Ficler and Goldberg [29]: sections 02–21, 22, and 23 for the training, development, and test sets, respectively.

Model

Both of the proposed top-down and bottom-up models were closely examined to present the differences in their behaviors and characteristics. The base encoders for the two models were almost identical to those used in Sections 4.5 and 5.5, except that the character-level convolutional neural networks [64] (CharCNNs) were additionally used, and their produced vectors were concatenated to the word and POS tag embeddings. The settings of the CharCNNs were as follows: the dimensionality of character embeddings was 100, window size of the CNN was 3, and the size of produced representations from the CNN was 100. Dropout was not applied to the character embeddings. As pretrained word embeddings, the encoder employed GloVe [85]. POS tags were obtained using the Stanford POS Tagger [107] with 10-way jackknifing. Other hyperparameters were identical to those in the preliminary experiments.

As a baseline method, the constituency parser proposed by Stern et al. [102] was also evaluated on the same dataset. However, in the experimental settings described here, no constituency annotation was available; only coordination boundaries could be used for training. Thus, the baseline parser was trained on parse trees converted from the coordination annotation using the CFG rules described in Section 5.4.2.

Evaluation metrics

The models were evaluated on the basis of their ability to identify conjuncts for each coordinator with precision, recall, and F1 measures. The predicted conjuncts were judged to be correct by the following agreement criteria:[1]

- **whole**: Matches at the beginning of the first conjunct and end of the last conjunct.
- **outer**: Matches in the first and last conjuncts.
- **inner**: Matches in the two conjuncts adjacent to the coordinator.
- **exact**: Matches in all the conjuncts.

In addition, special attention was paid to the evaluation of NP coordination, particularly for the phrases of types NX and NP, which are considered as NP coordination following the prior work [29].

6.1.2 Results

Table 6.1 shows the experimental results. The proposed bottom-up approach outperformed the proposed top-down approach, baseline, and existing methods

[1]Inner and outer have the same criteria when a coordinate structure consists of two conjuncts.

		Development			Test		
		Pre.	Rec.	F1	Pre.	Rec.	F1
		All Coordination					
bottom-up	whole	78.73	79.00	78.86	77.93	78.43	78.18
	outer	77.79	78.06	77.92	76.40	76.88	76.64
	inner	79.43	79.71	79.57	78.66	79.16	78.91
	exact	**77.67**	**77.94**	**77.81**	**76.40**	**76.88**	**76.64**
top-down	whole	77.97	79.71	78.83	76.77	78.79	77.77
	outer	73.81	75.47	74.63	70.83	72.70	71.75
	inner	75.66	77.35	76.50	72.60	74.52	73.55
	exact	73.47	75.11	74.28	70.65	72.52	71.57
Ficler+16	inner	72.34	72.25	72.29	72.81	72.61	72.7
Stern+17	whole	77.61	71.51	74.44	74.54	66.97	70.55
	outer	75.67	69.72	72.58	72.91	65.50	69.01
	inner	77.61	71.51	74.44	73.82	66.33	69.87
	exact	75.29	69.36	72.20	71.79	64.50	67.95
		NP Coordination					
bottom-up	whole	80.68	80.32	80.50	79.10	78.98	79.04
	outer	80.00	79.63	79.81	77.67	77.54	77.60
	inner	81.14	80.77	80.96	79.90	79.77	79.84
	exact	**79.77**	**79.40**	**79.58**	**77.67**	**77.54**	**77.60**
top-down	whole	80.27	80.09	80.18	81.49	81.36	81.43
	outer	77.75	77.57	77.66	76.71	76.59	76.65
	inner	79.58	79.40	79.49	77.67	77.54	77.60
	exact	77.29	77.11	77.20	76.55	76.43	76.49
Ficler+16	inner	75.17	74.82	74.99	76.91	75.31	76.1
Stern+17	whole	79.80	74.65	77.14	76.97	71.68	74.23
	outer	78.32	73.27	75.71	76.11	70.88	73.40
	inner	80.54	75.34	77.85	76.46	71.20	73.73
	exact	77.58	72.58	75.00	74.39	69.28	71.74

Table 6.1: Results on the PTB. The numbers in the row “Ficler+16” are taken from their paper [29].

for all criteria. The bottom-up approach was more accurate than the top-down approach because the former learns both the inner and outer boundaries of conjunct pairs, including those for sub-coordinators, while the latter learns only the coordination boundaries. The constituency parser developed by Stern et al. [102], which was trained not on constituency trees but on parse trees converted from the proposed CFG, performed poorly for all metrics compared to the proposed methods. Nevertheless, when evaluating the parsing performance on parse trees, it achieved 97.88 F1 score on the development set and 97.63 on the test set for agreement on labeled spans. These performances came from many plain trees that consist of binaries of a pre-terminal W (any word) and non-terminal N (non-coordination) in most parts, which are irrelevant to coordination and can be easily predicted. On the contrary, COORD nodes cannot be easily identified because they do not appear so frequently, and the parser fails to capture two conjuncts in a COORD node.

6.1.3 Discussion

Sentence-level evaluation

The proposed approaches were investigated for the ability of their systems to predict all coordinate structures in a sentence precisely. Sentences were categorized on the basis of following five groups:

All: All sentences that have at least a single coordinate structure.

- **Simple**: Sentences that have only one coordinate structure consisting of two conjuncts.
- **Complex**: Sentences that are categorized as Consecutive or Multiple.
  - **Consecutive**: Sentences that have at least one coordinate structure consisting of more than two conjuncts.
  - **Multiple**: Sentences that have multiple coordinate structures.

Sentences categorized as "All" are the union of the mutually exclusive sets of Simple and Complex, while the sets of Consecutive and Multiple are not disjoint and thus have the nonempty intersection.

	Category	Development	Test
bottom-up	All	496 / 673 = **73.69**	629 / 873 = **72.05**
	· Simple	378 / 481 = **78.58**	476 / 609 = **78.16**
	· Complex	118 / 192 = **61.45**	153 / 264 = **57.95**
	· Consecutive	44 / 66 = **66.66**	60 / 96 = **62.50**
	· Multiple	86 / 146 = **58.90**	104 / 197 = **52.79**
top-down	All	469 / 673 = 69.68	584 / 873 = 66.89
	· Simple	361 / 481 = 75.05	438 / 609 = 71.92
	· Complex	108 / 192 = 56.25	146 / 264 = 55.30
	· Consecutive	41 / 66 = 62.12	56 / 96 = 58.33
	· Multiple	78 / 146 = 53.42	101 / 197 = 51.26

Table 6.2: Complete match rates of coordinate structures per sentence on the PTB.

Table 6.2 shows the complete match rates. On both the development and test sets, the proposed bottom-up approach achieved gains compared to the proposed top-down approach on Simple coordination sentences. This might reflect the difference in which boundaries the two approaches learn. The inner- and outer-boundary scoring models learn to predict four boundaries of two spans, whereas the top-down model predicts only two outer boundaries on Simple coordination sentences. Because an appositive or adverbial phrase can appear between a coordinator and its conjunct, it can easily cause an error when two conjuncts are assumed to be next to a coordinator. The bottom-up approach also outperformed the top-down approach on Consecutive and Multiple coordination sentences. The top-down approach predicts a coordination span and then splits it into conjunct spans. Therefore, it can mistakenly segment coordination spans when false sub-coordinators appear in a sentence. In contrast, the bottom-up approach ascertains whether sub-coordinating words are true sub-coordinators or not, which can lead to more robust production of Consecutive sentences.

6.2 Experiments on the GENIA Treebank

6.2.1 Settings

Dataset

To evaluate the effectiveness of the proposed approaches in another domain, the GENIA Treebank beta [54] (GENIA) was used for experiments. The pre- and post-processing for punctuation and quotation were not applied because no irregular movement is found in the corpus. The models were evaluated through five-fold cross-validation, as in the work of Hara et al. [39].

Model

Experiments on the GENIA were conducted for both the proposed top-down and bottom-up approaches, and the settings were essentially the same as those used in the PTB experiments. In addition to the two default models, a variant of the bottom-up model, referred to as *bottom-up:sim+repl*, was also evaluated; it employs the similarity and replaceability features to its outer-boundary scoring model, which were originally proposed for the top-down model. These feature representations are relatively richer than those used in the original outer-boundary scoring model but require higher computational costs. As pretrained word embeddings, the proposed models used BioASQ [108], which is composed of 200-dimensional word vectors. Following Hara et al., POS tags were obtained as annotated in the corpus. When computing the loss for a given mini-batch, an L2 regularization term was added to it with strength 0.0001. Other hyperparameters were identical to those in the experiments on the PTB.

Evaluation metrics

The performance was measured by recall values for predicting coordinate structures using the aforementioned four criteria (whole, outer, inner, and exact); previous studies, on the other hand, evaluated their systems only on the basis of the whole criterion. Also, the models were evaluated according to syntactic categories.

		NP	VP	ADJP	S	PP	UCP	SBAR	ADVP	Others	All
	#	2317	465	321	188	167	60	56	21	3	3598
bottom-up :sim+repl	whole	74.88	70.32	89.40	60.10	70.65	56.66	78.57	90.47	33.33	74.43
	outer	74.62	70.10	89.40	59.04	70.65	56.66	76.78	90.47	33.33	74.15
	inner	77.90	72.47	90.34	60.10	71.25	58.33	78.57	90.47	33.33	76.79
	exact	**74.62**	**70.10**	**89.40**	**59.04**	**70.65**	56.66	**76.78**	**90.47**	**33.33**	**74.15**
bottom-up	whole	72.20	70.96	86.29	56.91	59.28	61.66	76.78	85.71	33.33	71.84
	outer	71.98	70.10	86.29	56.91	59.28	61.66	75.00	85.71	33.33	71.56
	inner	75.18	71.82	86.91	57.97	59.88	61.66	76.78	85.71	33.33	74.01
	exact	71.98	**70.10**	86.29	56.91	59.28	**61.66**	75.00	85.71	**33.33**	71.56
top-down	whole	74.88	73.54	87.53	56.91	72.45	58.33	82.14	90.47	33.33	74.68
	outer	73.71	69.67	87.22	50.00	66.46	58.33	60.71	90.47	33.33	72.42
	inner	76.26	71.39	87.85	51.59	67.06	61.66	64.28	90.47	33.33	74.56
	exact	73.62	69.67	87.22	50.00	66.46	58.33	60.71	**90.47**	**33.33**	72.37
Ficler+16	whole	65.08	71.82	74.76	17.02	56.28	51.66	91.07	80.95	33.33	64.14
Hara+09	whole	64.2	54.2	80.4	22.9	59.9	36.7	51.8	85.7	66.7	61.5

Table 6.3: Results on the GENIA. The numbers in the rows "Ficler+16" and "Hara+09" are taken from their papers [29, 39].

6.2.2 Results

Table 6.3 shows the results. The proposed bottom-up model with the similarity and replaceability features achieved the best overall performance on the exact criterion. Both the bottom-up models performed better on the inner criterion, whereas the proposed top-down model was better on the whole criterion. This performance reflects the differences between the algorithms of the two systems. The bottom-up approach builds a coordinate tree in a bottom-up manner and predicts inner conjuncts accurately, whereas the top-down approach predicts the entire span of a coordinate structure and splits it into conjuncts in a top-down manner. This is why the top-down model cannot predict coordinated clauses labeled as "S," which are likely to be longer and contain non-coordinating commas. The shortcoming of the bottom-up approach is that the bottom-up parsing may cause errors due to wrong decisions in the early stages of the parsing; this was observed as poor performance on the whole criterion. Compared with the existing methods, both the top-down and bottom-up approaches performed well for both NP and VP coordination. The proposed similarity feature representations can

Category		All
bottom-up :sim+repl	All	1674 / 2508 = **66.74**
	· Simple	1147 / 1507 = **76.11**
	· Complex	527 / 1001 = **52.64**
	· Consecutive	190 / 358 = 53.07
	· Multiple	417 / 822 = **50.72**
bottom-up	All	1602 / 2508 = 63.87
	· Simple	1109 / 1507 = 73.58
	· Complex	493 / 1001 = 49.25
	· Consecutive	179 / 358 = 50.00
	· Multiple	381 / 822 = 46.35
top-down	All	1628 / 2508 = 64.91
	· Simple	1119 / 1507 = 74.25
	· Complex	509 / 1001 = 50.84
	· Consecutive	192 / 358 = **53.63**
	· Multiple	385 / 822 = 46.83

Table 6.4: Complete match rates of coordinate structures per sentence on the GENIA.

capture similar conjuncts, such as in NP coordination, while the replaceability feature representations appear to help the models identify non-similar conjuncts, such as in VP and S coordination.

6.2.3 Discussion

Sentence-level evaluation

The models were also evaluated for their ability to identify all conjuncts in a sentence completely. Table 6.4 shows the results. The proposed bottom-up approach performed well, especially when using the similarity and replaceability features. This might be because NP coordination is much more frequent than other types of coordination in the GENIA corpus, and the similarity feature works particularly well on NP coordination, as Table 4.3 shows in Chapter 4. The results also

1.	Input	Last week , the banking company said it purchased both Freedom Savings & Loan Association , Tampa , Fla. , and University Federal Savings Association of San Antonio , Texas , for $ 169.4 million .
	Output	[Freedom Savings & Loan Association] , [Tampa] , [Fla.] , and [University Federal Savings Association of San Antonio , Texas ,]
	Gold	[Freedom Savings & Loan Association , Tampa , Fla. ,] and [University Federal Savings Association of San Antonio , Texas ,]
2.	Input	Traders can vary their strategies and execute their orders in any one of them .
	Output	[vary their strategies] and [execute their orders]
	Gold	[vary their strategies] and [execute their orders in any one of them]

Table 6.5: Incorrect predictions by the proposed top-down approach.

support the consistent production of complex coordination by the bottom-up approach. At the same time, conjuncts in NP coordination are easily identified by splitting a coordination span, which results in the relatively good recall value for the proposed top-down approach on Consecutive sentences.

6.3 Analysis

6.3.1 Qualitative evaluation

Top-down approach

Table 6.5 shows errors produced by the proposed top-down approach. In Example 1, the system successfully identified the correct coordination span but mistakenly divided it into wrong conjuncts. This type of mistake is observed everywhere, but often in NPs, such as those involved in apposition ("[Cara Operations, a food services concern], and [. . .]), relative clauses ("[Texas Air Corp., which owns Continental], and [. . .]"), dates ("[. . . Jan. 18, 1990], and [. . .]), locations ("[Columbus, Ohio], and [. . .]), and other types of named entities ("[Goldman, Sachs & Co.], and [. . .]"). In Example 2, the model failed to include the PP in the post-conjunct because it supposed that the PP modified both VPs. This type of error, known as a *PP attachment*, is one of the most difficult issues in

3.	Input	In addition to a general slowing of the computer industry , NCR , which sells automated teller machines and computerized cash registers , is also affected by the retail and financial sectors , “ areas of the economy that have generally not been robust , ” notes Sanjiv G. Hingorani , an analyst for Salomon Brothers Inc .
	Output	[teller machines] and [computerized cash registers] ; [retail] and [financial]
4.	Input	While profitable , it “ was n’t growing and was n’t providing a satisfactory return on invested capital , ” he says .
	Output	[was n’t growing] and [was n’t providing a satisfactory return on invested capital]
5.	Input	The Dow Jones Industrial Average jumped sharply yesterday to close at 2657.38 , panic did n’t sweep the world ’s markets , and investors large and small seemed to accept Friday ’s dizzying 190-point plunge as a sharp correction , not a calamity .
	Output	[The Dow Jones Industrial Average jumped sharply yesterday to close at 2657.38] , [panic did n’t sweep the world ’s markets] , and [investors [large] and [small] seemed to accept Friday ’s dizzying 190-point plunge as a sharp correction , not a calamity] .

Table 6.6: Correct predictions by the proposed bottom-up approach.

syntactic parsing.

Bottom-up approach

Tables 6.6 and 6.7 present example outputs of coordinate structures identified by the proposed bottom-up approach. Example 3 shows that the system identified the two independent coordinate structures. For the latter coordinate structure, the system correctly predicted ADJP coordination of “retail” and “financial” but not NP coordination of “the retail” and “financial sectors”. In Example 4, the system successfully captured imbalanced VP coordination of constituents governed by intransitive and transitive verbs. In Example 5, the system identified coordinated simple clauses, one of which embeds another coordinate structure. In Example 6, however, the system expected that the PP “at prevailing market

6.	Input	The company said it will buy additional shares “ from time to time in the open market or in private transactions at prevailing market prices . ”
	Output	[in the open market] or [in private transactions at prevailing market prices]
	Gold	[in the open market] or [in private transactions]
7.	Input	Dealers said morning activity was hectic as prices dropped in response to gains in the stock market and losses in Treasury securities , but trading slowed to moderate levels in the afternoon .
	Output	[Dealers said morning activity was hectic as prices dropped in response to [gains in the stock market] and [losses in Treasury securities]] , but [trading slowed to moderate levels in the afternoon] .
	Gold	Dealers said [morning activity was hectic as prices dropped in response to [gains in the stock market] and [losses in Treasury securities]] , but [trading slowed to moderate levels in the afternoon] .

Table 6.7: Incorrect predictions by the proposed bottom-up approach.

prices” was included by the right conjunct. The system could not identify the correct head “buy” of the PP, while the predicted conjunct was not easily refused because it seemed to be a valid constituent. In Example 7, the system predicted the nested NP coordinate structure correctly but failed to identify the correct clausal coordination. This might be caused by the omission of the clause marker “that”.

As seen in the above examples, the system is not good at determining whether phrases around coordination are included or not by the coordinate structure, while predicted conjuncts by the system tend to be valid as phrases. The bottom-up method must be improved to consider whether the conjuncts have the same relation (or dependency) to peripheral phrases.

Prediction on linguistic phenomenon involved in coordination

Table 6.8 demonstrates the system outputs of the proposed bottom-up approach for sentences in the PTB where linguistically unique phenomena occur. In Examples 8 and 9, every pair of conjuncts has a different type of syntactic category, all

8.	Gold (unlike cat.)	NCR said revenue declined both [in the U.S.] and [overseas] , reflecting a world-wide softening of the computer markets .
	Output ✓	[in the U.S.] and [overseas]
9.	Gold (unlike cat.)	America West , though , is [a smaller airline] and therefore [more affected by the delayed delivery of a single plane than many of its competitors would be] .
	Output ✓	[a smaller airline] and therefore [more affected by . . . would be] .
10.	Gold (non-constit.)	The magazine will reward with " page bonuses " advertisers who in 1990 [meet] or [exceed] their 1989 spending , as long as they [spent $ 325,000 in 1989] and [$ 340,000 in 1990] .
	Output ✗	[meet] or [exceed] ; [$ 325,000 in 1989] and [$ 340,000 in 1990]
11.	Gold (RNR)	The indicator [reached a peak in January 1929] and [then fell steadily [up to] and [through] the crash] .
	Output ✗	The indicator [reached a peak in January 1929] and [then fell steadily up [to] and [through] the crash] .
12.	Gold (RNR)	" I just do n't feel that the company [can really stand] or [would want] a prolonged walkout , " Tom Baker , president of Machinists ' District 751 , said in an interview yesterday .
	Output ✗	[can really stand] and [would want a prolonged walkout]
13.	Gold (gapping)	Among other [Asian] and [Pacific] markets , [Malaysia] and [Singapore] had the biggest losses , with [the Kuala Lumpur composite index in Malaysia falling 11.5 %] and [Singapore 's Straits Times Industrial Index down 10 %] .
	Output ✓	[Asian] and [Pacific] ; [Malaysia] and [Singapore] ; [the Kuala Lumpur composite index in Malaysia falling 11.5 %] and [Singapore 's Straits Times Industrial Index down 10 %]
14.	Gold (gapping)	PacifiCare Health Systems Inc. , proposed offering of 1.5 million common shares , of which [700,000 shares will be offered by PacifiCare] and [800,000 shares by UniHealth America Inc .] -LRB- PacifiCare 's 71 % -RRB- , via [Dillon , Read & Co. Inc.] , [Goldman , Sachs & Co.] and [Dean Witter Reynolds Inc] .
	Output ✗	[700,000 shares will be offered by PacifiCare] and [800,000 shares] ; [Dillon , Read & Co. Inc.] , [Goldman , Sachs & Co.] and [Dean Witter Reynolds Inc]
15.	Gold (gapping)	[Honeywell 's contract totaled $ 69.7 million] , and [IBM 's $ 68.8 million] .
	Output ✓	[Honeywell 's contract totaled $ 69.7 million] , and [IBM 's $ 68.8 million] .

Table 6.8: Predictions for linguistically unique examples.

of which were correctly identified by the system. The coordinate structure in Example 8 consists of a PP and ADVP, whereas that in Example 9 consists of an NP and ADJP. Example 10 has a coordinate structure, one of whose conjuncts does not qualify as a constituent. In the annotated corpus, two VPs *spent $ 325,000 in 1989* and *$ 340,000 in 1990* are considered to be in parallel, while the system returned *$ 325,000 in 1989* and *$ 340,000 in 1990* as coordinated elements. The verb *spent* is included in the annotated conjunct because the annotation was conducted on the treebank and thus conjuncts are forced to agree with the originally annotated phrase structures. It might be more natural to consider the system output as correct. In Examples 11 and 12, right node raising is observed; the NP *the crash* is an element shared by *up to* and *through* in Example 11, where the system recognized that *the crash* is shared but failed to include *up* as part of the conjunct, and the NP *a prolonged walkout* in Example 12 is an argument shared by *can really stand* and *would want*, but the system did not recognize the shared argument. The remaining of examples involve gapping. The system successfully identified conjuncts even though the head elements *falling* and *contract totaled* are elided in the second conjuncts in Examples 13 and 15, respectively. In Example 14, on the other hand, the system mistakenly excluded *by UniHealth America Inc.* due to the ellipsis.

6.3.2 Use of contextualized word embeddings

This section investigates the effectiveness of contextualized word embeddings. Contextualized word embeddings are beneficial to many NLP tasks because they encode contextual information into word representations, which can model the polysemy or context-sensitive meaning of words. The encoder accompanied with CharCNNs in the proposed top-down and bottom-up models additionally employed the most successful and widely used contextualized embeddings: BERT [22] and ELMo [86].[2] The top-down and bottom-up models with the extended encoder

[2]The following models were used:

- BERT (PTB): `https://storage.googleapis.com/bert_models/2018_10_18/uncased_L-24_H-1024_A-16.zip`
- ELMo (PTB): `https://s3-us-west-2.amazonaws.com/allennlp/models/elmo/2x4096_512_2048cnn_2xhighway/elmo_2x4096_512_2048cnn_2xhighway_weights.hdf5`
- BERT (GENIA): `https://s3-us-west-2.amazonaws.com/ai2-s2-research/scibert/tensorflow_models/scibert_scivocab_uncased.tar.gz`
- ELMo (GENIA): `https://s3-us-west-2.amazonaws.com/allennlp/models/elmo/contributed/pubmed/elmo_2x4096_512_2048cnn_2xhighway_weights_PubMed_only.hdf5`

were trained on the PTB and GENIA with the identical settings to those in Sections 6.1 and 6.2.

Tables 6.9 and 6.10 show the results on the PTB test set and the GENIA, respectively. For both approaches, BERT and ELMo gave significant performance gains, while BERT contributed much more than ELMo. One of differences between BERT and ELMo lies in their architectures; ELMo embeddings were trained with a BiLSTM encoder, whereas BERT embeddings were trained with a Transformer [110]. Because the top-down and bottom-up models employ BiLSTMs already, the models might benefit more from the Transformer, whose attention mechanism can serve as an alignment, as used in the methods of Kurohashi and Nagao [60] and Shimbo and Hara [100].

Tables 6.11 and 6.12 show the complete match rates on the PTB and GENIA, respectively. Both BERT and ELMo improved the performance of the proposed bottom-up model, especially on Complex sentences. This is partly because the contextualized embeddings help the model learn individual pairs of conjuncts and the bottom-up algorithm repeatedly inspects each pair. On the other hand, the proposed top-down model could predict coordination spans with great accuracy when using the contextualized embeddings, but they did not improve partitioning by the top-down approach; thus, identifying individual conjuncts remains problematic for that approach.

The top-down model with contextualized embeddings performed better than the bottom-up model without contextualized embeddings. This might give the impression that the top-down model with contextualized embeddings is sufficient and the structural constraints used in the bottom-up approach are unnecessary. However, the experimental results contradict this hypothesis; contextualized embeddings did achieve performance gains on the bottom-up approach as well. This indicates that the structural constraints in the bottom-up approach and contextualized embeddings can help the models solve different problems in different ways. Table 6.13 shows examples that the plain bottom-up model successfully solved but the top-down model with BERT still failed to solve, and vice versa. The bottom-up model performed better, especially on sentences containing structurally complex coordinate structures, whereas the contextualized embeddings play an important role to disambiguate conjunct boundaries that are particularly

		All Coordination				NP Coordination			
		Pre.	Rec.	F1	diff.	Pre.	Rec.	F1	diff.
	whole	85.27	85.89	85.58	+7.40	87.87	87.73	87.80	+8.76
bottom-up	outer	83.55	84.16	83.86	+7.22	86.28	86.14	86.21	+8.61
+BERT	inner	85.00	85.62	85.31	+6.40	87.55	87.42	87.49	+7.65
	exact	**83.55**	**84.16**	**83.86**	+7.22	**86.28**	**86.14**	**86.21**	+8.61
	whole	82.70	83.53	83.11	+4.93	85.03	85.03	85.03	+5.99
bottom-up	outer	81.08	81.89	81.48	+4.84	83.59	83.59	83.59	+5.99
+ELMo	inner	82.88	83.71	83.29	+4.38	85.03	85.03	85.03	+5.19
	exact	81.08	81.89	81.48	+4.84	83.59	83.59	83.59	+5.99
	whole	84.73	86.89	85.80	+8.03	89.95	89.80	89.88	+8.45
top-down	outer	77.19	79.16	78.16	+6.41	84.52	84.39	84.46	+7.81
+BERT	inner	78.26	80.25	79.24	+5.69	85.32	85.19	85.25	+7.65
	exact	76.92	78.88	77.89	+6.32	84.37	84.23	84.30	+7.81
	whole	82.97	85.16	84.05	+6.29	88.19	88.05	88.12	+6.69
top-down	outer	76.24	78.25	77.23	+5.48	83.09	82.96	83.02	+6.37
+ELMo	inner	77.57	79.61	78.58	+5.03	83.89	83.75	83.82	+6.22
	exact	76.06	78.07	77.05	+5.48	82.93	82.80	82.86	+6.37

Table 6.9: Results with contextualized embeddings on the PTB.

		NP	VP	ADJP	S	PP	UCP	SBAR	ADVP	Others	All	diff.
	#	2317	465	321	188	167	60	56	21	3	3598	-
bottom-up	whole	83.72	78.49	93.76	80.31	78.44	68.33	89.28	95.23	0.00	83.35	+8.92
:sim+repl	outer	83.64	78.27	93.76	80.31	78.44	68.33	89.28	95.23	0.00	83.26	+9.12
+BERT	inner	85.02	78.92	94.08	81.38	79.64	68.33	91.07	95.23	0.00	84.40	+7.62
	exact	**83.64**	**78.27**	93.76	**80.31**	**78.44**	**68.33**	**89.28**	**95.23**	0.00	**83.26**	+9.12
bottom-up	whole	81.39	75.91	92.83	69.14	76.64	60.00	89.28	90.47	33.33	80.62	+6.20
:sim+repl	outer	81.35	75.91	92.83	69.14	76.64	60.00	87.50	90.47	33.33	80.57	+6.42
+ELMo	inner	83.77	78.06	93.45	71.27	77.24	60.00	87.50	90.47	33.33	82.60	+5.81
	exact	81.35	75.91	92.83	69.14	76.64	60.00	87.50	90.47	**33.33**	80.57	+6.42
	whole	83.94	78.70	94.70	79.78	80.23	65.00	87.50	95.23	0.00	83.57	+8.89
top-down	outer	82.30	74.19	94.39	69.68	73.65	65.00	67.85	95.23	0.00	80.76	+8.34
+BERT	inner	83.08	74.83	94.39	72.34	73.65	65.00	71.42	95.23	0.00	81.54	+6.98
	exact	82.21	74.19	**94.39**	69.68	73.65	65.00	67.85	**95.23**	0.00	80.71	+8.34
	whole	82.77	77.41	93.45	72.34	79.64	61.66	94.64	90.47	33.33	82.18	+7.50
top-down	outer	81.35	73.33	93.14	62.76	72.45	61.66	73.21	90.47	33.33	79.54	+7.12
+ELMo	inner	82.90	74.19	93.45	63.82	72.45	63.33	75.00	90.47	33.33	80.79	+6.23
	exact	81.26	73.33	93.14	62.76	72.45	61.66	73.21	90.47	**33.33**	79.48	+7.12

Table 6.10: Results with contextualized embeddings on the GENIA.

	Category	+BERT	diff.	+ELMo	diff.
bottom-up	All	705 / 873 = **80.75**	+8.71	683 / 873 = 78.23	+6.19
	· Simple	521 / 609 = **85.55**	+7.39	496 / 609 = 81.44	+3.28
	· Complex	184 / 264 = 69.69	+11.74	187 / 264 = **70.83**	+12.88
	· Consecutive	73 / 96 = **76.04**	+13.54	71 / 96 = 73.95	+11.46
	· Multiple	128 / 197 = 64.97	+12.18	133 / 197 = **67.51**	+14.72
top-down	All	644 / 873 = 73.76	+6.87	639 / 873 = 73.19	+6.30
	· Simple	485 / 609 = 79.63	+7.72	477 / 609 = 78.32	+6.40
	· Complex	159 / 264 = 60.22	+4.92	162 / 264 = 61.36	+6.06
	· Consecutive	61 / 96 = 63.54	+5.21	60 / 96 = 62.50	+4.17
	· Multiple	111 / 197 = 56.34	+5.08	114 / 197 = 57.86	+6.60

Table 6.11: Complete match rates of coordinate structures per sentence on the PTB with the use of contextualized embeddings.

	Category	+BERT	diff.	+ELMo	diff.
bottom-up :sim+repl	All	1949 / 2508 = **77.71**	+10.96	1861 / 2508 = 74.20	+7.46
	· Simple	1256 / 1507 = **83.34**	+7.23	1233 / 1507 = 81.81	+5.71
	· Complex	693 / 1001 = **69.23**	+16.58	628 / 1001 = 62.73	+10.09
	· Consecutive	263 / 358 = **73.46**	+20.39	226 / 358 = 63.12	+10.06
	· Multiple	546 / 822 = **66.42**	+15.69	503 / 822 = 61.19	+10.46
top-down	All	1869 / 2508 = 74.52	+9.61	1824 / 2508 = 72.72	+7.81
	· Simple	1229 / 1507 = 81.55	+7.30	1199 / 1507 = 79.56	+5.31
	· Complex	640 / 1001 = 63.93	+13.09	625 / 1001 = 62.43	+11.59
	· Consecutive	246 / 358 = 68.71	+15.08	236 / 358 = 65.92	+12.29
	· Multiple	489 / 822 = 59.48	+12.65	477 / 822 = 58.02	+11.19

Table 6.12: Complete match rates of coordinate structures per sentence on the GENIA with the use of contextualized embeddings.

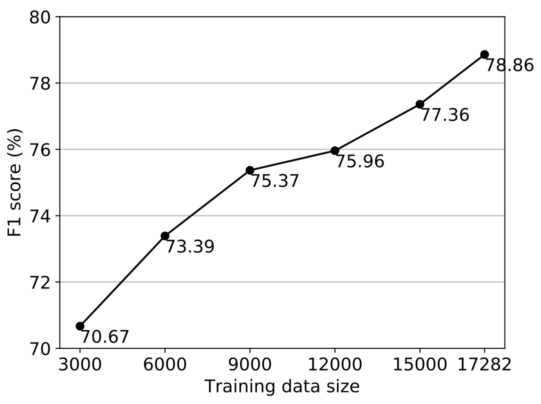
16.	top-down +BERT ✗	[Baseball] , [that game of the long haul] , [is the quintessential sport of the mean] , and [the mean ol' law caught up with the San Francisco Giants in the World Series last weekend] .
	bottom-up ✓	[Baseball , that game of the long haul , is the quintessential sport of the mean] , and [the mean ol' law caught up with the San Francisco Giants in the World Series last weekend] .
17.	top-down +BERT ✗	" [They 've been laggard] , [" he says] , ["] but [they 'll have to become more aggressive] . "
	bottom-up ✓	" [They 've been laggard] , " he says , " but [they 'll have to become more aggressive] . "

(a) Errors from the proposed top-down approach with BERT fixed by the plain proposed bottom-up approach.

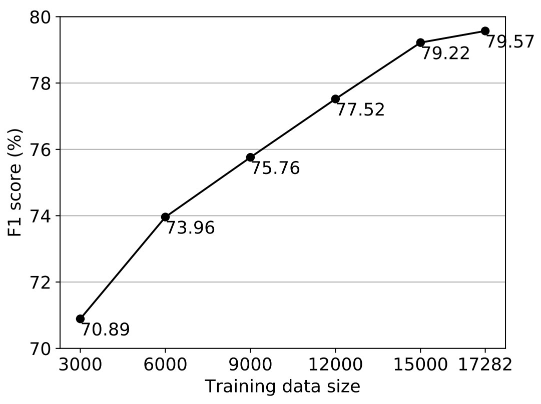
18.	bottom-up ✗	The rationale is that an interruption of trading will allow investors to reconsider their [strategies] , [calm sellers] and [lead buyers] to enter the market at indicated new price levels .
	top-down +BERT ✓	The rationale is that an interruption of trading will allow investors to [reconsider their strategies] , [calm sellers] and [lead buyers to enter the market at indicated new price levels] .
19.	bottom-up ✗	They combined for [25 hits] , [six home runs] and [24 runs batted in in the five games against the Cubs] .
	top-down +BERT ✓	They combined for [25 hits] , [six home runs] and [24 runs batted in] in the five games against the Cubs .

(b) Errors from the plain proposed bottom-up approach fixed by the proposed top-down approach with BERT.

Table 6.13: Different outputs between the proposed top-down approach with BERT and plain bottom-up approach.



(a) The top-down approach (whole).



(b) The bottom-up approach (inner).

Figure 6.1: F1 scores by different training data sizes.

sensitive to contexts, such as those involved in prepositional phrases (e.g., "NP and NP PP") and adverbial phrases (e.g., "VP and VP ADVP").

6.3.3 Impact of training dataset size

In the proposed top-down and bottom-up approaches, parameters of the neural networks are optimized through supervised learning. Training neural networks generally requires many training examples. This section investigates how the number of training examples affects the performances of the proposed approaches. Figure 6.1 shows the F1 scores of the proposed models for different training data sizes. In this experiment, training examples were randomly selected from the PTB training set, which has 17,282 sentences containing at least one coordinator key. Both models could generalize learned examples and predict coordination/conjunct spans for unseen data, even if they were trained on a small number of examples. This was also observed in the experimental results on the PTB (17,282 sentences) and the GENIA (80% of 2,508 sentences through five-fold cross-validation). Conversely, it might be difficult to improve the generalization performance just by increasing the number of training examples. This is partly because annotating coordinate structures is difficult for humans, and maintaining good quality consistently is an issue, as discussed in Section 2.4. The performance gains by contextualized word embeddings shown in Tables 6.9 and 6.10 indicate that exploiting implicit knowledge acquired through a huge amount of unlabeled data is an easier way to improve performance for coordination disambiguation.

Chapter 7

Conclusion

7.1 Summary

This dissertation proposed computational systems for coordination disambiguation in natural language.

Chapter 2 discussed the fundamentals of coordination, including its structural characteristics, unique behavior discussed in the linguistics literature, and orthography in English. This chapter also exemplified problems in syntactic parsing and introduce the task that this dissertation focused on.

Chapter 3 described prior research related to coordination. In the field of computational linguistics, many efforts have been made to deal with coordination in rule-based syntactic analysis systems. Due to advancements in statistical and machine learning methods for NLP, automatic methods for coordination disambiguation have been developed, many of which focus on the similarity between coordinated elements. This chapter also discussed the issues of other NLP tasks, including syntactic parsing, named entity recognition, and machine translation.

Chapter 4 proposed a top-down approach to coordination disambiguation. This approach attempts to detect the region of a coordinate structure as a whole and then retrieve individual conjuncts from it. The experimental results presented in this chapter demonstrated that the top-down method outperformed the existing best method with minimal use of external resources by using neural networks that were invented to exploit the similarity and replaceability properties between conjuncts.

Chapter 5 proposed a bottom-up parsing algorithm-based approach with production rules for coordination. Unlike the top-down approach, the bottom-up approach identifies the conjunct pair for each coordinating word and gradually constitutes coordinate structures so that they do not conflict with each other. This chapter suggested that the bottom-up approach performs better than the top-down approach.

Chapter 6 reported experiments on the two proposed approaches. Quantitative results indicated that both models successfully learned boundaries in coordinate structures, but the bottom-up model performed better by considering the structural constraints on coordination. Qualitative analysis revealed that the bottom-up model suggests coordinated elements whose boundaries seem to agree with phrasal boundaries and thus they can naturally connect to forward and backward contexts. This chapter concluded that there still remain ambiguities of context-dependent boundaries, such as which contextualized embeddings are not helpful enough to disambiguate and those even a human cannot easily solve.

7.2 Future work

The work described in this dissertation highlights the need for further work in the following areas.

Better conjunct segmentation

It is apparent that the drawback of the proposed top-down approach is straightforward division by the appearance of commas. As seen in Chapter 6, the top-down approach mistakenly divides whole coordination spans, although the model is able to predict those spans with good accuracy. With a better method for conjunct segmentation, accuracy on the outer, inner, and exact criteria would be greatly improved. One possible solution is to use a classification model for sub-coordinator keys, as used in the proposed bottom-up approach.

Reranking parse trees

The proposed top-down and bottom-up approaches significantly improve performances on the task of coordination disambiguation. However, they are still far from "perfect", even with contextualized embeddings, and there is room for further improvements. One possible method is reranking the n-best parse trees produced by the CKY algorithm using richer neural network models. As shown in Section 6.3, the bottom-up method returns fairly promising candidates. In fact, when retrieving the correct parse tree from an n-best list on the PTB development set, 84.08% ($n = 2$), 86.79% ($n = 4$), and 87.73% ($n = 8$) of correct coordinate structures were found (without using contextualized word embeddings); these numbers indicate the upper bound of the recall value on the exact criterion by reranking candidate trees. This indicates that re-evaluating n-best candidates with additional scoring would lead to better results.

Data augmentation and generation

One way to enhance the generalization capability of a neural network is to feed it more training data. However, annotating labels to text for supervised methods is generally laborious and costly. To remedy this issue, the similarity and replaceability properties of conjuncts could be used. Sentences with coordinate structures can be generated from sentences with no coordination by conjoining two or more similar phrases. Also, one conjunct can be replaced with a similar element taken outside of a sentence when conjunct boundaries are annotated. Similarly, conjuncts can exchange their positions in a sentence, which is a kind of data augmentation techniques, such as rotating or flipping images in computer vision tasks. For success in data generation and/or augmentation, keeping artificial data as high quality as possible is fairly important and requires development.

More substantial evaluation

Disambiguating coordination is a very difficult task even for humans, and the coordination annotations in the PTB and GENIA do not reflect such difficulties because there exist no confidence measures for labeling. Some coordinate structures are obvious, but others are relatively ambiguous, especially when a PP or ADVP appears around coordinated NPs and VPs. Given the ambiguities and potential lower agreements in the annotated conjunct spans, the performance evaluated on the gold labels does not always reflect the true coordination structures intended by the original authors or speakers. Thus, it seems to be more meaningful to evaluate how largely a predicted span overlaps with the corresponding gold span, for example, by using the Jaccard similarity coefficient.

Working on other languages

The proposed top-down and bottom-up approaches focus on English text. Although the two methods are not limited to a specific language or domain, small changes are needed when applying them to other languages. Specifically, the proposed top-down and bottom-up approaches require the following:

- Input texts are split into sentences that have already been segmented into words.
- Words that can be (sub-)coordinators are defined.

For the use of the bottom-up approach, the following condition is also necessary:

- The forms of coordinate structures are defined as CFG rules.

The two approaches can be applied to any language or domain given that these requirements are satisfied. Training the proposed neural networks requires many sentences with explicitly annotated conjunct boundaries. The GENIA corpus used in Chapter 6, for instance, contains 3,598 coordinate structures within 2,508 sentences, which indicates that at least more than a few thousand sentences with coordination should be prepared for training. In addition, POS tags or pretrained word embeddings would be beneficial in terms of accuracy, even though the models in the two proposed approaches do not necessarily require them.

For example, consider that applying the proposed bottom-up approach to Japanese language under the conditions above. "Balanced Corpus of Contemporary Written Japanese" [65] (BCCWJ; 『現代日本語書き言葉均衡コーパス』) is a corpus comprised of Japanese texts collected from a variety of genres, such as general books and magazines, newspapers, business reports, blogs, internet forums, textbooks, and legal documents. This corpus contains 14,368 coordinate structures within 57,109 sentences in some parts of the data [3], where coordination boundaries based on (*short unit*) words are explicitly marked. From these annotations and morphological information including POS, we can define words that could be coordinators and sub-coordinators. Unlike the English language, a coordinate structure with more than two conjuncts often appears in the form "[A] と [B] と [C]" which is equivalent to "[A] and [B] and [C]" in English. This requires an additional CFG rule similar to rule (2) in Table 5.1, such as "COORD $\rightarrow$ CONJ CC COORD". Therefore, the three requirements are satisfied and the proposed bottom-up approach as well as the proposed top-down approach could be applied to Japanese texts.

Specializing in specific domains

The experiments in this dissertation were only conducted on corpora collected from news articles and biomedical abstracts. The proposed top-down and bottom-up approaches can essentially be applied to texts in any domain, but additional changes are required to cover broader forms of coordination in some domains. In scientific papers, for example, coordinated elements are conjoined by special punctuation marks, such as – and /, and these should be treated in the conjunct segmentation for the top-down approach and the CFG rules and coordinator classification for the bottom-up approach. Because such markers usually connect noun phrases, we could devise special treatments for coordinate structures formed by them for easier and more accurate identification. It also would be interesting to use an explicit hierarchy of coordinate structures in some language or domain. In Japanese legal text, for example, disjunctive coordinators "又は" and "若しくは" are equivalent to "or" in English, but coordinate structures formed by "若しくは" are always embedded in other larger coordinate structures. In other words, a sequence of "A 又は B 若しくは C" must be interpreted as "(A 又は (B 若しくは C))". The same hierarchy is observed in conjunction; "並びに" is always superior to "及び", both of which are conjunctive coordinators equivalent to "and" in English. Such domain-specific conventions can be considered in the decoding by the top-down approach and as CFG rules in the bottom-up approach.

Application to other NLP tasks

The work in this dissertation focused only on coordination disambiguation. The performance on the tasks was significantly improved by the proposed approaches, but how can we leverage this achievement to solve “real” problems? Possible applications of coordination disambiguation include integration of the task and other NLP tasks, such as syntactic parsing, named entity recognition, and machine translation, as discussed in Chapter 3. Giving boundaries of coordinate structures to syntactic parsers might be helpful because they often make mistakes around coordination. However, coordination boundaries can even be harmful when used as constraints on syntactic parsing because the accuracies of coordination disambiguation and syntactic parsing have a large gap. Conversely, feeding syntactic information to a system for coordination disambiguation would improve its performance, as proposed by Ficler and Goldberg [29]. As another practical integration, the success on the GENIA in this work accommodates the challenge of recognizing named entities from elliptical coordination in scientific literature. By focusing on coordinated noun phrases, the proposed systems in this work can provide good practical performance in order to achieve promising results in the expansion of elliptical named entities.

Bibliography

[1] Rajeev Agarwal and Lois Boggess. A simple but useful approach to conjunct identification. In *Proceedings of the 30th Annual Meeting of the Association for Computational Linguistics*, pages 15–21. Association for Computational Linguistics, June 1992.

[2] Daniel Andor, Chris Alberti, David Weiss, Aliaksei Severyn, Alessandro Presta, Kuzman Ganchev, Slav Petrov, and Michael Collins. Globally normalized transition-based neural networks. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2442–2452. Association for Computational Linguistics, August 2016.

[3] Masayuki Asahara and Yuji Matsumoto. BCCWJ-DepPara: A syntactic annotation treebank on the 'balanced corpus of contemporary written Japanese'. In *Proceedings of the 12th Workshop on Asian Language Resources (ALR12)*, pages 49–58. The COLING 2016 Organizing Committee, December 2016.

[4] Samuel Bayer. The coordination of unlike categories. *Language*, 72(3): 579–616, 1996.

[5] Yoshua Bengio, Patrice Simard, and Paolo Frasconi. Learning long-term dependencies with gradient descent is difficult. *IEEE Transactions on Neural Networks*, 5(2):157–166, 1994.

[6] Ann Bies, Mark Ferguson, Karen Katz, and Robert MacIntyre. Bracketing guidelines for treebank II style Penn treebank project, 1995.

[7] Daniel M. Bikel. *On the Parameter Space of Generative Lexicalized Statistical Parsing Models.* PhD thesis, University of Pennsylvania, 2004.

[8] Taylor L. Booth. Probabilistic representation of formal languages. In *Proceedings of the 10th Annual Symposium on Switching and Automata Theory*, pages 74–81, 1969.

[9] Taylor L. Booth and Richard A. Thompson. Applying probability measures to abstract languages. *IEEE Transactions on Computers*, C-22(5):442–450, 1973.

[10] Ekaterina Buyko and Udo Hahn. Are morpho-syntactic features more predictive for the resolution of noun phrase coordination ambiguity than lexico-semantic similarity scores? In *Proceedings of the 22nd International Conference on Computational Linguistics*, pages 89–96. Coling 2008 Organizing Committee, August 2008.

[11] Ekaterina Buyko, Katrin Tomanek, and Udo Hahn. Resolution of coordination ellipses in biological named entities using conditional random fields. In *Proceedings of the 10th Conference of the Pacific Association for Computational Linguistics*, pages 163–171, 2007.

[12] Jeongmin Chae, Younghee Jung, Taemin Lee, Soonyoung Jung, Chan Huh, Gilhan Kim, Hyeoncheol Kim, and Heungbum Oh. Identifying non-elliptical entity mentions in a coordinated NP with ellipses. *Journal of Biomedical Informatics*, 47(C):139–152, February 2014.

[13] Francis Chantree, Adam Kilgarriff, Anne. de Roeck, and Alistair Willis. Disambiguating coordinations using word distribution information. In *Proceedings of Recent Advances in Natural Language Processing*, pages 287–294, 2005.

[14] Eugene Charniak and Mark Johnson. Coarse-to-fine n-best parsing and MaxEnt discriminative reranking. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*, pages 173–180. Association for Computational Linguistics, June 2005.

[15] Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using RNN encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734. Association for Computational Linguistics, October 2014.

[16] John Cocke and Schwartz J. T. Programming languages and their compilers: Preliminary notes. Technical report, Courant Institute of Mathematical Sciences, New York University, New York, NY, 1970.

[17] K. Bretonnel Cohen, Karin Verspoor, Helen Johnson, Chris Roeder, Philip Ogren, William Baumgartner, Elizabeth White, and Lawrence Hunter. High-precision biological event extraction with a concept recognizer. In *Proceedings of the BioNLP 2009 Workshop Companion Volume for Shared Task*, pages 50–58. Association for Computational Linguistics, June 2009.

[18] Michael Collins. *Head-Driven Statistical Models for Natural Language Parsing*. PhD thesis, University of Pennsylvania, 1999.

[19] Richard P. Cooper. Coordination in unification-based grammars. In *Proceedings of the fifth Conference of the European Chapter of the Association for Computational Linguistics*, pages 167–172. Association for Computational Linguistics, April 1991.

[20] Veronica Dahl and Michael C. McCord. Treating coordination in logic grammars. *American Journal of Computational Linguistics*, 9(2):69–91, 1983.

[21] Marie-Catherine de Marneffe, Bill MacCartney, and Christopher D. Manning. Generating typed dependency parses from phrase structure parses. In *Proceedings of the Fifth International Conference on Language Resources and Evaluation (LREC06)*, pages 449–454. European Language Resources Association (ELRA), May 2006.

[22] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language under-

standing. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics, June 2019.

[23] Markus Dickinson and W. Detmar Meurers. Prune diseased branches to get healthy trees! how to find erroneous local trees in a treebank and why it matters. In *Proceedings of the Fourth Workshop on Treebanks and Linguistic Theories*, 2005.

[24] Timothy Dozat and Christopher D. Manning. Deep biaffine attention for neural dependency parsing. In *Proceedings of the 5th International Conference on Learning Representations*, April 2017.

[25] Amit Dubey, Patrick Sturt, and Frank Keller. Parallelism in coordination as an instance of syntactic priming: Evidence from corpus-based modeling. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 827–834. Association for Computational Linguistics, October 2005.

[26] Amit Dubey, Frank Keller, and Patrick Sturt. Integrating syntactic priming into an incremental probabilistic parser, with an application to psycholinguistic modeling. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 417–424. Association for Computational Linguistics, July 2006.

[27] Jeffrey L. Elman. Finding structure in time. *Cognitive Science*, 14(2): 179–211, 1990.

[28] Jessica Ficler and Yoav Goldberg. Coordination annotation extension in the Penn tree bank. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 834–842. Association for Computational Linguistics, August 2016.

[29] Jessica Ficler and Yoav Goldberg. A neural network for coordination boundary prediction. In *Proceedings of the 2016 Conference on Empirical Methods*

in Natural Language Processing, pages 23–32. Association for Computational Linguistics, November 2016.

[30] Jessica Ficler and Yoav Goldberg. Improving a strong neural parser with conjunction-specific features. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 343–348. Association for Computational Linguistics, April 2017.

[31] Jenny Finkel, Shipra Dingare, Christopher D. Manning, Malvina Nissim, Beatrice Alex, and Claire Grover. Exploring the boundaries: gene and protein identification in biomedical text. *BMC Bioinformatics*, 6(S1):S5, May 2005.

[32] Sandiway Fong and Robert C. Berwick. New approaches to parsing conjunctions using prolog. In *Proceedings of the 23rd Annual Meeting of the Association for Computational Linguistics*, pages 118–126. Association for Computational Linguistics, July 1985.

[33] Kunihiko Fukushima. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological Cybernetics*, 36(4):193–202, 1980.

[34] Miriam Goldberg. An unsupervised model for statistically determining coordinate phrase attachment. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics*, pages 610–614. Association for Computational Linguistics, June 1999.

[35] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning.* MIT Press, 2016.

[36] Dan Gusfield. *Algorithms on Strings, Trees, and Sequences: Computer Science and Computational Biology.* Cambridge University Press, 1997.

[37] Atsushi Hanamoto, Takuya Matsuzaki, and Jun'ichi Tsujii. Coordination structure analysis using dual decomposition. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Lin-*

guistics, pages 430–438. Association for Computational Linguistics, April 2012.

[38] Jorge Hankamer. *Deletion in coordinate structures.* Garland Publishing, Inc., 1979.

[39] Kazuo Hara, Masashi Shimbo, Hideharu Okuma, and Yuji Matsumoto. Coordinate structure analysis with global structural constraints and alignment-based local features. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the Association for Computational Linguistics and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 967–975. Association for Computational Linguistics, August 2009.

[40] Kazuma Hashimoto, Caiming Xiong, Yoshimasa Tsuruoka, and Richard Socher. A joint many-task model: Growing a neural network for multiple NLP tasks. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1923–1933. Association for Computational Linguistics, September 2017.

[41] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, November 1997.

[42] Deirdre Hogan. Coordinate noun phrase disambiguation in a generative parsing model. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 680–687. Association for Computational Linguistics, June 2007.

[43] Deirdre Hogan. Empirical measurements of lexical similarity in noun phrase conjuncts. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions*, pages 149–152. Association for Computational Linguistics, June 2007.

[44] Xiuming Huang. Dealing with conjunctions in a machine translation environment. In *Proceedings of the first Conference of the European Chapter*

of the Association for Computational Linguistics, pages 81–85. Association for Computational Linguistics, September 1983.

[45] Christian Jacquemin, Judith L. Klavans, and Evelyne Tzoukermann. Expansion of multi-word terms for indexing and retrieval using morphology and syntax. In *35th Annual Meeting of the Association for Computational Linguistics and 8th Conference of the European Chapter of the Association for Computational Linguistics*, pages 24–31. Association for Computational Linguistics, July 1997.

[46] Yangfeng Ji and Jacob Eisenstein. Discriminative improvements to distributional sentence similarity. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 891–896. Association for Computational Linguistics, October 2013.

[47] Kyle Johnson. Gapping is not (VP-) ellipsis. *Linguistic Inquiry*, 40(2): 289–328, 2009.

[48] Daniel Jurafsky and James H. Martin. *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition.* Prentice Hall, second edition edition, 2008.

[49] Ronald M. Kaplan and John T. Maxwell III. Constituent coordination in lexical-functional grammar. In *Proceedings of the 12th Conference on Computational Linguistics - Volume 1*, pages 303—305. Association for Computational Linguistics, 1988.

[50] Tadao Kasami. An efficient recognition and syntax-analysis algorithm for context-free languages. Technical Report AFCRL-65-758, Air Force Cambridge Research Laboratory, Bedford, MA, 1965.

[51] Daisuke Kawahara and Sadao Kurohashi. Probabilistic coordination disambiguation in a fully-lexicalized Japanese parser. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 306–314, Prague, Czech Republic, June 2007. Association for Computational Linguistics.

[52] Daisuke Kawahara and Sadao Kurohashi. Coordination disambiguation without any similarities. In *Proceedings of the 22nd International Conference on Computational Linguistics*, pages 425–432. Coling 2008 Organizing Committee, August 2008.

[53] Daisuke Kawahara and Sadao Kurohashi. Generative modeling of coordination by factoring parallelism and selectional preferences. In *Proceedings of the Fifth International Joint Conference on Natural Language Processing*, pages 456–464. Asian Federation of Natural Language Processing, November 2011.

[54] Jin Dong Kim, Tomoko Ohta, Yuka Tateisi, and Jun'ichi Tsujii. GENIA corpus—a semantically annotated corpus for bio-textmining. *Bioinformatics*, 19(suppl 1):i180–i182, 2003.

[55] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *Proceedings of the 3rd International Conference on Learning Representations*, 2015.

[56] Donald W. Kosy. Parsing conjunctions deterministically. In *Proceedings of the 24th Annual Meeting of the Association for Computational Linguistics*, pages 78–84. Association for Computational Linguistics, July 1986.

[57] Sandra Kübler, Erhard Hinrichs, Wolfgang Maier, and Eva Klett. Parsing coordinations. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, pages 406–414. Association for Computational Linguistics, March 2009.

[58] Sadao Kurohashi. Analyzing coordinate structures including punctuation in English. In *Proceedings of the fourth International Conference on Parsing Technologies*, pages 136–147, 1995.

[59] Sadao Kurohashi and Makoto Nagao. Dynamic programming method for analyzing conjunctive structures in Japanese. In *Proceedings of the 15th International Conference on Computational Linguistics - Volume 1*, pages 170–176, 1992.

[60] Sadao Kurohashi and Makoto Nagao. A syntactic analysis method of long Japanese sentences based on the detection of conjunctive structures. *Computational Linguistics*, 20(4):507–534, 1994.

[61] Yann LeCun, Bernhard Boser, John S. Denker, Donnie Henderson, Richard E. Howard, Wayne Hubbard, and Lawrence D. Jackel. Backpropagation applied to handwritten zip code recognition. *Neural Computation*, 1(4):541–551, 1989.

[62] Leonardo Lesmo and Pietro Torasso. Analysis of conjunctions in a rule-based parser. In *23rd Annual Meeting of the Association for Computational Linguistics*, pages 180–187. Association for Computational Linguistics, July 1985.

[63] Vladimir I. Levenshtein. Binary codes capable of correcting deletions, insertions and reversals. *Cybernetics and Control Theory*, 10(8):707–710, 1966. Original in *Doklady Akademii Nauk SSSR*, 163(4):845–848, 1965.

[64] Xuezhe Ma and Eduard Hovy. End-to-end sequence labeling via bi-directional LSTM-CNNs-CRF. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, pages 1064–1074. Association for Computational Linguistics, August 2016.

[65] Kikuo Maekawa, Makoto Yamazaki, Toshinobu Ogiso, Takehiko Maruyama, Hideki Ogura, Wakako Kashino, Hanae Koiso, Masaya Yamaguchi, Makiro Tanaka, and Yasuharu Den. Balanced corpus of contemporary written japanese. *Language resources and evaluation*, 48(2):345–371, 2014.

[66] Wolfgang Maier, Sandra Kübler, Erhard Hinrichs, and Julia Kriwanek. Annotating coordination in the Penn treebank. In *Proceedings of the Sixth Linguistic Annotation Workshop*, pages 166–174. Association for Computational Linguistics, July 2012.

[67] Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. Building a large annotated corpus of English: The Penn treebank. *Computational Linguistics*, 19(2):313–330, 1993.

[68] Paweł Mazur and Robert Dale. Named entity extraction with conjunction disambiguation. In *Proceedings of the Fifth International Conference on Language Resources and Evaluation*, pages 1752–1755. European Language Resources Association (ELRA), May 2006.

[69] Stephen Merity, Nitish Shirish Keskar, and Richard Socher. Regularizing and optimizing LSTM language models. In *Proceedings of the 6th International Conference on Learning Representations*, April 2018.

[70] Tomas Mikolov, Kai Chen, Greg S. Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. In *Proceedings of the 1st International Conference on Learning Representations*, May 2013.

[71] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems 26*, pages 3111–3119. Curran Associates, Inc., December 2013.

[72] George A Miller. *WordNet: An electronic lexical database.* MIT press, 1998.

[73] David Milward. Coordination in an axiomatic grammar. In *Proceedings of the 13th International Conference on Computational Linguistics - Volume 3*, pages 207–212. Association for Computational Linguistics, 1990.

[74] Alan Munn. *Topics in the Syntax and Semantics of Coordinate Structures.* PhD thesis, University of Maryland at College Park, 1993.

[75] Preslav Nakov and Marti Hearst. Using the web as an implicit training set: Application to structural ambiguity resolution. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 835–842. Association for Computational Linguistics, October 2005.

[76] Goran Nenadic, Irena Spasic, and Sophia Ananiadou. Mining biomedical abstracts: What's in a term? In *The First International Joint Conference on Natural Language Processing.* Asian Federation of Natural Language Processing, March 2004.

[77] Jens Nilsson, Joakim Nivre, and Johan Hall. Graph transformations in data-driven dependency parsing. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 257–264. Association for Computational Linguistics, July 2006.

[78] Jens Nilsson, Joakim Nivre, and Johan Hall. Generalizing tree transformations for inductive dependency parsing. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 968–975. Association for Computational Linguistics, June 2007.

[79] Joakim Nivre, Marie-Catherine de Marneffe, Filip Ginter, Yoav Goldberg, Jan Hajič, Christopher D. Manning, Ryan McDonald, Slav Petrov, Sampo Pyysalo, Natalia Silveira, Reut Tsarfaty, and Daniel Zeman. Universal dependencies v1: A multilingual treebank collection. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC16)*, pages 1659–1666. European Language Resources Association (ELRA), May 2016.

[80] Philip Ogren. Improving syntactic coordination resolution using language modeling. In *Proceedings of the 2010 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Student Research Workshop*, pages 1–6. Association for Computational Linguistics, June 2010.

[81] Philip Victor Ogren. *Coordination Resolution In Biomedical Texts.* PhD thesis, University of Colorado Boulder, 2011.

[82] Hideharu Okuma, Kazuo Hara, Masashi Shimbo, and Yuji Matsumoto. Bypassed alignment graph for learning coordination in Japanese sentences. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the Association for Computational Linguistics and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 5–8. Association for Computational Linguistics, August 2009.

[83] Akitoshi Okumura and Kazunori Muraki. Symmetric pattern matching analysis for English coordinate structures. In *Fourth Conference on Applied*

Natural Language Processing, pages 41–46. Association for Computational Linguistics, October 1994.

[84] Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. On the difficulty of training recurrent neural networks. In *Proceedings of the 30th International Conference on Machine Learning*, volume 28 of *Proceedings of Machine Learning Research*, pages 1310–1318, Atlanta, Georgia, USA, June 2013. PMLR.

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

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

[87] Peter G. Peterson. Coordination: consequences of a lexical-functional account. *Natural Language and Linguistic Theory*, 22(3):643–679, 2004.

[88] Slav Petrov, Leon Barrett, Romain Thibaux, and Dan Klein. Learning accurate, compact, and interpretable tree annotation. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 433–440. Association for Computational Linguistics, July 2006.

[89] Maja Popović and Sheila Castilho. Are ambiguous conjunctions problematic for machine translation? In *Proceedings of the International Conference on Recent Advances in Natural Language Processing*, pages 959–966. INCOMA Ltd., September 2019.

[90] Paul Martin Postal. *On Raising: One Rule of English Grammar and its Theoretical Implications.* MIT Press, 1974.

[91] Adwait Ratnaparkhi, Salim Roukos, and R. Todd Ward. A maximum entropy model for parsing. In *In Proceedings of the International Conference on Spoken Language Processing*, pages 803–806, 1994.

[92] Carol Raze. A computational treatment of coordinate conjunctions. *American Journal of Computational Linguistics*, September 1976.

[93] Philip Resnik. Semantic similarity in a taxonomy: An information-based measure and its application to problems of ambiguity in natural language. *Journal of Artificial Intelligence Research*, 11(1):95–130, July 1999.

[94] Philip Stuart Resnik. *Selection and Information: A Class-Based Approach to Lexical Relationships*. PhD thesis, USA, 1993.

[95] John Robert Ross. *Constraints on variables in syntax*. PhD thesis, Massachusetts Institute of Technology, 1967.

[96] David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. Learning representations by back-propagating errors. *Nature*, 323(6088):533–536, 1986.

[97] Ivan Sag, Gerald Gazdar, Thomas Wasow, and Steven Weisler. Coordination and how to distinguish categories. *Natural Language and Linguistic Theory*, 3:117–171, 1985.

[98] Swarnadeep Saha and Mausam . Open information extraction from conjunctive sentences. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 2288–2299. Association for Computational Linguistics, August 2018.

[99] Anoop Sarkar and Aravind Joshi. Coordination in tree adjoining grammars: Formalization and implementation. In *Proceedings of the 16th International Conference on Computational Linguistics - Volume 2*, pages 610—615. Association for Computational Linguistics, 1996.

[100] Masashi Shimbo and Kazuo Hara. A discriminative learning model for coordinate conjunctions. In *Proceedings of the 2007 Joint Conference on*

Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL), pages 610–619. Association for Computational Linguistics, June 2007.

[101] Nitish Srivastava, Geoffrey E Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15:1929–1958, 2014.

[102] Mitchell Stern, Jacob Andreas, and Dan Klein. A minimal span-based neural constituency parser. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 818–827. Association for Computational Linguistics, July 2017.

[103] Kai Sheng Tai, Richard Socher, and Christopher D. Manning. Improved semantic representations from tree-structured long short-term memory networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1556–1566. Association for Computational Linguistics, July 2015.

[104] Yuka Tateisi, Akane Yakushiji, Tomoko Ohta, and Jun'ichi Tsujii. Syntax annotation for the GENIA corpus. In *Proceedings of the Second International Joint Conference on Natural Language Processing (Companion Volume: Posters/Demos and tutorial abstracts)*, pages 220–225. Asian Federation of Natural Language Processing, October 2005.

[105] Zhiyang Teng and Yue Zhang. Two local models for neural constituent parsing. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 119–132. Association for Computational Linguistics, August 2018.

[106] Hiroki Teranishi, Hiroyuki Shindo, and Yuji Matsumoto. Coordination boundary identification with similarity and replaceability. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 264–272. Asian Federation of Natural Language Processing, November 2017.

[107] Kristina Toutanova, Dan Klein, Christopher D. Manning, and Yoram Singer. Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1*, pages 173–180. Association for Computational Linguistics, May 2003.

[108] George Tsatsaronis, Michael Schroeder, Georgios Paliouras, Yannis Almirantis, Ion Androutsopoulos, Eric Gaussier, Patrick Gallinari, Thierry Artieres, Michael R Alvers, Matthias Zschunke, and Axel-Cyrille Ngonga Ngomo. BioASQ: A challenge on large-scale biomedical semantic indexing and question answering. In *AAAI Fall Symposium Series: Information Retrieval and Knowledge Discovery in Biomedical Text*, 2012.

[109] David Vadas and James Curran. Adding noun phrase structure to the Penn treebank. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 240–247. Association for Computational Linguistics, June 2007.

[110] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc., December 2017.

[111] Li Wan, Matthew Zeiler, Sixin Zhang, Yann Le Cun, and Rob Fergus. Regularization of neural networks using DropConnect. In *Proceedings of the 30th International Conference on Machine Learning*, volume 28 of *Proceedings of Machine Learning Research*, pages 1058–1066, Atlanta, Georgia, USA, June 2013. PMLR.

[112] Edwin Williams. Across-the-board rule application. *Linguistic Inquiry*, 9 (1):31–43, 1978.

[113] William A. Woods. An experimental parsing system for transition network grammars. *American Journal of Computational Linguistics*, 1973.

[114] Akifumi Yoshimoto, Kazuo Hara, Masashi Shimbo, and Yuji Matsumoto. Coordination-aware dependency parsing (preliminary report). In *Proceedings of the 14th International Conference on Parsing Technologies*, pages 66–70. Association for Computational Linguistics, July 2015.

[115] Daniel H. Younger. Recognition and parsing of context-free languages in time n^3. *Information and Control*, 10(2):189–208, 1967.

List of Publications

Journal Papers

1. 寺西 裕紀, 進藤 裕之, 渡辺 太郎, 松本 裕治. 局所的モデルと CKY アルゴリズムによる並列構造解析. 自然言語処理, Vol.27 No.4. 2020 年. (in press)

2. 寺西 裕紀, 進藤 裕之, 松本 裕治. 語系列の類似性・可換性の特徴表現による並列句の範囲同定. 自然言語処理, Vol.25 No.4, pages 441–462. 2018 年.

Conference Papers (refereed)

1. Hiroki Teranishi, Hiroyuki Shindo, and Yuji Matsumoto. Decomposed Local Models for Coordinate Structure Parsing. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long and Short Papers)*, pages 3394–3403. Association for Computational Linguistics, Minneapolis, Minnesota, USA, June 2019.

2. Hiroki Teranishi, Hiroyuki Shindo, and Yuji Matsumoto. Coordination Boundary Identification with Similarity and Replaceability. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 264–272. Asian Federation of Natural Language Processing, Taipei, Taiwan, November 2017.

3. Yuta Kobayashi, Hiroki Teranishi, Masashi Shimbo and Yuji Matsumoto. Learning scientific paper representations from text and citation graphs. In

Proceedings of the First International Workshop on SCIentific DOCument Analysis (SCIDOCA 2016), Kanagawa, Japan, November 2016.

Other Publications

1. 寺西 裕紀, 進藤 裕之, 松本 裕治. 細分化された局所的モデルによる並列構造の構文解析. 言語処理学会第 25 回年次大会. 愛知, 2019 年 3 月.
2. 寺西 裕紀, 進藤 裕之, 松本 裕治. 文脈自由文法とニューラルネットワークを用いた並列構造木の CKY 構文解析. 情報処理学会第 237 回自然言語処理研究会. 北海道, 2018 年 9 月.
3. 寺西 裕紀, 進藤 裕之, 松本 裕治. マルチタスク学習による遷移型依存構文解析. 言語処理学会第 24 回年次大会. 岡山, 2018 年 3 月.
4. 寺西 裕紀, 進藤 裕之, 松本 裕治. ニューラルネットワークに基づく並列句表現の学習と構造解析. 情報処理学会第 232 回自然言語処理研究会. 東京, 2017 年 7 月.

Appendix

A.1 List of syntactic labels

This information is from “Bracketing Guidelines for Treebank II Style Penn Treebank Project” [6].

Clause level

S	Simple declarative clause, i.e., one that is not introduced by a (possible empty) subordinating conjunction or a wh-word and that does not exhibit subject–verb inversion.
SBAR	Clause introduced by a (possibly empty) subordinating conjunction.
SBARQ	Direct question introduced by a wh-word or wh-phrase. Indirect questions and relative clauses should be bracketed as SBAR, not SBARQ.
SINV	Inverted declarative sentence, i.e., one in which the subject follows the tensed verb or modal.
SQ	Inverted yes/no question, or main clause of a wh-question, following the wh-phrase in SBARQ.

Phrase level

ADJP	Adjective Phrase.
ADVP	Adverb Phrase.

CONJP	Conjunction Phrase.
FRAG	Fragment.
INTJ	Interjection. Corresponds approximately to the part-of-speech tag UH.
LST	List marker. Includes surrounding punctuation.
NAC	Not a Constituent; used to show the scope of certain prenominal modifiers within an NP.
NP	Noun Phrase.
NX	Used within certain complex NPs to mark the head of the NP. Corresponds very roughly to N-bar level but used quite differently.
PP	Prepositional Phrase.
PRN	Parenthetical.
PRT	Particle. Category for words that should be tagged RP.
QP	Quantifier Phrase (i.e., complex measure/amount phrase); used within NP.
RRC	Reduced Relative Clause.
UCP	Unlike Coordinated Phrase.
VP	Verb Phrase.
WHADJP	Wh-adjective Phrase. Adjectival phrase containing a wh-adverb, e.g., how hot.
WHAVP	Wh-adverb Phrase. Introduces a clause with an NP gap. May be null (containing the 0 complementizer) or lexical, containing a wh-adverb, such as how or why.
WHNP	Wh-noun Phrase. Introduces a clause with an NP gap. May be null (containing the 0 complementizer) or lexical, containing some wh-word, e.g., who, which book, whose daughter, none of which, or how many leopards.
WHPP	Wh-prepositional Phrase. Prepositional phrase containing a wh-noun phrase (such as of which or by whose authority) that either introduces a PP gap or is contained by a WHNP.

X	Unknown, uncertain, or unbracketable. X is often used for bracketing typos and in bracketing the...the-constructions.

Word level

CC	Coordinating conjunction.
CD	Cardinal number.
DT	Determiner.
EX	Existential there.
FW	Foreign word.
IN	Preposition or subordinating conjunction.
JJ	Adjective.
JJR	Adjective, comparative.
JJS	Adjective, superlative.
LS	List item marker.
MD	Modal.
NN	Noun, singular or mass.
NNS	Noun, plural.
NNP	Proper noun, singular.
NNPS	Proper noun, plural.
PDT	Predeterminer.
POS	Possessive ending.
PRP	Personal pronoun.
PRP$	Possessive pronoun (prolog version PRP-S).
RB	Adverb.
RBR	Adverb, comparative.
RBS	Adverb, superlative.
RP	Particle.
SYM	Symbol.

TO	To.
UH	Interjection.
VB	Verb, base form.
VBD	Verb, past tense.
VBG	Verb, gerund or present participle.
VBN	Verb, past participle.
VBP	Verb, non-3rd person singular present.
VBZ	Verb, 3rd person singular present.
WDT	Wh-determiner.
WP	Wh-pronoun.
WP$	Possessive wh-pronoun (prolog version WP-S).
WRB	Wh-adverb.