博士論文

Learning Semantic Textual Relatedness using Natural Deduction Proof

（自然演繹に基づく論理推論を用いた
文間関連性の評価）

谷中　瞳

UNIVERSITY OF TOKYO

DOCTORAL THESIS

Learning Semantic Textual Relatedness using Natural Deduction Proof

---

Hitomi YANAKA

A thesis submitted in fulfillment of the requirements
for the degree of Doctor of Engineering

in the

Department of Systems Innovation
School of Engineering

July, 2018

UNIVERSITY OF TOKYO

Abstract

School of Engineering

Department of Systems Innovation

Doctor of Engineering

Learning Semantic Textual Relatedness

using Natural Deduction Proof

by Hitomi YANAKA

Capturing textual entailment and similarity is one of the most important problems in natural language processing. This task is related to many applications such as question answering. To solve this problem, machine learning-based approaches often use shallow information such as words and characters. However, it is not clear whether they are capable of accounting for functional meanings such as negation and quantifier and capturing the whole meaning of a sentence correctly. Conversely, logic-based approaches have been successful in capturing functional meanings by using logical semantic representations but their symbolic nature does not offer graded notions of textual similarity. To achieve advantages over these two approaches, hybrid approaches have been proposed for learning textual entailment and similarity. However, a more effective way to combine logic-based approaches with machine learning-based approaches is desired. In this thesis, I explore a hybrid approach to capturing semantic relations between two sentences by careful treatment of proof processes. The key idea is that the *proof processes reflect some aspects of semantic textual relations, and they are useful for capturing textual entailment and similarity more precisely*. I propose a hybrid approach to learning textual entailment and similarity by combining shallow features with features extracted from natural deduction proofs of bidirectional entailment relations between sentences. I evaluated my approach with the datasets for two tasks about semantic textual relations: recognizing textual entailment (RTE) and semantic textual similarity (STS). Experiments showed that my approach outperformed previous logic-based approaches in the STS task. Furthermore, my approach achieved state-of-the-art performance on the RTE task. The evaluations indicate that features derived from proof processes are effective features for learning textual similarity and entailment. The evaluations also indicate that handling phrase-level semantic relations in logical inference is a crucial problem in capturing semantic relations between sentences more precisely. To solve this problem, I propose a phrase abduction mechanism, which detects the lack of phrasal knowledge in logical inference by the careful management of variable sharing during the proof processes. The experiments showed that the phrase abduction mechanism compensated for a lack of phrasal knowledge and improved the accuracy of logical inference in the RTE task.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Prospects of capturing semantic relations

Language is used in everyday life. We use language to communicate with each other. With the development of information technologies and the availability of large text datasets, researchers strongly expect that natural languages can be analyzed computationally. Natural language processing (NLP) is the field that studies how to enable computers to analyze and understand human languages. In this field, the establishment of a method for calculating whether one sentence is semantically related to another or not is one of the most important core problems. In this thesis, I consider how to quantify a semantic relation between sentences computationally.

This core problem is closely related to many NLP applications such as question answering, information retrieval, and text summarization.

Question answering

This task is to build systems that automatically answer questions asked in natural languages from text. Given the question "*Where was Barack Obama born?*", the question answering system returns the answer "*Barack Obama was born in Honolulu.*" Here, the calculation of a semantic relation between a question and an answer can be used to validate candidate answers of a question answering system. Candidate answers can be determined by testing how similar a question is to candidate answers, or whether the meaning of a question is included in candidate answers. Previous studies (Vo, Magnolini, and Popescu, 2015; Sacaleanu et al., 2008; Harabagiu and Hickl, 2006) have applied the calculation of semantic relations for question answering systems.

Information retrieval

This activity is to find material that satisfies an information need from a collection of documents. Recently, retrieval of short texts from a large number of web documents has become important for many web applications (e.g., web search and ad matching). Previous works (Clinchant, Goutte, and Gaussier, 2006; Kenter and Rijke, 2015) have used a semantic relation between a target sentence and a source sentence for information retrieval to evaluate if the retrieved document contains the target information.

Text summarization

This task is to create an accurate and fluent summary of a longer text document by retaining its important points while avoiding redundancy. Previous works (Mogren, Kågebäck, and Dubhashi, 2015; Bentivogli et al., 2009) have applied measurement of semantic relations between sentences in the text to the task of text summarization.

In summary, capturing a semantic relation between sentences provides a common generic framework that can be used by these various NLP applications.

1.2 Task description

1.2.1 Entailment and similarity

In NLP tasks, we can capture a semantic relation between sentences from two points of view: *entailment* and *similarity*. Figure 1.1 depicts the conceptual diagram of these two kinds of semantic relations.

Consider the semantic relations between the sentences below:

(1) a. *The boy sings*

b. *The man sings*

If we judge the sentence (1a) to be true, we also judge the sentence (1b) to be true. In such cases, we say the meaning of the sentence (1a) *entails* the meaning of the sentence (1b). In this setting, we capture the meaning of a sentence as a set of possible interpretations. We consider the semantic relation as if the meaning of the sentence

(1b) includes the meaning of the sentence (1a) as described in the left side of Figure 1.1. This discrete semantic relation between sentences is called as *entailment*[1].

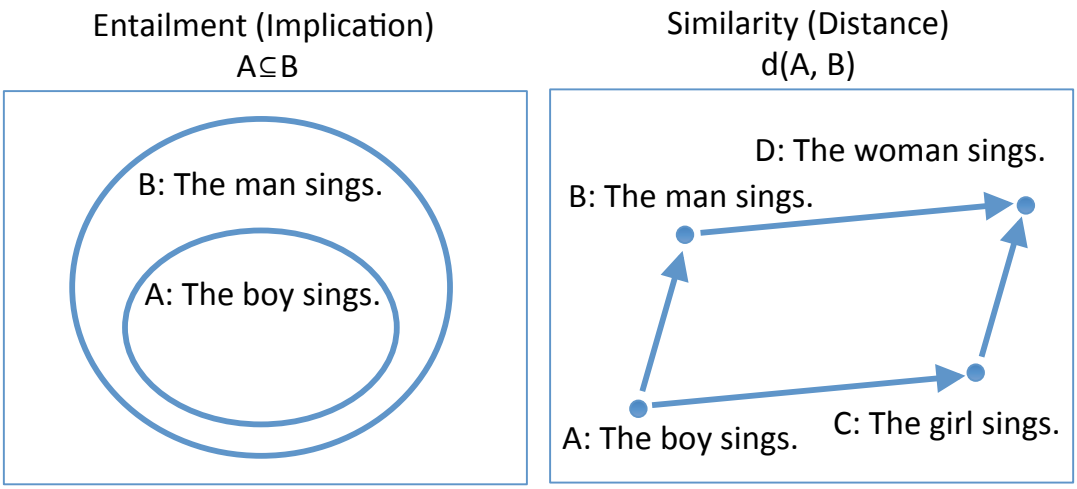

FIGURE 1.1: Conceptual diagram of entailment and similarity.

Now let us consider the semantic relation between the sentences (2a) and (2b), in addition to the semantic relation between the sentences (1a) and (1b):

(2) a. *The girl sings*

b. *The woman sings*

We assume the relatedness between the sentence (1a) and the sentence (1b) is the same as the relatedness between the sentence (2a) and the sentence (2b), i.e., a word *man* is a hypernym of a word *boy* and the same is true of the words *woman* and *girl*. In addition, the meaning of the sentence (1a) comes closer to the meaning of the sentence (1b) than to that of the sentence (2a). In such cases, we say the meaning of the sentence (1a) is more *similar* to the meaning of the sentence (1b) than to that of the sentence (2a). In this setting, we capture the meaning of a sentence as a vector in a certain vector space described in the right side of Figure 1.1. We consider the semantic relation as the distance between the meaning of one sentence and that of another. This graded semantic relation between sentences is called as *similarity*.

[1]An entailment relation is not limited to the relation between two sentences. $P_1 \wedge \ldots \wedge P_n \subseteq C$ describes that the meanings of sentences $P_1 \ldots P_n$ entail the meaning of another sentence C, where the meanings of multiple sentences are described as the meaning of each sentence connected with a conjunction ($\wedge$).

1.2.2 Recognizing textual entailment

Recognizing textual entailment (RTE) is the task of judging whether the meaning of one sentence entails the meaning of another sentence. This task is one of the most challenging NLP tasks. RTE was first introduced by Dagan and Glickman (2006) and developed to evaluate systems for natural language inference. In the main RTE task, two text fragments are given, and we judge whether one text fragment logically follows from another text fragment automatically (Dagan et al., 2013).

As an example of an RTE problem, let us consider the following four sentences:

(3) a. *An onion is being sliced by a man*

b. *There is no man slicing an onion*

c. *A man is eating onion slices*

d. *An onion is being cut by a man*

The sentence (3a) *entails* the meaning of the sentence (3d). In other words, if we assume the sentence (3d) as a hypothesis H and the sentence (3a) as a text T, the meaning of H can be inferred by a human from the meaning of T, as interpreted in the context of T. The sentence (3b) *contradicts* the meaning of the sentence (3d). This means the context of T does not lead to any conclusion about H. We cannot judge both the entailment relation and the contradiction between the sentence (3c) and the sentence (3d). In this thesis, I refer to this condition as *neutral*.

RTE tasks are usually considered as classification problems concerned with a two-class classification, i.e., entailment or not (Dagan, Glickman, and Magnini, 2006; Giampiccolo et al., 2007), or a three-class classification, i.e., *yes* (entailment), *no* (contradiction), or *unknown* (neutral) (Giampiccolo et al., 2008; Bentivogli et al., 2009; Bentivogli et al., 2010; Bentivogli et al., 2011). The three-class classification is more complex, and thus it appears to be a major RTE classification. In this thesis, I consider this three-class classification.

Performance of RTE systems is evaluated using gold-standard datasets and evaluation metrics. There are four kinds of evaluation metrics used for RTE systems: accuracy, precision, recall, and F1-score. TP (true positives) and FP (false positives) are the numbers of pairs that have been correctly or incorrectly, respectively, classified as entailment or contradiction pairs. TN (true negatives) and FN (false negatives) are the numbers of pairs that have been correctly or incorrectly, respectively, classified as neutral pairs. $TP + FP$ is the number of all prediction results and $TP + FN$ is

the number of all correct (gold) labels. Therefore, in RTE, the precision describes the proportion of positive predictions that are correct, the recall describes the proportion of positive problems that are classified correctly, and accuracy describes the ratio between true predictions and total problems. The F1-score is the harmonic mean of the precision and the recall. Following previous studies, I use the accuracy as the main evaluation metric in this thesis.

$$precision = \frac{TP}{TP + FP} \tag{1.1}$$

$$recall = \frac{TP}{TP + FN} \tag{1.2}$$

$$accuracy = \frac{correctly\ classified\ pairs}{all\ pairs} \tag{1.3}$$

$$F1\text{-}score = \frac{2 \cdot precision \cdot recall}{precision + recall} \tag{1.4}$$

1.2.3 Semantic textual similarity

Semantic textual similarity (STS) (Agirre et al., 2012; Agirre et al., 2013) is the task of capturing a similarity between sentences. This task is one of the most critical tasks in NLP and information retrieval. STS measures the degree of semantic equivalence between two sentences. The difference between STS and RTE is that STS considers symmetric and graded equivalence between the sentences.

For example, let us consider the following two sentences:

(4) a. *There are small children*

b. *There are two children*

We cannot judge if the relation between these two sentences is entailment or contradiction, while we capture these sentences are similar.

The similarity score is annotated by humans, and its scale is designed to be accessible by reasonable human judges. Table 1.1 shows one of the annotation guidelines (Agirre et al., 2013; Cer et al., 2017). The ordinal scale guides human annotation, ranging from 0 for no meaning overlap to 5 for meaning equivalence. Intermediate values reflect interpretable levels of partial overlap in meaning.

As with the evaluation of RTE systems, prediction performance of STS models is evaluated using gold-standard datasets and evaluation metrics. There are three major evaluation metrics for STS: the Pearson correlation coefficient γ, Spearman's

Similarity	Definition	Example
5	The two sentences are completely equivalent, as they mean the same thing.	*The bird is bathing in the sink* *Birdie is washing itself in the water basin*
4	The two sentences are mostly equivalent, but some unimportant details differ.	*Two boys on a couch are playing video games* *Two boys are playing a video game*
3	The two sentences are roughly equivalent, but some important information differs/missing.	*John said he is considered a witness but not a suspect* *"He is not a suspect anymore." John said*
2	The two sentences are not equivalent, but share some details.	*They flew out of the nest in groups* *They flew into the nest together*
1	The two sentences are not equivalent, but are on the same topic.	*The woman is playing the violin* *The young lady enjoys listening to the guitar*
0	The two sentences are completely dissimilar	*The black dog is running through the snow* *A race car driver is driving his car through the mud*

TABLE 1.1: Similarity scores with explanations and English examples defined in Agirre et al. (2013) and Cer et al. (2017).

rank correlation coefficient ρ, and the mean squared error (MSE) between predicted semantic textual similarity scores y_i and gold scores x_i. The definitions of these evaluation metrics are as follows: d_i is the pairwise distances of the ranks of the variables x_i and y_i, and n is the number of sentence pairs.

$$r = \frac{\sum_{i=1}^{n}(x_i - \overline{x})(y_i - \overline{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \overline{x})^2(y_i - \overline{y})^2}} \tag{1.5}$$

$$\rho = 1 - \frac{6\sum d_i^2}{n(n^2-1)} \tag{1.6}$$

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - x_i)^2 \tag{1.7}$$

1.3 Problem statement

How can we calculate a semantic relation between sentences computationally? For example, consider a semantic relation between the following two sentences:

(5) a. *There is no child sawing logs in the park*

b. *No boy is cutting wood outside*

In this sentence pair, we judge that the sentence (5a) entails the sentence (5b). We also judge they are similar sentences. To make these judgements about entailment and similarity, we assume lexical meanings of content words or phrases. In this example, *child* and *boy* have a hyponym relation. "*sawing logs*" and "*cutting wood*" are paraphrases. Meanwhile, we assume functional meanings of logical or functional words, e.g., the meaning of the phrase including negation "*There is no* A ..." is equivalent to the meaning of the phrase "*No* A *is* ...".

As I described above, words can be divided into two broad categories of meaning: content (lexical) words and logical or functional words. In English grammar, content words are called as open classes and logical or functional words are called as closed classes (Jurafsky and Martin, 2009). Open classes are those that are continually coined. For instance, the class of nouns is potentially infinite, since it is continually being expanded as new scientific discoveries are made, new ideas are explored, and so on. Due to this characteristic of open classes, lexical meanings of open classes are generally defined in dictionaries or lexical databases. There are four major open classes that occur in English: nouns, verbs, adjectives, and adverbs.

By contrast, closed classes are those that have relatively fixed members. For example, prepositions (e.g., *in*, *of*) are a closed class because there is a fixed set of them in English; new prepositions are rarely coined. Functional words are grammatical words (e.g., *of*, *it*, *and*), which tend to be very short, occur frequently, and play an important role in grammar. Logical words are words whose meanings can be described with logical operators, such as negation and quantifier. Both functional and logical words compose semantic structures and thus the functional meanings of closed classes are sensitive to syntactic structures.

In summary, to calculate a semantic relation between sentences computationally, we should consider how to represent and calculate the meanings of sentences which is determined by the interactions between the lexical meanings of content words and the functional meanings of functional or logical words.

To analyze the meanings of sentences computationally, there are mainly two approaches to representing the meanings of sentences: vector representations and logical formula representations. In the former approach, the meanings of words in a dataset are represented by vectors and the meanings of sentences are learned by a supervised model trained with the dataset. Semantic similarity between sentence vector representations is calculated by the inner product between them. This approach is called a machine learning-based approach. In the latter approach, the meanings of sentences are represented as logical formulas and a semantic relation between the logical formulas is directly calculated by using logical inference. This approach is called a logic-based approach.

In machine learning-based approaches, vector-based sentence representation models have been widely used to compare and rank words, phrases, or sentences using various similarities (Wong and Raghavan, 1984; Mitchell and Lapata, 2010; Le

and Mikolov, 2014). Recently, neural network-based sentence representation models (Mueller and Thyagarajan, 2016; Hill, Cho, and Korhonen, 2016; Yin and Schütze, 2017) have been proposed for learning textual entailment and similarity. These models achieved high accuracy in learning textual entailment and similarity by training from large datasets. However, these machine learning-based approaches often use shallow information, such as words and characters, and it is not clear whether they can account for the functional meaning of logical words such as negation and quantification.

To describe this problem, let us consider the meaning of the following sentence pair:

(6) a. *Tom did not meet some of the players*

b. *Tom did not meet any of the players*

If functional words such as *some* or *any* are ignored or represented as the same vector, then these sentences would be represented by identical vectors. However, the sentence (6a) implies that there is a player who Tom did not meet, whereas the sentence (6b) means that Tom did not meet anyone. Therefore, the sentences have different meanings.

By contrast, logic-based approaches have been successful in representing the meanings of complex sentences (including the functional meanings) as logical formulas, which have had a positive impact on RTE tasks (Mineshima et al., 2015; Mineshima et al., 2016; Abzianidze, 2015; Abzianidze, 2016). However, purely logic-based approaches only assess an entailment relation or contradiction between sentences, lacking some flexibility for predicting graded notions of semantic textual similarity.

To achieve advantages over both of logic-based approaches and machine learning-based approaches, hybrid approaches have been proposed for learning both textual entailment and similarity. In previous hybrid approaches, The Meaning Factory (Bjerva et al., 2014) and UTexas (Beltagy et al., 2014) have improved accuracy by using logic-based features derived from the entailment results of first-order theorem proving combined with shallow features such as sentence lengths. These previous hybrid approaches opened a door to a fusion of logic and machine learning to

predict both textual similarities and entailments, though there is still room for improvement in combining logic-based approaches with machine learning-based approaches effectively. In this thesis, I develop a hybrid approach to capturing both textual similarities and entailments.

I tackle two main problems in logic-based approaches to calculating semantic relations between sentences. The first problem is that logic-based approaches have difficulty offering graded notions of semantic textual similarity with high accuracy. As we saw above, purely logic-based approaches only provide a binary semantic relation (entailment or contradiction) between sentences. More specifically, while the previous hybrid approaches above attempt to offer graded notions of semantic textual similarity, they focus only on proof results or logical formulas, which are only a part of the information obtained from proofs. Machine learning-based approaches are still much more successful in assessing semantic textual similarity than logic-based approaches and thus there should be a more effective way for combining logic-based approaches with machine learning-based approaches.

The second problem is that logic-based approaches have difficulty capturing phrase-level semantic relations. With genuine logical inference only, logic-based approaches fail to capture the meaning of content words. Accounting for lexical relations between content words or phrases when doing logical inference remains a crucial problem. Many previous logic-based approaches (Mineshima et al., 2015; Mineshima et al., 2016; Abzianidze, 2015; Bjerva et al., 2014; Beltagy et al., 2014) use knowledge databases such as WordNet (Miller, 1995) to identify lexical relations within a sentence pair. While this solution has been successful in handling word-level paraphrases, its extension to phrase-level semantic relations is still an unsolved problem.

1.4 Intended contributions

I solve these two problems for calculating semantic relations with *a careful treatment of the theorem proving process*. Considering the conception of proof-theoretic semantics (Bekki and Mineshima, 2017), not only the entailment results but also the *theorem proving process* can be considered as features for learning both textual entailment and similarity. That is, by taking into account not only whether a theorem is proved but also *how* it is proved, we can capture the semantic relation in more depth. The key

idea is that the *proof processes reflect some aspects of semantic relations between sentences*. This idea contributes to capturing textual entailment and similarity more precisely than previous approaches.

To solve the first problem of predicting semantic textual similarity with high accuracy, I propose a novel method of learning both textual entailment and similarity using proof processes. I propose a hybrid approach to learning both textual entailment and similarity by combining shallow features with features extracted from natural deduction proofs of bidirectional entailment relations between sentences. Experimental results with two evaluation datasets show that my approach achieved higher accuracy in predicting semantic textual similarity than previous logic-based approaches that ignore these proof processes. Furthermore, my approach achieved state-of-the-art performance in predicting textual entailment. The evaluation results indicate that features derived from proof processes are effective for learning both textual entailment and similarity.

To solve the second problem of considering phrasal knowledge in natural language inferences, I propose a phrase abduction mechanism. In consideration of phrasal knowledge in logical inferences, it is important to capture phrase-to-phrase relations included in a sentence pair. To do that, my method identifies phrase correspondences through natural deduction proofs of semantic relations for a given sentence pair. Experimental results showed that extracted paraphrases using proof processes improves the accuracy of the RTE task.

The contributions of this thesis are summarized as follows:

1. I propose a new hybrid approach to learning semantic textual similarity from proof processes to obtain high performance for the STS task (Yanaka et al., 2017).

2. I show that my hybrid approach is general and also can be applied to learning textual entailment, thereby obtaining the highest performance for the RTE task (Yanaka et al., 2018b).

3. I propose a new method to detect phrase correspondences using proof processes of a semantic relation between sentences (Yanaka et al., 2018a).

1.5 Thesis organization

This thesis is structured as follows.

Chapter 2 provides a common background on how to obtain semantic representations of sentences and how to judge their semantic relations using natural deduction proofs. First, I introduce previous works on semantic parsing systems and inference systems and describe the motivation for selecting Combinatory Categorial Grammar (CCG) (Steedman, 2000) for syntax and higher-order logical formulas for semantics. Next, I describe how to translate sentences to their logical representations via CCG syntactic parsing and semantic parsing. Lastly, I explain natural deduction proofs used for proving semantic relations between sentences.

In Chapter 3, I address my hybrid approach to learning textual entailment and similarity by combining shallow features with features extracted from natural deduction proofs. First, I discuss some related works on learning textual entailment and similarity. Then, I explain how to conduct proofs and extract features that are predictive of both textual entailment and similarity in my approach. Lastly, I evaluate my hybrid approach to both STS and RTE tasks and illustrate performance comparisons among the extracted features.

In Chapter 4, I address the phrase abduction mechanism to inject phrasal knowledge for natural language inference. First, I discuss previous inference systems combined with lexical knowledge and previous paraphrase identification approaches. Next, I describe how I attempt a proof with the phrase abduction mechanism. Lastly, I evaluate the phrase abduction mechanism for the RTE task and present analysis results of the extracted phrases.

Chapter 5 concludes the thesis with a discussion of open issues and future research.

Chapter 2

Background

In this chapter, I introduce common background about how to translate sentences into their semantic representations and how to capture their semantic relations by using logical inference.

To describe the motivation for selecting logical representations for semantic representations, I first introduce two main streams for representing the meaning of natural languages: *distributional semantics* and *logical semantics*. In distributional semantics, which has been studied in computational linguistics, the meaning of a word is induced based on its usage in large corpora, and the meaning of a sentence is composed of the meanings of the words. On the other hand, in logical semantics, which has been studied in formal semantics for a long time, the meaning of a sentence is described using logical formulas. While distributional semantics has been successful in modeling the meanings of content words, logical semantics is necessary to capture the meaning of functional words.

Considering these two observations, I selected logical representations as semantic representations of sentences for the following two reasons. First, as described above, logical representations are more expressive in representing the meaning of natural languages. This is described by Lewis and Steedman (2013):

> Semantic operators, such as determiners, negation, conjunctions, modals, tense, mood, aspect, and plurals are ubiquitous in natural language, and are crucial for high performance on many practical applications—but current distributional models struggle to capture even simple examples.

As logical representations can represent the meanings of these logical or functional operators, they potentially cover the wide variety of natural language phenomena. The second reason is concerned with logical inference: logical representations easily

connect to logical inference, which captures semantic relations between sentences more deeply and robustly.

2.1 Related work about semantic composition

Logical representations for semantic representations have been widely studied in the fields of both linguistics and natural language processing. Table 2.1 shows comparisons with several works about semantic composition based on formal language theory.

	Syntax	Semantics	Logic	Prover	Language
Bos and Katja(2005) Bjerva et al. (2014)	CCG	ND/DRT	FOL	FOL provers	English
Beltagy (2016)	CCG	ND/DRT	FOL	Resolution (MRS)	English
Abzianidze (2015)	CCG	LLF	Natural Logic	Tableau	English
Mineshima et al. (2015)	CCG	ND	HOL	Natural Deduction	English Japanese
Moot (2010)	TLG	ND/DRT	FOL	N/A	French
Butler et al. (2012)	PCFG	D/SCT	FOL	N/A	English Japanese

TABLE 2.1: Comparison among semantics.
D: Davidsonian Event Semantics,
ND: Neo-Davidsonian Event Semantics, LLF: Lambda Logical Form.

In previous semantic parsing systems, the semantic parsing system developed by Moot (2010) is based on Type Logical Grammar (TLG) (Carpenter, 1998) for syntax, and Discourse Representation Theory (DRT) (Kamp and Reyle, 1993) for semantics. The supported language of this system is limited to French. Butler and Yoshimoto (2012)'s system uses Probabilistic Context-Free Grammar (PCFG) (Johnson, 1998) for syntax and Davidsonian Event Semantics (Davidson, 1967) combined with Scope Control Theory (SCT) (Butler, 2010) for semantics. This semantic parser is available for both English and Japanese. In PCFG, a probability is assigned to each production rule. However, these two semantic parsing systems are not directly connected with an inference component.

In previous semantic parsing systems connected with specific provers, the inference system developed by Bos and Markert (2005) and Bjerva et al. (2014) uses the semantic parsing system Boxer (Bos, 2008) to obtain logical semantic representations and uses multiple theorem provers and model builders for inference in first-order logic. In Boxer, CCG is used for syntax and DRT is used for semantics.

The inference system developed by Beltagy et al. (2016) also uses Boxer for logical semantic representations. In this system, modified resolution principle (Robinson, 1965) is used for inference in first-order logic.

The inference system of Abzianidze (2015) and Abzianidze (2016) (LangPro) uses CCG for syntax, natural logic for semantics, and a tableau prover for inference in higher-order logic. The supported language of this system is English.

The inference system of Mineshima et al. (2015) and Mineshima et al. (2016) (ccg2lambda) uses CCG for syntax, higher-order logic combined with Neo-Davidsonian Event Semantics (Parsons, 1990) for semantics, and natural deduction for inference in higher-order logic. The supported languages of this system include both Japanese and English.

Regarding an inference system not based on formal language theory, a dependency-based logical inference system (Tian, Miyao, and Matsuzaki, 2014; Dong, Tian, and Miyao, 2014) has been proposed. In dependency-based models, a sentence is mapped to its dependency structure, which is represented by linguistic units (e.g., words) connected to each other by directed links. Recently, dependency parsers (Chen and Manning, 2014) have been developed, which enables mapping sentences to their logical forms. However, dependency-based logical forms are less expressive because they do not capture all the linguistic phenomena that formal semantics introduced in Table 2.1 can represent. The main limitation of this logical representation is that it fails to represent any phenomena that require scope such as negation (e.g., not, no) and relative clauses (*wh*-clauses). Recent work using dependency parsers (Reddy et al., 2016a) improved this limitation, but this system needs special rules for the translation of conjunctions, relative clauses, and *wh*-questions.

In consideration of these semantic parsers and logical inference systems, I follow ccg2lambda (Mineshima et al., 2015; Mineshima et al., 2016) and use CCG for syntax and higher-order logic for semantics in this thesis. There are four motivations for selecting CCG for syntax. The first motivation is the existence of multiple robust CCG parsers. The second motivation is the transparency between syntactic categories and semantic types for semantic composition. The third motivation is the maintainability of semantic composition with a small number of combinatory rules. The fourth motivation is the expressive power of CCG for a wide range of linguistic phenomena.

In terms of semantics, Abzianidze (2016)'s semantic representations are based

on natural logic. Natural language inference based on natural logic was first elaborated by MacCartney and Manning (2007) and developed by MacCartney (2009). There are three characteristics of natural logic (Lakoff, 1970). The first characteristic is that natural logic uses formulas that resemble linguistic surface forms. The second characteristic is that natural logic is able to express linguistic semantics. The third characteristic is that natural logic can be used for the valid inference of natural language. These characteristics show that the reasoning and the grammar of natural languages are strongly related to each other in natural logic. Thus, natural logic is capable of directly reasoning about monotonicity. However, the previous natural logic based system (MacCartney, 2009) was limited to single-premise inference. The system developed by Abzianidze (2016) addresses this problem and enables semantic representation for a wide range of linguistic phenomena by natural logic. Although the expressive power of this system is comparable to that of my system, the natural logic used in this system is in a non-standard form, which causes two problems. The first problem is that semantic composition is rule-based and a lot of pre/post-processing for obtaining semantic representations is required in CCG syntactic analysis. The second problem is that the natural logic requires the definition of new inference rules for each logical or functional word, such as *every*, *all*, and *no* and for which generic theorem provers are not reusable. Also, the second problem makes proving processes complex, which is unsuitable for extracting proving processes or injecting lexical knowledge in my method.

In contrast with the systems developed by Bos (2008), Bjerva et al. (2014), Beltagy et al. (2016), Moot (2010), and Butler and Yoshimoto (2012), I select higher-order logical formulas, which are more expressive than first-order logical formulas. As an example comparison between higher-order logic and first-order logic, consider the following sentence pair.

(7) a. *Some student might come*

b. *Some student comes*

In this example, the text (7a) includes the modal auxiliary expression, *might*. In such a case, we have to distinguish modal contexts from actual contexts and we have to

infer that the text (7a) does not entail the hypothesis (7b). Here, the first-order representation[1] and higher-order representation of the text (7a) are described as follows:

FOL: $\exists x(\mathbf{student}(w_0, x) \wedge \exists w_1(Rw_0w_1 \wedge \mathbf{come}(w_1, x))$

HOL: $\exists x(\mathbf{student}(x) \wedge \mathbf{might}(\mathbf{come}(x)))$

The first-order representation and higher-order representation of the hypothesis (7b) can be described as follows:

FOL: $\exists x(\mathbf{student}(w_0, x) \wedge \mathbf{come}(w_0, x))$

HOL: $\exists x(\mathbf{student}(x) \wedge \mathbf{come}(x))$

Compared with the higher-order representation, we can see that the first-order representation introduces additional quantifiers and variables, which induces the complexity in its semantic representations and logical inference.

Furthermore, higher-order logic is more expressive than first-order logic in other linguistic phenomena, e.g., generalized quantifier, veridical and anti-veridical predicates, attitude verbs, and non-affirmative adjectives. Therefore, in the formal semantics of natural language, it is generally assumed that adequate semantic representations of natural language demand higher-order logic or type theory (Carpenter, 1998). This expressiveness of higher-order logic also contributes to the possibility of extracting more features about semantic representations in learning textual entailment and similarity, which I describe in Chapter 3. While it is thought that inference based on higher-order logic is hopelessly inefficient for practical applications (Bos, 2009), ccg2lambda (Mineshima et al., 2015; Mineshima et al., 2016) uses a proof-assistant Coq (Bertot and Castran, 2010) for efficient theorem-proving and thus the inference speed is competitive with first-order logic based inference systems.

In summary, there are two motivations for selecting higher-order logical formulas for semantic representations. The first motivation is its maintainability of semantic representation. The second motivation is its adequacy in formal semantics of natural language.

[1]Formulas of modal logic can be translated into formulas of first-order logic by standard translation (Blackburn, Rijke, and Venema, 2001). In this representation, modal operators are described as world variables (e.g., w_0, w_1) and its relation R.

2.2 Syntax and semantics

In this section, I describe how to translate sentences to higher-order logical formulas via CCG syntactic parsing and semantic parsing by using ccg2lambda (Mineshima et al., 2015; Mineshima et al., 2016). As a preliminary objective to map sentences to their semantic representations, syntactic parsing is performed. To convert syntax structures to logical forms, I assume a transparent interface between syntax and semantics following Montague (1973). To represent the meaning of natural language by logical representations robustly, I formalize the syntax with CCG (Steedman, 2000) and the semantics with lambda calculus.

CCG is a modern syntactic theory that provides a completely transparent interface between syntax and semantics. The meaning of natural languages can be obtained based on a lexicon consisting of triplets of the form (i.e., a word, its CCG category, and its semantic representation) and a small number of combinatory rules for CCG trees. Semantic templates give a lexical entry for each word, which consists of a syntactic category in CCG and a semantic representation described as lambda terms.

There are two forms of syntactic categories. One is a basic category such as N (noun), NP (noun phrase), PP (preposition phrase), and S (sentence). The other is a functional category described as the form of X/Y or $X\backslash Y$, which defines a functor with an argument Y and a result X, representing meta-variables over syntactic categories. $X\backslash Y$ indicates a function that returns X when it is combined with Y from its left hand side, and X/Y is combined with Y from its right hand side to become X.

Combinatory rules specify the syntactic behaviors of words and compositional rules for the CCG trees. Figure 2.1 shows some combinatory rules of CCG. For example, applying the backward application rule ($>$), one word, which has X/Y as its syntactic category and f as its meaning is combined with the other word, which has Y as its syntactic category and a as its meaning. This gives rise to a word with X as its syntactic category and fa as its meaning.

As an example of CCG derivation trees, Figure 2.2 shows the CCG derivation tree of the sentence "*A woman sings loudly*." In this CCG derivation tree, "*a woman*" has a category NP and "*sings loudly*" has a category $S\backslash NP$, which is combined with NP on the left hand side by applying the forward application rule ($<$), resulting in a sentence S.

$$\frac{X/Y : f \quad Y : a}{X : fa} > \qquad \frac{Y : a \quad X\backslash Y : f}{X : fa} <$$

$$\frac{f : X/Y \quad g : Y/Z}{\lambda x.f(gx) : X/Z} >\mathbf{B} \qquad \frac{g : Y\backslash Z \quad f : X\backslash Y}{\lambda x.f(gx) : X\backslash Z} <\mathbf{B}$$

$$\frac{f : X/Y \quad g : (Y/W)/Z}{\lambda z.\lambda w.f(gzw) : (X/W)/Z} >\mathbf{B}^2 \qquad \frac{g : (Y\backslash W)\backslash Z \quad f : X\backslash Y}{\lambda z.\lambda w.f(gzw) : (X\backslash W)\backslash Z} <\mathbf{B}^2$$

$$\frac{f : X/Y \quad g : Y\backslash Z}{\lambda x.f(gx) : X\backslash Z} >\mathbf{B}_\times \qquad \frac{g : Y/Z \quad f : X\backslash Y}{\lambda x.f(gx) : X/Z} <\mathbf{B}_\times$$

FIGURE 2.1: Some combinatory rules of CCG: forward/backward application rules ($<, >$), forward/backward composition rules ($> \mathbf{B}, < \mathbf{B}$), generalized forward/backward composition rules ($> \mathbf{B}^2, < \mathbf{B}^2$), forward/backward crossed composition rules ($> \mathbf{B}_\times, < \mathbf{B}_\times$).

$$\dfrac{\dfrac{\dfrac{\text{A}}{NP/N} \quad \dfrac{\text{woman}}{N}}{NP} > \quad \dfrac{\dfrac{\text{sings}}{S\backslash NP} \quad \dfrac{\text{loudly}}{(S\backslash NP)\backslash(S\backslash NP)}}{S\backslash NP} <}{S} <$$

FIGURE 2.2: An example of CCG derivation tree.

For parsing sentences into CCG syntactic trees, I use statistical CCG parsers trained on CCGBank (Hockenmaier and Steedman, 2007), which is a large treebank of CCG derivations semi-automatically obtained from phrase-structure trees of the Penn Treebank (Marcus, Marcinkiewicz, and Santorini, 1993). Lexical categories used in the CCG parsers are derived from those of CCGBank. For example, S_{dcl}, S_{ng} and S_{pss} categories are intended for declarative, gerund and passive sentences, respectively. While combinatory rules used in the CCG parsers are different from each other, they commonly use the basic combinatory rules from Steedman (2012): a forward application rule, a backward application rule, a forward composition rule, a backward crossed composition rule, a generalized forward composition rule, and a generalized backward crossed composition rule.

In the development of previous systems for deriving semantic representations based on CCG (Beltagy et al., 2016; Martínez-Gómez et al., 2017), it has been shown that the accuracy of CCG parsing can be improved by combining multiple CCG parsers. In my system, three wide-coverage CCG parsers, C&C (Clark and Curran, 2007), EasyCCG (Lewis and Steedman, 2014) and depccg (Yoshikawa, Noji, and Matsumoto, 2017) are used for converting tokenized sentences into CCG syntactic trees. While other systems based on CCG use only two CCG parsers, I combine three CCG parsers and thus I can obtain correct parsing results more robustly.

After obtaining each parser's CCG derivation tree, I map it into its semantic representation. CCG parsers sometimes produce parsing errors due to the wrong part-of-speech (POS) tag being applied in preprocessing of annotating POS tags to target sentences. Therefore, I select parsing results that succeed in mapping input sentences to their semantic representations. When the output of all parsers can be proved, I select a parsing result following the order of parsing accuracy on CCGBank: depccg, EasyCCG and C&C.

Other alternative syntactic theories include Lexical-Functional Grammar (LFG) (Kaplan and Bresnan, 1995), Head-Driven Phrase Structure Grammar (HPSG) (Pollard and Sag, 1994) and Tree-adjoining Grammars (TAG) (Joshi and Schabes, 1997). However, compared with CCG, these formalisms are not explored for grounded semantic parsing, and this can be an obstacle to obtaining semantic representations robustly.

Furthermore, there are two other motivations for selecting CCG as syntax. First, CCG can handle complex syntactic phenomena such as unbounded and long-range dependencies, coordination, and non-projective constructions without additional post-processing steps (Clark, Hockenmaier, and Steedman, 2002). Specifically, with functional categories and a small number of combinatory rules, CCG can handle a wide range of bounded and unbounded dependencies, whether a dependency relation is in the same tensed clause as its head or outside the clause, respectively. Second, as each syntactic derivation corresponds directly to a lambda-calculus structure that represents the meaning of a word, CCG is suitable for semantic composition from syntactic structures.

Next, I describe the details of semantic parsing used in this thesis. For semantic representations, I use higher-order logic combined with Neo-Davidsonian Event Semantics (Parsons, 1990). For example, a sentence containing a transitive verb in (8a), a quantifier in (9a), and a quantifier with negation in (10a) is analyzed as follows:

(8) a. *Bob surprised Susan*

b. $\exists y_1(\mathbf{surprise}(y_1) \wedge (\mathbf{subj}(y_1) = \mathbf{bob}) \wedge (\mathbf{dobj}(y_1) = \mathbf{susan}))$

(9) a. *Some women are singing loudly*

b. $\exists x_1(\mathbf{woman}(x_1) \wedge \exists y_1(\mathbf{sing}(y_1) \wedge (\mathbf{subj}(y_1) = x_1) \wedge \mathbf{loudly}(y_1)))$

(10) a. *No women are singing loudly*

b. $\neg\exists x_1(\mathbf{woman}(x_1) \wedge \exists y_1(\mathbf{sing}(y_1) \wedge (\mathbf{subj}(y_1) = x_1) \wedge \mathbf{loudly}(y_1)))$

In these examples, we can see every verb is decomposed into a one-place predicate over events. Adverbs and prepositions are also represented as predicates over events. In this thesis, I use x_i as a variable for entities and y_i for events. A set of functional expressions such as the subject (**subj**) and the direct object (**dobj**) is related to the events. For example, I describe the fact that the term x_1 is the subject of the verb represented by the event y_1 as $\mathbf{subj}(y_1) = x_1$. I also describe the fact that the term x_2 is the direct object of the verb represented by the event y_1 as $\mathbf{dobj}(y_1) = x_2$. These functional expressions are called as semantic (thematic) roles, which define relations between an event and its term. Some definitions about semantic roles between an event and its term have been proposed (Peter and Wilkins, 1984; Ray, 1990). In this thesis, I reflect relations which can be detected from CCG derivation trees based on CCGBank to semantic representations.

This analysis provides a uniform way of capturing the semantic relation (e.g., hypernym, synonym) between verbs. Consider the following two sentences and their semantic representations:

(11) a. *A woman boils onions*

b. $\exists y_1(\mathbf{boil}(y_1) \wedge (\mathbf{subj}(y_1) = \mathbf{woman}) \wedge (\mathbf{dobj}(y_1) = \mathbf{onion}))$

(12) a. *A woman cooks*

b. $\exists y_1(\mathbf{cook}(y_1) \wedge (\mathbf{subj}(y_1) = \mathbf{woman}) \wedge (\mathbf{dobj}(y_1) = \mathbf{onion}))$

For instance, the hypernym relation between the verb *boil* and the verb *cook* is represented as a simple axiom $\forall y(\mathbf{boil}(y) \rightarrow \mathbf{cook}(y))$. This is possible because both verbs are analyzed as one-place predicates over events, rather than as predicates with different arities such as $\mathbf{boil}(x_1, y_1)$ and $\mathbf{cook}(y_1)$. In addition, both arguments and adjuncts of verb phrases are analyzed as event predicates in this semantics and thus Neo-Davidsonian Event Semantics can represent the meaning more flexibly and expressively than previous Davidsonian Event Semantics (Davidson, 1967). For these reasons, this semantics is suitable for natural language inference with lexical knowledge.

A semantic representation of each word is described as lambda terms defined in semantic templates. Semantic representations of a sentence are obtained by combining each lambda term of a word in accordance with the combinatory rules specified in the CCG tree and then by applying β-conversion. As an example of a semantic

representation, Figure 2.3 shows the CCG derivation tree and semantic representation of the sentence "*No women are singing loudly.*". We can see that the final formula in Figure 2.3 is equal to the formula (10b).

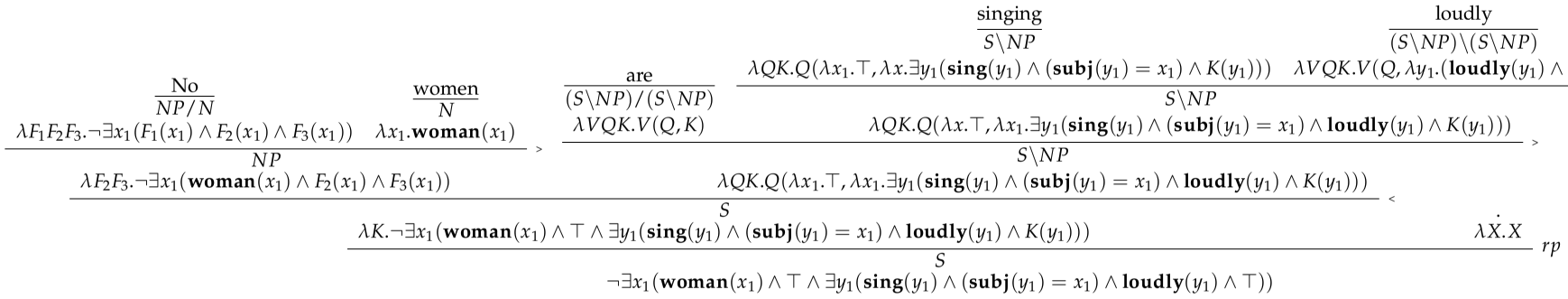

FIGURE 2.3: A CCG derivation tree and semantic representation of the sentence *No women are singing loudly.*

Here, I describe how we obtain the semantic representation in Figure 2.3 and its implementation. The set of semantic templates is defined manually based on formal semantics in a YAML file. In this thesis, the validation of semantic templates used for semantic representations was carried out exclusively on the trial split of the SICK dataset, which is used for the main evaluation (see the details in Section 3.5.1).

The template applies to a node of the CCG derivation tree. Each template has two required attributes: `category` and `semantics`. The attribute `category` is a CCG syntactic category, and the attribute `semantics` is a lambda term in NLTK semantics format (Garrette and Klein, 2009). The example of the semantic template is as follows:

```
- category : N

  semantics : λE.λx.E(x)
```

Applying this semantic template to a leaf whose base word is *woman* and whose syntactic category is N would produce the expression $(\lambda E.\lambda x.E(x))(\mathit{woman})$, which is β-reduced to $\lambda x.\mathbf{woman}(x)$. Here, the base form "*woman*" substitutes all occurrences of the variable E in the semantic expression.

When a template is applied to a CCG inner node (i.e., a node with children), lambda abstraction is applied to the semantics of the children.

```
- category : NP/N

  semantics : λE.λF1.λF2.λF3.¬∃x.(F1(x) ∧ F2(x) ∧ F3(x))

  surf : no
```

In Figure 2.3, the template above produces a rule for the surface word *no* from N to negative NP, and when applied to the CCG node whose child's semantics is $\lambda x_1.\mathbf{woman}(x_1)$, it will produce, after β-reduction, the formula $\lambda F_2 F_3.\neg\exists x_1.(\mathbf{woman}(x_1) \wedge F_2(x) \wedge F_3(x_1))$. Here, the child's semantics $\lambda x_1.\mathbf{woman}(x_1)$ substitutes all occurrences of the variable F_1. The newly composed semantic representation $\lambda F_2 F_3.\neg\exists x_1.(\mathbf{woman}(x_1) \wedge F_2(x_1) \wedge F_3(x_1))$ now expects another predicate (a verb) as an argument F (i.e., **sing**), which will be filled in the next step of the composition.

In this thesis, I assign to determiners a semantic term that has an extra predicate variable to derive correct truth conditions for quantificational sentences following

Mineshima et al. (2015). For example, the semantics of a determiner *every* can be described as $\lambda F \lambda G \lambda H. \forall x(Fx \wedge Gx \rightarrow Hx)$, in a way that is similar to the continuation semantics for event predicates (Champollion, 2015). The extra predicate variable G can be filled by the semantically empty predicate $\lambda x. \top$ (where $\top$ denotes the tautology) in a verb phrase.

2.3 Natural deduction

This section introduces how to evaluate the semantic relation between sentences. To capture the semantic relation between two sentences represented by logical formulas, I use logical inference. It remains unclear which logic is expressive enough to represent accurate natural language inference. For logical inference, I use a Gentzen-style natural deduction system (Prawitz, 1965; Troelstra and Schwichtenberg, 2000). The first motivation for using natural deduction is that it can potentially extend to higher-order logical inference. In formal semantics of natural language, it is generally assumed that adequate semantic representations of natural language demand higher-order logic or type theory (Carpenter, 1998). Thus, natural deduction is suitable for natural language inference based on standard higher-order logic (type theory). The second motivation is that natural deduction system is suitable for injecting axioms from external knowledge to fill the gap between a premise and a conclusion during the theorem-proving process (Martínez-Gómez et al., 2017). When I infer the semantic relation between sentences, I inject external knowledge such as commonsense or lexical knowledge if necessary. Injecting axioms during the theorem-proving process demonstrates such natural language inference.

In this section, subsection 2.3.1 explains the outline of natural deduction proofs. Subsection 2.3.2 describes how to implement natural deduction proof. Subsection 2.3.3 introduces inference rules used in my proof strategy. Subsection 2.3.4 describes proof processes for proving the entailment relation. Finally, subsection 2.3.5 describes proof processes for proving the contradiction.

2.3.1 Outline of natural deduction

Natural deduction systems, a type of logical systems, retain the "natural form" of logic and do not restrict themselves to any subset of the connectives nor any normal form representation (Pelletier, 1999; Pelletier and Hazen, 2012).

Another characteristic of natural deduction systems is that they have two types of inference rules: introduction rules and elimination rules. Introduction rules are the rules for installing new main operators in formulas, while elimination rules are the rules for reducing the complexity of formulas by stripping off the main logical operator. For example, the introduction rule and the elimination rule for an implication $\rightarrow$ are described in Figure 2.4.

$$\frac{A \quad A \rightarrow B}{B}\ {\rightarrow}E \qquad\qquad \frac{\begin{array}{c}[A]\\ \vdots \\ B\end{array}}{A \rightarrow B}\ {\rightarrow}I$$

FIGURE 2.4: Introduction rule (I-rule) and elimination rule (E-rule) for implication ($\rightarrow$).

In these two proof trees, material above the horizontal line represents the *premises*, and that below represents the *conclusion* (or *goal*) of the inference. The elimination rule for an implication $\rightarrow$ states that if two premises A and $A \rightarrow B$ exist, we can assure B. This rule is also known as *modus ponens*. The introduction rule for an implication $\rightarrow$ states that if we have one temporary assumption A and succeed in deriving B, then this temporary assumption is closed or "discharged" and the conclusion to $A \rightarrow B$ is made.

In using a natural deduction system for natural language inference, we first map a sentence pair (T, H) to a pair of logical formulas (T', H'). T' is initially set to the premise P and H' is set to the goal G to be proved. The typical proving strategy is to decompose them into a set of subformulas with no logical connectives using inference rules. The premise P is decomposed into a pool of premises $\mathcal{P} = \{\mathbf{p}_i(\theta_i) \mid i \in \{1, \ldots, m\}\}$, where each $\mathbf{p}_i(\theta_i)$ is a subformula and θ_i is a list of terms appearing in $\mathbf{p}_i(\theta_i)$. The goal G is also decomposed into a set of sub-goals $\mathcal{G} = \{\mathbf{g}_j(\theta'_j) \mid j \in \{1, \ldots, n\}\}$, where θ'_j is a list of terms appearing in $\mathbf{g}_j(\theta'_j)$. The proof is performed by searching for a premise $\mathbf{p}_i(\theta_i)$ whose predicate matches that of a sub-goal $\mathbf{g}_j(\theta'_j)$. If such a premise is found, then variables in θ'_j are unified to those in θ_i and the sub-goal $\mathbf{g}_j(\theta'_j)$ can be removed from $\mathcal{G}$. If all the sub-goals can be removed, we prove $T' \rightarrow H'$. In the presence of two or more variables with the same predicate, there might be multiple possible variable unifications. Modern theorem provers explore these multiple possibilities in search of a configuration that proves a theorem.

2.3.2 Proof assistant

A proof assistant is a computer system that assists the users to develop formal proofs. For a natural deduction theorem proving, I use the proof assistant Coq (Bertot and Castran, 2010), which can be used for efficient theorem-proving for natural language inference using both first-order and higher-order logic (Mineshima et al., 2015). I use Coq's built-in tactics for first-order inference. The additional axioms and tactics specialized for natural language constructions can be written in Ltac (Delahaye, 2000). We can run Coq with full automation by combining a set of pre-defined tactics with user-defined proof-search tactics and using its interactive mode. This achieves efficient automatic inference.

2.3.3 Inference rule

In a natural deduction proof, the premise P and the goal G are decomposed according to inference rules. Figure 2.5 shows the major inference rules used in this thesis. As described in Subsection 2.3, inference rules in natural deduction are divided into two types: introduction rules and elimination rules. In terms of decomposing a premise and a goal, introduction rules specify how to prove a formula in the goal, decomposing a goal formula into smaller sub-goals. Elimination rules specify how to use a premise, decomposing a formula in the pool of premises into smaller ones.

For example, the introduction rule for implication ($\rightarrow$-INTRO) states that if the sub-goal is an implication of the form $A \rightarrow B$, this sub-goal $A \rightarrow B$ can be divided into one premise A and one sub-goal B.

Note that we have to consider *eigenvariable condition* in inference rules for universal quantifier ($\forall$) and existential quantifier ($\exists$) in Figure 2.5. In the $\forall$-Intro rule, $y \equiv x$ or $y \notin fv(\mathcal{A})$, and y is not free in any assumption open in the premise, where $fv(\mathcal{A})$ is the set of free variables in $\mathcal{A}$. Also, in the $\exists$-Elim rule, $y \equiv x$ or $y \notin fv(\mathcal{A})$, and y is not free in the goal nor in any assumption open in the premise except in $\mathcal{A}[x/y]$. See Troelstra and Schwichtenberg (2000) for more details on the treatment of other logical operators.

$$\begin{array}{l} G : \mathcal{A} \wedge \mathcal{B} \\ \quad \downarrow \wedge\text{-INTRO} \\ G_1 : \mathcal{A} \\ G_2 : \mathcal{B} \end{array} \qquad \begin{array}{l} P : \mathcal{A}_1 \wedge \mathcal{A}_2 \wedge \cdots \wedge \mathcal{A}_n \\ \quad \downarrow \wedge\text{-ELIM} \\ P_1 : \mathcal{A}_1,\ P_2 : \mathcal{A}_2,\ \ldots,\ P_n : \mathcal{A}_n \end{array}$$

$$\begin{array}{l} G : \mathcal{A} \to \mathcal{B} \\ \quad \downarrow \to\text{-INTRO} \\ \dfrac{P : \mathcal{A}}{G : \mathcal{B}} \end{array} \qquad \begin{array}{l} P_1 : \mathcal{A} \to \mathcal{B} \\ P_2 : \mathcal{A} \\ \quad \downarrow \to\text{-ELIM} \\ P : \mathcal{B} \end{array}$$

$$\begin{array}{l} G : \exists x \mathcal{A}(x) \\ \quad \downarrow \exists\text{-INTRO} \\ G_1 : \mathcal{A}(t) \end{array} \qquad \begin{array}{l} P : \exists x \mathcal{A}(x) \\ \quad \downarrow \exists\text{-ELIM} \\ P_1 : \mathcal{A}(y) \end{array} \qquad \begin{array}{l} P_1 : \mathcal{A}(t) \\ P_2 : t = u \\ \quad \downarrow =\text{-ELIM} \\ P : \mathcal{A}(u) \end{array}$$

$$\begin{array}{l} \dfrac{P : \mathcal{C}}{G : \mathcal{A} \vee \mathcal{B}} \\ \quad \downarrow \vee\text{-INTRO} \\ \dfrac{P : \mathcal{C}}{G_1 : \mathcal{A}} \end{array} \qquad \begin{array}{l} \dfrac{P : \mathcal{C}}{G : \mathcal{A} \vee \mathcal{B}} \\ \quad \downarrow \vee\text{-INTRO} \\ \dfrac{P : \mathcal{C}}{G_2 : \mathcal{B}} \end{array} \qquad \begin{array}{l} P : \mathcal{A} \vee \mathcal{B} \\ \quad \downarrow \vee\text{-ELIM} \\ P_1 : \mathcal{A} \\ P_2 : \mathcal{B} \end{array}$$

$$\begin{array}{l} G : \forall x \mathcal{A}(x) \\ \quad \downarrow \forall\text{-INTRO} \\ G_1 : \mathcal{A}(y) \end{array} \qquad \begin{array}{l} P : \forall x \mathcal{A}(x) \\ \quad \downarrow \forall\text{-ELIM} \\ P_1 : \mathcal{A}(t) \end{array}$$

FIGURE 2.5: Inference rules used in natural deduction. $P, P_1, \ldots P_n$ are formulas in the premise, while G, G_1, G_2 are formulas in the goal. The initial formulas are at the top, with the formulas obtained by applying the inference rules shown below.

2.3.4 Proving entailment relation

As an illustration of how my natural deduction proof works for calculating the semantic relation between sentences, consider the case of proving the entailment relation between the following sentence pair:

(13) a. *A man is singing in a bar*

b. *A man is singing*

The sentences (13a) and (13b) are mapped onto logical formulas A' and B' based on event semantics via CCG-based semantic composition, as follows.

$$A' \ : \ \exists y_1 \exists x_1 \exists x_2 (\mathbf{man}(x_1) \wedge \mathbf{sing}(y_1) \wedge (\mathbf{subj}(y_1) = x_1) \wedge \ \mathbf{bar}(x_2) \wedge \mathbf{in}(y_1, x_2))$$

$$B' \ : \ \exists y_1 \exists x_1 (\mathbf{man}(x_1) \wedge \mathbf{sing}(y_1) \wedge (\mathbf{subj}(y_1) = x_1))$$

When we attempt a natural deduction proof of the entailment relation $A' \rightarrow B'$, we set the logical formula A' as the premise and the logical formula B' as the goal of the proof. Then A' and B' are decomposed by using inference rules described in the previous subsection.

$$\frac{P_0 : \ \exists y_1 \exists x_1 \exists x_2 (\mathbf{man}(x_1) \wedge \mathbf{sing}(y_1) \wedge (\mathbf{subj}(y_1) = x_1) \wedge \ \mathbf{bar}(x_2) \wedge \mathbf{in}(y_1, x_2))}{G_0 : \exists y_1 \exists x_1 (\mathbf{man}(x_1) \wedge \mathbf{sing}(y_1) \wedge (\mathbf{subj}(y_1) = x_1))}$$

$$\downarrow \ \exists\text{-ELIM } (P_0)\times 3 \ , \ \exists\text{-INTRO } (G_0)\times 2$$

$$\frac{P_1 : \ \mathbf{man}(x_1) \wedge \mathbf{sing}(y_1) \wedge (\mathbf{subj}(y_1) = x_1) \wedge \ \mathbf{bar}(x_2) \wedge \mathbf{in}(y_1, x_2)}{G_1 : \mathbf{man}(x_1) \wedge \mathbf{sing}(y_1) \wedge (\mathbf{subj}(y_1) = x_1)}$$

$$\downarrow \ \wedge\text{-ELIM } (P_1)\times 4, \ \wedge\text{-INTRO } (G_1)\times 2$$

$$\frac{P_2 : \ \mathbf{man}(x_1), \ P_3 : \mathbf{sing}(y_1), \ P_4 : \mathbf{subj}(y_1) = x_1, P_5 : \mathbf{bar}(x_2), \ P_6 : \mathbf{in}(y_1, x_2)}{G_2 : \mathbf{man}(x_1), \ G_3 : \mathbf{sing}(y_1), \ G_4 : \mathbf{subj}(y_1) = x_1}$$

$$\downarrow$$

$$\frac{P_2 : \ \mathbf{man}(x_1), \ P_3 : \mathbf{sing}(y_1), \ P_4 : \mathbf{subj}(y_1) = x_1, P_5 : \mathbf{bar}(x_2), \ P_6 : \mathbf{in}(y_1, x_2)}{G_2 : \cancel{\mathbf{man}(x_1)}, \ G_3 : \cancel{\mathbf{sing}(y_1)}, \ G_4 : \cancel{\mathbf{subj}(y_1) = x_1}}$$

FIGURE 2.6: The proof process for proving an entailment relation.

The proof process for $A' \rightarrow B'$ is shown in Figure 2.6. Here, A' is initially set to the premise P_0 and B' to the goal G_0. P_0 and G_0 are then decomposed using elimination rules ($\wedge$-ELIM, $\exists$-ELIM) and introduction rules ($\wedge$-INTRO, $\exists$-INTRO). Then we obtain a set of premise formulas $\mathcal{P} = \{P_2, P_3, P_4, P_5, P_6\}$ and a set of sub-goals $\mathcal{G} = \{G_2, G_3, G_4\}$. The proof is performed by searching for a premise P_i whose predicate and arguments match those of a given sub-goal G_j. If such a logical premise is found, the sub-goal is removed. In this example, the sub-goals G_2, G_3, and G_4 match the premises P_2, P_3, and P_4, respectively. Thus, $A' \rightarrow B'$ can be proved.

2.3.5 Proving contradiction

The proof strategy illustrated here can be straightforwardly applied to proving the contradiction. In natural deduction, a negative formula of the form $\neg A$ can be defined as $A \rightarrow$ **False** ("the formula A implies the contradiction"), by using a propositional constant **False** to encode the contradiction. Thus, the inference rules for negation ($\neg$) can be taken as special cases of implication rules, as shown in Figure 2.7.

$$\dfrac{\begin{array}{c} G : \neg\mathcal{A} \\ \downarrow \neg\text{-INTRO} \\ P : \mathcal{A} \end{array}}{G_1 : \textbf{False}} \qquad \dfrac{P : \neg\mathcal{A}}{\begin{array}{c} G : \textbf{False} \\ \downarrow \neg\text{-ELIM} \\ G_1 : \mathcal{A} \end{array}}$$

FIGURE 2.7: Inference rules of negation.

As an illustration of proving the contradiction, let us consider the following sentence pair:

(14) a. *No man is singing*

b. *There is a man singing loudly*

Figure 2.8 shows the proof process. The sentences (14a) and (14b) are first mapped to P_0 and P_1, respectively, via compositional semantics and the goal G_0 is set to **False**. Second, by applying the elimination rule for negation to P_0, P_0 is set to G_1. Third, by decomposing P_1 and G_1 using elimination rules ($\wedge$-ELIM, $\exists$-ELIM) and introduction rules ($\wedge$-INTRO, $\exists$-INTRO), we can obtain the set of premises $\mathcal{P} = \{P_3, P_4, P_5, P_6\}$ and the set of sub-goals $\mathcal{G} = \{G_3, G_4, G_5\}$. Lastly, from P_3, P_4 and P_5, we can remove these sub-goals and then derive the contradiction.

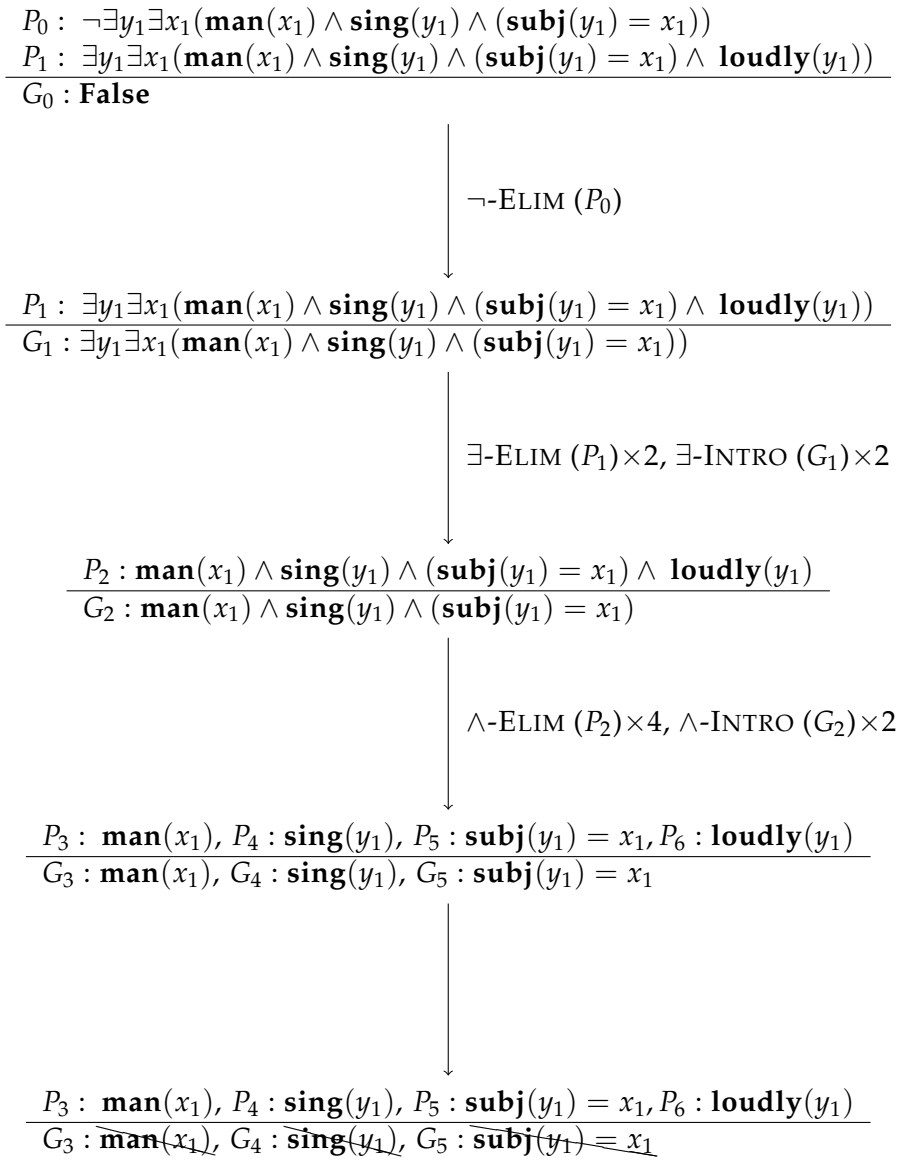

FIGURE 2.8: Proof process for proving a contradiction.

I introduce another example of proving the contradiction. Consider this sentence pair:

(15) a. *The woman is not wearing glasses or a headdress*

b. *A woman is wearing an Egyptian headdress*

Figure 2.9 shows the proof process for proving the contradiction. The sentences (15a) and (15b) are mapped to P_0 and P_1, respectively, via compositional semantics and the goal G_0 is set to **False**. Second, by applying the elimination rule for negation to P_0, P_0 is set to G_1. By decomposing P_1 and G_1 using elimination rules ($\wedge$-ELIM, $\exists$-ELIM) and introduction rules ($\wedge$-INTRO, $\exists$-INTRO), we can obtain the set of premises $\mathcal{P} = \{P_3, P_4, P_5, P_6, P_7, P_8\}$ and the set of sub-goals $\mathcal{G} = \{G_3, G_4, G_5, G_6, G_7\}$. From P_3, P_4, P_7 and P_8, we can remove the sub-goals G_3, G_4, G_6 and G_7. After applying the introduction rule for disjunction ($\vee$) to the sub-goal G_5, we can remove the sub-goal G_5 and then derive the contradiction. The difference between Figure 2.8 and Figure 2.9 is that Figure 2.9 describes how to treat the disjunctive expression *not wearing glasses or a headdress* in the sentence (15a).

P_0 : $\neg\exists e_1\exists x_1\exists x_2(\mathbf{woman}(x_1) \wedge \mathbf{wear}(e_1) \wedge (\mathbf{glass}(x_2) \vee \mathbf{headdress}(x_2)) \wedge (\mathbf{subj}(e_1) = x_1)) \wedge (\mathbf{obj}(e_1) = x_2))$
P_1 : $\exists e_1\exists x_1\exists x_2(\mathbf{woman}(x_1) \wedge \mathbf{wear}(e_1) \wedge \mathbf{egyptian}(x_2) \wedge \mathbf{headdress}(x_2) \wedge (\mathbf{subj}(e_1) = x_1) \wedge (\mathbf{obj}(e_1) = x_2))$

G_0 : **False**

$\downarrow$ $\neg$-ELIM (P_0)

P_1 : $\exists e_1\exists x_1\exists x_2(\mathbf{woman}(x_1) \wedge \mathbf{wear}(e_1) \wedge \mathbf{egyptian}(x_2) \wedge \mathbf{headdress}(x_2) \wedge (\mathbf{subj}(e_1) = x_1) \wedge (\mathbf{obj}(e_1) = x_2))$

G_1 : $\exists e_1\exists x_1\exists x_2(\mathbf{woman}(x_1) \wedge \mathbf{wear}(e_1) \wedge (\mathbf{glass}(x_2) \vee \mathbf{headdress}(x_2)) \wedge (\mathbf{subj}(e_1) = x_1)) \wedge (\mathbf{obj}(e_1) = x_2))$

$\downarrow$ $\exists$-ELIM (P_1)×3, $\exists$-ELIM (G_1)×3

P_2 : $\mathbf{woman}(x_1) \wedge \mathbf{wear}(e_1) \wedge \mathbf{egyptian}(x_2) \wedge \mathbf{headdress}(x_2) \wedge (\mathbf{subj}(e_1) = x_1) \wedge (\mathbf{obj}(e_1) = x_2)$

G_2 : $\mathbf{woman}(x_1) \wedge \mathbf{wear}(e_1) \wedge (\mathbf{glass}(x_2) \vee \mathbf{headdress}(x_2)) \wedge (\mathbf{subj}(e_1) = x_1)) \wedge (\mathbf{obj}(e_1) = x_2)$

$\downarrow$ $\wedge$-ELIM (P_2)×5, $\wedge$-INTRO (G_2)×4

$P_3 : \mathbf{woman}(x_1)$, $P_4 : \mathbf{wear}(e_1)$, $P_5 : \mathbf{egyptian}(x_2)$
$P_6 : \mathbf{headdress}(x_2)$, $P_7 : \mathbf{subj}(e_1) = x_1$, $P_8 : \mathbf{obj}(e_1) = x_2$

$G_3 : \mathbf{woman}(x_1)$, $G_4 : \mathbf{wear}(e_1)$, $G_5 : \mathbf{glass}(x_2) \vee \mathbf{headdress}(x_2)$,
$G_6 : \mathbf{subj}(e_1) = x_1$, $G_7 : \mathbf{obj}(e_1) = x_2$

$\downarrow$

$P_3 : \mathbf{woman}(x_1)$, $P_4 : \mathbf{wear}(e_1)$, $P_5 : \mathbf{egyptian}(x_2)$
$P_6 : \mathbf{headdress}(x_2)$, $P_7 : \mathbf{subj}(e_1) = x_1$, $P_8 : \mathbf{obj}(e_1) = x_2$

~~$G_3 : \mathbf{woman}(x_1)$~~, ~~$G_4 : \mathbf{wear}(e_1)$~~, $G_5 : \mathbf{glass}(x_2) \vee \mathbf{headdress}(x_2)$,
~~$G_6 : \mathbf{subj}(e_1) = x_1$~~, ~~$G_7 : \mathbf{obj}(e_1) = x_2$~~

$\downarrow$ $\vee$-INTRO (G_5)

$P_3 : \mathbf{woman}(x_1)$, $P_4 : \mathbf{wear}(e_1)$, $P_5 : \mathbf{egyptian}(x_2)$
$P_6 : \mathbf{headdress}(x_2)$, $P_7 : \mathbf{subj}(e_1) = x_1$, $P_8 : \mathbf{obj}(e_1) = x_2$

~~$G_3 : \mathbf{woman}(x_1)$~~, ~~$G_4 : \mathbf{wear}(e_1)$~~, ~~$G_5 : \mathbf{headdress}(x_2)$~~
~~$G_6 : \mathbf{subj}(e_1) = x_1$~~, ~~$G_7 : \mathbf{obj}(e_1) = x_2$~~

FIGURE 2.9: Proof process for proving a contradiction including a disjunctive expression.

2.4 Word abduction

Sub-goals may remain unproved when T logically does not entail H, i.e., when there are no premise predicates $\mathbf{p}_i$ that are matched with $\mathbf{g}_j$. In this case, the system attempts on-the-fly word axiom injection, which was first developed by Martínez-Gómez et al. (2017). I call this word axiom injection mechanism *word abduction*.

If there is a premise $\mathbf{p}_i(\theta_i)$ whose predicate has a linguistic relation (according to external linguistic knowledge) with that of a sub-goal $\mathbf{g}_j(\theta'_j)$, then variables in θ'_j are unified with those in θ_i and the sub-goal $\mathbf{g}_j(\theta'_j)$ can be removed from $\mathcal{G}$. A linguistic ontology or knowledge database is used for external linguistic knowledge. In the original word axiom injection mechanism, two linguistic knowledge databases, i.e., WordNet (Miller, 1995) and VerbOcean (Chklovski and Pantel, 2004) are used.

As an illustration of this word abduction mechanism, let us consider the following sentence pair:

(16) a. *A white and brown cat walks*

b. *A white and brown cat moves*

The proof process for proving the entailment relation between these two sentences is shown in Figure 2.10. Here, the sentence (16a) is initially map to the premise formula P_0 and the sentence (16b) to the goal formula G_0. P_0 and G_0 are then decomposed using elimination rules ($\wedge$-ELIM, $\exists$-ELIM) and introduction rules ($\wedge$-INTRO, $\exists$-INTRO). Then we obtain a set of premise formulas $\mathcal{P} = \{P_2, P_3, P_4, P_5, P_6\}$, and a set of sub-goals $\mathcal{G} = \{G_2, G_3, G_4, G_5, G_6\}$. The proof is performed by searching for a premise P_i whose predicate and arguments match those of a given sub-goal G_j. If such a logical premise is found, the sub-goal is removed. In this example, the sub-goals G_2, G_3, G_4, and G_6 match the premises P_2, P_3, P_4, and P_6, respectively.

Here, the sub-goal G_5 : $\mathbf{move}(y_1)$ is still unprovable. One can assume that *a white and brown cat* is the subject of both *move* and *walk* and that the word *move* is the hypernym of the word *walk*. Thus, the sub-goal G_5 : $\mathbf{move}(y_1)$ and the premise P_5 : $\mathbf{walk}(y_1)$ share the same variable y_1 and there is a linguistic relationship between their predicates **move** and **walk**. In such cases, word abduction takes place and the axiom $\forall x(\mathbf{walk}(x) \rightarrow \mathbf{move}(x))$ is generated. By applying the axiom, the entailment relation between the sentence (16a) and the sentence (16b) is proved.

Compared with other proof systems, such as the resolution proof systems (Beltagy et al., 2016; Bjerva et al., 2014) and the tableau proof system (Abzianidze, 2016)

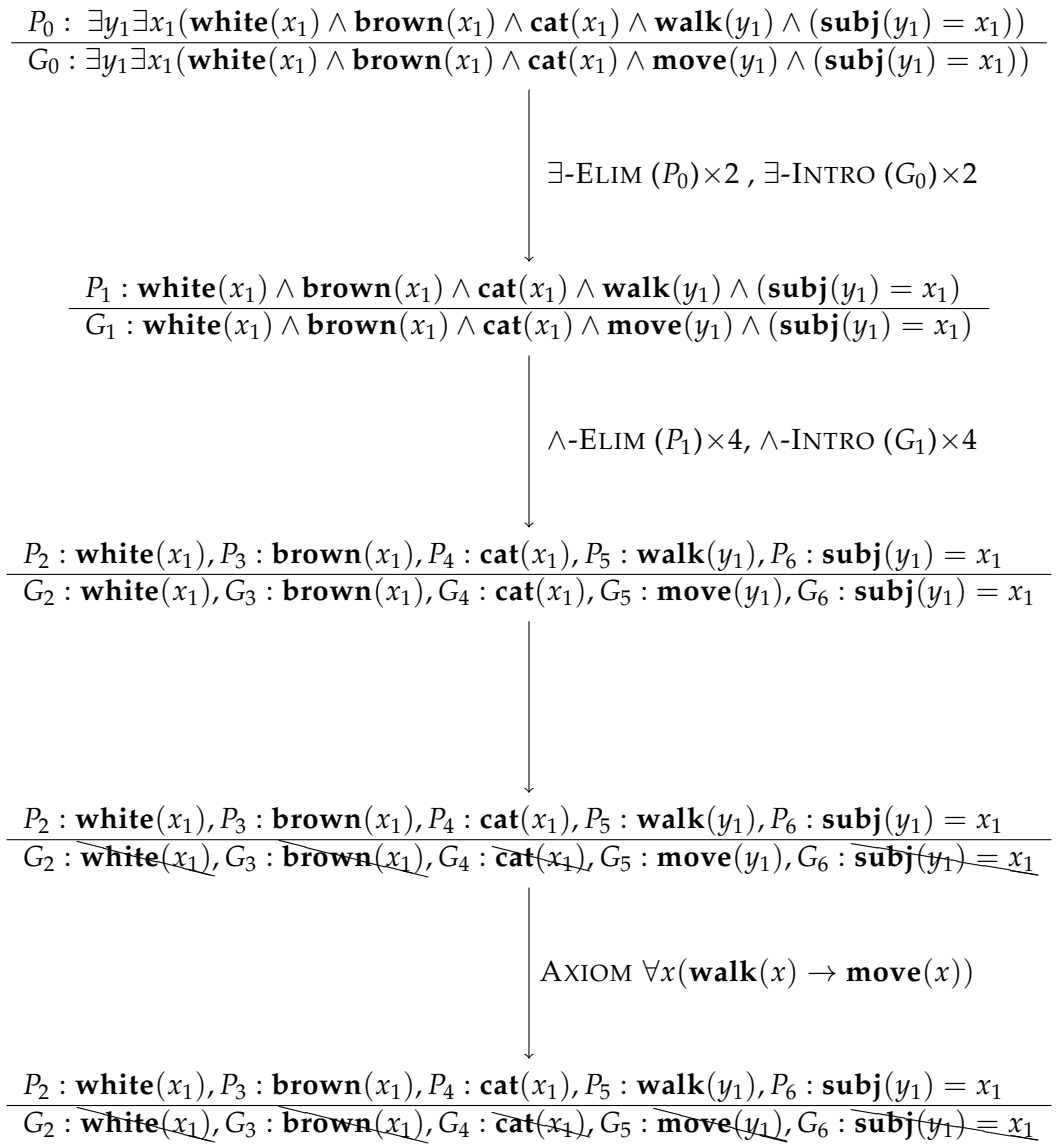

FIGURE 2.10: The proof process for word abduction.

introduced in Section 2.1, the distinction between the premise and the conclusion to be proved is always clear when carrying out the proof in the natural deduction system. Thus, the natural deduction system enables the injection of an axiom that is necessary to fill the gap between the premise and the conclusion during the proof process on-the-fly.

Chapter 3

Learning textual entailment and similarity using proof processes

3.1 Motivation and related work

In this chapter, I describe my research about learning textual entailment and similarity. There are three main approaches to capturing semantic relations between sentences: logic-based approaches, machine learning-based approaches, and hybrid approaches. First, I introduce previous studies about machine learning-based approaches, which use supervised learning algorithms. Second, I introduce previous studies about logic-based approaches, which uses logical inference systems to capture semantic relations between sentences. Third, I introduce previous studies about hybrid approaches of logic-based and machine learning-based approaches. Lastly, I describe the motivation for my hybrid approach to learning textual entailment and similarity.

3.1.1 Machine learning-based approaches

In machine learning-based approaches, vector-based models of semantic composition have been widely studied with regards to calculating semantic textual similarity (STS). Mitchell and Lapata (2008) and Mitchell and Lapata (2010) proposed a sentence vector model involving word vector addition or component-wise multiplication. Addition and multiplication are commutative and associative operations and thus ignore word order. To obtain vector-based semantic representations more precisely, Polajnar, Rimell, and Clark (2015) proposed a discourse-based sentence vector model considering extra-intra sentential context. Also, a categorical compositional distributional semantic model has been developed for recognizing textual

entailment (RTE) and for calculating STS (Grefenstette and Sadrzadeh, 2011; Kartsaklis, Kalchbrenner, and Sadrzadeh, 2014; Kartsaklis and Sadrzadeh, 2016). In this model, the dimensions of the vectors encode model-theoretic structures rather than observed co-occurrences, even though they are not strictly hybrid systems as they do not include contextual distributional information. However, these previous studies are mostly concerned with the structures of basic phrases or sentences and do not address logical and functional words such as negations and connectives. Grefenstette (2013) represents logical constructs using vectors and tensors, but concludes that they do not adequately capture logical structures, in particular, quantifiers.

Neural network-based models of semantic composition (Mueller and Thyagarajan, 2016; Hill, Cho, and Korhonen, 2016; Tai, Socher, and Manning, 2015; Zhou, Liu, and Pan, 2016; He and Lin, 2016) have also been proposed. Although these neural network-based models achieve higher performance in both RTE and STS tasks, they lack three basic and simple capabilities that we get from symbolic representations.

The first missing capability is interpretability; being able to inspect the meaning representation and interpret how the end task is performed. This is easy in logic-based approaches because the meanings of sentences are represented by logical formulas and we can detect where the logical inference fails in the middle of proof processes. However, it is difficult in deep learning approaches because the meanings of sentences are represented by vectors of real values. Thus, it is not self-evident if neural network-based model also can capture logical and functional words like logical formulas. The end-to-end nature of neural network-based models introduces challenges in the diagnosis of the reasons that make two sentences to be similar or dissimilar from each other. These diagnosis capabilities may play an important role in making the system explainable and also guide future system improvements in a more precise manner.

The second missing capability is the ability to use existing resources such as knowledge bases to assist inference. For example, neural network-based models are known to have difficulties in the detection of antonyms such as *big* and *small*. However, it is easy to identify the antonym relation between *big* and *small* using lexical knowledge such as WordNet (Miller, 1995), and we can use this knowledge in logical inference, instead of having to train on hundreds of training examples with the words *big* and *small*.

The third missing capability is that neural network-based models cannot be easily trained on small datasets and little memory. Recently, a large dataset for natural language inference (Bowman et al., 2015; Williams, Nangia, and Bowman, 2017) has been proposed for developing neural network-based models. However, considering that we apply the model to each domain-specific NLP application, we have to prepare large and cleaned domain-specific datasets respectively. In addition, whether we can prepare a large dataset that covers all linguistic phenomena is still an open problem.

3.1.2 Logic-based approaches

Purely logic-based approaches such as ccg2lambda (Mineshima et al., 2015; Martínez-Gómez et al., 2016; Mineshima et al., 2016) and LangPro (Abzianidze, 2015; Abzianidze, 2016) introduced in Section 2.1 have been successful in representing the meanings of complex sentences, having had a positive impact for RTE tasks. Another advantage of logic-based approaches is that the accuracy of logic-based approaches does not depend on the size of the dataset. That is, logic-based approaches judge the entailment relations directly through logical inference and they do not require the training of a model with a large dataset (i.e., logic-based approaches are intrinsically based on unsupervised learning). Indeed, compared with neural-network-based approaches, logic-based approaches (Mineshima et al., 2015; Abzianidze, 2016) have been successful in performing high accuracy with the FraCaS dataset (Cooper et al., 1994), which includes only about 300 cases.

However, purely logic-based approaches only provide an intrinsically binary output (i.e., if a statement is true or not), which prevents them from capturing the graded aspects of meaning in language, such as semantic textual similarity. The graded aspects of meaning are also discussed in linguistics. As for the graded aspects of meaning in language, Edmonds and Hirst (2002) write:

> Absolute synonymy, if it exists at all, is quite rare. Absolute synonyms would be able to be substituted one for the other in any context in which their common sense is denoted with no change to truth value, communicative effect, or "meaning" (however "meaning" is defined).

The authors introduce two words, *daddy* and *father*, as an example of this claim. They are known as synonyms, while *daddy* expresses a stronger feeling of intimacy

than *father*. This means the sentence *I called, "my daddy"* entails the other sentence *I called, "my father"* if we assume *daddy* and *father* as absolute synonyms, while they are dissimilar in terms of the emotion of the speaker. This means that the function for capturing the graded aspects of meanings is necessary in natural language inference. Also, capturing the graded aspects of meanings is necessary in NLP applications such as information retrieval. In these applications, huge source texts are compared with the target text to judge whether they are related or not. Under such a condition, the semantic textual relation should be handled not as a binary decision (entailment or contradiction), but as a similarity.

3.1.3 Hybrid approaches

To compensate for the problems associated with logic-based approaches, hybrid approaches of logic and machine learning have been explored. The hybrid approach was first developed by Bos and Markert (2005). They translated a text and its hypothesis into first-order logical formulas using Boxer (Bos, 2008), and employed off-the-shelf inference tools such as a theorem prover and a model builder to detect entailment relations. Unfortunately, this work also reported that the genuine theorem prover correctly proves less than 6% of the RTE problems. The main reason for its low performance is a lack of lexical and background knowledge. However, their experiments showed that the combination of a machine learning classifier and the features extracted from the inference tools are more successful than the tools alone.

Subsequently, The Meaning Factory (Bjerva et al., 2014) outperformed this previous work. This system also uses Boxer for their semantic representations and translates two sentences into first-order logical formulas. In RTE tasks, this system uses theorem provers and model builders to check if one sentence entails the other, or if the sentences are contradictory. In STS tasks, the system follows a supervised approach to solve the regression problem of determining the similarity between each given sentence pair. The system uses a variety of features, ranging from simpler ones such as word overlap, to more complex ones in the form of deep semantic features and features derived from a compositional distributional semantic model. For the form of deep semantic features, in particular, the proportion of overlap between logical formulas and the judgment output (entailment, contradiction, or neutral) from the logical inference system are used. These features are used for training a random forest regressor model, and this hybrid approach succeeded in obtaining high

performance in the STS task.

UTexas (Beltagy et al., 2014) uses Markov Logic Networks (MLNs) (Richardson and Domingos, 2006) for RTE tasks and Probabilistic Soft Logic (PSL) (Broecheler, Mihalkova, and Getoor, 2010) for STS tasks. This system also uses Boxer for their logical semantic representations. In this system, each ground atom in the logical formulas has a probability based on the distributional semantics of a word. The weights of the logical formulas are calculated from the probabilities of their ground atoms and are extracted as features. In RTE tasks, the conditional probabilities of proving an entailment relation or contradiction between sentences are calculated using these features. These probabilities are then mapped to a final relation by an SVM classifier. In STS tasks, a probability of each ground atom has a continuous truth value on the interval [0, 1] rather than a binary truth value as used in MLNs.

These previous studies improved accuracy by using logic-based features derived from logical formulas and inference results of first-order theorem proving in addition to using shallow features such as sentence lengths. However, there is still the open problem of how to effectively combine a logic-based approach with a machine learning-based approach. As described above, the previous hybrid approaches only focus on the output from their inference systems, such as the overlap of logical formulas and the (probabilistic) inference results, to assess textual entailment and similarity. However, there is much more information that can be obtained from the inference system other than the inference result. That is, the processes to obtain the final logical formulas and their inference process are also potentially valuable information about the semantic relations between sentences.

3.1.4 Motivation for using proving process

In this thesis, I determine semantic relations between sentences based on the conception of proof-theoretic semantics (Bekki and Mineshima, 2017). The key idea is that not only the inference results but also the *theorem proving process* can represent semantic relations between sentences. That is, by taking into account not only whether a theorem is proved but also *how* it is proved, we can capture semantic relations between sentences in more depth. My hypothesis is that *observing proof processes when testing the semantic relations is useful for capturing textual entailment and similarity more precisely*.

To illustrate my hypothesis about the relation between semantic relations and proving processes more concretely, I describe two examples of proving processes for proving an entailment relation. First, consider the proving process for proving an entailment relation between the following sentences (17a) and (17b):

(17) a. *A man is walking in the rain*

b. *A man is walking*

The sentences (17a) and (17b) are mapped onto logical formulas A'_1 and A'_2 based on event semantics via CCG-based semantic composition, as follows.

$$A'_1 : \exists y_1 \exists x_1 \exists x_2 (\mathbf{man}(x_1) \wedge \mathbf{walk}(y_1) \wedge \mathbf{in}(y_1, x_2) \wedge \mathbf{rain}(x_2) \wedge (\mathbf{subj}(y_1) = x_1))$$

$$A'_2 : \exists y_1 \exists x_1 (\mathbf{man}(x_1) \wedge \mathbf{walk}(y_1) \wedge (\mathbf{subj}(y_1) = x_1))$$

Here, we can see A'_1 entails A'_2. The proving process of the attempted proof $A'_1 \rightarrow A'_2$ is described in Figure 3.1. A'_1 is initially set to the premise P_0 and A'_2 to the goal G_0. the premise P_0 and the goal G_0 are then decomposed into the premise P_1 and the sub-goal G_1, respectively, using the elimination and introduction rules for existential quantifier ($\exists$-ELIM, $\exists$-INTRO). We can remove the sub-goal G_1 from the premise P_1 and prove $A'_1 \rightarrow A'_2$.

$$\frac{P_0 : \ \exists y_1 \exists x_1 \exists x_2 (\mathbf{man}(x_1) \wedge \mathbf{walk}(y_1) \wedge \mathbf{in}(y_1, x_2) \wedge \mathbf{rain}(x_2) \wedge (\mathbf{subj}(y_1) = x_1))}{G_0 : \exists y_1 \exists x_1 (\mathbf{man}(x_1) \wedge \mathbf{walk}(y_1) \wedge (\mathbf{subj}(y_1) = x_1))}$$

$$\Big\downarrow \ \exists\text{-ELIM } (P_0)\times 3,\ \exists\text{-INTRO } (G_0)\times 2$$

$$\frac{P_1 : \ \mathbf{man}(x_1) \wedge \mathbf{walk}(y_1) \wedge \mathbf{in}(y_1, x_2) \wedge \mathbf{rain}(x_2) \wedge (\mathbf{subj}(y_1) = x_1)}{G_1 : \mathbf{man}(x_1) \wedge \mathbf{walk}(y_1) \wedge (\mathbf{subj}(y_1) = x_1)}$$

$$\Big\downarrow$$

$$\frac{P_1 : \ \mathbf{man}(x_1) \wedge \mathbf{walk}(y_1) \wedge \mathbf{in}(y_1, x_2) \wedge \mathbf{rain}(x_2) \wedge (\mathbf{subj}(y_1) = x_1)}{G_1 : \mathbf{man}(x_1) \wedge \cancel{\mathbf{walk}(y_1) \wedge (\mathbf{subj}(y_1) = x_1)}}$$

FIGURE 3.1: The proof process for proving the entailment relation $A'_1 \rightarrow A'_2$.

Next, consider the proving process for proving an entailment relation between the following sentences (18a) and (18b). Note that the sentence (18a) does not logically entail the sentence (18b).

(18) a. *A man is walking in the rain*

b. *A man and a woman are singing*

Sentences (18a) and (18b) are mapped onto logical formulas A'_1 and A'_3 based on event semantics via CCG-based semantic composition, as follows.

$A'_1 : \exists y_1 \exists x_1 \exists x_2(\mathbf{man}(x_1) \wedge \mathbf{walk}(y_1) \wedge \mathbf{in}(y_1, x_2) \wedge \mathbf{rain}(x_2) \wedge (\mathbf{subj}(y_1) = x_1))$

$A'_3 : \exists y_1 \exists x_1(\mathbf{man}(x_1) \wedge \mathbf{sing}(y_1) \wedge \mathbf{woman}(x_2) \wedge \mathbf{sing}(y_2) \wedge (\mathbf{subj}(y_1) = x_1) \wedge (\mathbf{subj}(y_2) = x_2))$

If we attempt the proof $A'_1 \rightarrow A'_3$, its proof process can be described in Figure 3.2. Here, the four sub-goals G_3, G_4, G_5, and G_8 remain. However, assuming four axioms $\forall y_1(\mathbf{walk}(y_1) \rightarrow \mathbf{sing}(y_1))$, $\forall x_1 \forall x_2(\mathbf{man}(x_1) \rightarrow \mathbf{woman}(x_2))$, $\forall y_1 \forall y_2(\mathbf{walk}(y_1) \rightarrow \mathbf{sing}(y_2))$ and $\forall x_2 \forall y_2(\mathbf{subj}(y_2) = x_2))$ can be generated forcibly, despite the fact that these axioms are not logically or lexically correct, we can prove the entailment relation between A'_1 and A'_3 forcibly. These two figures show if two sentences are less similar, we have to add more axioms and apply inference rules to prove their entailment relation. This demonstrates my hypothesis that not only the proving results (i.e., entailment, contradiction, or neutral), but also the proving processes reveal information about semantic relations between sentences.

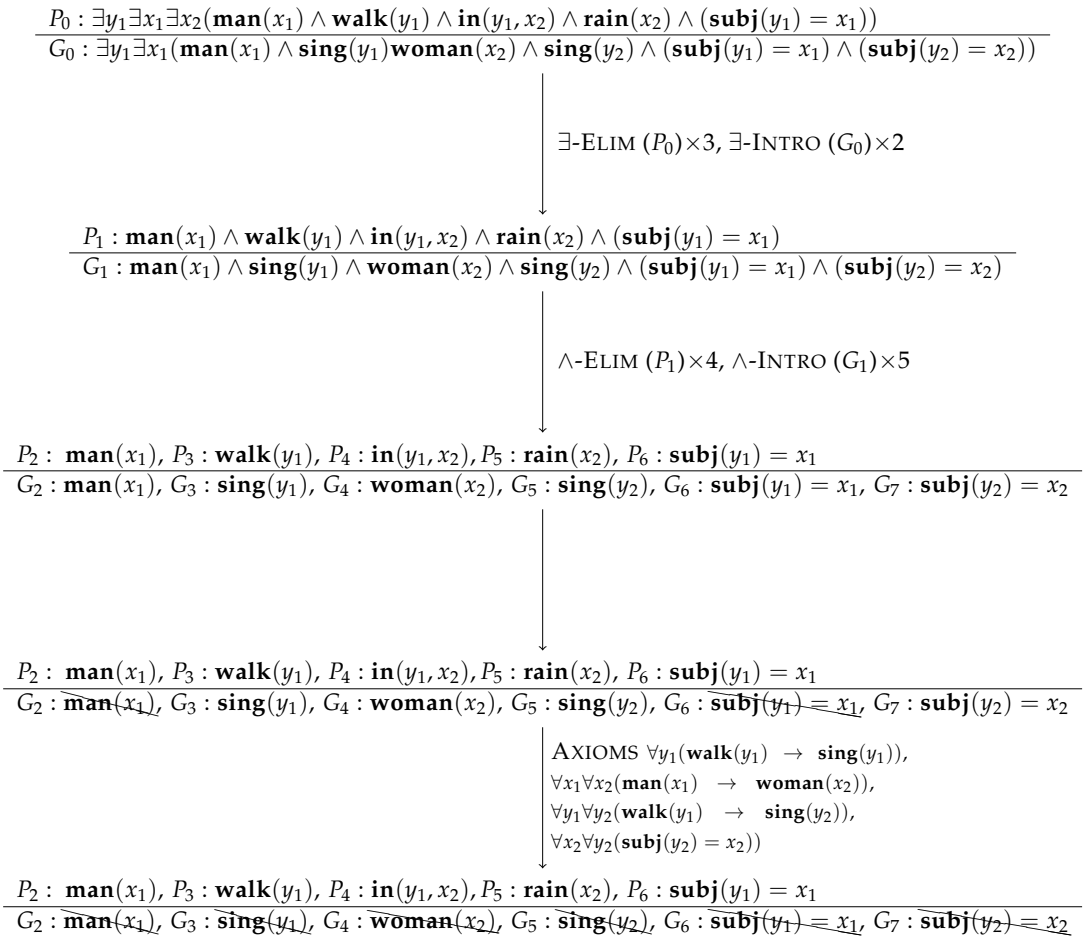

FIGURE 3.2: The proof process for proving the entailment relation $A'_1 \rightarrow A'_3$.

3.2 System overview

I capture the semantic relation between the sentence pair (A, B) as a function of the provability of bidirectional entailment relations for (A, B) and combine it with shallow features. Figure 3.3 shows an overview of my system for learning textual entailment and similarity from logical proofs. This system is mainly implemented using Python, except for CCG syntactic parsers and a prover.

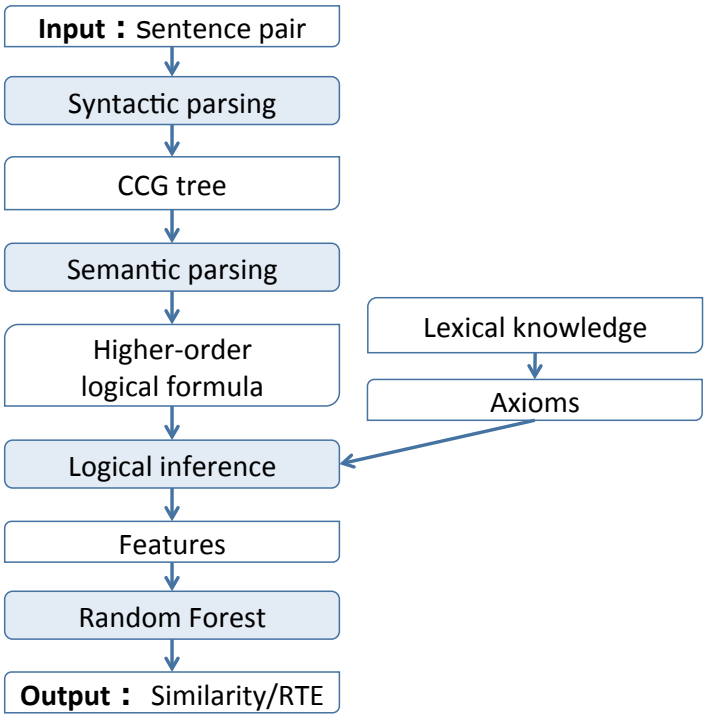

FIGURE 3.3: System overview.

First, sentences are parsed into syntactic trees based on CCG. Second, semantic representations are obtained by combining lambda terms in accordance with the meaning composition rules specified in the CCG tree. Third, after obtaining logical formulas A' and B' from A and B, proofs of the bidirectional entailment relations, $A' \rightarrow B'$ and $B' \rightarrow A'$ are attempted. There are two intentions for proving the bidirectional entailment relations. The first intention is to capture both entailment relations and similarities. Compared with entailment relations, textual similarity is directionless. Thus, I attempt to prove the bidirectional entailment relations and capture the textual similarity features. The second intention is to obtain more information from the proving process than that which can be inferred from a unidirectional entailment proving process. Compared with the proof in one direction,

we can obtain more information about semantic relations between sentences from bidirectional proofs.

If the initial natural deduction proofs fail, we rerun the proof process with relevant external axioms. If the proofs fail again, unproved sub-goals are skipped until the proof is completed. After that, features for learning textual entailment and similarity are extracted by quantifying the provability of the bidirectional entailment relations.

Next, features for learning textual entailment and similarity are extracted from the proofs during the theorem-proving process. For natural deduction proofs, I use a proof-assistant Coq as described in Section 2.3.2. When a proof is successful, Coq outputs the resulting proof (a proof term), from which we can extract detailed information such as the number of proof steps and the types of inference rules used. While multiple proof paths can be considered in each successful proof, I simply extract information from the proof path that Coq outputs[1]. In addition to the inference result (i.e., entailment, contradiction, or neutral), information about the proof process can be used as features for learning textual entailment and similarity.

Finally, I combine extracted logic-based features with non-logic-based features and train the model for predicting textual entailment and similarity. I did a pre-experiment with three regression models: logistic regression, support vector model and random forest model. I found that random forest model was the most effective, and the random forest model was therefore selected for learning textual entailment and similarity, with its hyperparameters being optimized by grid search. In the regression model for STS, the mean squared error (MSE) was used to measure the prediction performance. In the classifier model for RTE, the Gini coefficient was used to measure the prediction performance.

Regarding the brief description about random forest model (Breiman, 2001), this model first draws first n_{tree} bootstrap samples from the training data. For each of the bootstrap samples, an unpruned regression tree is grown by choosing the best split among all predictors at each node, and the best split is chosen from among those variables. The test score is predicted by aggregating the predictions of the average

[1]Coq's output depends on the depth of proof search and tactics. In this thesis, I consistently set the number of the depth of proof search to three, and use tactics which are validated by the trial split of the SICK dataset.

of n_{tree} trees. There are two characteristics of the random forest model. One characteristic is that this model is robust with respect to noise. The other characteristic is that this model does not overfit the training data easily. Given these characteristics, the random forest model is potentially suitable for my hybrid approach, where different types of features, including logic-based features and non-logic-based features, are used.

3.3 Proof for a hybrid approach

To describe the details of the proof for extracting features about semantic relations, let us consider the sentence pair A, B:

$$\begin{aligned} A &: \textit{A kitten plays in a house colored in green} \\ B &: \textit{A young cat plays in a green house} \end{aligned}$$

First, through syntactic analysis and semantic composition, we can translate A and B into logical formulas A' and B', as shown below:

$$\begin{aligned} A' : \exists y_1 \exists y_2 \exists x_1 \exists x_2 \exists x_3 (&\mathbf{kitten}(x_1) \wedge \mathbf{play}(y_1) \wedge (\mathbf{subj}(y_1) = x_1) \\ &\wedge \mathbf{house}(x_2) \wedge \mathbf{color}(y_2) \wedge (\mathbf{dobj}(y_2) = x_2) \wedge \mathbf{in}(y_1, x_2) \wedge \mathbf{green}(x_3) \wedge \mathbf{in}(y_2, x_3)) \\ B' : \exists y_1 \exists x_1 \exists x_2 (&\mathbf{young}(x_1) \wedge \mathbf{cat}(x_1) \wedge \mathbf{play}(y_1) \wedge (\mathbf{subj}(y_1) = x_1) \\ &\wedge \mathbf{green}(x_2) \wedge \mathbf{house}(x_2) \wedge \mathbf{in}(y_1, x_2)) \end{aligned}$$

Then, we attempt a natural deduction proof without using external axioms, aiming to prove entailment relations, $A' \rightarrow B'$. If both fail, then we check whether A' contradicts B', which amounts to proving the negation of the original conclusion, namely $A' \rightarrow \neg B'$. The motivation for attempting to prove contradiction is that semantic textual similarity tends to be higher when the negation of the conclusion can be proved, compared with the case where neither the conclusion nor its negation can be proved. As described in Section 3.5.1, in the Sentences Involving Compositional Knowledge (SICK) dataset (Marelli et al., 2014), 70% of the sentence pairs annotated as contradictory are assigned a similarity score in the range $[3, 5)$.

Following the proof process for proving the entailment relation described in Figure 2.6, we can obtain a set of premises $\mathcal{P}$ and a set of unproved sub-goals $\mathcal{G}$. In this

step, existential quantifiers and conjunctions in A' and B' are eliminated by applying the inference rules and the variables whose predicates of $A'$are in common with those of B' are unified.

$$\begin{aligned}\mathcal{P} &= \{\mathbf{kitten}(x_1), \mathbf{play}(y_1), \mathbf{subj}(y_1) = x_1, \\ &\quad \mathbf{house}(x_2), \mathbf{color}(y_2), \mathbf{dobj}(y_2) = x_2, \mathbf{in}(y_1, x_2), \mathbf{green}(x_3), \mathbf{in}(y_2, x_3)\} \\ \mathcal{G} &= \{\mathbf{young}(x_1), \mathbf{cat}(x_1), \mathbf{green}(x_2)\}\end{aligned}$$

If all sub-goals can be removed from $\mathcal{G}$, we can prove $A' \rightarrow B'$. In this example, we can see the three sub-goals $\mathbf{young}(x_1)$, $\mathbf{cat}(x_1)$, and $\mathbf{green}(x_2)$ still remain in $\mathcal{G}$, and we fail to prove $A' \rightarrow B'$. As above, if we fail to prove the entailment relation or contradiction, that is, if we cannot prove the conclusion or its negation, we can identify unproved sub-goals that are not matched by any predicate in the premise. To remove sub-goals that cannot be proved using genuine logical inference alone, it is necessary to inject axioms that represent the relations between premises and the target sub-goals from external knowledge. Therefore, in the next step, we attempt to prove $A' \rightarrow B'$ using axiom injection.

In axiom injection, unproved sub-goals are candidates to form axioms. In the example above, the two sub-goals $\mathbf{young}(x_1)$ and $\mathbf{cat}(x_1)$ are still unproved. The problem is to select premise candidates semantically related to these sub-goals. In generating axioms, I improved a previous word axiom injection mechanism (Martínez-Gómez et al., 2017). The purpose of improving a word axiom injection mechanism is to generate axioms more correctly and obtain more accurate proof processes. There are three points of improvements: improving the rule for selecting axiom candidates, increasing lexical knowledge with distributed word embeddings and evaluating axiom scores.

First, I describe how to improve the rule for selecting axiom candidates. I focus only on predicates that share at least one argument with both the premise and the conclusion. This means that an axiom can be generated only if there is a predicate p in the pool of premises and a predicate q in a sub-goal and p and q share a variable in an argument position, possibly with the same semantic role (e.g., subject, direct object or indirect object). In this example, the two unproved sub-goals $\mathbf{young}(x_1)$ and $\mathbf{cat}(x_1)$ share the same variable x_1 with the premise $\mathbf{kitten}(x_1)$. If we can confirm the semantic relation between *young cat* and *kitten* from lexical knowledge, we

can generate the axiom between *young cat* and *kitten*.

Second, I describe how to increase lexical knowledge with distributed word embeddings. The semantic relationships between the predicates in the premise and those in the conclusion are checked using lexical knowledge. At first, I check the linguistic relationship using WordNet, a lexical database of words grouped into sets of synonyms. In addition to grouping synonyms, linguistic relations connecting groups are listed in WordNet. Linguistic relations between predicates are checked in the following order: inflections, derivationally related forms, synonyms, antonyms, hypernyms, hyponyms, and similarities. Table 3.1 shows the examples of each linguistic relation included in WordNet. One advantage of using logical inference is that it is a powerful representation that can correctly represent these different linguistic relations, such as antonyms and synonyms.

linguistic relation	word1	word2
inflections	*play*	*plays*
derived forms	*short*	*shortness*
synonyms	*short*	*brief*
antonyms	*short*	*long*
hypernyms	*apple*	*fruit*
hyponyms	*food*	*meat*
similarities	*cute*	*pretty*

TABLE 3.1: Examples of each linguistic relation included in WordNet.

However, the WordNet database contains only 155,327 words, which is too limited to cover all linguistic relations between content words. To increase the coverage of vocabulary, I use distributed word embeddings trained with large corpora, which is known as Word2Vec (Mikolov et al., 2013). Word2Vec is the neural network model that takes a text corpus as input and produces word vectors as output. This model is based on the distributional hypothesis (Harris, 1954), where the meaning for each word is determined by its nearby words. This model is composed of two algorithms for computing word representations: continuous bag-of-words (CBOW) and skip-gram. These two algorithms are complementary. That is, the CBOW architecture predicts the current word given its context while the continuous skip-gram architecture predicts the context given the current word. As this model is successful in representing the word meaning as vectors in low dimensional spaces, Word2Vec is widely used for predicting the similarity of words and phrases (Dumais, 1997; Mitchell and

Lapata, 2010). Thus, I select Word2Vec to compensate for the lack of lexical knowledge. If the linguistic relationship between the predicates in the premise and those in the conclusion cannot be found in WordNet, I check the linguistic relationship using Word2Vec. In this thesis, I compared two large corpora (Google News Corpus and Wikipedia) in preliminary experiments. I select a 200-dimensional word vector pre-trained with Google News Corpus. Google News Corpus contains about 3 billion words, which is much larger than the number of words in WordNet.

Third, I describe how to select correct axioms from axiom candidates. I select plausible axioms by evaluating axiom scores. In this thesis, a similarity score between predicates is assumed to be an *axiom score*. The axiom candidates whose scores are highest are selected as axioms to inject into the proof. Also, if axiom candidates have scores lower than the threshold, I assume they have no linguistic relationship and they are not injected into the proof. Both axiom scores take values in the range of 0.0 to 1.0. The threshold of the score of an axiom is set to 0.25. When an axiom is generated from WordNet, the score of the axiom generated with WordNet is defined as the inverse of the length of the shortest path that connects the senses in the is-a (hypernym/hyponym) taxonomy in WordNet. When an axiom is generated by using Word2Vec, the score of the axiom is defined as the cosine similarity between word vectors represented by Word2Vec. If we fail to prove the entailment relation using the axiom injection mechanism, we attempt to prove the negation of the conclusion using the axiom injection mechanism.

In this example, the two unproved sub-goals $\mathbf{young}(x_1)$ and $\mathbf{cat}(x_1)$ share the same variable x_1 with the premise $\mathbf{kitten}(x_1)$. We calculate the similarity between **young** and **kitten** and the similarity between **cat** and **kitten**. Both similarities are over the threshold and thus we can confirm the semantic relation between *young cat* and *kitten* from lexical knowledge. Two axioms $\forall x.(\mathbf{kitten}(x) \rightarrow \mathbf{young}(x))$ and $\forall x.(\mathbf{kitten}(x) \rightarrow \mathbf{cat}(x))$ are generated, and we can thus remove two sub-goals $\mathbf{young}(x_1)$ and $\mathbf{cat}(x_1)$. Regarding the sub-goal $\mathbf{green}(x_2)$, there are no premises that share the same variable with this sub-goal. However, the predicate *green* is in common with the premise $\mathbf{green}(x_3)$. In such a special case, we generate the axiom $\forall x.(\mathbf{green}(x))$ and remove the sub-goal $\mathbf{green}(x_2)$, even if they do not share the same variable. In summary, the proof process for proving the entailment relation $A' \rightarrow B'$ is described by Figure 3.4.

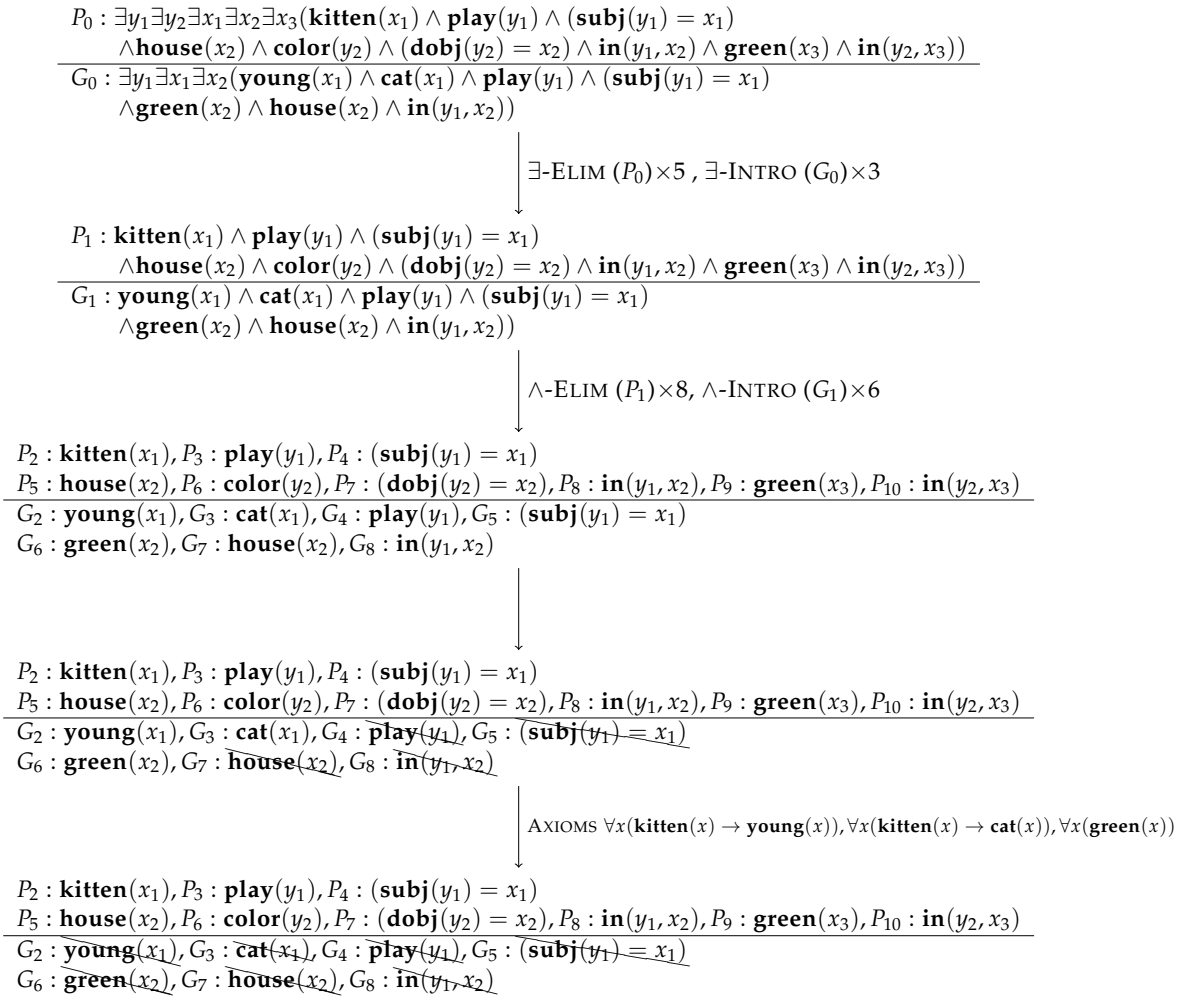

FIGURE 3.4: The proof process for proving the entailment relation $A' \rightarrow B'$.

Next, we attempt the proof in the opposite direction, i.e., $B' \rightarrow A'$, in the same way. Here, B' is set to the premise and A' is set to the conclusion.

$$B' : \exists y_1 \exists x_1 \exists x_2 (\mathbf{young}(x_1) \wedge \mathbf{cat}(x_1) \wedge \mathbf{play}(y_1) \wedge (\mathbf{subj}(y_1) = x_1)$$
$$\wedge \mathbf{green}(x_2) \wedge \mathbf{house}(x_2) \wedge \mathbf{in}(y_1, x_2))$$
$$A' : \exists y_1 \exists y_2 \exists x_1 \exists x_2 \exists x_3 (\mathbf{kitten}(x_1) \wedge \mathbf{play}(y_1) \wedge (\mathbf{subj}(y_1) = x_1)$$
$$\wedge \mathbf{house}(x_2) \wedge \mathbf{color}(y_2) \wedge (\mathbf{dobj}(y_2) = x_2) \wedge \mathbf{in}(y_1, x_2) \wedge \mathbf{green}(x_3) \wedge \mathbf{in}(y_2, x_3))$$

Again, following the proof process described in Figure 2.6, we can obtain the following set of premises $\mathcal{P}$ and a set of unproved sub-goals $\mathcal{G}$:

$$\mathcal{P} = \{\mathbf{young}(x_1), \mathbf{cat}(x_1), \mathbf{play}(y_1), \mathbf{subj}(y_1) = x_1, \mathbf{in}(y_1, x_2), \mathbf{green}(x_2), \mathbf{house}(x_2)\}$$
$$\mathcal{G} = \{\mathbf{kitten}(x_1), \mathbf{color}(y_2), \mathbf{in}(y_2, x_3), \mathbf{green}(x_3), \mathbf{dobj}(y_2) = x_2\}$$

If all sub-goals can be removed from $\mathcal{G}$, we can prove $B' \rightarrow A'$. We can see five sub-goals still remain in $\mathcal{G}$ and we fail to prove $B' \rightarrow A'$. Thus we attempt to prove $B' \rightarrow A'$ using word axiom injection similarly. Here, the one sub-goal $\mathbf{kitten}(x_1)$ and the two premises $\mathbf{young}(x_1)$ and $\mathbf{cat}(x_1)$ share the same variable x_1. Thus we calculate the similarity between **kitten** and **young** and the similarity between **kitten** and **cat**. Accordingly, the similarity between **kitten** and **cat** is higher than that between **kitten** and **young** and we can generate the axiom $\forall x.(\mathbf{cat}(x) \rightarrow \mathbf{kitten}(x))$. Note that this axiom is not logically correct. Like this case, axioms based on the similarity (i.e., the axiom score) are not always logically correct because semantic similarity does not have semantic directions. However, the lack of vocabulary is a more crucial problem and lexical relations are not strictly determined in some cases of natural language inference. Thus, I combine a word embedding model with lexical knowledge for the vocabulary expansion and use an axiom score for selecting correct axioms.

With this axiom, the sub-goal $\mathbf{kitten}(x_1)$ can be removed. Regarding the sub-goal $\mathbf{green}(x_3)$, the predicate *green* is in common with the premise $\mathbf{green}(x_2)$. In that case, we generate the axiom $\forall x.(\mathbf{green}(x))$ and remove the sub-goal $\mathbf{green}(x_3)$ exceptionally, even if there are no premises that do not share the same variable with the sub-goal.

Here, the three sub-goals $\mathbf{color}(y_2)$, $\mathbf{in}(y_2, x_3)$, and $\mathbf{dobj}(y_2) = x_2$ do not match any of the premises, so the attempted proof of $B' \rightarrow A'$ still fails. If the proof by axiom injection fails because of a lack of lexical knowledge, I obtain information about semantic relations from partial proofs by simply accepting the unproved sub-goals and forcibly completing the proof. The intention of forcibly completing the proof is to obtain information from a partial proof. In this example, we forcibly complete the proof of $B' \rightarrow A'$ by skipping these three unproved sub-goals. In summary, the proof process for the proof for the reverse entailment relation $B' \rightarrow A'$ is described by Figure 3.5.

After all bidirectional proofs are completed, the information about the generated axioms and skipped sub-goals in Figure 3.4 and Figure 3.5 is used to create features. These proofs are performed by an automated prover implemented in a proof assistant Coq.

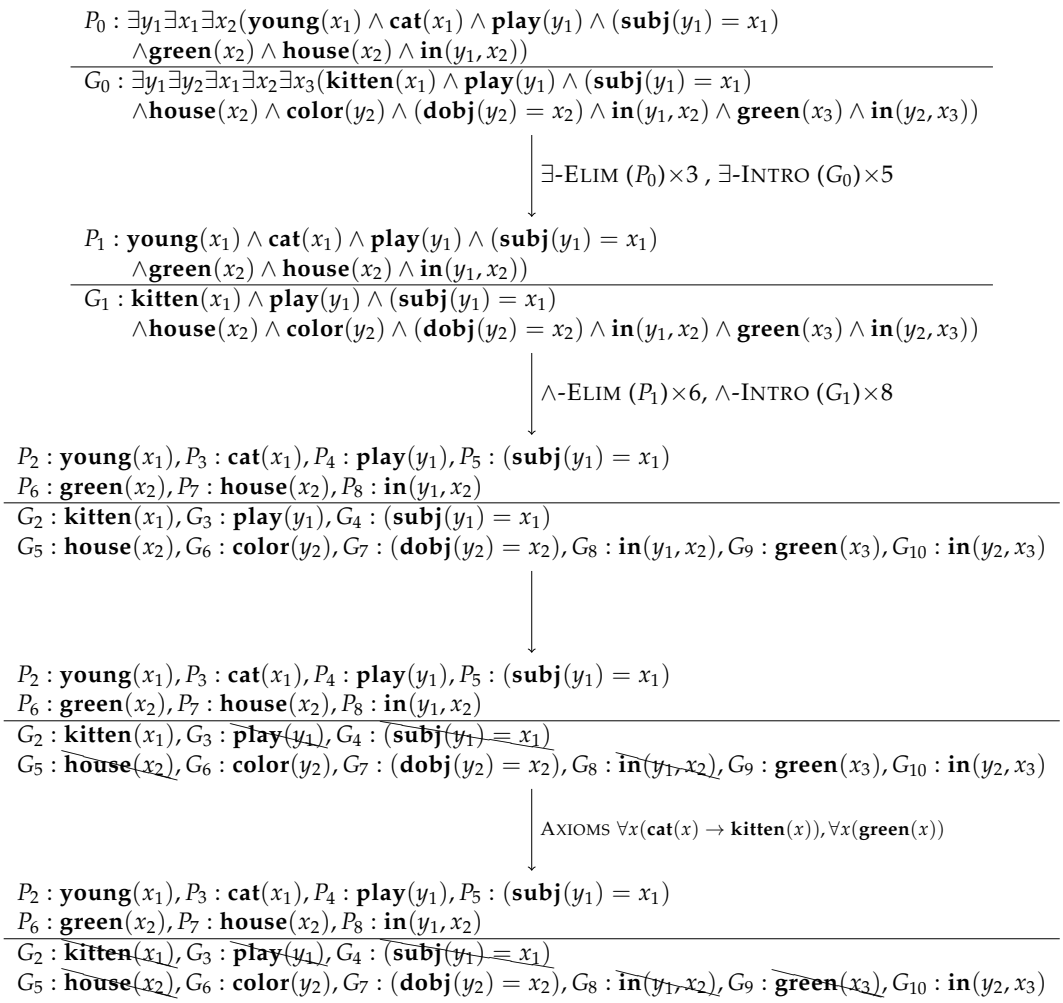

FIGURE 3.5: The proof process for proving the entailment relation $B' \rightarrow A'$.

3.4 Feature selection

To maximize the performance of predicting entailment relations and similarities, I adopt a hybrid approach where I use both logic-based features extracted from the natural deduction proof and non-logic-based features. All features are scaled to the range $[0, 1]$.

3.4.1 Logic-based features

In this thesis, I design 15 logic-based features, 12 of which are derived from the bidirectional natural deduction proofs. That is, six features are extracted from the direct proof ($A' \to B'$) and another six from the reverse proof ($B' \to A'$). The remaining three features are derived from semantic representations of the sentence pairs. Since the raw proof paths from Coq's output included redundant proof paths or noisy paths, I designed features that characterize natural deduction proofs from the raw proof paths. These 15 logic-based features are instantiated from the following nine feature types:

Logical inference result

As described in Section 3.5.1, sentence pairs with the entailment label tend to be scored much more highly, and sentence pairs with the contradiction label tend to be scored a little more highly than the average similarity. Considering this tendency, I include features to distinguish the case where either the conclusion or its negation can be proved from the case where neither can be proved. I deal with the logical inference result (i.e., entailment, contradiction, or neutral) as features with different dimensions respectively. If the logical inference result matches any result, the feature is set to 1.0 in its dimension, and if not, the feature is set to 0.0.

Score of axioms

I assume the hypothesis that if axiom scores used in the proof are scored highly, a sentence pair is more similar. Therefore, I use the score of an axiom and the number of axioms appearing in the proof to create features. As described in the previous section, the score of an axiom generated with WordNet is defined as the inverse of the length of the shortest path that connects the senses in the is-a (hypernym/hyponym) taxonomy in WordNet. The score of an axiom generated based on the threshold of the word similarity calculated using Word2Vec

is defined as the word cosine similarity. When multiple axioms are used in the proof, the average of the scores of the axioms is extracted as a feature. If the proof can be completed without using axioms, the feature is set to 1.0.

Proved sub-goals

Given that proofs can be obtained either by proving all the sub-goals or skipping unproved sub-goals, I use the proportion of proved sub-goals as a feature. My assumption is that if there are more unproved sub-goals, then the sentence pair is less similar. When there are m logical formulas in the premise pool and n proved sub-goals, I set the feature to n/m. If the theorem can be proved without skipping any sub-goals, the feature is set to 1.0. Here, there are two types of logical formulas that may remain as sub-goals: predicates (e.g., $\mathbf{girl}(x_1)$, $\mathbf{on}(y_1, x_3)$) and functional relations between terms (e.g., $\mathbf{subj}(y_1) = x_1$) Therefore, I count the number of sub-goals by each type of logical formula, and treat it as a feature of a different dimension. It may be the case that the number of sub-goals is so large that some sub-goals remain unproved even after axiom injection. Since the proportion of unproved sub-goals decreases by axiom injection, I use the proportion of unproved sub-goals both with and without axiom injection as features.

Semantic roles in unproved sub-goals

Subject or object words can strongly affect the similarity of sentence pairs compared with prepositions. Therefore, the number of each semantic role in unproved sub-goals, like $\mathbf{subj}(y_1)$ in Figures 2.6 and 2.8, is used as a feature in a different dimension respectively. Here, I count three semantic roles: subjective (**subj**), direct objective (**dobj**), and indirect objective (**iobj**).

Proof steps

As shown in Section 3.1, in general, complex theorems are difficult to prove, and in such cases, the sentence pairs are considered to be less similar. I therefore use the number of Coq's proof steps, namely the number of inference rule applications in a given proof, as a feature.

Inference rules

The complexity of a natural deduction proof can be measured in terms of the inference rules used for each proof step. I therefore extract the relative frequency with which each inference rule is used in the proof as a feature in a

different dimension respectively. I check seven inference rules used in the natural deduction proof using Coq (cf. Figure 2.5): introduction and elimination rules for conjunction ($\wedge$-INTRO, $\wedge$-ELIM), implication ($\rightarrow$-INTRO, $\rightarrow$-ELIM), and existential quantification ($\exists$-INTRO, $\exists$-ELIM) and the elimination rule for equality ($=$-ELIM).

Predicate overlap

Intuitively, the more predicates that overlap between the premise and the conclusion, the more likely it is that the inference can be proved. I therefore use the proportion of predicates that overlap between the premise and the conclusion as a feature.

Semantic type overlap

Each semantic representation in higher-order logic has a semantic type, such as Entity for entities and Prop for propositions. As is the case with the predicate overlap, the more types that overlap between the premise and the conclusion, the more likely it is that the inference can be proved. As with predicates, I use the degree of semantic type overlap between the premise and the conclusion as a feature.

Existence of negative clauses

Whether or not the premise or conclusion contains negative clauses is an effective measure of semantic relations between sentences. In semantic representations, negative clauses are represented by the negation operator $\neg$, so I check for negation operators in the premise and the conclusion and set this feature to 1.0 if either contains one.

3.4.2 Non-logic-based features

To capture the semantic relation between sentences more accurately, I combine logic-based features with non-logic-based features, extracted from surface information or by using external knowledge. I select the following nine non-logic-based features.

Noun/verb overlap

I assume that sentences tend to be similar if their degrees of overlap of the noun and verb is high. I extract and lemmatize all nouns and verbs from the

sentence pairs and use the degrees of overlap of the noun and verb lemmas as features.

Part-of-speech overlap

It might be the case that the similarity between syntactic structures affects semantic relations between sentences. Therefore, I obtain part-of-speech (POS) tags for all words in the sentence pairs by first tokenizing them with the Penn Treebank Project tokenizer[2] and then POS tagging them with the C&C POS tagger (Curran and Clark, 2003). The degree of overlap between the sentences' POS tags is used as a feature.

Synset overlap

If relatively synonymous words are used in a sentence pair (e.g., woman and lady), these words will belong to the same synset. Therefore, I have the hypothesis that the textual similarity tends to be high if synset overlap between sentences is high. For each sentence in the pair, I obtain the set containing all the synonym lemmas (i.e., the synset) for the words in the sentence. The degree of overlap between the sentences' synsets is used as a feature.

Synset distance

For each word in the first sentence, I compute the maximum path similarity between its synset and the synset of any other word in the second sentence. Then, I use the average of maximum path similarities as a feature.

Sentence length

If a conclusion is much longer than a premise, there will possibly be many subgoals in the proof. If the lengths of both a conclusion and a premise are too long, there will also possibly be many proof steps in the proof. This means the average length of the sentence pair and the difference in length affect semantic relations between the sentences. I therefore use the average of the sentence lengths and the difference in length between a premise and a conclusion as features.

String similarity

Especially in the case containing derivation words such as "*I comprehend this phrase*" and "*I have a comprehension of this phrase*," the string similarity affects

[2]ftp://ftp.cis.upenn.edu/pub/treebank/public_html/tokenization.html

the semantic relations between such sentences. Therefore, I use the similarity of the sequence of characters within the sentence pairs as a feature. The Python *Difflib*[3] function returns the similarity between two sequences as a floating-point value in the range $[0, 1]$. This measure is given by $2.0 * M/T$, where T is the total number of elements in both sequences and M is the number of matches. This feature is 1.0 if the sequences are identical and 0.0 if they have no characters in common.

Sentence similarity from vector space models

As introduced in Section 3.1, vector space models are widely used to calculate textual similarity especially in information retrieval applications. In this model, a sentence is represented using a vector (possibly derived from a count vector), which is an effective way to calculate the superficial similarity between sentences. I calculate three kinds of sentence similarity by using three major vector space models, TF-IDF, latent semantic analysis (LSA) (Deerwester et al., 1990), and latent Dirichlet allocation (LDA) (Blei, Ng, and Jordan, 2003), and I use them as features, respectively. The dimension of each sentence vector is 200. I use these cosine similarities as features.

Mapping cost of CCG derivation tree

I have a hypothesis that two sentences are similar if their CCG derivation trees are similar. Therefore, I compute the mapping cost of their CCG derivation trees by using a tree-to-tree mapping algorithm proposed by Martínez-Gómez and Miyao (2016). A value obtained by dividing the mapping cost by the total number of nodes of the derivation tree of the sentence pair is used as a feature.

Existence of passive clauses

The existence of passive clauses has an influence on textual similarity. For example, the sentence "*The game of basketball consists of a ball being dunked by a man with a jersey*" perfectly entails the sentence "*A man with a jersey is dunking the ball at a basketball game*," while their similarity is annotated as 4.2 in the SICK dataset. In this example, the topic of the sentence is changed by the use of passive clauses (i.e., the topic of the first sentence is "*a man*," while the topic of the second sentence is "*the game*"), affecting the similarity.

[3]https://docs.python.org/3.5/library/difflib.html

In CCG trees, passive clauses are represented using the syntactic category $S_{pss}\backslash NP$. I check the occurrence of passive clauses in the premise and conclusion, and if either of them contains a passive clause, then the feature is set to 1.0.

3.5 Experiments

3.5.1 Dataset for evaluation experiments

To evaluate models and systems for learning textual entailment and similarity, datasets for RTE and STS tasks have been developed. I first introduce some gold-standard datasets for RTE and STS tasks.

The SemEval-2014 Task1 SICK dataset, which is one of the main evaluation datasets for this thesis, is a dataset for studying both STS and RTE. It was originally developed for evaluating compositional distributional semantics, so it contains the lexical, syntactic, and semantic phenomena that a compositional distributional semantics framework is expected to account for. Particularly, this dataset contains logically challenging expressions such as quantifiers, negations, conjunctions, and disjunctions.

The SICK dataset was built starting from two baseline datasets: the 8K ImageFlickr dataset (Rashtchian et al., 2010) and the SemEval-2012 STS MSR-Video Description dataset (Agirre et al., 2012). These two baseline datasets contain sentences that describe the same picture or video, and thus contain many paraphrases and generic terms. First, the original sentences from these baseline datasets were normalized to remove unwanted linguistic phenomena. Then, the normalized sentences were expanded to obtain up to three new sentences; (i) a sentence with a similar meaning, (ii) a sentence with a logically contradictory or at least highly contrasting meaning, and (iii) a sentence that contains most of the same lexical items, but has a different meaning. Finally, all the sentences generated in the expansion phase were paired to the normalized sentences in order to obtain the final SICK dataset.

The SICK dataset contains 4,500 problems for training, 500 for trial and 4,927 for testing. These sentence pairs are manually annotated with three types of labels, *yes* (entailment), *no* (contradiction), or *unknown* (neutral), as well as a semantic similarity score in the range [1,5] determined using crowdsourcing. The average number of words per sentence is 10. Table 3.2 shows some examples in the SICK dataset.

Sentence Pair	Entailment	Similarity
A dog is chasing another and is holding a stick in its mouth *A dog is chasing another and is holding a piece of wood in its mouth*	Yes	4.6
There is no dog running and carrying an object in its mouth *Two dogs are running and carrying an object in their mouths*	No	3.5
The girl, who is little, is combing her hair into a pony tail *A man in a red shirt is doing a trick with the rollerblades*	Unknown	1.2
Two men are taking a break from a trip on a snowy road *Two men are holding bikes and standing on the side of a road covered of snow*	Unknown	3.8

TABLE 3.2: Examples in the SICK dataset with different entailment labels and similarity scores.

Table 3.3 shows the distribution of gold entailment labels and similarity scores in the whole SICK dataset. Regarding the relation between similarities and gold entailment labels, sentence pairs whose gold labels are *no* tend to be scored a little more highly than the average, whereas those whose labels are *unknown* have a wide range of scores. Sentence pairs whose gold labels are *yes* clearly tend to be scored more highly. Regarding the distribution of gold entailment labels, the majority of the problems are labeled as neutral.

Similarity	Yes	No	Unknown	Total
[1,2) range	0%	0%	10%	10%
[2,3) range	1%	0%	13%	14%
[3,4) range	10%	1%	28%	39%
[4,5) range	3%	27%	7%	37%
Total	14%	28%	58%	100%

TABLE 3.3: Distribution of gold labels and similarity scores in the whole SICK dataset.

The SemEval-2012 MSR-vid dataset is the second evaluation dataset I used for the STS task. This dataset contains 1,500 sentence pairs with a 750/750 training/test split. The average number of words per sentence is six, which is fewer than that of the SICK dataset. All sentence pairs are annotated with similarity scores in the range [0, 5]. Table 3.4 shows some examples in the MSR dataset.

Sentence Pair	Similarity
The man hit the other man with a stick *The man spanked the other man with a stick*	4.2
There is no man eating some food *A man is playing a flute*	2.7
A woman is slicing big pepper *A dog is moving its mouth*	0.0

TABLE 3.4: Examples in the MSR-video dataset.

The Stanford Natural Language Inference (SNLI) dataset (Bowman et al., 2015) is a large dataset for the RTE task. It contains over 570K sentence pairs. Hypotheses were written by crowd workers who were given the premise and asked to write one definitely true sentence, one possibly true sentence, and one definitely false sentence for given premises that were written as hypotheses. However, it is not concerned with logically challenging expressions; the semantic relationships between a premise and a hypothesis are often limited to synonym/hyponym lexical substitution, replacement of short phrases, or exact word matching. This is because hypotheses are often parallel to the premise in structures and vocabularies. A recent study on the SNLI dataset (Tsuchiya, 2018) reports that this dataset has a hidden bias which allows prediction of entailment labels from hypothesis sentences even if no context information is given by a premise.

The Multi-Genre Natural Language Inference (MultiNLI) dataset (Williams, Nangia, and Bowman, 2017) is also a large dataset designed for use in the development and evaluation of machine learning models for sentence understanding. Compared with the SNLI dataset, the MultiNLI dataset represents both written and spoken language across a range of styles, degrees of formality, and topics. This dataset contains over 390K sentence pairs. As shown in this sentence pair below, there are many ungrammatical cases in the MultiNLI dataset, especially the cases derived from spoken speech:

(19) a. *yeah well i'm a hot weather person i'm i can take the heat but i don't like the cold*
b. *I do not like warm weather at all*

As my method is based on syntactic analysis, such ungrammatical cases are out-of-scope in this thesis.

The multi premise entailment dataset (Lai, Bisk, and Hockenmaier, 2017) is an RTE dataset containing multiple premise sentences. This dataset contains 10,000 cases with a $8,000/1,000/1,000$ training/development/test split. Each case consists of four premise sentences (captions from the same 30K ImageFlickr), one hypothesis sentence (a simplified 30K ImageFlickr caption), and one label (entailment, contradiction, or neutral) that indicates the relationship between the set of four premises and the hypothesis. This label is based on a consensus of five crowdsourced judgments. However, a hypothesis sentence is too simplified (the length of a hypothesis

sentence is three or four words in many cases) and this dataset is not concerned with logical expressions.

The FraCaS dataset (Cooper et al., 1994) is a dataset designed to evaluate theories of formal semantics. The whole dataset is divided into nine sections, each devoted to linguistically and logically challenging problems. Each problem consists of one or more premise sentences and one hypothesis sentence. However, it is confined to genuine logical inference and thus does not contain problems requiring lexical knowledge. In addition, this dataset contains only 346 cases, which were not designed for machine learning approaches.

For these reasons, I mainly chose two datasets for evaluating my system: the SICK dataset and the MSR-video dataset.

3.5.2 Experimental setting

I evaluated my system in both RTE and STS tasks using the SemEval-2014 SICK dataset. I used the training and trial split of the SICK dataset for training the random forest model and evaluated the prediction performance of the test split. Regarding the STS task with the SICK dataset, I compared my system with the following systems: the state-of-the-art neural network-based system **Siamese-LSTM** (Mueller and Thyagarajan, 2016); the **SemEval-2014 best** system (Zhao, Zhu, and Lan, 2014); and two logic-based systems: namely **The Meaning Factory** (Bjerva et al., 2014) and **UTexas** (Beltagy et al., 2014). As described in Section 1.2.3, I used three standard evaluation metrics from the SemEval-2014 shared task: Pearson correlation coefficient γ, Spearman's rank correlation coefficient ρ, and MSE.

Regarding the RTE task with the SICK dataset, I compared my system with the following systems: the state-of-the-art neural network-based system **GRU** (Yin and Schütze, 2017); the **SemEval-2014 best** system (Lai and Hockenmaier, 2014); and three logic-based systems: **The Meaning Factory** (Bjerva et al., 2014), **UTexas** (Beltagy et al., 2014) and the previous system **ccg2lambda** with word abduction (Martínez-Gómez et al., 2017). I used precision, recall and accuracy (described in Section 1.2.2) as evaluation metrics.

I also evaluated my system in the STS task using the SemEval-2012 MSR-video dataset to compare my system with the **SemEval-2012 best** system (Bär et al., 2012) and the logic-based **UTexas** system (Beltagy, Erk, and Mooney, 2014). I used the training split of the MSR-video dataset for training the random forest model and

evaluated the prediction performance of the test split. I used the standard evaluation metric from the SemEval-2012 shared task, i.e., Pearson correlation coefficient γ as the evaluation metric in the STS task with the MSR-video dataset.

3.5.3 Comparison with other systems

Semantic textual similarity

Table 3.5 shows the evaluation results of my experiment with the SICK dataset on the STS task.

	γ	ρ	MSE
Siamese-LSTM	0.882	0.835	0.229
My system (WordNet)	0.842	0.799	0.299
My system (WordNet+Word2Vec)	0.841	0.798	0.300
My system (Word2Vec)	0.822	0.768	0.329
The Meaning Factory	0.827	0.772	0.322
UTexas	0.714	0.674	0.499

TABLE 3.5: Results of the test split of SICK STS.

Although the state-of-the-art neural network-based system yielded the best results overall, my system achieved higher scores than SemEval-2014 submissions, including the two logic-based systems (i.e., The Meaning Factory and UTexas). As I mentioned above, the sentence pairs annotated as *unknown* produced a wide range of scores. The Pearson correlation of the *unknown* portion of the SICK dataset was 0.766, which suggests that my logic-based system can also be applied to sentences with the neutral label. Regarding the difference in lexical knowledge, my system provided the best performance when only WordNet was used as lexical knowledge. Using Word2Vec as linguistic knowledge often causes excessive generation of axioms and induces incorrect proofs, especially in cases where it is necessary to inject phrasal knowledge into the proof. I describe the detail of this error phenomena in the last subsection (error analysis) of this chapter. Also, the Word2Vec model relies on the distributional hypothesis and cannot distinguish antonyms from synonyms. This causes generation of wrong axioms.

For example, the correct similarity score of the sentence pair ID 677 in the test dataset was 3.6 and the prediction score when using WordNet was the same, while the prediction score when using Word2Vec was 4.1.

A dog, which is black, and a white one are ignoring each other in the street

A dog, which is black, and a white one are staring at each other in the street

The semantic relation between these sentences is the contradiction because the word *ignore* is the antonym of the word *stare*. When using WordNet, I generated the antonym axiom $\forall x(\mathbf{ignore}(x) \rightarrow \mathbf{stare}(x) \rightarrow \mathbf{False})$ and prove the contradiction. Conversely, when using Word2Vec, I wrongly generated only the synonym axiom $\forall x(\mathbf{ignore}(x) \rightarrow \mathbf{stare}(x))$ and prove the entailment relation. For these reasons, the performance was the best when only WordNet was used as lexical knowledge for the SICK dataset.

Table 3.6 shows the evaluation results of my experiment with the MSR-video dataset on the RTE task.

	γ
SemEval-2012 Best Score	0.873
My system	0.853
UTexas	0.830

TABLE 3.6: Results of the test split of MSR-video STS.

These results also indicate that my logic-based system achieved higher performance than the other logic-based systems. Regarding the difference in lexical knowledge, my system provided the best performance when both WordNet and Word2Vec were used as lexical knowledge. Compared with the SICK dataset, there are many short sentence pairs in the MSR-video dataset and there are not many cases where phrasal knowledge should be considered in logical inferences. Thus, using Word2Vec in the MSR-video dataset was an effective way to increase vocabulary. This result also indicates that adequate lexical knowledge depends on the characteristics of the target dataset. Accordingly, my hybrid system does not depend on types of lexical knowledge and it is easy to combine with various lexical knowledge such as domain-specific knowledge.

Recognizing textual entailment

Table 3.7 shows the evaluation result of the experiment with the SICK dataset on the RTE task. My system achieved the state-of-the-art accuracy on the RTE task when only WordNet or both WordNet and Word2Vec were used for the lexical knowledge. As with the evaluation result in the STS task, my system provided the best accuracy when only WordNet was used for lexical knowledge. The precision increased and the recall decreased when using both WordNet and Word2Vec since the number of

proofs of correct entailment relations increased and the number of proofs of correct contradiction decreased by using Word2Vec.

	Precision	Recall	Accuracy
My system (WordNet)	0.906	0.797	0.877
My system (WordNet+Word2Vec)	0.919	0.775	0.874
My system (Word2Vec)	0.870	0.733	0.841
GRU	–	–	0.871
SemEval2014 Best Score	0.816	0.819	0.846
ccg2lambda	0.970	0.636	0.831
The Meaning Factory	0.936	0.606	0.816
UTexas	–	–	0.734

TABLE 3.7: Results of the test split of SICK RTE.

3.5.4 Feature ablation and isolation

Semantic textual similarity

Table 3.8 shows ablation results with the SICK dataset on the STS task. In my ablation study, I removed each feature from all features and trained the model. The intention of this ablation study is to assess the prediction performance of each feature. While there was no change in the prediction performance when each feature was removed, the inference result was the most effective feature of logic-based features in the ablation analysis. This result indicates that the similarity of a sentence pair tends to be higher when the gold label of RTE is entailment or contradiction, as described in Section 3.5.1. The cosine similarity of vector space models was the most effective feature of non-logic-based features. This result indicates that the surface information is more effective for the prediction of textual similarity.

As shown in Table 3.8, when all logic-based features were removed in training, the performance significantly decreased, compared to the case when all non-logic-based features were removed. This result indicates that logic-based features have more effects that improve prediction performance than non-logic-based features. When the features derived from the process of theorem proving (i.e., proved sub-goals, semantic roles of unproved sub-goals, proof steps, inference rules, and axiom score) were removed, the prediction performance decreased more significantly, compared with the case when the features were derived from the logical formulas

	γ	ρ	MSE
- Logical inference result	0.8355	0.7900	0.3077
- Predicate overlap	0.8376	0.7976	0.3072
- Score of axioms	0.8388	0.7963	0.3044
- Proved sub-goals	0.8407	0.7946	0.3013
- Proof steps	0.8409	0.7982	0.3008
- Semantic type overlap	0.8410	0.7982	0.3007
- Unproved sub-goals' case	0.8414	0.7995	0.3000
- Negative clauses	0.8420	0.7988	0.2990
- Inference rules	0.8430	0.8005	0.2972
- Vector space model	0.8288	0.7925	0.3208
- Noun/verb overlap	0.8373	0.7934	0.3048
- Passive clauses	0.8396	0.7959	0.3027
- Sentence length	0.8397	0.7978	0.3026
- String similarity	0.8403	0.7970	0.3018
- Synset overlap	0.8405	0.7974	0.3012
- Synset distance	0.8406	0.7982	0.3011
- Tree mapping cost	0.8406	0.7980	0.3013
- Part-of-speech overlap	0.8412	0.7983	0.3003
- Proving Process	0.8326	0.7900	0.3146
- Logical Formulas	0.8375	0.7958	0.3073
- Only logic-based	0.7847	0.7237	0.3937
- Only non logic-based	0.8114	0.7820	0.3483
All	0.8420	0.7988	0.2990

TABLE 3.8: Ablation results on SICK STS.

(i.e., predicate overlap, type overlap, and existence of negation). This result suggests that features extracted from the process of theorem proving contribute to the performance of predicting textual similarity.

Table 3.9 shows the evaluation results when training the regressor with each feature group in isolation with the SICK dataset on the STS task. The performance was the highest when only the feature of predicate overlap was used. This result indicates that a logical formula represents linguistic information in a sentence more precisely. In tree mapping cost and passive clauses, the Pearson correlation was almost 0. This result indicates that these features have no effect for the prediction of textual similarity. However, regarding similarity, 76% of sentence pairs in the SICK dataset are annotated in the range of [3, 5]. This causes the MSE result to be almost 1.0 although the tree mapping cost and passive clauses are not effective features.

	γ	ρ	MSE
Predicate overlap	0.6818	0.6105	0.5454
Proved sub-goals	0.6302	0.6012	0.6144
Inference rules	0.5611	0.5360	0.6988
Logical inference result	0.5493	0.5425	0.7117
Unproved sub-goals' case	0.5233	0.4965	0.7402
Proof steps	0.4381	0.4386	0.8296
Score of axioms	0.3787	0.3621	0.8732
Semantic type overlap	0.2333	0.2137	0.9641
Negative clauses	0.1615	0.2061	0.9924
Noun/verb overlap	0.6560	0.5598	0.5805
Vector space model	0.6086	0.5260	0.6602
String similarity	0.4857	0.4786	0.7788
Synset distance	0.4331	0.3953	0.8279
Synset overlap	0.4191	0.3849	0.8422
Part-of-speech overlap	0.3504	0.3484	0.8940
Sentence length	0.2621	0.2685	0.9490
Tree mapping cost	0.0347	0.0208	1.0179
Passive clauses	0.0987	0.1087	1.0092
Only logic-based	0.8114	0.7820	0.3483
Only non logic-based	0.7847	0.7237	0.3937
All	0.8420	0.7988	0.2990

TABLE 3.9: Results when training the regressor with each feature group in isolation on SICK STS.

Recognizing textual entailment

Table 3.10 shows ablation results with the SICK dataset on the RTE task. As is the same with ablation results on the STS task, there was no significant fluctuation in accuracy in the case when each feature was removed alone. This result indicates that logic-based features have more effects to improve accuracy than non-logic-based features. When the features derived from the process of theorem proving (i.e., proved sub-goals, semantic roles of unproved sub-goals, proof steps, inference rules, and axiom score) were removed, the accuracy decreased more significantly, compared with the case when the features derived from the logical formulas (i.e., predicate overlap, type overlap, and existence of negation). This result suggests that features extracted from the process of theorem proving contribute to the performance of predicting entailment/contradiction labels.

Among logic-based features, the proof step was the most effective feature in the ablation result. I discuss this result by comparing the ablation result on RTE task with that on STS task. Superficial overlap such as cosine similarity between sentence vectors strongly affects the prediction performance on STS task, while semantic overlap such as the proof step strongly effects the accuracy on RTE task. This indicates the difference between the RTE task and STS task: while the RTE task requires deep

	Precision	Recall	Accuracy
- Proof steps	0.9176	0.7345	0.8569
- Proved sub-goals	0.9123	0.7782	0.8729
- Inference rule	0.9202	0.7768	0.8755
- Unproved sub-goals' case	0.9120	0.7853	0.8755
- Negation clauses	0.9155	0.7853	0.8765
- Logical inference result	0.9122	0.7877	0.8767
- Score of axioms	0.9110	0.7905	0.8771
- Semantic type overlap	0.9267	0.8027	0.8771
- Predicate overlap	0.9138	0.7891	0.8778
- Passive clauses	0.9134	0.7745	0.8710
- Vector space model	0.9016	0.7848	0.8712
- String similarity	0.9125	0.7853	0.8753
- Noun/verb overlap	0.9051	0.7947	0.8763
- Part-of-speech overlap	0.9161	0.7863	0.8769
- Tree mapping cost	0.9073	0.7971	0.8780
- Sentence length	0.9145	0.7095	0.8782
- Synset distance	0.9079	0.7980	0.8784
- Synset overlap	0.9069	0.7976	0.8786
- Proving Process	0.9038	0.7247	0.8238
- Logical Formulas	0.9125	0.7900	0.8771
- Only logic-based	0.7618	0.6474	0.7869
- Only non-logic-based	0.9255	0.7491	0.8659
All	0.9058	0.7971	0.8774

TABLE 3.10: Ablation results on SICK RTE.

semantic analysis, the STS task is based on distributional analysis.

Furthermore, the prediction accuracy decreased when the axiom score feature was removed, whereas accuracy increased when the synset distance feature or the overlap rate of synsets feature was removed. I consider the reason for this pattern by comparing the axiom score with synset distances and the overlap rate of synsets. To calculate the synset distance feature and the overlap rate of synsets feature, I consider distances or overlap about for possible word pairs in a premise and conclusion. On the other hand, to calculate the axiom score feature, I consider similarities for word pairs only necessary for logical inference. This indicates that the axiom score represents the meaning relatedness of content words more correctly than the synset distance and the overlap rate of synsets. Thus, the axiom score feature is a more effective feature for predicting entailment/contradiction labels in the RTE task.

Table 3.11 shows the evaluation result when training the classifier with each feature group in isolation with the SICK dataset on the RTE task. As shown in Table 3.11, the accuracy was highest when only the inference result was used as the feature in training. Regarding the five kinds of features (i.e., type overlap, existence of passive clauses, part-of-speech overlap, sentence lengths, and mapping cost of

	Precision	Recall	Accuracy
Logical inference result	0.5628	0.5104	0.7002
Proved sub-goals	0.5995	0.5913	0.7219
Score of axioms	0.7292	0.3512	0.7158
Inference rule	0.6371	0.6026	0.7123
Predicate overlap	0.5628	0.5104	0.7002
Proof steps	0.6465	0.4529	0.6908
Unproved sub-goals' case	0.7242	0.1620	0.6177
Negation clauses	0.6036	0.2990	0.6110
Semantic type overlap	-	0.0000	0.5651
Vector space model	0.5439	0.5400	0.6839
Noun/verb overlap	0.7399	0.6737	0.6544
String similarity	0.4897	0.3701	0.6413
Synset overlap	0.4557	0.2302	0.6075
Synset distance	0.4062	0.2081	0.5911
Passive clauses	0.9211	0.0330	0.5790
Part-of-speech overlap	0.3269	0.0160	0.5661
Sentence length	0.4619	0.0513	0.5649
Tree mapping cost	0.2222	0.0009	0.5645
Only logic-based	0.9255	0.7491	0.8659
Only non logic-based	0.7618	0.6474	0.7869
All	0.9058	0.7971	0.8774

TABLE 3.11: Results when training the regressor with each feature group in isolation on SICK RTE.

syntactic trees) the recall score was remarkably low, around less than 0.1. In the case when these features were used in isolation, my system predicted neutral labels in almost all sentence pairs. This result suggests that these features do not contribute to accuracy when these features are used in isolation.

3.5.5 Evaluation by each gold label

Semantic textual similarity

Table 3.12 shows the evaluation results for each gold similarity score on SICK STS, where my proposed method achieved the highest performance in the range $[4, 5]$. About 80% of sentence pairs whose gold similarity scores are in the range $[4, 5]$ are annotated as entailment or contradiction in RTE. This fact indicates that my proposed method predicts similarity scores more accurately in sentence pairs that have logical relations.

Gold similarity score	Cases	γ	ρ	MSE
$[1,2)$	300	0.4631	0.4522	0.7203
$[2,3)$	800	0.2829	0.2849	0.3980
$[3,4)$	2000	0.2280	0.2104	0.2028
$[4,5]$	1654	0.6185	0.6408	0.2673
All	4927	0.8420	0.7988	0.2290

TABLE 3.12: Evaluation results for each gold similarity score (SICK STS).

Recognizing textual entailment

Table 3.13 shows evaluation results for each gold label on SICK RTE. Here I evaluated the results based on precision score, recall score, and F1-score in three-class (i.e., *yes*, *no*, or *unknown*) classification. This result shows that my proposed method got the highest accuracy in predicting contradiction labels of a sentence pair. In the SICK dataset, negation expressions are included in many sentence pairs whose gold labels are annotated as contradiction. Therefore, this result indicates my proposed method has potential to capture the meaning of negation expressions.

Gold label	Cases	Precision	Recall	F1-score
Unknown	2793	0.8597	0.9391	0.8977
No	720	0.9476	0.8075	0.8720
Yes	1414	0.8855	0.7918	0.8360
All	4927	0.8801	0.8774	0.8761

TABLE 3.13: Evaluation results for each gold label (SICK RTE).

3.5.6 Evaluation by linguistic phenomena

For 4927 sentence pairs in the SICK test data, I compared the similarity scores predicted by my proposed method with those predicted by the state-of-the-art model based on deep learning (Mueller and Thyagarajan, 2016). My proposed method predicted similarity scores of 2,666 sentence pairs more accurately. Furthermore, I classified these 2,666 sentence pairs based on their linguistic phenomena to analyze which linguistic phenomenon in a sentence pair can be captured by my proposed method. Table 3.14 shows the result of counting the number of sentence examples, including negation, quantifier, conjunction, and relative pronouns included in the whole test data.

The number of sentence examples for which my proposed method achieved better predictions and its percentage divided by the number of examples including each linguistic phenomena in the whole test data are shown in Table 3.15. This result shows my method tends to predict similarity scores whose sentences include negation, quantifiers and conjunctions more accurately than the deep learning-based model.

Linguistic phenomena	Cases	Keyword	Example
negation	544	*no*, *not* (*n't*), *nobody*	*There is no man in a black jacket doing tricks on a motorbike* *A person in a black jacket is doing tricks on a motorbike*
quantifier	720	*many*, *few*, *some*, *any*, *all*, *every*, numeral	*Some cheerleaders are dancing* *A female cheerleader is sitting on the knee of a male cheerleader*
conjunction	1044	*and*, *or*	*A man and a woman are walking through the woods* *The man and woman are walking*
relative pronoun	158	*what*, *which*, *who*, *when*, *where*, *how*	*A small child is showing excitement on a swing set at the park* *The girl is standing behind the little woman who is swinging*

TABLE 3.14: Examples of linguistic phenomena.

Linguistic phenomena	Cases	Proportion	Example
negation	308	57%	*Nobody is playing ping pong* *Two people are playing ping pong*
quantifier	409	57%	*A few men in a competition are running outside* *The man is racing for the lead*
conjunction	625	60%	*The woman is not wearing glasses or a headdress* *The woman is wearing glasses and a black headdress*
relative pronoun	80	51%	*A dog, which is brown, and a black one are racing in the grass* *A dark black dog and a light brown dog are playing in the backyard*

TABLE 3.15: Evaluation results for each linguistic phenomenon.

3.5.7 Positive examples and error analysis

Table 3.16 shows some examples for which the prediction score was better when using logic-based features than when using non-logic-based ones.

For IDs 642 and 1360, one sentence contains a passive clause, while the other sentence does not. In such cases, the sentence pairs are not superficially similar. By using logical formulas based on event semantics, my method enabled the sentence containing the passive clause to be interpreted correctly and judge that the passive and non-passive sentences are similar to each other.

In ID 891, one sentence contains a negative clause while the other does not. Using shallow features, the word overlap is small and the prediction score was much lower than the correct score. My logic-based method, however, interpreted the first sentence as a negative existential formula of the form $\neg\exists x\mathcal{P}(x)$ and the second sentence as an existential formula $\exists x\mathcal{P}'(x)$. Thus, it could easily handle the semantic difference between the positive and negative sentences.

In ID 1158, by contrast, the proportion of word overlap is so high that the prediction score with non-logic-based features was much higher than the correct score. My method, however, was able to prove the contradiction using an antonym axiom of the form $\forall x(\mathbf{remove}(x) \rightarrow \neg\mathbf{add}(x))$ from WordNet and thus predict the score correctly.

In ID 59, the proportion of word overlap is low, so the prediction score with non-logic-based features was lower than the correct score. My method, however, was able to prove the partial entailment relations for the sentence pair and thus predict the score correctly. Here the logic-based method captured the common meaning of the sentence pair: both sentences refer to kids playing in the leaves.

Finally, in ID 71, the prediction score with non-logic-based features was much higher than the correct score. There are two reasons for this phenomenon: negations tend to be omitted in non-logic-based features such as TF-IDF and the proportion of word overlap is high. However, as logical formulas and proofs can handle negative clauses correctly, my method was able to predict the score correctly.

ID	Sentence Pair	Gold	Pred +logic	Pred -logic	Entailment
642	*A person is climbing a rock with a rope, which is pink* *A rock is being climbed by a person with a rope, which is pink*	5.0	4.9	4.1	Yes
1360	*The machine is shaving the end of a pencil* *A pencil is being shaved by the machine*	4.7	4.6	3.8	Yes
891	*There is no one on the shore* *A bunch of people is on the shore*	3.6	3.7	2.6	No
1158	*A woman is removing ingredients from a bowl* *A woman is adding ingredients to a bowl*	3.3	3.5	4.1	No
59	*Kids in red shirts are playing in the leaves* *Three kids are jumping in the leaves*	3.9	3.8	3.1	Unknown
71	*There is no child lying in the snow and making snow angels* *Two people in snowsuits are lying in the snow and making snow angels*	3.3	3.3	4.1	Unknown

TABLE 3.16: Examples for which the regressor trained only with logic-based features performs better than when using non-logic features. "Gold": correct score, "Pred+logic": prediction score only with logic-based features, "Pred-logic": prediction score only with non-logic-based features.

Table 3.17 shows examples where using only logic-based features produced erroneous results. In ID 2831, two word axioms $\forall x(\mathbf{lady}(x) \rightarrow \mathbf{girl}(x))$ and $\forall x(\mathbf{pencil}(x))$ were generated but my system failed to prove the entailment relation between the sentences due to the lack of phrasal knowledge between *pencil on eyeshadow* and *use an eye pencil on her eyelid*.

In ID 3974, the two word axioms $\forall x(\mathbf{awaken}(x) \rightarrow \mathbf{wake}(x))$ and $\forall x(\mathbf{awaken}(x) \rightarrow \mathbf{up}(x))$ were generated. However, the score of the word axiom $\forall x(\mathbf{awaken}(x) \rightarrow \mathbf{up}(x))$ was low (0.25), and the prediction score was thus lower than the correct score. Likewise, in ID 4833, two word axioms were generated but the score of the word axiom $\forall x(\mathbf{file}(x) \rightarrow \mathbf{do}(x))$ was very low (0.09), and the prediction score was thus negatively affected. In these cases, it is necessary to consider phrase-level axioms and their scores such as $\forall x(\mathbf{awaken}(x) \rightarrow \mathbf{wake\_up}(x))$ and $\forall x(\mathbf{file\_nail}(x) \rightarrow \mathbf{do\_manicure}(x))$.

In ID 1941, my system wrongly proved bidirectional entailment relations by adding three word axioms, so the prediction score was much higher than the correct score. Although I control the word axiom injection by setting an axiom score threshold, there is a limit to finding the best threshold. Thus, to prevent generating unnecessary word axioms and to generate correct axioms instead, a mechanism for detecting the lack of phrasal knowledge in the proof processes is necessary.

ID	Sentence Pair	STS Gold	STS System	RTE Gold	RTE System	Generated Axioms
2831	*The lady is penciling on eyeshadow* *The girl is using an eye pencil on her eyelid*	4.4	2.7	yes	unk	$\forall x(\mathbf{lady}(x) \rightarrow \mathbf{girl}(x))$ $\forall x(\mathbf{pencil}(x))$
3974	*A girl is awakening* *A girl is waking up*	4.9	3.6	yes	unk	$\forall x(\mathbf{awaken}(x) \rightarrow \mathbf{wake}(x))$ $\forall x(\mathbf{awaken}(x) \rightarrow \mathbf{up}(x))$
4833	*A girl is filing her nails* *A girl is doing a manicure*	4.2	1.8	yes	unk	$\forall x(\mathbf{nail}(x) \rightarrow \mathbf{manicure}(x))$ $\forall x(\mathbf{file}(x) \rightarrow \mathbf{do}(x))$
1941	*A woman is putting the baby into a trash can* *A person is putting meat into a skillet*	1.0	3.3	unk	yes	$\forall x(\mathbf{woman}(x) \rightarrow \mathbf{person}(x))$ $\forall x(\mathbf{trash}(x) \rightarrow \mathbf{skillet}(x))$ $\forall x(\mathbf{baby}(x) \rightarrow \mathbf{meat}(x))$

TABLE 3.17: Error examples when training the regressor only with logic-based features.

Chapter 4

Phrase Abduction

4.1 Motivation and Related Work

The error analysis in Chapter 3 indicates that handling phrasal knowledge in logical inferences is a crucial problem. As described in previous sections, it is difficult to capture the meanings of content words or phrases using genuine logical inference alone. To account for lexical relations between content words or phrases, previous logic-based approaches use knowledge databases such as WordNet (Miller, 1995) to identify lexical relations within a sentence pair. Also, considerable research efforts have been focused on the identification and extraction of paraphrases. Thus, I first review previous logical inference systems that are combined with lexical knowledge and describe the problem of handling phrasal knowledge in logical inference. Second, I introduce previous works that have identified and collected paraphrases. Lastly, I describe the motivation for my proposed method for handling phrasal knowledge in logical inference; I call this method *phrase abduction*.

4.1.1 RTE systems combined with lexical knowledge

The RTE system developed by Abzianidze (2016) is a purely logic-based RTE system that uses CCG parsers and a tableaux-based prover. The system uses WordNet as lexical knowledge and adds missing knowledge manually from the training dataset. However, this technique requires considerable human effort and is not extended to handle phrasal knowledge.

Martínez-Gómez et al. (2017) proposed an RTE system with an on-the-fly axiom injection mechanism guided by a natural deduction theorem prover. Pairs of unprovable sub-goals and plausible single premises are identified by means of a variable unification routine and then linguistic relations between their logical predicates

are checked using lexical knowledge such as WordNet and VerbOcean (Chklovski and Pantel, 2004). Although this mechanism succeeds in capturing word-to-word relations in a sentence pair, the proof strategy of selecting only premises that share an argument with the sub-goal perfectly is not suitable for capturing phrase-to-phrase or word-to-phrase relations.

To describe this issue, let us consider the proof of the entailment relation between the following sentences:

(20) a. *A band is playing on a stage*

b. *A band is playing onstage*

The sentences (20a) and (20b) are mapped onto logical formulas T' and H' based on event semantics via CCG-based semantic composition, as follows.

$$T' : \exists y_1 \exists x_1 \exists x_2 (\mathbf{band}(x_1) \wedge \mathbf{play}(y_1) \wedge (\mathbf{subj}(y_1) = x_1) \wedge \mathbf{on}(y_1, x_2) \wedge \mathbf{stage}(x_2))$$

$$H' : \exists y_1 \exists x_1 (\mathbf{band}(x_1) \wedge \mathbf{play}(y_1) \wedge (\mathbf{subj}(y_1) = x_1) \wedge (\mathbf{onstage}(y_1))$$

We attempt to prove the entailment relation $T' \rightarrow H'$. By applying inference rules of a natural deduction proof, we obtain the following sets of premises $\mathcal{P}$ and sub-goals $\mathcal{G}$:

$$\mathcal{P} = \{P_1 : \mathbf{band}(x_1),\ P_2 : \mathbf{play}(y_1), P_3 : \mathbf{on}(y_1, x_2),\ P_4 : \mathbf{stage}(x_2)\}$$

$$\mathcal{G} = \{G_1 : \mathbf{onstage}(y_1)\}$$

In this proof, the sub-goal $\mathbf{onstage}(y_1)$ remains. Next, in the previous word axiom injection mechanism with lexical knowledge, we search for a premise that shares the same argument with the sub-goal. In this example, people assume the sub-goal $\mathbf{onstage}(y_1)$ is the paraphrase of the premise $\mathbf{on}(y_1, x_2) \wedge \mathbf{stage}(x_2))$. To capture this paraphrase, we have to detect the relation between $\mathbf{on}(y_1, x_2) \wedge \mathbf{stage}(x_2))$ in the premise and $\mathbf{onstage}(y_1)$ in the sub-goal. However, as the premise $\mathbf{on}(y_1, x_2)$ does not match the argument with the sub-goal $\mathbf{onstage}(y_1)$, it fails to generate this phrasal axiom.

Bjerva et al. (2014) proposed an RTE system where WordNet relations are translated to logical formulas and used as axioms for word-to-word knowledge in theorem proving. For phrasal knowledge, PPDB (Ganitkevitch, Van Durme, and Callison-Burch, 2013) is used to rephrase an input sentence pair instead of translating paraphrases into axioms. However, this solution ignores logical contexts that might be necessary when applying phrasal knowledge. Moreover, it does not apply to discontinuous phrases.

For example, consider the following sentence pair:

(21) a. *A hurdle is being leapt by a horse that has a rider on its back*

b. *A horse and its rider are leaping over a barrier*

If we prove the entailment relation between these sentences, we have to capture the paraphrase *X leap a hurdle* and *X leap over a barrier*. To capture this paraphrase, we need to translate the original phrase *A hurdle is being leapt* to the normalized phrase *leap a hurdle* by applying normalization between passive voices and active voices.

Consider another example below:

(22) a. *A black dog and a small white and black dog are looking up at a kitchen countertop*

b. *A large dog and a small dog are standing next to the kitchen counter and are investigating*

If we prove the entailment relation between these sentences, we have to capture the paraphrase *X look up at Y* and *X stand next to Y and investigate*, which is difficult to rephrase by simply checking phrasal knowledge such as PPDB. In this thesis, I refer to paraphrases for which normalization is necessary for capturing *discontinuous paraphrases*.

The RTE system developed by Beltagy et al. (2016) is based on probabilistic logic. This system assigns distributional similarity scores to any words in a sentence pair. This system also uses WordNet and PPDB as lexical knowledge. To increase their coverage of phrasal knowledge, this system uses a resolution strategy to align clauses and literals in a sentence pair. These alignments also constrain how the unaligned fragments of a sentence pair may correspond to each other, reducing the problem to a word or phrasal entailment recognition using a statistical classifier.

However, this strategy only considers one possible set of alignments between fragments of a sentence pair, which causes a lack of coverage when there are repetitions of content words and meta-predicates.

Consider the following sentences:

(23) a. *Soccer players are kicking a soccer ball into the goal*

b. *There are no soccer players kicking a soccer ball into the goal*

To prove the entailment relation between the sentences, we have to make alignments by distinguishing the first word *soccer* (used in the phrase *soccer players*) in the sentence (23a) and (23b) from the second word *soccer* (used in the phrase *soccer ball*), respectively, which may fail in single one alignment.

4.1.2 Paraphrase identification

In this subsection, I introduce previous studies about the identification and extraction of paraphrases. One successful technique is associated with bilingual pivoting (Bannard and Callison-Burch, 2005; Zhao et al., 2008), in which alternative phrase translations are used as paraphrases at a certain probability. However, this technique requires large bilingual parallel corpora; moreover, word alignment errors likely cause noisy paraphrases. Another strategy is to extract paraphrases is monolingual phrase alignment between syntactic trees (Arase and Tsujii, 2017). This method identifies syntactic paraphrases under linguistically motivated grammar. The main difference between previous studies and my method is that they typically attempted an alignment between words or syntactic trees, whereas my method performs an alignment between meaning representations, which enables the acquisition of more general paraphrases by distinguishing functional words from content words. This point is important in distinguishing among different semantic relations (e.g., antonyms and synonyms). In addition, word and syntactic alignments potentially ignore coreferences, making it difficult to find relations between many-to-many sentences. Semantic alignments enable this because coreferences must refer to the same variable as the original entity.

The second way to extract paraphrases is to calculate the similarity between phrases using an additive composition model and selecting the phrase pair whose with maximum similarity paraphrases. Additive composition models (Mitchell and Lapata, 2010) present a method for computing meanings of phrases that utilizes

the average of vector representations of the constituent words. In this model, distributional vector representations are used to combine word vectors to obtain the meanings of compositional phrases (Socher et al., 2011). Currently, however, the additive composition model is theoretically limited to representing short phrases (i.e., phrases composed of two or three words) (Tian, Okazaki, and Inui, 2016; Tian, Okazaki, and Inui, 2017). In addition, it is also difficult to distinguish antonyms from synonyms in this model.

4.1.3 Motivation for phrase abduction

There are three main difficulties that prevent effective identification and use of phrasal linguistic knowledge in logical inference. The first difficulty is the presence of out-of-context phrase relations in popular databases such as the Paraphrase Database (PPDB) (Ganitkevitch, Van Durme, and Callison-Burch, 2013). PPDB may suggest paraphrases that do not adhere to the context of the relevant text segments nor to their semantic structure, which might be problematic.

The second difficulty is finding semantic phrase correspondences between the relevant text segments. Typical approaches only rely on surface (Beltagy et al., 2013) or syntactic correspondences (Arase and Tsujii, 2017), often producing inaccurate alignments that significantly impact logical inference capabilities. Instead, a mechanism to compute semantic phrase correspondences could potentially produce, if available, more coherent phrase pairs and solve the recurring issue of discontinuity.

The third difficulty is the intrinsic lack of coverage of databases for logical inference despite their large size. Whereas there is a relatively small number of possible word-to-word correspondences and thus their semantic relations can be enumerated, the same is not true for all phrase pairs that might be of interest. One alternative is to use functions of the infinite domain (e.g., cosine similarity) between phrase representations (Tian, Okazaki, and Inui, 2016), but these techniques are still under development, and we have not seen definitive successful applications when combined with logical inference systems.

In this chapter, I tackle these three problems. I propose an automatic phrase abduction mechanism to inject phrasal knowledge during the proof construction process. In addition, I consider multiple alignments by backtracking the decisions on variable and predicate unifications, which is a more flexible strategy. I represent logical formulas using graphs since this is a general formalism that is easy to visualize

and analyze. However, I use *natural deduction* (see Section 2.3) as a proof system instead of Markov Logic Networks for inference. In addition, I extract phrasal knowledge only from the natural deduction proof, not using supervised classification. Some research has investigated graph operations for semantic parsing (Reddy, Lapata, and Steedman, 2014; Reddy et al., 2016b) and abstractive summarization (Liu et al., 2015); I contribute to these ideas by proposing a subgraph mapping algorithm that is useful for performing natural language inference.

4.2 Phrase abduction

4.2.1 Logical formulas and graph representations

I formalize the phrase abduction mechanism by describing proof processes using graphs. In this subsection, I describe some definitions for corresponding logical semantic representations with graph representations.

For instance, the sentence *A girl is skipping rope on a sidewalk* can be analyzed as the following logical formula in Neo-Davidsonian Event Semantics.

$$\begin{aligned}\exists x_1 \exists x_2 \exists x_3 \exists y_1 (&\mathbf{girl}(x_1) \wedge \mathbf{rope}(x_2) \wedge \mathbf{sidewalk}(x_3) \wedge \mathbf{skip}(y_1) \\ &\wedge (\mathbf{subj}(y_1) = x_1) \wedge (\mathbf{dobj}(y_1) = x_2) \wedge \mathbf{on}(y_1, x_3))\end{aligned}$$

In this semantics, all content words (e.g., *girl* and *skip*) are represented as one-place predicates, e.g., $\mathbf{girl}(x_1)$ and $\mathbf{skip}(y_1)$. For functional words, a preposition like *on* is represented as a two-place predicate, e.g., $\mathbf{on}(y_1, x_3)$. A small set of semantic roles such as **subj** and **obj** is used as functional terms and equality ($=$) is used to connect an event and its participant, as in $\mathbf{subj}(y_1) = x_1$.

To be precise, the set of *atomic formulas* $\mathcal{A}$ in this event semantics is defined by the rule

$$\mathcal{A} ::= \mathbf{F}(t) \mid \mathbf{G}(t, u) \mid t = u \tag{4.1}$$

where $\mathbf{F}(t)$ is a one-place predicate (for content words), $\mathbf{G}(t, u)$ is a two-place predicate (for prepositions), and t and u are terms. A term is defined as a constant, a variable, or a functional term of the form $f(t)$, where f is a semantic role and t is a term. I call $\mathbf{F}(t)$ and $\mathbf{G}(t, u)$ *basic predicates*.

I call a formula constructed by conjunctions ($\wedge$) and existential quantifiers ($\exists$) a *basic formula* in event semantics. Thus, a set of basic formulas φ in event semantics is defined as:

$$\varphi ::= \mathcal{A} \mid \varphi \wedge \varphi \mid \exists t\, \varphi \tag{4.2}$$

The first example shows an instance of a basic formula, which captures the predicate-argument structure of a sentence.

On top of the system of basic formulas, we have a full language of event semantics with negation ($\neg$), disjunction ($\vee$), implication ($\rightarrow$), and universal quantifier ($\forall$). These operators are used to represent additional logical features.

There is a natural correspondence between basic formulas constructed in terms of $\wedge$ and $\exists$ in event semantics and directed acyclic graphs (DAGs). Figure 4.1 shows an example. See Liang, Jordan, and Klein (2011) and Jones (2016) for some variants of graphical representations of logical formulas.

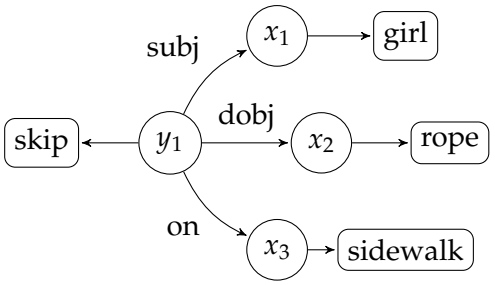

FIGURE 4.1: A graph for the event semantics formula.

In the graph representation, constants and variables correspond to vertices; both two-place predicates for prepositions (e.g., $\mathbf{on}(y_1, x_1)$) and functional terms for semantic roles (e.g., $\mathbf{subj}(y_1) = x_1$) are represented as edges. A one-place predicate $\mathbf{F}(t)$ in a logical formula can be represented as a functional relation $\mathbf{isa}(t, \mathbf{F})$, where **isa** is an expression relating a term t and a predicate **F** represented as a vertex. The **isa** edges are unlabeled for simplicity. Here, bound variables (e.g., x_1) correspond to vertices and functional relations (e.g., $\mathbf{subj}(y_1) = x_1$, $\mathbf{dobj}(y_1) = x_2$ and $\mathbf{on}(y_1, x_3)$) to edges. A one-place predicate $\mathbf{F}(t)$ in a logical formula can be understood as a functional relation $\mathbf{isa}(t, \mathbf{F})$, where **isa** is an expression relating a variable and a predicate represented as a vertex. The **isa** edges are unlabeled for simplicity.

4.2.2 Graph-based formulation of a theorem proving routine

To describe the phrase abduction mechanism, let us consider the proof $T \rightarrow H$ in the following sentence pair:

T : *A lady is cutting up some meat precisely*

H : *Some meat is being cut into pieces by a woman*

Figure 4.2 gives an outline of my graph-based formulation of a theorem proving routine. The first step is to obtain the graphical meaning representations of T' and H'. To begin with, the input sentence pair (T, H) is mapped onto a pair of formulas, (T', H') through CCG syntactic parsing and semantic composition. T' is initially set to the premise P, and H' to the goal G. Note that these are basic formulas, and they are thus decomposed to the following sets of formulas $\mathcal{P}$ and $\mathcal{G}$, respectively:

$$\mathcal{P} = \{\mathbf{lady}(x_1), \mathbf{meat}(x_2), \mathbf{cut}(y_1), \mathbf{up}(y_1), \mathbf{precisely}(y_1), \mathbf{subj}(y_1)=x_1, \mathbf{obj}(y_1)=x_2\}$$

$$\mathcal{G} = \{\mathbf{woman}(x_3), \mathbf{meat}(x_4), \mathbf{cut}(y_2), \mathbf{piece}(x_5), \mathbf{into}(y_2, x_5), \mathbf{subj}(y_2)=x_3, \mathbf{obj}(y_2)=x_4\}$$

Steps 1 to 3 in Figure 4.2 demonstrate the variable unification routine and word axiom injection using graphs. In the graph representations, vertices are variables (e.g., entities x_i or events y_j) or predicates (e.g., **lady**, **woman**) and edges are semantic roles (e.g., **subj**, **obj**) or prepositions (e.g., **into**). Note that in step 1, all variables in formulas in $\mathcal{P}$ or $\mathcal{G}$ are initially different.

In step 2, we run a theorem proving mechanism that uses graph terminal vertices as anchors to unify variables between formulas in $\mathcal{P}$ and those in $\mathcal{G}$. The premise $\mathbf{meat}(x_2)$ in $\mathcal{P}$ matches the predicate **meat** of the sub-goal $\mathbf{meat}(x_4)$ in $\mathcal{G}$ and the variable unification $x_4 := x_2$ is applied (and similarly for the sub-goal $\mathbf{cut}(y_2)$ in $\mathcal{G}$ with the variable unification $y_2 := y_1$). Then, a set of formulas in $\mathcal{G}$ is updated as follows:

$$\mathcal{P} = \{\mathbf{lady}(x_1), \mathbf{meat}(x_2), \mathbf{cut}(y_1), \mathbf{up}(y_1), \mathbf{precisely}(y_1), \mathbf{subj}(y_1)=x_1, \mathbf{obj}(y_1)=x_2\}$$

$$\mathcal{G} = \{\mathbf{woman}(x_3), \cancel{\mathbf{meat}(x_2)}, \cancel{\mathbf{cut}(y_1)}, \mathbf{piece}(x_5), \mathbf{into}(y_1, x_5), \mathbf{subj}(y_1)=x_3, \cancel{\mathbf{obj}(y_1)=x_2}\}$$

In step 3, we use the previously described variable unification on y_1, the **subj**

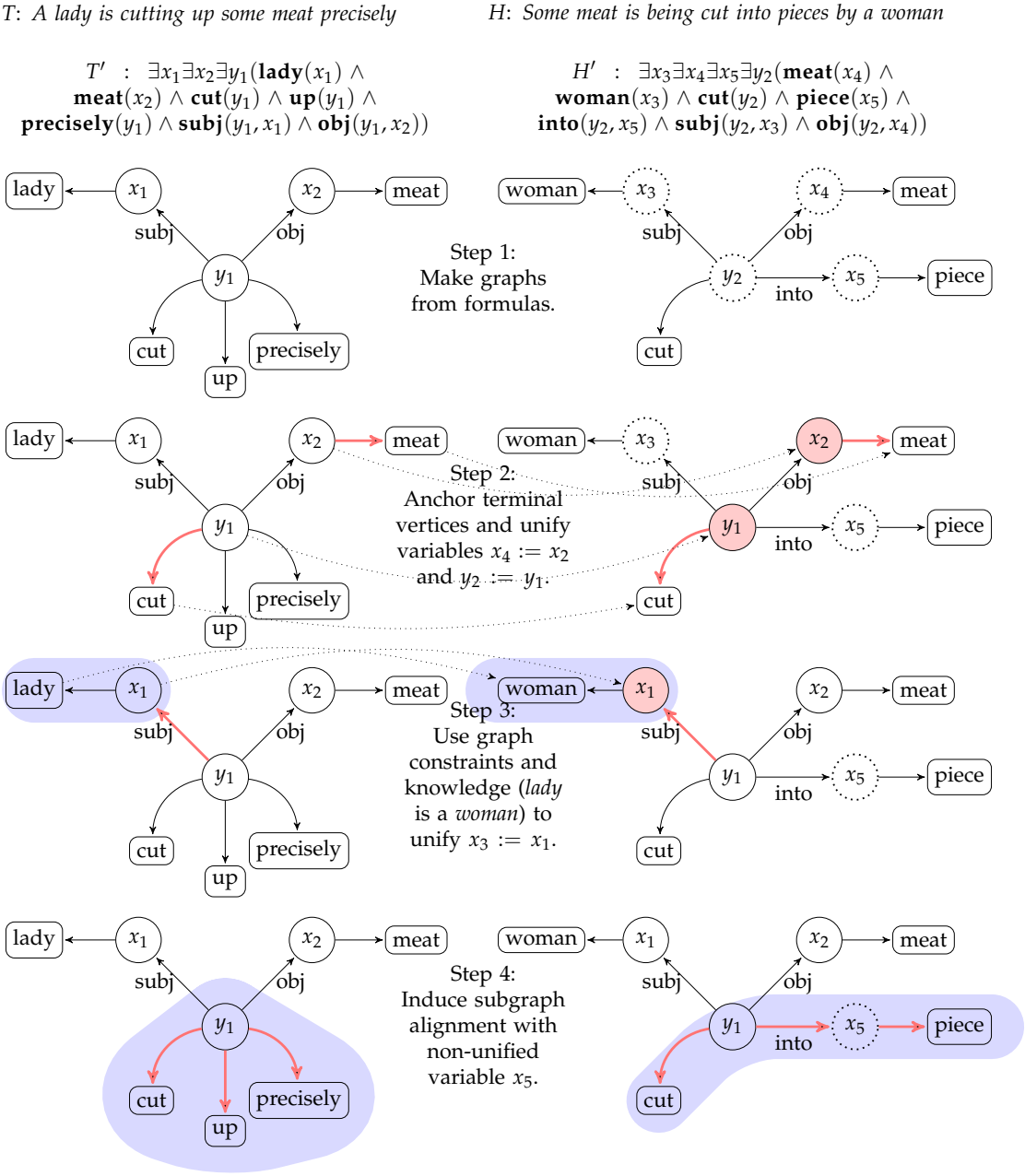

FIGURE 4.2: A graph representation of a theorem proving routine on basic formulas and variable unification. Dotted circles represent non-unified variables at each step, whereas edges without labels are attributes. The graph on the left side shows the set of premises $\mathcal{P}$, and the graph on the right side shows the set of sub-goals $\mathcal{G}$. Colored subgraphs represent a word or a phrase to which the axiom injection mechanism applies.

edge in $\mathcal{P}$ and $\mathcal{G}$ and the axiom $\forall x.\mathbf{lady}(x) \rightarrow \mathbf{woman}(x)$ from external lexical knowledge to infer that $x_3 := x_1$. Then, a set of formulas in $\mathcal{G}$ is updated as follows:

$$\mathcal{P} = \{\mathbf{lady}(x_1), \mathbf{meat}(x_2), \mathbf{cut}(y_1), \mathbf{up}(y_1), \mathbf{precisely}(y_1), \mathbf{subj}(y_1) = x_1, \mathbf{obj}(y_1) = x_2\}$$

$$\mathcal{G} = \{\cancel{\mathbf{woman}(x_1)}, \cancel{\mathbf{meat}(x_2)}, \cancel{\mathbf{cut}(y_1)}, \mathbf{piece}(x_5), \mathbf{into}(y_1, x_5), \cancel{\mathbf{subj}(y_1) = x_1}, \cancel{\mathbf{obj}(y_1) = x_2}\}$$

4.2.3 Problem of phrase pair detection

There is one critical reason that the word-to-word axiom injection described in Section 2.3 fails to detect phrase-to-phrase correspondences. That is, the natural deduction mechanism decomposes the goal G into atomic sub-goals that are then proved *one-by-one* (word-by-word), independent of each other except for the variable unification effect. This mechanism is particularly problematic when we attempt to prove phrases that resist decomposition, two-place predicates (e.g., $\mathbf{into}(x, y)$), or failures in variable unification (e.g., due to inaccurate semantics). Thus, I propose a method to detect phrase-to-phrase correspondence through natural deduction proofs.

I detect phrase-to-phrase entailing relations between T and H by finding alignments between the subgraphs of their meaning representations when $T' \rightarrow H'$ or $T' \rightarrow \neg H'$ hold. Finding subgraph alignments is a generalization of the subgraph isomorphism problem, which is NP-complete (Emmert-Streib, Dehmer, and Shi (2016), provides a good overview). I approximate a solution to this problem by using a combination of a backtracking variable unification[1] and a deterministic graph search of the neighborhood of non-unified variables.

Using the running example in Figure 4.2, step 4 displays the subgraph alignment. Here, the set of premises and sub-goals $\mathcal{P}, \mathcal{G}$ are as follows:

$$\mathcal{P} = \{\mathbf{lady}(x_1), \mathbf{meat}(x_2), \mathbf{cut}(y_1), \mathbf{up}(y_1), \mathbf{precisely}(y_1), \mathbf{subj}(y_1) = x_1, \mathbf{obj}(y_1) = x_2\}$$

$$\mathcal{G} = \{\cancel{\mathbf{woman}(x_1)}, \cancel{\mathbf{meat}(x_2)}, \cancel{\mathbf{cut}(y_1)}, \mathbf{piece}(x_5), \mathbf{into}(y_1, x_5), \cancel{\mathbf{subj}(y_1) = x_1}, \cancel{\mathbf{obj}(y_1) = x_2}\}$$

The variable x_5 in the graph of $\mathcal{G}$ cannot be unified with any variable in the graph of $\mathcal{P}$. This is a very common case in natural language inference, as there might be concepts in H that are not directly supported by concepts in T.

In this research, I propose spanning a subgraph starting at non-unified variables (e.g., x_5 in $\mathcal{G}$) whose boundaries are semantic roles (e.g., **subj**, **obj**). Its candidate semantics from $\mathcal{P}$ are then the attributes of its corresponding unified variables from $\mathcal{G}$ (e.g., *cut up precisely* $\rightarrow$ *cut into pieces*).

[1] A variable unification can be seen as a variable renaming (or vertex relabeling, in the context of graphs), as implemented in most modern theorem provers.

4.2.4 Graph-based formulation for phrase abduction

To formalize this solution I introduce some graph notation. Let $V = \mathcal{V}^u \cup \mathcal{V}^{\bar{u}} \cup \mathcal{L}$ be the set of vertices, where $\mathcal{V}^u$ is the set of unified variables (e.g. x_1, x_2, y_1), $\mathcal{V}^{\bar{u}}$ is the set of non-unified variables (e.g. x_5), and $\mathcal{L}$ is a set of predicates (e.g., *lady*, *woman*). Let E be the set of labeled, directed edges $\langle v, l, v' \rangle$, where $v, v' \in V$, and l are labels that may represent a functional relation **isa**, a preposition, or a semantic role. I denote a set of two-place predicates for prepositions as PREP and a set of functional terms for semantic roles as ARGS; e.g., ARGS $= \{\mathbf{subj}, \mathbf{obj}\}$. A graph that represents $\mathcal{P}$ is then a tuple $G_\mathcal{P} = \langle V_\mathcal{P}, E_\mathcal{P} \rangle$, and similarly, for $\mathcal{G}$, $G_\mathcal{G} = \langle V_\mathcal{G}, E_\mathcal{G} \rangle$.

I define a function to span a subgraph in the neighborhood of non-unified variables $v \in \mathcal{V}^{\bar{u}}_\mathcal{G}$ in the graph of $\mathcal{G}$. I call a connected set of edges in which no semantic roles appear a *phrase set*. A phrase set is defined as follows:

$$\{\langle v, l, v' \rangle \mid l \notin \text{ARGS}\} \tag{4.3}$$

Let $E(x)$ be the phrase set in E such that each vertex is connected to x with an incoming or outgoing edge, that is,

$$E(x) = \{(v_i, l, v_k) \in E \mid (x = v_i \vee x = v_k) \wedge l \notin \text{ARGS}\} \tag{4.4}$$

Note that $E(x)$ induces a subgraph in a given graph G and the condition $l \notin$ ARGS sets the boundaries of the subgraph by excluding the semantic roles of verb phrases. Given two phrase sets E and E', I say E' is reachable from E, written $E \sim E'$, if E and E' share at least one variable vertex, that is, $fv(E) \cap fv(E') \neq \emptyset$. Let $\sim^*$ be the transitive closure of $\sim$. Given a set of edges $E_\mathcal{G}$ and a variable v, I define the *extended phrase set*, written Reach(v), as follows:

$$\text{Reach}(v) = \{e \in E \mid E_\mathcal{G}(v) \sim^* E\} \tag{4.5}$$

That is, Reach(v) is the set of edges e that can be reached from v without crossing an edge with a semantic role label. This function defines a partition or equivalence class for non-unified variables $v \in \mathcal{V}^{\bar{u}}_\mathcal{G}$, and each of these partitions induces a (possibly discontinuous) phrase in $\mathcal{G}$ that remains unproved.

The corresponding subgraph in $\mathcal{P}$ to each of these partitions is given by the vertices and edges connected with a path of length one to the unified variables that

appear in Reach(v). That is,

$$\mathrm{Corr}(v) = \left\{ e \in E_{\mathcal{P}}(v'), v' \in V_{\mathcal{G}}^{[v]} \cap V_{\mathcal{P}} \right\} \tag{4.6}$$

where $V_{\mathcal{G}}^{[v]}$ denotes the vertices in the subgraph of $\mathcal{G}$ induced by the partition Reach(v).

A subgraph alignment between $\mathcal{P}$ and $\mathcal{G}$ is given by the pair of $\langle \mathrm{Corr}(v), \mathrm{Reach}(v) \rangle$ for all $v \in \mathcal{V}_{\mathcal{G}}^{\bar{u}}$, where the phrases can be read from the predicates in the vertices and edges labeled with prepositions.

I define a mapping $(\cdot)^\bullet$ from a labeled edge $\langle v, l, v' \rangle$ to an atomic formula as follows.

$$\langle v, l, v' \rangle^\bullet = \begin{cases} v'(v) & \text{if } l \text{ is } \mathbf{isa} \\ l(v, v') & \text{if } l \in \mathrm{PREP} \\ l(v) = v' & \text{if } l \in \mathrm{ARGS} \end{cases} \tag{4.7}$$

Let E be a set of labeled edges, and let $E^\bullet$ be $\left\{ \langle v, l, v' \rangle^\bullet \mid \langle v, l, v' \rangle \in E \right\}$. The phrasal axiom generated for each non-unified variable $v \in \mathcal{V}_{\mathcal{G}}^{\bar{u}}$ is defined as

$$\forall \theta_{\mathrm{C}}. (\bigwedge \mathrm{Corr}(v)^\bullet \rightarrow \exists \theta_{\mathrm{R}}. (\bigwedge \mathrm{Reach}(v)^\bullet)) \tag{4.8}$$

where θ_{C} is a set of free variables appearing in $\mathrm{Corr}(v)^\bullet$ (which includes v) and θ_{R} is a set of free variables appearing in $\mathrm{Reach}(v)^\bullet$ but not in $\mathrm{Corr}(v)^\bullet$.

In Figure 4.2, the only non-unified variable in the sub-goal in step 4 is x_5, that is, $\mathcal{V}_{\mathcal{G}}^{\bar{u}} = \{x_5\}$. Then, starting from the variable x_5, Reach(x_5) is

$$\{ \langle y_1, \mathbf{into}, x_5 \rangle, \langle x_5, \mathbf{isa}, \mathbf{piece} \rangle \}.$$

Now $V_{\mathcal{G}}^{[x_5]} = \{y_1, x_5\}$, and thus Corr$(x_5)$ is

$$\{ \langle y_1, \mathbf{isa}, \mathbf{cut} \rangle, \langle y_1, \mathbf{isa}, \mathbf{up} \rangle, \langle y_1, \mathbf{isa}, \mathbf{precisely} \rangle \}.$$

Finally, the following is the axiom generated from $\langle \mathrm{Corr}(x_5), \mathrm{Reach}(x_5) \rangle$[2].

$$\forall y_1(\mathbf{cut}(y_1) \wedge \mathbf{up}(y_1) \wedge \mathbf{precisely}(y_1) \rightarrow \exists x_5(\mathbf{into}(y_1, x_5) \wedge \mathbf{piece}(x_5))).$$

4.2.5 Another formulation for phrase abduction

The phrase abduction mechanism can be generally applied to logical inference, and it can thus also be formalized without graphs. In this subsection, I describe another way to formalize phrase abduction. Here, I define a set of basic predicates including variable x in a pool of premises $\mathcal{P}$ and sub-goals $\mathcal{G}$ as a *phrase set*. $fv(\varphi)$ is a set of free variables included in basic predicates φ, ψ.

$$\mathcal{P}_x \stackrel{def}{\equiv} \{\varphi \in \mathcal{P} \mid x \in fv(\varphi)\} \quad \mathcal{G}_x \stackrel{def}{\equiv} \{\psi \in \mathcal{G} \mid x \in fv(\psi)\} \tag{4.9}$$

Next, I call $\Phi \sim \Psi$, as Φ can *reach* Ψ. That means a phrase set Ψ and Φ shares a free variable. If Φ is a set of logical formulas, I define $fv(\Phi) \stackrel{def}{\equiv} \bigcup_{\varphi \in \Phi} fv(\varphi)$.

$$\Phi \sim \Psi \quad \stackrel{def}{\equiv} \quad fv(\Phi) \cap fv(\Psi) \neq \emptyset \tag{4.10}$$

$\sim^*$ is a transitive closure of $\sim$. I define an extended phrase set including a variable x in a pool of premises $\mathcal{P}$ and sub-goals $\mathcal{G}$ as follows:

$$\mathcal{P}^*_x \stackrel{def}{\equiv} \{\varphi \in \Phi \mid \mathcal{P}_x \sim^* \Phi\} \quad \mathcal{G}^*_x \stackrel{def}{\equiv} \{\psi \in \Psi \mid \mathcal{G}_x \sim^* \Psi\} \tag{4.11}$$

Considering a pair of $\mathcal{P}^*_x$ and $\mathcal{G}^*_x$ for each variable x included in $fv(\mathcal{G})$, a phrase-to-phrase axiom candidate is determined as the formula (4.12) by calculating $U = fv(\mathcal{P}^*_x), V = fv(\mathcal{G}^*_x) - fv(\mathcal{P}^*_x)$, where $U = u_1, \ldots, u_m, V = v_1, \ldots, v_n$

$$\forall u_1, \ldots, u_m \, ((\bigwedge \mathcal{P}^*_x) \rightarrow \exists v_1 \ldots v_n \, (\bigwedge \mathcal{G}^*_x)) \tag{4.12}$$

In step 4 of Figure 4.2, the variable x_5 in $\mathcal{G}$ cannot unify to any variable in $\mathcal{P}$. Consider a phrase set and an enhanced phrase set for each variable included in $fv(\mathcal{G}) = \{y_1, x_5\}$. Here, $\mathcal{G}_{y_1}$ can *reach* $\mathcal{G}_{x_5}$, and the enhanced phrase set $\mathcal{G}^*_{y_1}, \mathcal{G}^*_{x_5}$ is

[2] Note that this axiom is logically equivalent to

$$\forall y_1(\mathbf{cut}(y_1) \wedge \mathbf{up}(y_1) \wedge \mathbf{precisely}(y_1) \rightarrow \exists x_5(\mathbf{cut}(y_1) \wedge \mathbf{into}(y_1, x_5) \wedge \mathbf{piece}(x_5)))$$

indicated in the colored subgraphs in step 4 of Figure 4.2.

represented as $\mathcal{G}_{y_1} \cup \mathcal{G}_{x_5}$.

$$\begin{aligned} \mathcal{G}_{y_1} &= \{\textbf{into}(y_1, x_5)\} \\ \mathcal{G}_{x_5} = \mathcal{G}^*_{x_5} = \mathcal{G}^*_{y_1} &= \{\textbf{into}(y_1, x_5), \textbf{piece}(x_5)\} \\ \mathcal{P}_{y_1} = \mathcal{P}^*_{y_1} &= \{\textbf{cut}(y_1), \textbf{up}(y_1), \textbf{precisely}(y_1)\} \end{aligned}$$

Hence, the phrasal axiom about variables $fv(\mathcal{G}) = \{y_1, x_5\}$ can be detected as follows:

$$\forall y_1(\textbf{cut}(y_1) \wedge \textbf{up}(y_1) \wedge \textbf{precisely}(y_1) \rightarrow \exists x_5(\textbf{into}(y_1, x_5) \wedge \textbf{piece}(x_5))).$$

4.2.6 Non-basic formulas

If formulas P and G are not basic formulas (i.e., they contain logical operators other than conjunction ($\wedge$) and existential quantifiers ($\exists$)), they are decomposed according to inference rules of natural deduction. As described in Section 2.3.3, there are two types of inference rules: introduction rules decompose a goal formula into smaller sub-goals and elimination rules decompose a formula in the pool of premises into smaller ones. By applying inference rules described in Figure 2.5, a proof of non-basic formulas appearing in sub-goals can be decomposed into a set of subproofs that only have basic formulas in sub-goals. If a universal quantifier appears in premises, it is treated in the same way as other premises.

For example, consider the following sentence pair with the gold label "no" (contradiction):

$$\begin{aligned} T &: \textit{A man is not cutting a potato} \\ H &: \textit{A man is slicing a potato into pieces} \end{aligned}$$

Figure 4.3 shows the proof process of $T' \rightarrow \neg H'$. To prove the contradiction, the formulas T' and $\neg H'$ are set to P and G, respectively. Then, the negation in G is removed by applying the introduction rule ($\neg$-$INTRO$) to G. Here, **False** is the propositional constant denoting the contradiction. In the second stage of the proof, the goal is to prove **False** in G_0 from the two premises P and P_0. By applying the elimination rule ($\neg$-$ELIM$) to P, we can eliminate the negation from P, resulting in the new goal G_1.

As both the premise P_0 and the sub-goal G_1 are basic formulas, the procedure described in the previous sections applies to the pair (P_0, G_1); these basic formulas are decomposed into atomic ones, and then word-to-word abduction generates the desired axiom $\forall y_1(\mathbf{cut}(y_1) \rightarrow \mathbf{slice}(y_1))$. Finally, the graph alignment applies in the same way as described in Figure 4.2, which generates the phrasal axiom:

$$\forall y_1(\mathbf{cut}(y_1) \rightarrow \exists x_5(\mathbf{into}(y_1, x_5) \wedge \mathbf{piece}(x_5)))$$

Using this axiom, one can complete the proof of the contradiction between T' and H'.

$$\frac{\begin{array}{l} P : \neg\exists y_1 \exists x_1(\mathbf{man}(x_1) \wedge \mathbf{cut}(y_1) \wedge \mathbf{potato}(x_2) \\ \quad \wedge(\mathbf{subj}(y_1) = x_1) \wedge (\mathbf{obj}(y_1) = x_2)) \end{array}}{\begin{array}{l} G : \neg\exists y_1 \exists x_1 \exists x_2 \exists x_3(\mathbf{man}(x_1) \wedge \mathbf{slice}(y_1) \wedge \mathbf{potato}(x_2) \\ \quad \wedge \mathbf{into}(y_1, x_3) \wedge \mathbf{piece}(x_3) \wedge (\mathbf{subj}(y_1) = x_1) \wedge (\mathbf{obj}(y_1) = x_2) \end{array}}$$

$$\Big\downarrow \neg\text{-INTRO } (G_0)$$

$$\frac{\begin{array}{l} P : \neg\exists y_1 \exists x_1(\mathbf{man}(x_1) \wedge \mathbf{cut}(y_1) \wedge \mathbf{potato}(x_2) \\ \quad \wedge(\mathbf{subj}(y_1) = x_1) \wedge (\mathbf{obj}(y_1) = x_2) \\ P_0 : \exists y_1 \exists x_1 \exists x_2 \exists x_3(\mathbf{man}(x_1) \wedge \mathbf{slice}(y_1) \wedge \mathbf{potato}(x_2) \\ \quad \wedge \mathbf{into}(y_1, x_3) \wedge \mathbf{piece}(x_3) \wedge (\mathbf{subj}(y_1) = x_1) \wedge (\mathbf{obj}(y_1) = x_2) \end{array}}{G_0 : \mathbf{False}}$$

$$\Big\downarrow \neg\text{-ELIM } (P)$$

$$\frac{P_0 : \exists y_1 \exists x_1 \exists x_2 \exists x_3(\mathbf{man}(x_1) \wedge \mathbf{slice}(y_1) \wedge \mathbf{potato}(x_2) \\ \quad \wedge \mathbf{into}(y_1, x_3) \wedge \mathbf{piece}(x_3) \wedge (\mathbf{subj}(y_1) = x_1) \wedge (\mathbf{obj}(y_1) = x_2)}{\begin{array}{l} G_1 : \exists y_1 \exists x_1(\mathbf{man}(x_1) \wedge \mathbf{cut}(y_1) \wedge \mathbf{potato}(x_2) \\ \quad \wedge(\mathbf{subj}(y_1) = x_1) \wedge (\mathbf{obj}(y_1) = x_2) \end{array}}$$

FIGURE 4.3: Contradiction proof process for non-basic formulas.

4.3 Experiments

4.3.1 Experimental setting

To evaluate the utility of the phrase abduction mechanism in logical inference, I evaluated my method with the RTE dataset. For an RTE dataset, I selected the SICK dataset, which contains logically challenging problems involving quantifiers, negation, conjunction, and disjunction, as well as inferences with lexical and phrasal knowledge.

I first attempted the proof using the training and trial split of the SICK dataset and extracted phrasal knowledge. In RTE tasks, we need to consider a directional semantic relation between words such as hypernym and hyponym to prove an entailment relation or contradiction. Hence, to extract phrasal knowledge for RTE tasks, I used the training and trial split of the dataset whose gold label is *entailment* or *contradiction*, excluding those having the *neutral* label. Next, I evaluated the system performance using the test split of the SICK dataset with the phrasal knowledge extracted from the training and trial split.

I compared phrase abduction with different experimental conditions. **No axioms** is my system without axiom injection. **W2W** is my system with the previous word abduction (Martínez-Gómez et al., 2017). **P2P** is my system with phrase abduction; **W2W+P2P** combines phrase abduction with word abduction. **W2W+RF** is my hybrid system using logical inference and the random forest model described in Chapter 3.

In addition, I compared my system with three purely logic-based (unsupervised) approaches: **The Meaning Factory** (Bjerva et al., 2014), **LangPro** (Abzianidze, 2015), and **UTexas** (Beltagy et al., 2014). I also compared my system with machine learning-based approaches: the current state-of-the-art deep learning model **GRU** (Yin and Schütze, 2017), a log-linear regression model **SemEval-2014 best** (Lai and Hockenmaier, 2014), and a hybrid system combining a logistic regression model and probabilistic logic **PL+eclassif** (Beltagy et al., 2016).

4.3.2 Extracted paraphrases

I extracted 9,445 axioms from the training and trial split of the SICK dataset. The proving time average to extract phrasal axioms was only 3.0 seconds in per sentence pair. My phrase-pair identification is a polynomial-time instance of the graph matching problem where the vertex cover set (maximum number of variables in a phrase) is bounded to a small constant (I ignored phrases with more than five predicates). Regarding the complexity of variable unification, I only do first-order proving in the SICK dataset, which is computationally fast. However, efficient higher-order theorem proving is also possible if I restrict the higher-order constructions as described by Huet (1975) and Mineshima et al. (2015).

Table 4.1 shows some examples of paraphrases I extracted from the natural deduction proof. In particular, the examples of verb phrases show my method has

potential for capturing long paraphrases. Each paraphrase in Table 4.1 is not contained in WordNet and PPDB. There are many instances of non-contiguous phrases in the SICK dataset, in particular, verb-particle phrases. As shown in Table 4.1, my semantic method can identify non-contiguous phrases through the variable unification process, which is one of the main advantages over other shallow/syntactic methods. In addition, Table 4.1 shows my method is not limited to hypernym or hyponym relations, but is capable of detecting antonym phrases.

Kind	Text	Hypothesis
noun phrase	***A blond woman*** *is sitting on the roof of a yellow vehicle and two people are inside*	***A woman with blond hair*** *is sitting on the roof of a yellow vehicle and two people are inside*
	A man is singing and playing a ***musical instrument***	*A man is singing and playing a* ***guitar***
	A ***big green ball*** *is knocking a potato*	*A* ***large green colored ball*** *is hitting a potato*
verb phrase	*The person is* ***setting fire to*** *the cameras*	*Some cameras are being* ***burned*** *by a person* ***with a blow torch***
	A man and a woman are ***walking*** *together* ***through the woods***	*A man and a woman are hiking* ***through a wooded area***
	A badger is ***shrewdly digging the earth***	*The badger isn't* ***burrowing a hole***
	A woman is ***sewing with machine***	*A woman is* ***using machine made for sewing***
	Two people are ***standing in the ocean***	*Two people are* ***wading through the water***
prepositional phrase	*A child, who is small, is outdoors climbing steps outdoors* ***in an area full of grass***	*A small child is outdoors climbing steps* ***in a grassy area***
	Two women are wearing bikinis and are walking ***on the sand***	*There are no women wearing bikinis* ***on the sandy beach***
	A group of people are standing ***around a sound mixing table***	*Some people is looking* ***at sound equipment***
antonym phrase	*A woman is* ***putting make-up on***	*The woman is* ***removing make-up***
	A crowd of people is ***near*** *the water*	*A crowd of people is* ***far from*** *the water*

TABLE 4.1: Examples of phrase alignments by phrasal axiom injection.

4.3.3 Comparison with other systems

Table 4.2 shows the experimental results.

	Prec.	Rec.	Acc.
W2W+RF	0.906	0.797	0.877
GRU	–	–	0.871
PL+eclassif	–	–	0.851
SemEval2014 Best Score	0.816	0.819	0.846
The Meaning Factory	0.936	0.606	0.816
LangPro	0.980	0.581	0.814
UTexas	–	–	0.804
W2W+P2P	0.842	0.773	0.843
W2W	0.971	0.636	0.831
P2P	0.856	0.721	0.830
No axioms	0.989	0.465	0.767

TABLE 4.2: RTE results on the SICK dataset.

The results show that combination of word abduction and phrase abduction improved the accuracy. When the **W2W+P2P** result is substituted for the **W2W** result, there is a 0.011 increase in accuracy (from 0.831 to 0.843). The accuracy of **P2P** is almost equal to that of **W2W**. This is because the recall improves from 0.636 to 0.721, while the precision decreases from 0.971 to 0.856. The increase in false positive cases appeared to cause this result; some details of false positive cases are described in the next subsection. **W2W+P2P** outperformed other purely logic-based systems. The machine learning-based approaches including my hybrid system **W2W+RF** outperformed **W2W+P2P**. Unlike these approaches, parameter estimation is not used in my phrase abduction method. This suggests that my method has the potential to increase accuracy by combining my hybrid system described in Chapter 3 with my phrase abduction.

4.3.4 Positive examples and error analysis

Table 4.3 shows some positive and negative examples on RTE with the SICK dataset. For ID 9491, the sentence pair requires the paraphrase from *a field of brown grass* to *a grassy area*. This relation is not included in previous lexical knowledge bases. The phrasal axiom injection can correctly generate this paraphrase from a natural deduction proof and the system correctly proves the entailment relation.

ID 2367 is also a positive example of phrasal axiom injection. The phrasal axiom between *set fire to cameras* and *burn cameras with a blow torch* was extracted from the

following sentence pair with the *entailment* label:

T_1 : *Some cameras are being burned by a person with a blow torch*

H_1 : *The person is setting fire to the cameras*

This example shows that the semantic alignment succeeds in acquiring a general paraphrase by separating logical expressions such as *some* from content words and also by taking into account syntactic structures such as the passive-active alternation.

For ID 3628, the axiom shown in Table 4.3 was extracted from the following sentence pair with the *entailment* label:

T_1 : *A woman is putting meat in a pan*

H_1 : *Someone is dropping the meat into a pan*

However, the phrase *drop over* does not entail the phrase *drop into*, and a proof for the inference is over-generated in ID 3628. I extracted all possible phrasal axioms from the training and trial split of the SICK dataset, so noisy axioms can be extracted as a consequence of multiple factors, such as parsing errors or potential disambiguation in the SICK dataset. One possible solution for decreasing such noisy axioms would be to use additive composition models (Tian, Okazaki, and Inui, 2016) and asymmetric learnable scoring functions to calculate the confidence of these asymmetric entailing relations between phrases.

ID 96 is also an example of over-generation of axioms. The first axiom, $\forall y_1(\mathbf{jump}(y_1) \rightarrow \exists x_1(\mathbf{in}(y_1, x_1) \wedge \mathbf{air}(x_1)))$ was extracted from the proof of $T_1 \rightarrow H_1$:

T_1 : *A child in a red outfit is jumping on a trampoline*

H_1 : *A little boy in red clothes is jumping in the air*

The second axiom $\forall y_1(\mathbf{man}(y_1) \rightarrow \mathbf{biker}(y_1))$ was extracted from the proof of $T_2 \rightarrow H_2$:

T_2 : *A man on a yellow sport bike is doing a wheelie and a friend on a black bike is catching up*

H_2 : *A biker on a yellow sport bike is doing a wheelie and a friend on a black bike is catching up*

Although these axioms play a role in the proofs of $T_1 \rightarrow H_1$ and $T_2 \rightarrow H_2$, the wrong axiom $\forall y_1(\mathbf{man}(y_1) \rightarrow \mathbf{biker}(y_1))$ causes the over-generation of a proof for the inference in ID 96. The correct axiom would instead be $\forall x_1 \forall y_1(\mathbf{man}(y_1) \wedge \mathbf{on}(y_1, x_1) \wedge \mathbf{bike}(x_1) \rightarrow \mathbf{biker}(y_1))$. In this case, it is necessary to bundle predicates in a noun-phrase by specifying the types of a variable (i.e., entity or event) when making phrase alignments.

For ID 408, the word *explorer* is not contained in the training and trial split and hence the relevant axiom $\forall x_1(\mathbf{explorer}(x_1) \rightarrow \mathbf{people}(x_1))$ was not generated. While my logic-based method enables identification of semantic phrase correspondences in a sentence pair in an unsupervised way, the next step is to predict unseen paraphrases of this type.

ID	Sentence Pair	Gold	Pred	Axiom
9491	*A group of four brown dogs are playing in a field of brown grass* *Four dogs are playing in a grassy area*	Yes	Yes	$\forall x_1(\mathbf{field}(x_1) \land \mathbf{brown}(x_1) \land \mathbf{grass}(x_1)$ $\rightarrow \mathbf{grassy}(x_1) \land \mathbf{area}(x_1))$
2367	*A person is burning some cameras with a blow torch* *The person is setting fire to the cameras*	Yes	Yes	$\forall x_1 \forall y_1(\mathbf{burn}(y_1) \land \mathbf{with}(y_1, x_1) \land \mathbf{blow\_torch}(x_1) \land \mathbf{camera}(\mathbf{obj}(y_1))$ $\rightarrow \mathbf{set}(y_1) \land \mathbf{fire}(\mathbf{obj}(y_1)) \land \mathbf{to}(y_1, \mathbf{obj}(y_1)) \land \mathbf{camera}(\mathbf{obj}(y_1)))$
3628	*A pan is being dropped over the meat* *The meat is being dropped into a pan*	Unk	Yes	$\forall y_1(\mathbf{pan}(\mathbf{obj}(y_1)) \rightarrow \mathbf{into}(y_1, \mathbf{obj}(y_1)))$
96	*A man is jumping into an empty pool* *There is no biker jumping in the air*	Unk	No	$\forall y_1(\mathbf{jump}(y_1) \rightarrow \exists x_1(\mathbf{in}(y_1, x_1) \land \mathbf{air}(x_1)))$ $\forall y_1(\mathbf{man}(y_1) \rightarrow \mathbf{biker}(y_1))$
408	*A group of explorers is walking through the grass* *Some people are walking*	Yes	Unk	

TABLE 4.3: Positive and negative examples on RTE from the SICK dataset.

Chapter 5

Conclusion

5.1 Summary of thesis

In this thesis, I explored how to calculate textual entailment and similarity between sentences using natural deduction proofs. My hypothesis is that *observing proof processes when testing the semantic relations is useful for capturing textual entailment and similarity more precisely*.

In Chapter 3, I developed a new hybrid approach to learning textual entailment and similarity by using proof processes. I attempted a natural deduction proof with my logical inference system, aiming to prove bidirectional entailment relations between sentences. Then, I combined logic-based features extracted from proof processes for proving the bidirectional entailment relations with non-logic-based features.

In Chapter 4, I developed a phrasal abduction mechanism with a careful treatment of the theorem proving process. I proposed an automatic phrase abduction mechanism to inject phrasal knowledge during the proof construction process. In addition, I considered multiple alignments by backtracking the decisions on variable and predicate unifications, which is a more flexible strategy. I represented logical formulas using directed acyclic graphs, since this is a general formalism that is easy to visualize and analyze. Then, I formalized a theorem proof routine and variable unification using graphs. I detected phrase-to-phrase semantic relations between the sentences by finding alignments between the subgraphs of their meaning representations when the proofs of an entailment relation or contradiction hold.

I summarize the findings of this thesis as follows:

1. Proof processes help to capture syntactic structures and logical relations in sentences.

2. Proof processes are effective features for STS and RTE tasks.

3. Proof processes can be useful for sentences with the neutral label.

4. Proof processes help to detect various phrasal knowledge.

Proof processes help to capture syntactic structures and logical relations

Experiment results showed that my hybrid approach to learning textual similarity from proof processes achieved competitive performance on the semantic textual similarity (STS) task. Experiment results also showed that my hybrid approach could be applied to the recognizing textual entailment (RTE) task and achieved state-of-the-art performance on the RTE task. The case analysis indicates that my approach has the capability of reflecting the syntactic structures or logical relations in sentences, which outperformed state-of-the-art performance.

Proof processes are effective features for STS and RTE tasks

The feature ablation study for both STS and RTE tasks showed that my proposed logic-based features have more impact on improving the accuracy than non-logic-based features. In particular, the evaluation results indicate that proof processes are effective features for learning textual similarity and entailment.

Proof processes can be useful for sentences with the neutral label

Experiments showed that the Pearson correlation of the "neutral" portion of the SICK dataset was 0.766, which suggests that my hybrid approach can be applied not only to sentence pairs with entailment/contradiction labels but also to sentence pairs with neutral labels.

Proof processes help to detect various phrasal knowledge

Experiment results demonstrated that my phrasal abduction mechanism automatically detects various phrase correspondences including antonym phrases and non-contiguous phrases, which is one of the main advantages over other shallow/syntactic methods. Experiment results also showed that the phrasal abduction mechanism compensated for the lack of phrasal knowledge in logical inference and achieved the best performance in the RTE task among purely logic-based approaches.

5.2 Future work

In this section, I list future work directions related to this thesis.

Combination of a hybrid approach and a phrase abduction mechanism

One direction of the future work is to explore how to combine my hybrid approach described in Chapter 3 with phrase abduction described in Chapter 4. To do that, we have to consider how to calculate the score of a phrasal axiom. As shown in section 4.1, an additive composition model (Mitchell and Lapata, 2010) is used for calculating the lexical similarity between phrases, though this model is theoretically limited in that it only represents short phrases or contiguous phrases (Tian, Okazaki, and Inui, 2016; Tian, Okazaki, and Inui, 2017). Thus, it is necessary to establish a precise method for calculating the lexical similarity between various phrases, including long phrases and non-contiguous phrases.

Expansion of language coverage

While the supported language of my hybrid approach is English, the original inference system ccg2lambda also supports Japanese (Mineshima et al., 2016). Thus, my hybrid approach can be enhanced to support Japanese without difficulty. Also, the semantic representation is theoretically independent of languages and another direction of the future work is to expand the list of supported languages of my approach.

Improvement of a hybrid approach using neural network

In my hybrid approach, I manually implemented functions for extracting features from proof processes. However, with the recent development of neural network-based approaches, feature learning can be used for extracting features from proof processes. Thus, another direction of the future work is to improve feature extraction from proof processes using neural network.

Modification of a hybrid approach for other NLP applications

On the other hand, compared with neural network-based approaches, there are two advantages of my hybrid approach: interpretability and customizability. Regarding interpretability, my approach not only can assess the relevancy of sentences but also explain the sources of similarity/entailment by referring to information on sub-goals in the proof. Regarding customizability, we can

easily customize each component of my hybrid approach (i.e., semantic parsing, inference and knowledge injection) for many NLP applications; consider applying my approach to a domain-specific sentence. We can reflect domain characteristics into logical semantic representations by modifying semantic templates. In addition, my axiom injection mechanism is available regardless of kinds of lexical knowledge databases, and thus it is capable of injecting domain knowledge as axioms in logical inference. While machine learning-based approaches generally require a large annotated dataset, my approach can do these customizations regardless of the size of a dataset.

Given these two capabilities of interpretation and customization of my approach, another direction of the future work is to refine my method so that it can be used in other NLP applications such as question answering, information retrieval and text summarization. My approach potentially enables calculation of the semantic relations among multiple sentences, which may benefit these applications. To apply my approach to question answering, we have to consider how to map interrogative sentences to logical formulas. Interrogative sentences can be also analyzed by CCG parsers and thus my approach can be applicable to question answering systems by modifying semantic templates. Regarding information retrieval and text summarization, my approach might be applied by adding some functions for splitting long sentences into short sentences or ranking the importance of sentences in the target text.

Acknowledgements

During work on my thesis, there were many people who helped and supported me, both inside and outside my academic life. Here I would like to thank them. My deepest appreciation goes to my supervisors, Prof. Seiichi Koshizuka and Prof. Daisuke Bekki. They formed a great advising team, and I am grateful for their continuous intellectual and personal support. Their joint support was truly vital for my research.

I would like to thank the members of my advising committee, Prof. Kazuhiro Aoyama, Prof. Kiyoshi Izumi, and Prof. Taro Kanno. This thesis benefitted from our discussions and their thoughtful comments. I also would like to thank Prof. Koji Mineshima and researcher Pascual Martínez-Gómez. Without their persistent help, this thesis would not have been possible. I also thank all the members of Bekki Laboratory and Koshizuka-Shibata Laboratory for their great friendship and immense assistance.

I appreciated the educational and financial support from JST CREST Grant Number JPMJCR1301 and AIP Challenge Program, Japan. All the fantastic experiences that I had would not have been possible without this support. I also would like to offer my special thanks to Prof. Sadao Kurohashi and Prof. Kentaro Inui. I have had the strong support and encouragement from them.

Special thanks to researcher Masashi Yoshikawa, who gave me insightful comments in my thesis. I also thank to researcher Kimitaka Asatani, who gave me constructive advices and warm encouragement. I would like to thank Prof. Yukio Ohsawa. Lastly, I deeply appreciate my family for encouraging me to step onto a brand new path to success.

Bibliography

Abzianidze, Lasha (2015). "A Tableau Prover for Natural Logic and Language". In: *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pp. 2492–2502.

— (2016). "Natural Solution to FraCaS Entailment Problems". In: *Proceedings of the 5th Joint Conference on Lexical and Computational Semantics*, pp. 64–74.

Agirre, Eneko et al. (2012). "SemEval-2012 Task 6: A Pilot on Semantic Textual Similarity". In: *Proceedings of the 6th International Workshop on Semantic Evaluation (SemEval-2012)*, pp. 385–393.

Agirre, Eneko et al. (2013). "*SEM 2013 shared task: Semantic Textual Similarity". In: *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 1: Proceedings of the Main Conference and the Shared Task: Semantic Textual Similarity*, pp. 32–43.

Arase, Yuki and Jun'ichi Tsujii (2017). "Monolingual Phrase Alignment on Parse Forests". In: *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pp. 1–11.

Bannard, Colin and Chris Callison-Burch (2005). "Paraphrasing with Bilingual Parallel Corpora". In: *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pp. 597–604.

Bär, Daniel et al. (2012). "UKP: Computing Semantic Textual Similarity by Combining Multiple Content Similarity Measures". In: *Proceedings of the 6th International Workshop on Semantic Evaluation (SemEval-2012)*, pp. 435–440.

Bekki, Daisuke and Koji Mineshima (2017). "Context-Passing and Underspecification in Dependent Type Semantics". In: *Modern Perspectives in Type Theoretical Semantics*. Ed. by Stergios Chatzikyriakidis and Zhaohui Luo. Studies of Linguistics and Philosophy. Springer, pp. 11–41.

Beltagy, Islam, Katrin Erk, and Raymond Mooney (2014). "Probabilistic Soft Logic for Semantic Textual Similarity". In: *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, pp. 1210–1219.

Beltagy, Islam et al. (2013). "Montague Meets Markov: Deep Semantics with Probabilistic Logical Form". In: pp. 11–21.

Beltagy, Islam et al. (2014). "UTexas: Natural Language Semantics using Distributional Semantics and Probabilistic Logic". In: *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval-2014)*, pp. 796–801.

Beltagy, Islam et al. (2016). "Representing Meaning with a Combination of Logical and Distributional Models". In: *Computational Linguistics* 42.4, pp. 763–808.

Bentivogli, Luisa et al. (2009). "The Fifth PASCAL Recognizing Textual Entailment Challenge". In: *Proceedings of the Second Text Analysis Conference, TAC*.

Bentivogli, Luisa et al. (2010). "The Sixth PASCAL Recognizing Textual Entailment Challenge". In: *Proceedings of the Third Text Analysis Conference, TAC*.

— (2011). "The Seventh PASCAL Recognizing Textual Entailment Challenge". In: *Proceedings of the Fourth Text Analysis Conference, TAC*.

Bertot, Yves and Pierre Castran (2010). *Interactive Theorem Proving and Program Development: Coq'Art The Calculus of Inductive Constructions*. New York, USA: Springer.

Bjerva, Johannes et al. (2014). "The Meaning Factory: Formal Semantics for Recognizing Textual Entailment and Determining Semantic Similarity". In: *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval-2014)*, pp. 642–646.

Blackburn, Patrick, Maarten de Rijke, and Yde Venema (2001). *Modal Logic*. New York, NY, USA: Cambridge University Press.

Blei, David M., Andrew Y. Ng, and Michael I. Jordan (2003). "Latent Dirichlet Allocation". In: *Journal of Machine Learning* 3, pp. 993–1022.

Bos, Johan (2008). "Wide-Coverage Semantic Analysis with Boxer". In: *Semantics in Text Processing. STEP 2008 Conference Proceedings*. Ed. by Johan Bos and Rodolfo Delmonte. Vol. 1. Research in Computational Semantics. College Publications, pp. 277–286.

— (2009). "Applying automated deduction to natural language understanding". In: *Journal of Applied Logic* 7.1, pp. 100 –112.

Bos, Johan and Katja Markert (2005). "Recognising Textual Entailment with Logical Inference". In: *Proceedings of the Conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pp. 628–635.

Bowman, Samuel R. et al. (2015). "A large annotated corpus for learning natural language inference". In: *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pp. 632–642.

Breiman, Leo (2001). "Random Forests". In: *Machine Learning* 45, pp. 5–32.

Broecheler, Matthias, Lilyana Mihalkova, and Lise Getoor (2010). "Probabilistic Similarity Logic". In: *Uncertainty in Artificial Intelligence*.

Butler, Alastair (2010). "Current Research in the Semantics/Pragmatics Interface, Vol.23". In: Bingley: Emerald. Chap. The Semantics of Grammatical Dependencies.

Butler, Alastair and Kei Yoshimoto (2012). "Banking Meaning Representations from Treebanks". In: *Linguistic Issues in Language Technology* 7, pp. 1–22.

Carpenter, Bob (1998). *Type-logical Semantics*. Cambridge, MA, USA: MIT Press.

Cer, Daniel et al. (2017). "SemEval-2017 Task 1: Semantic Textual Similarity Multilingual and Crosslingual Focused Evaluation". In: *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pp. 1–14.

Champollion, Lucas (2015). "The interaction of compositional semantics and event semantics". In: *Linguistics and Philosophy* 38.1, pp. 31–66.

Chen, Danqi and Christopher D. Manning (2014). "A Fast and Accurate Dependency Parser using Neural Networks." In: *Proceedings of the 52th Annual Meeting on Association for Computational Linguistics*, pp. 740–750.

Chklovski, Timothy and Patrick Pantel (2004). "VerbOcean: Mining the Web for Fine-Grained Semantic Verb Relations". In: *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pp. 33–40.

Clark, Stephen and James R. Curran (2007). "Wide-coverage efficient statistical parsing with CCG and log-linear models". In: *Computational Linguistics* 33.4, pp. 493–552.

Clark, Stephen, Julia Hockenmaier, and Mark Steedman (2002). "Building Deep Dependency Structures with a Wide-coverage CCG Parser". In: *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, pp. 327–334.

Clinchant, Stéphane, Cyril Goutte, and Eric Gaussier (2006). "Lexical Entailment for Information Retrieval". In: *Advances in Information Retrieval*. Ed. by Mounia Lalmas et al.

Cooper, Robin et al. (1994). "FraCaS–A Framework for Computational Semantics". In: *Deliverable* D6.

Curran, James R and Stephen Clark (2003). "Investigating GIS and smoothing for maximum entropy taggers". In: *Proceedings of the tenth conference on European chapter of the Association for Computational Linguistics-Volume 1*, pp. 91–98.

Dagan, Ido and Bernardo Glickman Orenand Magnini (2006). "The PASCAL Recognising Textual Entailment Challenge". In: *Machine Learning Challenges. Evaluating Predictive Uncertainty, Visual Object Classification, and Recognising Tectual Entailment*, pp. 177–190.

Dagan, Ido, Oren Glickman, and Bernardo Magnini (2006). "The PASCAL Recognising Textual Entailment Challenge". In: *Proceedings of the First International Conference on Machine Learning Challenges: Evaluating Predictive Uncertainty Visual Object Classification, and Recognizing Textual Entailment*, pp. 177–190.

Dagan, Ido et al. (2013). *Recognizing Textual Entailment: Models and Applications*. Synthesis Lectures on Human Language Technologies. Morgan & Claypool Publishers.

Davidson, Donald (1967). "The Logical Form of Action Sentences". In: *The Logic of Decision and Action*. Ed. by Nicholas Rescher. Reprinted in **Davidson80** pages 105–148. Pittsburgh, PA: University of Pittsburgh Press, pp. 81–95.

Deerwester, Scott et al. (1990). "Indexing by latent semantic analysis". In: *Journal of the American Society for Information Science* 41.6, pp. 391–407.

Delahaye, David (2000). "A Tactic Language for the System Coq". In: *Logic for Programming and Automated Reasoning*, pp. 85–95.

Dong, Yubing, Ran Tian, and Yusuke Miyao (2014). "Encoding Generalized Quantifiers in Dependency-based Compositional Semantics". In: *Proceedings of the 28th Pacific Asia Conference on Language, Information, and Computation*, pp. 585–594.

Dumais, Susan (1997). "A Solution to Plato's Problem: The Latent Semantic Analysis Theory of Acquisition, Induction and Representation of Knowledge". In: 104(2), pp. 211–240.

Edmonds, Philip and Graeme Hirst (2002). "Near-synonymy and Lexical Choice". In: *Computational Linguistics* 28.2, pp. 105–144.

Emmert-Streib, Frank, Matthias Dehmer, and Yongtang Shi (2016). "Fifty years of graph matching, network alignment and network comparison". In: *Information Sciences* 346-347.Supplement C, pp. 180–197.

Ganitkevitch, Juri, Benjamin Van Durme, and Chris Callison-Burch (2013). "PPDB: The Paraphrase Database". In: *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologie*, pp. 758–764.

Garrette, Dan and Ewan Klein (2009). "An Extensible Toolkit for Computational Semantics". In: *Proceedings of the Eighth International Conference on Computational Semantics*, pp. 116–127.

Giampiccolo, Danilo et al. (2007). "The Third PASCAL Recognizing Textual Entailment Challenge". In: *Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing*, pp. 1–9.

Giampiccolo, Danilo et al. (2008). "The Fourth PASCAL Recognizing Textual Entailment Challenge". In: *Proceedings of the First Text Analysis Conference, TAC*.

Grefenstette, Edward (2013). "Towards a Formal Distributional Semantics: Simulating Logical Calculi with Tensors". In: *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 1: Proceedings of the Main Conference and the Shared Task: Semantic Textual Similarity*, pp. 1–10.

Grefenstette, Edward and Mehrnoosh Sadrzadeh (2011). "Experimental Support for a Categorical Compositional Distributional Model of Meaning". In: *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pp. 1394–1404.

Harabagiu, Sanda and Andrew Hickl (2006). "Methods for Using Textual Entailment in Open-domain Question Answering". In: *Proceedings of the 21st International Conference on Computational Linguistics and the 44th Annual Meeting of the Association for Computational Linguistics*, pp. 905–912.

Harris, Zellig (1954). "Distributional structure". In: 10.23, pp. 146–162.

He, Hua and Jimmy Lin (2016). "Pairwise Word Interaction Modeling with Deep Neural Networks for Semantic Similarity Measurement". In: *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 937–948.

Hill, Felix, Kyunghyun Cho, and Anna Korhonen (2016). "Learning Distributed Representations of Sentences from Unlabelled Data". In: *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 1367–1377.

Hockenmaier, Julia and Mark Steedman (2007). "CCGbank: A corpus of CCG derivations and dependency structures extracted from the penn treebank". In: *Computational Linguistics* 33.3, pp. 355–396.

Huet, Gérard Pierre (1975). "A Unification Algorithm for Typed lambda-Calculus". In: *Theoretical Computer Science* 1, pp. 27–57.

Johnson, Mark (1998). "PCFG Models of Linguistic Tree Representations". In: *Computational. Linguistics* 24.4, pp. 613–632.

Jones, Bevan Keeley (2016). "Learning words and syntactic cues in highly ambiguous contexts". PhD thesis. The University of Edinburgh.

Joshi, Aravind K. and Yves Schabes (1997). "Handbook of Formal Languages, Vol. 3". In: ed. by Grzegorz Rozenberg and Arto Salomaa. New York, NY, USA: Springer-Verlag New York, Inc. Chap. Tree-adjoining Grammars, pp. 69–123.

Jurafsky, Daniel and James H. Martin (2009). *Speech and Language Processing (2Nd Edition)*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc.

Kamp, Hans and Uwe Reyle (1993). *From Discourse to Logic. Introduction to Modeltheoretic Semantics of Natural Language, Formal Logic and Discourse Representation Theory*. Dordrecht: Kluwer.

Kaplan, Ronald M. and Joan Bresnan (1995). *Lexical-Functional Grammar: A Formal System for Grammatical Representation*.

Kartsaklis, Dimitri, Nal Kalchbrenner, and Mehrnoosh Sadrzadeh (2014). "Resolving Lexical Ambiguity in Tensor Regression Models of Meaning". In: *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, pp. 212–217.

Kartsaklis, Dimitri and Mehrnoosh Sadrzadeh (2016). "Distributional Inclusion Hypothesis for Tensor-based Composition". In: *Proceedings of the 26th International Conference on Computational Linguistics: Technical Papers*, pp. 2849–2860.

Kenter, Tom and Maarten de Rijke (2015). "Short Text Similarity with Word Embeddings". In: *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*, pp. 1411–1420.

Lai, Alice, Yonatan Bisk, and Julia Hockenmaier (2017). "Natural Language Inference from Multiple Premises". In: *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pp. 100–109.

Lai, Alice and Julia Hockenmaier (2014). "Illinois-LH: A Denotational and Distributional Approach to Semantics". In: *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pp. 329–334.

Lakoff, George (1970). "Linguistics and natural logic". In: *Synthese* 22.1, pp. 151–271.

Le, Quoc V. and Tomas Mikolov (2014). "Distributed Representations of Sentences and Documents". In: *Proceedings of the 31th International Conference on Machine Learning*, pp. 1188–1196.

Lewis, Mike and Mark Steedman (2013). "Combined Distributional and Logical Semantics." In: *Transactions of the Association for Computational Linguistics* 1, pp. 179–192.

— (2014). "A* CCG Parsing with a Supertag-factored Model". In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pp. 990–1000.

Liang, Percy, Michael I Jordan, and Dan Klein (2011). "Learning dependency-based compositional semantics". In: *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pp. 590–599.

Liu, Fei et al. (2015). "Toward Abstractive Summarization Using Semantic Representations." In: *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 1077–1086.

MacCartney, Bill (2009). *Natural language inference*. Ph.D. thesis.

MacCartney, Bill and Christopher D. Manning (2007). "Natural Logic for Textual Inference". In: *Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing*, pp. 193–200.

Marcus, Mitchell P., Mary Ann Marcinkiewicz, and Beatrice Santorini (1993). "Building a Large Annotated Corpus of English: The Penn Treebank". In: *Computational. Linguistics* 19.2, pp. 313–330.

Marelli, Marco et al. (2014). "A SICK Cure for The Evaluation of Compositional Distributional Semantic Models". In: *Proceedings of the 9th International Conference on Language Resources and Evaluation*, pp. 216–223.

Martínez-Gómez, Pascual and Yusuke Miyao (2016). "Rule Extraction for Tree-to-Tree Transducers by Cost Minimization". In: *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pp. 12–22.

Martínez-Gómez, Pascual et al. (2016). "ccg2lambda: A Compositional Semantics System". In: *Proceedings of the 2016 System Demonstrations of the Association for Computational Linguistics*, pp. 85–90.

— (2017). "On-demand Injection of Lexical Knowledge for Recognising Textual Entailment". In: *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics*, pp. 710–720.

Mikolov, Tomas et al. (2013). "Distributed Representations of Words and Phrases and their Compositionality". In: *Advances in Neural Information Processing Systems 26*. Ed. by C. J. C. Burges et al. Curran Associates, Inc., pp. 3111–3119.

Miller, George A. (1995). "WordNet: A Lexical Database for English". In: *Communications of the ACM* 38.11, pp. 39–41.

Mineshima, Koji et al. (2015). "Higher-order logical inference with compositional semantics". In: *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pp. 2055–2061.

Mineshima, Koji et al. (2016). "Building compositional semantics and higher-order inference system for a wide-coverage Japanese CCG parser." In: *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pp. 2236–2242.

Mitchell, Jeff and Mirella Lapata (2008). "Vector-based Models of Semantic Composition". In: *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics*, pp. 236–244.

— (2010). "Composition in distributional models of semantics". In: *Cognitive Science* 34.8, pp. 1388–1429.

Mogren, Olof, Mikael Kågebäck, and Devdatt Dubhashi (2015). "Extractive Summarization by Aggregating Multiple Similarities". In: *Proceedings of the International Conference Recent Advances in Natural Language Processing*, pp. 451–457.

Montague, Richard (1973). "The Proper Treatment of Quantification in Ordinary English". In: *Approaches to Natural Language*. Ed. by K. J. J. Hintikka, J. Moravcsic, and P. Suppes. Dordrecht: Reidel, pp. 221–242.

Moot, Richard (2010). "Wide-Coverage French Syntax and Semantics using Grail". In: *TALN 2010*.

Mueller, Jonas and Aditya Thyagarajan (2016). "Siamese Recurrent Architectures for Learning Sentence Similarity". In: *Proceedings of the 30th AAAI Conference on Artificial Intelligence*, pp. 2786–2792.

Parsons, Terence (1990). *Events in The Semantics of English: a Study in Subatomic Semantics*. Cambridge, USA: MIT Press.

Pelletier, Francis Jeffry (1999). "A Brief History of Natural Deduction". In: *History and Philosophy of Logic* 20, pp. 1–31.

Pelletier, Francis Jeffry and Allen P. Hazen (2012). "A History of Natural Deduction". In: *Logic: A History of its Central Concepts*. Ed. by Dov M. Gabbay, Francis Jeffry Pelletier, and John Woods. Vol. 11. Handbook of the History of Logic. North-Holland, pp. 341–414.

Peter, W. Culicover and Wendy K. Wilkins (1984). *Locality in linguistic theory*. Florida, USA: Academic Press.

Polajnar, Tamara, Laura Rimell, and Stephen Clark (2015). "An Exploration of Discourse-Based Sentence Spaces for Compositional Distributional Semantics". In: *Proceedings of the 1st Workshop on Linking Computational Models of Lexical, Sentential and Discourse-level Semantics*, pp. 1–11.

Pollard, Carl and Ivan A. Sag (1994). *Head-Driven Phrase Structure Grammar*. Chicago: The University of Chicago Press.

Prawitz, Dag (1965). *Natural Deduction – A Proof-Theoretical Study*. Stockholm, Sweden: Almqvist & Wiksell.

Rashtchian, Cyrus et al. (2010). "Collecting Image Annotations Using Amazon's Mechanical Turk". In: *Proceedings of the NAACL HLT 2010 Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk*, pp. 139–147.

Ray, Jackendo (1990). *Semantic Structures*. Cambridge, USA: The MIT Press.

Reddy, Siva, Mirella Lapata, and Mark Steedman (2014). "Large-scale Semantic Parsing without Question-Answer Pairs". In: *Transactions of the Association for Computational Linguistics* 2, pp. 377–392.

Reddy, Siva et al. (2016a). "Transforming Dependency Structures to Logical Forms for Semantic Parsing". In: *Transactions of the Association of Computational Linguistics* 4, pp. 127–141.

— (2016b). "Transforming Dependency Structures to Logical Forms for Semantic Parsing". In: *Transactions of the Association for Computational Linguistics* 4, pp. 127–140.

Richardson, Matthew and Pedro Domingos (2006). "Markov Logic Networks". In: *Machine Learning* 62, pp. 107–136.

Robinson, John A. (1965). "A Machine-Oriented Logic Based on the Resolution Principle". In: *Journal of the ACM* 12.1, pp. 23–41.

Sacaleanu, Bogdan et al. (2008). "Entailment-based Question Answering for Structured Data". In: *Coling 2008: Companion volume: Demonstrations*, pp. 173–176.

Socher, Richard et al. (2011). "Dynamic Pooling and Unfolding Recursive Autoencoders for Paraphrase Detection". In: *Advances in Neural Information Processing Systems 24*. Curran Associates, Inc., pp. 801–809.

Steedman, Mark (2000). *The Syntactic Process*. Cambridge, USA: MIT Press.

— (2012). *Taking Scope: The Natural Semantics of Quantifiers*. Cambridge, USA: MIT Press.

Tai, Kai Sheng, Richard Socher, and Christopher D. Manning (2015). "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)*, pp. 1556–1566.

Tian, Ran, Yusuke Miyao, and Takuya Matsuzaki (2014). "Logical Inference on Dependency-based Compositional Semantics". In: *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 79–89.

Tian, Ran, Naoaki Okazaki, and Kentaro Inui (2016). "Learning Semantically and Additively Compositional Distributional Representations". In: *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, pp. 1277–1287.

— (2017). "The mechanism of additive composition". In: *Machine Learning* 106.7, pp. 1083–1130.

Troelstra, Anne and Helmut Schwichtenberg (2000). *Basic Proof Theory*. Cambridge University Press.

Tsuchiya, Masatoshi (2018). "Performance Impact Caused by Hidden Bias of Training Data for Recognizing Textual Entailment". In: *Proceedings of the 11th International Conference on Language Resources and Evaluation*.

Vo, Ngoc Phuoc An, Simone Magnolini, and Octavian Popescu (2015). "FBK-HLT: An Application of Semantic Textual Similarity for Answer Selection in Community Question Answering". In: *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pp. 231–235.

Williams, Adina, Nikita Nangia, and Samuel R. Bowman (2017). "A Broad-Coverage Challenge Corpus for Sentence Understanding through Inference". In: *CoRR* abs/1704.05426. URL: http://arxiv.org/abs/1704.05426.

Wong, S. K. M. and Vijay V. Raghavan (1984). "Vector Space Model of Information Retrieval: A Reevaluation". In: *Proceedings of the 7th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 167–185.

Yanaka, Hitomi et al. (2017). "Determining Semantic Textual Similarity using Natural Deduction Proofs". In: *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pp. 692–702.

— (2018a). "Acquisition of Phrase Correspondences using Natural Deduction Proof". In: *Proceedings of The 16th Annual Conference of North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

— (2018b). "Learning Semantic Textual Relatedness using a Natural Deduction Proofs". In: *Journal of Natural Language Processing* 25.3.

Yin, Wenpeng and Hinrich Schütze (2017). "Task-Specific Attentive Pooling of Phrase Alignments Contributes to Sentence Matching". In: *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pp. 699–709.

Yoshikawa, Masashi, Hiroshi Noji, and Yuji Matsumoto (2017). "A* CCG Parsing with a Supertag and Dependency Factored Model". In: *Proceedings of the 55nd Annual Meeting of the Association for Computational Linguistics*, pp. 277–287.

Zhao, Jiang, Tiantian Zhu, and Man Lan (2014). "ECNU: One Stone Two Birds: Ensemble of Heterogenous Measures for Semantic Relatedness and Textual Entailment". In: *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval-2014)*, pp. 271–277.

Zhao, Shiqi et al. (2008). "Combining Multiple Resources to Improve SMT-based Paraphrasing Model". In: *Proceedings of the 46rd Annual Meeting on Association for Computational Linguistics*, pp. 1021–1029.

Zhou, Yao, Cong Liu, and Yan Pan (2016). "Modelling Sentence Pairs with Tree-structured Attentive Encoder". In: *Proceedings of the 26th International Conference on Computational Linguistics*, pp. 2912–2922.