Master Thesis

Automatic Induction of Background Knowledge Axioms for Recognising Textual Entailment

Annisa Ihsani
2218577

Research Master Linguistics
Erasmus Mundus European Masters Program in Language and Communication Technologies (LCT)
Faculty of Arts
University of Groningen

July 22, 2012

Supervisor: Johan Bos

Automatic Induction of Background Knowledge Axioms for Recognising Textual Entailment

Annisa Ihsani

M.Sc. Dissertation

Department of Intelligent Computer Systems
Faculty of Information and Communication Technology
University of Malta
July 22, 2012

Supervisor: Michael Rosner

Submitted in partial fulfillment of the requirements for the Degree of European Master of Science in Human Language Science and Technology (HLST)

M.SC.(HLST)
FACULTY OF INFORMATION AND COMMUNICATION TECHNOLOGY
UNIVERSITY OF MALTA

Declaration

Plagiarism is defined as "the unacknowledged use, as one's own work, of work of another person, whether or not such work has been published, and as may be further elaborated in Faculty or University guidelines". (University Assessment Regulations, 2009, Regulation 39 (b)(i), University of Malta).

I, the undersigned, declare that the Master's dissertation submitted is my own work, except where acknowledged and referenced.

I understand that the penalties for making a false declaration may include, but are not limited to, loss of marks; cancellation of examination results; enforced suspension of studies; or expulsion from the degree programme.

Annisa Ihsani
Student Name

Signature

CSA5310
HLST Dissertation
Course Code

Automatic Induction of Knowledge Axioms for Recognising Textual Entailment
Title of work submitted

Date: July 22, 2012

Abstract

One of the challenges in recognising textual entailment (RTE) with logical inference is the lack of appropriate background knowledge. In this thesis we attempt to make explicit general knowledge by automatically extracting them from RTE Challenge corpora. Our approach departs from deep semantic analysis using Discourse Representation Structure, resulting in a background knowledge axiom that expresses a paraphrase of relationships between two entities X and Y in the form of an implication such as X *is the wife of* $Y \Rightarrow X$ *is married to* Y. Evaluation on an existing RTE system shows that these axioms provide useful knowledge which helps the inference engines in predicting textual entailment and improves the result of correct predictions made by the system.

Acknowledgments

I owe many debts of gratitude to my supervisor Prof. Johan Bos for his guidance and support in supervising me to finish this thesis and also for writing, with Patrick Blackburn, the book that changed my academic career towards NLP. I would also like to thank my second supervisor in Malta, Dr. Mike Rosner. This thesis would not have been possible without their supervisions.

I am immensely grateful to the LCT consortium and EU for making this study possible for me. Thanks for all support and useful discussions to my friends Milos, Mariya, Dima, Jelke, and Lili. I especially thank my fiance for his suggestions and encouragement during this thesis writing.

Finally, my thanks go to my parents. Without them I would not have made it this far.

Contents

Chapter 1

Introduction

1.1 Motivation

One of the characteristics of human communications is the use of different linguistic expressions when communicating a meaning. Many NLP tasks are dedicated to address this phenomenon; that is, deciding whether a collection of text documents semantically overlap or lead to the same meaning. Providing a general framework that captures these tasks is the Recognising Textual Entailment (RTE) Challenge proposed by the PASCAL Network (Dagan et al., 2006). Given a pair containing a text fragment T and a hypothesis H, the challenge is defined as deciding whether T entails H. That is, T is said to entail H if H reasonably follows or can be inferred from T. The following examples taken from the test set of the First RTE Challenge illustrate the given pairs and the case of positive and negative entailment, respectively.

RTE-1 Test (1640)

T:	Today's highlight in history: On Nov. 24, 1963, in a scene captured on live network television, Dallas nightclub owner Jack Ruby shot and mortally wounded Lee Harvey Oswald as the accused assassin of President John F. Kennedy was being escorted by police to an armored truck at the Dallas municipal building for transfer to the county jail.
H:	Jack Ruby killed Lee Harvey Oswald.

RTE-1 Test (1820)

T:	The recent G8 summit, which was first held on July 13-16, 1975, took place on Sea Island on June 8-10.
H:	The recent G8 summit took place on July 13-16, 1975.

Since its first appearance in 2005, many systems have participated in the RTE Challenge. One of the approaches includes the use of logical inference to show entailment between T and H (Tatu and Moldovan, 2006; Bos and Markert, 2005). A problem with this approach is that the system often suffers from the lack of appropriate background knowledge needed to draw inferences. In the previous RTE pair of positive entailment, for example, we would expect the action of shooting and mortally wounding to somehow correspond with killing. While humans take general knowledge for granted and often omit them from the texts, this knowledge must be made explicit to the RTE systems. Several participating systems employ available resources such as WordNet (Fellbaum, 2010) to provide lexical knowledge. This task, however, requires a more extensive coverage of background knowledge. For this reason, we would like to come up with a technique to automatically generate background knowledge rules needed in recognising textual entailment.

1.2 Problem

An important feature in recognising textual entailment is the frequency of word overlap between the text and the hypothesis. The idea is that pairs that have a lot of word overlap, in which T and H are more similar, are more likely to indicate positive entailment than those that do not. However, there is sometimes a gap between the phrase in T and that in H, e.g. in T we might have *X is the wife of Y*, while in H this sentence is paraphrased into *X is married to Y*. Another case would be the information embedded in noun compounds such as *Greenpeace founder Don White*, which points out to the proposition *Don White is the founder of Greenpeace*.

We believe that the dataset of the RTE Challenge contains knowledge that can be extracted in order to fill this gap. For this reason, we propose a method of automatically inducing background knowledge rules from the available resources. We will use labelled data from RTE Challenge corpora, therefore our approach will be supervised. The background knowledge rules shall be referred to as *axioms* throughout the rest of the discussion; this is slightly different from the term in logic, where an axiom is defined as a self-evident truth or statement that is universally accepted as true. Here we use the term axiom in a weaker sense, i.e. as a rule that states plausible knowledge.

Since semantic entailment itself is defined as a truth relation such that T is said to entail H if and only if whenever T is true then H is also true (Saeed, 2003), our axioms will be modeled in the form of logical implication. To generate these axioms, we pick from

T and H a pair of overlapping words, say X and Y, and observe how they are related to each other. The pattern of our axioms will be of the following implication:

$$X\ path_t\ Y \Rightarrow X\ path_h\ Y;$$

where $path_t$ and $path_h$ represent the meaning relation between X and Y in T and H, respectively. From the previous example, an instance of these paths would be *is the wife of* and *is married to*. Our work is then driven by the following research question: how can we develop a method to automatically obtain background knowledge axioms in logical form from labelled data?

1.3 Objective

The objective of this thesis is to provide a technique to automatically generate background knowledge axioms from text-hypothesis pairs. It especially aims at paraphrasing and recovering information between two entities which might help in recognising textual entailment. To see whether these resulting axioms actually provide useful background knowledge in recognising textual entailment, we evaluate the axioms using an existing RTE system called Nutcracker[1].

1.4 Structure of the Thesis

The rest of this thesis is organised as follows. In Chapter 2, we discuss various works related to knowledge extraction from text documents. Chapter 3 gives an overview of the approach we are going to implement. Chapter 4 explains the theoretical background of our work, including the semantic representation using Discourse Representation Theory, approach to RTE with logical inference, and path finding algorithm called Breadth First Search. In Chapter 5 we give a step-by-step description of our approach in automatic induction of background knowledge axioms, starting from the raw text-hypothesis pair until the candidate axioms are produced. We conduct the experiments for evaluating the axioms using Nutcracker RTE system and report the results in Chapter 6. Finally, we end the discussion in Chapter 7 where we talk about conclusion and further work.

[1] `http://svn.ask.it.usyd.edu.au/trac/candc/wiki/nutcracker`

Chapter 2

Related Work

Recent research works have shown that there is a growing interest in the area of automatic extraction of background knowledge. In this section we will discuss some relevant works on this area.

2.1 DIRT

Lin and Pantel (2001) described an unsupervised algorithm, called DIRT, for discovering inference rules for question answering. An inference rule expresses the relationship between two entities, which can be a paraphrase such as *X is the author of Y* $\approx$ *X wrote Y*, or other relationships that have closely related meaning like *X caused Y* $\approx$ *Y is blamed on X*. This work is based on the Distributional Hypothesis, which states that words that occurred in the same contexts tend to have similar meanings. Instead of words, they adopted this hypothesis to apply on paths in dependency trees, which are obtained from parsing the sentences from text corpora. The parsing was done using Minipar parser, which constructed all possible parse trees for a given sentence and output the dependency tree with the highest ranking. A path was formed from a link between two words in a dependency tree, thus representing a binary semantic relationship between two content words. For example, in the following sentence:

(2.1) John found a solution to the problem

the path between *John* and *problem* is represented by:
N:subj:V ← find → V:obj:N → solution → N:to:N.

The words *John* and *problem*, called slot fillers, are not included in the path, so this path can be read as *X finds solution to Y*. Since the variables X and Y are to be instantiated by entities, the slot fillers must be of category nouns. The slot fillers act as the contexts for finding the similarity between paths, based on the underlying assumption that paths that tend to occur in the same contexts usually have similar meanings. The similarity between paths is measured based on common features they share; most similar paths lead to the discovery of inference rules. A few example of most similar paths for *X solves Y* are: *Y is solved by X*, *X resolves Y* and *X find a solution to Y*. Some incorrect paths are also found, e.g. *Y is blamed for X*, *X creates Y*, etc.

The inference rules were extracted 1 GB of newpaper text parsed by Minipar, resulting in 7 million parse trees. The evaluation was done by comparing the inference rules generated by DIRT and paraphrases produced by humans on the several questions from TREC-8 Question-Answering Track (Voorhees, 1999). The result showed that there was very little overlap between the manually-written paraphrases and the top-40 most similar paths generated by DIRT for each question, suggested that it was as difficult for humans to generate paraphrases as it was for machines. The DIRT output, however, allowed humans to identify correct inference rules, hence useful in adding knowledge which were missing from manually-generated set.

Inferential Selectional Preferences (ISP)

Underspecified variables in inference rules such as those produced by DIRT can lead to problematic instantiations. For example, the inference rule *X visits Y* ⇒ *X travels to Y* may be good for *John visits England* ⇒ *John travels to England*, but given "John visits Maria" it is not likely to conclude "John travels to Maria". In the work of Pantel et al. (2007), the problem is formulated as follows: given an inference rule $p_i \Rightarrow p_j$ and the instance $< x, p_i, y >$, the system's task is to determine whether $< x, p_j, y >$ is valid.

Their approach in filtering out such incorrect inference rules was based on selectional preferences (Resnik, 1993). They introduced *relational selectional preference* (RSPs) of a binary semantic relation p between entities X and Y as the semantic classes $C(x)$ and $C(y)$ of the words that can be instantiated by X and Y, respectively. For instance, given the relation *X buys Y*, the semantic classes $C(x)$ can be $\{individual, organisation, ...\}$ while the semantic classes $C(y)$ can be $\{food, clothes, ...\}$.

To calculate the relational selectional preferences of a path/relation p, two models were implemented:

- Joint Relational Model (JRM). This was done by first discovering every instance of $< x, p, y >$ in a corpus, thus obtaining the semantic classes $C(x)$ and $C(y)$. For $c(x) \in C(x)$ and $c(y) \in C(y)$ that occur together, every triple $< c(x), p, c(y) >$ is a candidate selectional preference for p. These candidates are then ranked based on their frequencies.
- Independent Relational Model (IRM). This method differs from JRM in that it computes the semantic classes of the arguments independently in order to address data sparseness. Instead of the triple $< c(x), p, c(y) >$, this method accumulates the triple $< c(x), p, * >$ and $< *, p, c(y) >$.

The selectional preferences of the inference rule $p_i \Rightarrow p_j$ are defined by computing the intersection between the RSPs of p_i and p_j. Two models were implemented for this purpose. The Joint Inferential Model (JIM) calculates the SPs of $p_i \Rightarrow p_j$ based on the SPs of p_i and p_j obtained from JRM. For example, the SPs of p_i "X visits Y" by JRM are $< Person, p_i, Location >$ and $< Person, p_i, Person >$, while the SPs of p_j "X travels to Y" are $< Person, p_j, Location >$. Therefore the SPs for *X visits Y* $\Rightarrow$ *X travels to Y* are the intersection of both sets, i.e. $< Person, Location >$. The second model, called Independent Inferential Model (IIM) is similar, but the SPs of p_i and p_i are those obtained from IRM instead of JRM.

Given the inference rule $p_i \Rightarrow p_j$ and the instance $< x, p_i, y >$ as input, the validity of $< x, p_j, y >$ is determined according to three different ISP algorithms (Pantel et al., 2007). These algorithms were evaluated on DIRT inference rules using semantic classes discovered from CBC clustering algorithm (Pantel and Lin, 2002) and WordNet. A comparison to three baseline systems (accept all, reject all, and randomly accept/reject inferences) shows that all the ISP algorithms perform better than the baseline systems, thus showing that learning SPs can be useful in filtering out incorrect inferences. It was shown that the system performed better when using semantic classes from CBC (highest accuracy 59%) than WordNet, since the former has wider lexical coverage.

Learning Directionality

DIRT algorithm produces symmetric inference rules in the form of *X* $path_i$ *Y* $\approx$ *X* $path_j$ *Y*, although asymmetric rules, such as *X* $path_i$ *Y* $\Rightarrow$ *X* $path_j$ *Y*, are often considered

more likely. Bhagat et al. (2007) presented an unsupervised approach to learning the directionality of inference rules (LEDIR) based on the so-called Directional Hypothesis, stating for two binary semantic relations that occur in similar contexts, the one occurs in more contexts (hence more general) is most likely to be implied by the other one. The similarity of the two paths were calculated based on their selectional preferences.

In this work, Bhagat et al. (2007) implemented the same two methods for calculating the selectional preferences of a path/relation p as used by Pantel et al. (2007), namely the Joint Relational Model (JRM) and the Independent Relational Model (IRM). The focus of LEDIR was on two tasks: 1) determining if an inference rule is plausible; 2) if so, learning the directionality of the inference rule. For the first part, an overlap coefficient between path p_i and p_j, denoted $sim(p_i, p_j)$, was calculated according to the selectional preferences of both paths. If $sim(p_i, p_j)$ satistfies an empirically determined threshold value α, then the rule is plausible, otherwise implausible. The directionality of a plausible inference rule was determined by comparing the ratio between the selectional preferences of p_i and p_j against another empirically determined threshold β.

Similar to that in Inferential Selectional Preferences (Pantel et al., 2007), the evalution was also conducted on DIRT inference rules using semantic classes from CBC clustering and WordNet. The output of LEDIR was to be compared against three baseline systems (each inference rule was tagged with random, most frequent, and bidirectional directionality assignment). The result showed that LEDIR outperformed all the three baselines in accuracy, with the highest accuracy (48%) achieved using IRM model and semantic classes from CBC. It was inspected that the system failed to filter out incorrect DIRT rules containing antonymy, such as *X loves Y* $\approx$ *X hates Y*. This is because both DIRT and LEDIR applied the distributional hypothesis and antonymous paths often occur in the same contexts (in the case of LEDIR, take the same set of semantic classes), hence antonymous rules were wrongly considered as plausible.

Applications

The inference rules produced by DIRT have been used in many NLP applications for providing background knowledge. One such work for the task of recognising textual entailment was conducted by Marsi et al. (2007). The purpose of their work was solely to investigate the usefulness of DIRT inference rules in RTE, hence no other sources of background knowledge were used.

The DIRT data was formed in clusters, where a cluster consists of a unique dependency

path, called source path, and a list of equivalent translations, called translation path. Substitution was performed by first matching the text part of an RTE pair with the source path. If there is such a match, the next step is to check if there is a corresponding translation path that matches with the hypothesis. If this is also successful, then the matching part of T is substituted with the translation path. Since DIRT rules were ordered according to likelihood, the substitution only concerned the most likely paths. Finally, entailment was predicted by aligning the substitued text with the hypothesis. If the proportion of the aligned words exceeds a certain threshold, this indicates positive entailment, otherwise negative entailment.

This method resulted in a total substitution of 108 positive entailment pairs and 75 negative entailment pairs from the development set of RTE-3[1]. Example rules involved in the substitutions are *X makes Y* $\approx$ *X sells Y* and *X win Y* $\approx$ *X is Y champion*. The substitution also included a few incorrect paraphrases such as *X feeds Y* $\approx$ *Y feeds X*. Not all of the substitutions were relevant with respect to entailment; moreover, substitutions on negative pairs were considered to be counterproductive.

Evaluation was conducted by comparing the performance of an existing RTE system (Marsi et al., 2006) without and with DIRT paraphrase rules. In comparison to the baseline system, i.e. without paraphrasing, the result showed that the experiments with DIRT inference rules contributed to a small improvement in accuracy. For the development set of RTE-3, the accuracy increased by more than 1%, while in the test set this improved by 0.5%.

Clark and Harrison (2008) applied DIRT inference rules in their RTE system BLUE. The system consists of an inference module, which employs background knowledge resources, i.e. WordNet and DIRT database, in determining entailment. The rules were written in implication form. If the antecedent subsumes (is more general than) the text part of an RTE pair, then the inference rule is applied, asserting the conclusion of the rule to that RTE pair.

Due to the massive size of DIRT database (12 million rules), around 1000 inference rules were triggered on a sentence. Some rules led to correct entailment prediction, such as *X stars as Y* $\Rightarrow$ *X portrays Y*, *X is sold to Y* $\Rightarrow$ *X is taken over by Y*, and so on. On the other hand, DIRT database also contains noise or bad rules that resulted in incorrect entailment prediction, e.g. *Y occurs in X* $\Rightarrow$ *something dies in X of Y*, *X is caused of Y* $\Rightarrow$ *Y is caused by X*, etc.

[1] `http://pascallin.ecs.soton.ac.uk/Challenges/RTE3`

The RTE system was evaluated on RTE-4 dataset in three different experiment settings. In predicting entailment using inference, the highest accuracy rate was 67%, but this was obtained in the run where the system only managed to make prediction for a small coverage of the test suite, i.e. 6.2%.

Another example of DIRT inference rules application in RTE was described in the work of Dinu and Wang (2009). Similar to Marsi et al. (2007), the focus of this work is to assess the application of DIRT inference rules in RTE, rather than to deal with the RTE task itself. The inference rules were first refined according to lexical relations provided by WordNet. Consider as an example the following inference rule:

(2.2) *X face threat of Y* $\approx$ *X at risk of Y*

An extension method was carried out by replacing a lexical item with its synonymous word, thus producing a number of new rules with different combinations of lexical resources. For example, the word *risk* is related to *danger* via WordNet synonymy, hence the new rule *X face the threat of Y* $\approx$ *X at danger of Y*. In addition, the antonym relation was also used in order to eliminate incorrect rules. As discussed previously, one of the drawbacks of DIRT algorithm is that it produced rules containing sentences with opposite meaning due to the high number of occurences in the same contexts. In this work, rules containing antonymous words with identical patterns were eliminated.

It was shown that the inference rules produced with the refinement method matched with more RTE pairs, that is, the coverage was around twice that of the original DIRT rules without refinement, with precision around 67%. In comparison to the baseline system (entailment predicted based on word overlap feature), experiments on RTE-2 and RTE-3 test suite with the help from inference rules resulted in a small improvement on precision (around 1%).

2.2 Rules Extraction from Text

Schubert (2002) attempted at deriving general knowledge from text corpora by extracting implicit propositions from sentences and making generalisations. For example, from the sentence:

(2.3) He entered the house through its open door

the following knowledge can be inferred: a male can enter a house, houses can have doors, doors can be open, etc. Departing from this idea, Clark and Harrison (2009) developed a system called DART. This system extracts knowledge from parse trees and stores them in a data structure called tuples. A tuple consists of a type symbol and arguments. An argument can be a root form of a word or another tuple. There are 12 predefined types of tuple symbols; some examples are as follows:

- (AN "small" "hotel"), meaning "hotels can be small"
- (NV "bus" "carry"), meaning "buses can carry (something/someone)"
- (VN "find" "spider"), meaning "spiders can be found"
- (QN "year" "contract"), meaning "contracts can be measured in years"
- (VPN "refer" "to" "business"), meaning "referring can be to businesses."

The DART databse consists of 23 million tuples extracted from two large text corpora. The plausibility of these tuples are scored with respect to their frequencies. One of the NLP tasks in which the DART tuples were evaluated is RTE. Particularly, the tuples were used to assess the plausibility of variable instantiations of DIRT inference rules applied in the task. In a way, this is the same problem as discussed in Inferential Selectional Preferences (Section 2.1). That is, certain inference rule instantions may lead to incorrect conclusion due to the lack of restrictions of what the variables can be. Consider for example a DIRT rule *X shoots Y* $\Rightarrow$ *X injures Y*. Say X and Y are instantiated with *Fred* and *gun*, respectively. The antecedent can be expressed in DART tuples (NV "person" "shoot") and (VN "shoot" "gun"), while the conclusion can be expressed in (NV "person" "shoot") and (VN "injure" "gun"). The low count of the last tuple in DART database indicates that it is not likely for guns to be injured, hence implausible instantion of the inference rule. It was shown that in comparison to DIRT original rules (average precision 59.2% on RTE-3 and 65% on RTE-4), the RTE system performed better using DIRT rules with DART confidence (average precision 64.1% on RTE-3 and 72.8% on RTE-4).

Gordon and Schubert (2011) described a technique for generating background knowledge as conditional expressions based on certain semantic patterns. Such patterns include *disconfirmed expectations* which indicate a mismatch between expectations and reality. Consider the sentence

(2.4) The ship weighed anchor and ran out her big guns, but did not fire a shot.

The word 'but' here suggests that the conclusion is the opposite of what the people expect. For this reason, the polarity of the conclusion is negated, resulting in the rule: *If a ship weighs anchor and runs out her big guns, then it may fire a shot.* Another semantic pattern that they observe is the use of 'but' in contrasting positive and negative words, such as "He is very clever but eccentric". Unlike the previous case, the polarity of the conclusion here must be reserved. In addition to these patterns, the pattern that describes *expected outcomes* are also used in generating entailment rules. For example, the sentence

(2.5) He stood before her in the doorway, evidently expecting to be invited in

yields the rule: *If a male stands before a female in the doorway, then he may expect to be invited in.* The evaluation of this work is done by human assessment involving a rule quality measure in the scale of 1 to 5 which reflects the level of agreement from high to low, respectively. Some of the rules that achieved high quality score (1) include *if a male looks around, then he may hope to see someone* and *if a pain is great, it may not be manageable.*

A method for discovering hyponymy lexical relation was presented by Hearst (1992). From large text corpora, certain semantic patterns were observed to extract hyponymy relation. Consider the folowing sentence:

(2.6) The bow lute, such as the Bambara ndang, is plucked and has an individual curved neck for each string.

Without knowing precisely what Bambara ndang or a bow lute is, we can infer from the sentence that Bambara ndang is a kind of bow lute, hence the hyponymy relation:

hyponym("Bambara ndang", "bow lute")

Hearst (1992) used several lexico-syntactic patterns to indicate the hyponymy relation:

1. *NP such as* {*NP* ,}* {*or* | *and*} *NP*, as exemplified in sentence (2.6)
2. *such NP as* {*NP* ,}* {*or* | *and*} *NP*, e.g. *works by such authors as Herrick, Goldsmith, and Shakespeare*
3. *NP* {, *NP*} *{,} *or other NP*, e.g. *bruises, wounds, broken bones, or other injuries*

4. *NP* {, *NP*} * {,} *and other NP*, e.g. *temples, treasuries, and other important civic buildings*

5. *NP* {,} *including* {*NP*,} * {*or* | *and*} *NP*, e.g. *all common-law countries, including Canada and England*

6. *NP* {,} *especially* {*NP*,} * {*or* | *and*} *NP*, e.g. *most European countries, especially France, England, and Spain.*

New patterns were learned by selecting a list of terms between which a lexical relation is known to hold, for instance *hyponymy(apple,fruit)*, and observing the commonalities of patterns where these terms occur in the corpus. Hyponymy pairs were extracted from a corpus of 8.6 million words, resulting in 152 hyponymy relations which were then compared to entries in WordNet hierarchy. From 106 relations in which both terms were registered in WordNet, 61 relations were found in both sets, including *hyponym(granite,rocks)*, *hyponym(scallop,bivalves)*, and *hyponym(nylon,fabrics)*. Some found relations were considered problematic due to underspecification or context dependency, e.g. *hyponym(steatornis,species)* and *hyponym(aircarft,target)*.

2.3 Paraphrases from Parallel Corpora

Barzilay and McKeown (2001) used a collection of multiple parallel English translations of novels as a source for extracting paraphrases. This is based on the idea that different translators might use different words to express the same meaning from the original source. As an example, they used sentences from two different translations of *Madame Bovary* as follows:

translation 1: *Emma burst into tears and he tried to comfort her, saying things to make her smile.*

translation 2: *Emma cried, and he tried to console her, adorning his words with puns.*

From these parallel sentences, paraphrases such as *(burst into tears,cried)* and *(comfort,console)* can be extracted. The research focused on two kinds of paraphrases:

- lexical or multi-word paraphrases
- morpho-syntactic rules, which express paraphrase patterns based on syntactic and

morphological structure of the extracted paraphrases. These rules are represented using parts-of-speech tags.

Eleven translations of five books were included in the corpus. Translated sentences that come from the same source sentence were first aligned using dynamic programming. From the aligned pair, they selected words that appear in both sentences as an example of positive paraphrasing. For instance, in the previous translations these would be *word1 = word2 = Emma* or *word1 = word2 = tried*. Context rules were created by looking for the context at certain length to the left and to the right of these words in the sentences. The paraphrases were obtained by matching these context rules in the sentence pairs. Examples of the resulting paraphrases are as follows:

- lexical paraphrases: *(countless, lots of)*, *(refuse, say no)*, *(sudden appearance, apparition)*
- morpho-syntactic paraphrases (indices indicate equal words):
  - (NN_0 POS NN_1) $\leftrightarrow$ (NN_1 IN DT NN_0), e.g. *King's son* $\leftrightarrow$ *son of the king*
  - (IN NN_0) $\leftrightarrow$ (VB_0), e.g. *in bottles* $\leftrightarrow$ *bottled*
  - (VB_0 RB_1) $\leftrightarrow$ (RB_1 VB_0), e.g. *suddenly came* $\leftrightarrow$ *came suddenly*

Similar work on extracting paraphrases was conducted by Ibrahim et al. (2003). Like Barzilay and McKeown (2001), the domain of paraphrase extraction was a corpus of parallel English translations. However, while the latter extracted the paraphrases by looking at a certain context length from a matching word between the translations, this method used syntactic structures as applied in DIRT so as to capture more variety of paraphrases, for example those with long distance relationships.

The method works by first aligning the translations in the corpus. To evaluate this alignment, gold standard was created by manually aligning 454 sentences of two different translations. The aligned corpus produced by the automatic alignment algorithm was compared to this gold standard, showing precision of 93% and recall of 88%.

After the alignment, the corpus was parsed to create the syntactic structures. Matching words between translations, or anchors, were extracted from these structures and scored according to certain heuristics, including exact string match, noun and matching pronouns, unique semantic classes, and so on. From each anchor pair, the shortest path between the two anchors was considered as a candidate paraphrase. The paths containing conjunction and punctuation were discarded. Two factors contribute to the increase

of the scores of these candidate paraphrases: the frequency of anchors with respect to a candidate paraphrase and the variety of different anchors from which the paraphrase was produced.

This method resulted in 5925 paraphrases with average length of 3.26 words excluding the anchors. For the evaluation, 130 unique paraphrases were chosen randomly and judged by three human judges, scoring in average precision of 41.2%. Some of the paraphrases produced by this method include:

- *X's Y* $\Rightarrow$ *Y of X* (paraphrase with highest score)
- *X rush over to Y* $\Rightarrow$ *X run to Y*
- *X fit to give Y* $\Rightarrow$ *X appropriate to supply Y*

Pang et al. (2003) described a syntax-based algorithm for building Finite State Automata (FSA) in order to derive and represent paraphrases. The data used for extracting the paraphrases was also a collection of parallel English translations, i.e. Multiple-Translation Chinese Corpus containing translations from 11 agencies, hence every sentence group consists of 11 different translations.

Every sentence in a sentence group was parsed into a parse tree. The resulting 11 parse trees were then merged into a parse forest following a top-down merging algorithm. Figure 2.1 illustrates the merging procedure for two parse trees. Tree 1 and Tree 2 represent two parallel sentences "12 persons were killed" and "twelve people died", respectively. After the merging, the parse forest was mapped into an FSA by traversing the forest top-down. Alternative paths was created for every merged node, thus an alternative path between two nodes is a paraphrase of the other path between the same nodes. The resulting FSA is shown in the bottom of Figure 2.1.

Evaluation of the paraphrases involved a comparison with the paraphrases produced in Barzilay and McKeown (2001). A total of 600 paraphrases (300 from each method) were presented to four human judges for three measurement choices: correct, partially correct, and incorrect. The overall result showed that the syntax-based method scored higher (81% correct) than the latter (66% correct).

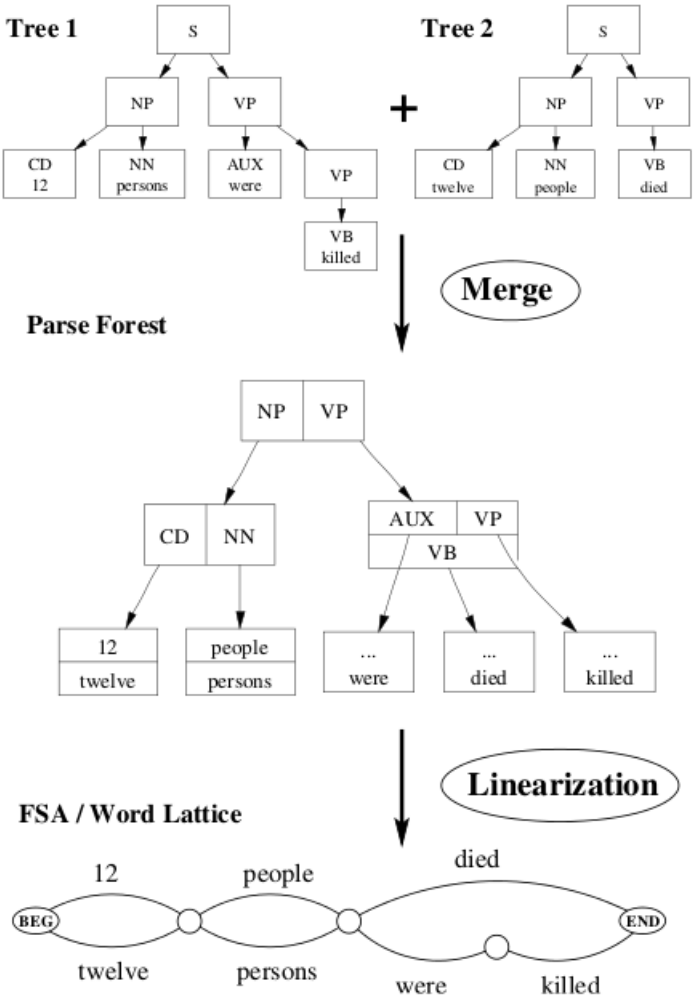


Figure 2.1: Merging Procedure and FSA construction (Pang et al., 2003)

2.4 Semantic Relation in Noun Compounds

Noun compounds often contain implicit knowledge between the composing nouns. For example, *leather shoes* can be interpreted as *shoes made from leather*, while *children book* can be interpreted as *book targetted for children.* Kim and Baldwin (2005) described a supervised approach to identifying the semantic relationships in binary noun compounds (noun compounds made up of two nouns). Their work is based on measuring the similarity of a noun compound with pre-tagged noun compounds.

Binary noun compounds were retrieved from a corpus, excluding those that consist of proper nouns and those that are part of larger compounds. The rightmost noun in the noun compound was referred to as head noun, while the rest was referred to as the modifier. The similarity between two binary noun compounds, e.g. *leather shoes* and *plastic bag*, was measured by calculating the product of the similarity between the modifiers (*leather* and *plastic*) and the similarity between the head nouns (*shoes* and *bags*). They applied WordNet::Similarity package (Pedersen et al., 2004) to calculate the similarity between two words.

For classifying the semantic relationships, 20 pre-existing semantic relations (Barker and Szpakowicz, 1998) were used. A few examples are as follows (N_1 denotes modifier, N_2 denotes head noun):

- AGENT: N_2 is performed by N_1, e.g. *military assault*
- BENEFICIARY: N_1 benefits from N_2, e.g. *student price*
- MATERIAL: N_2 is made of N_1, e.g. *plastic bag*
- POSSESSOR: N_1 has N_2, e.g. *national debt*
- PURPOSE: N_2 is meant for N_1, e.g. *children book*

For the training set, the semantic relations of the binary noun compound were annotated manually. The semantic relation of a binary noun compound n in test set was determined by calculating the similarity between n and every instance of noun compound in the training data. Say n is most similar to noun compound b (that is, n and b has the highest similarity value), then the semantic relation of n is assigned the same as that of b.

A different approach to the same task was later conducted by Kim and Baldwin (2006).

They introduced the so-called *seed verbs* in order to identify the semantic relation of binary noun compounds. Seed verbs are verbs that correspond to semantic relations, for example the verbs *possess*, *own*, *have*, and *belong to* corresponds to the semantic relation POSSESSOR, while the verbs *do* and *perform* correspond to the semantic relation AGENT.

In this work, 2166 binary noun compounds were first extracted, excluding those with proper nouns. Three corpora were parsed in order to identify verbs along with their subject, object, and whether the voice was active or passive. The result was then filtered by selecting only those which subject and object co-occur with the head noun and modifier of the extracted binary noun compounds. After the filtering, the verbs were mapped to the seed verbs based on WordNet::Similarity and Moby's Thesaurus[2]. The semantic relation for a noun compound was determined by selecting the best-fitting semantic relation which corresponds to each seed verb. Compared to their previously discussed approach (Kim and Baldwin, 2005) on the same set, which led to 48% accuracy, this method showed a significant improvement (53.3%).

The two methods we have discussed previously used a fixed set of semantic relations in interpreting the implicit knowledge in noun compounds. In contrast, Peñas and Ovchinnikova (2012) presented an unsupervised method for recovering the knowledge embedded in noun compounds without a predefined set of semantic relations. Given a list of arguments a, they extracted from large document collection the tuples $< p, a >$, where p is a predicate, and store them in a data structure called Proposition Store. This proposition is similar to that in DART (Clark and Harrison, 2009), in that the arguments have a predetermined set of syntactic relations between them. Some examples of propositions that contain the arguments *bomb* and *attack* are as follows:

- NPN *bomb in attack*
- NVPN *bomb explode in attack*
- NVNPN *bomb kill people in attack*

The output of the system is the predicate p which has the highest probability of $P(p|a)$. For cases where an argument is a proper name, in which case it may not have been observed before, it is abstracted over its named-entity class. The evalution was performed on RTE-2 development and test set, where 87 RTE pairs contain noun compounds that are responsible for determining entailment. The Proposition Store provides paraphrases

[2] `http://icon.shef.ac.uk/Moby/mthes.html`

for 63% of non-trivial cases in the dataset. The paraphrases found include *Berlin's landmark* ↔ *landmark in Berlin*, *people live in Germany* ↔ *Germany's people*, and *comments made by Wells* ↔ *David Wells' comments*; while missing paraphrases include *head of branch* ↔ *head commands branch* and *browser called Mosaic* ↔ *browser Mosaic*.

2.5 Summary

In the previous sections we have discussed various approaches to automatically extracting background knowledge from text corpora. We described in detail different methods for finding implicit knowledge based on certain syntactic and semantic patterns in sentences, including noun compounds, and learning paraphrases from parallel corpora of foreign novel translations. Of all these approaches, we consider DIRT-like inference rules to be most suitable for the RTE task, especially since there has been many applications of the rules in RTE. However, DIRT and the other methods discussed so far only encode general knowledge at syntactic level, which makes it difficult to express more complicated relationships. For this reason, we shall develop a technique which automatically extracts background knowledge while taking into account the analysis on semantic level. We shall elaborate on this approach in the next chapter.

Chapter 3

Proposed Approach

3.1 Basic Idea

Our target axioms shall represent the knowledge about if-then relations between two entities X and Y, for example *if X is the wife of Y, then X is married to Y*. Since we want to model knowledge, the axioms will be extracted only from RTE pairs with positive entailment. RTE pairs with negative entailment can act as a filter for the resulting axioms. That is, if doing inference on a negative entailment pair with the help from an axiom yields a positive entailment prediction, there is a possibility that the axiom contain incorrect knowledge. In the next sections of this chapter we will give an overview of our method, including identification of the key entities, refinement of the axioms through abstraction and filtering, and implementation plan.

3.2 Anchors

We illustrate our proposed approach using the following text-hypothesis pairs as examples. Every pair are taken from the RTE Challenge dataset, labelled with the challenge number and the set in which it is taken, i.e. test or development set, and followed by the example number.

We begin by choosing a pair of overlapping words in T and H. These words shall be referred to as *anchors* throughout the rest of the discussion. Consider the following example:

RTE-2 Test (58)

T: Across the Atlantic, on July 13, a radical Islamic cleric named Ali Al-Timimi was sentenced to life in prison, in Virginia, for soliciting treason.
H: Ali Al-Timimi is imprisoned in Virginia.

There are three overlapping words in this example: *Ali*, *Al-Timimi*, and *Virginia*. For now let us assume that we have access to a semantic representation that analyses *Ali Al-Timimi* as one proper name, therefore we treat *Ali Al-Timimi* as one anchor. Given anchor pair <Ali Al-Timimi, Virginia>, we can form an implication with T as the antecedent and H as the consequence and get the following axiom:

(3.1) If Ali Al-Timimi was sentenced to life in prison, in Virginia, then Ali Al-Timimi is imprisoned in Virginia

Some examples may have more than one pair of anchors. Consider Example 339 as follows:

RTE-2 Test (339)

T: When an earthquake rumbled off the coast of Hokkaido in Japan in July of 1993, the resulting tsunami hit just three to five minutes later, killing 202 people who were trying to flee for higher ground.
H: An earthquake occurred on the coast of Hokkaido, Japan.

In the example above, there are four possible anchors, namely: *earthquake*, *coast*, *Hokkaido*, and *Japan*. This results in $C(2,4) = 6$ possible combinations of anchor pair. The possible axioms generated are as follows, with the anchors shown in italic:

(3.2) *earthquake* rumbled off the *coast* ⇒ *earthquake* occurred on the *coast*

(3.3) *earthquake* rumbled off the coast of *Hokkaido* ⇒ *earthquake* occurred on the coast of *Hokkaido*

(3.4) *earthquake* rumbled off the coast of Hokkaido, *Japan* ⇒ *earthquake* occurred on the coast of Hokkaido, *Japan*

(3.5) *coast* of *Hokkaido* ⇒ *coast* of *Hokkaido*

(3.6) *coast* of Hokkaido, *Japan* ⇒ *coast* of Hokkaido, *Japan*

(3.7) *Hokkaido*, *Japan* ⇒ *Hokkaido*, *Japan*

So far we have considered only noun phrase as anchors. However, this needs not be the case as the following example shows:

RTE-1 Development (585)

T:	Former tennis star Vitas Gerulaitis died in such fashion in September, from a lethal carbon monoxide buildup related to the faulty installation of a propane heater.
H:	Vitas Gerulaitis died of carbon monoxide poisoning.

Based on our definition of anchors so far, *died* would be a possible anchor for the example above. However, we discover that using *died* and other verbs in general as anchors do not result in informative axioms. In fact, meaningful axioms would be produced with nouns as the anchors since they refer to entities.

3.3 Abstraction and Filtering

Recall the axiom shown in (3.1). Obviously it is not likely to encounter many occurrences of *Ali Al-Timimi* in the RTE dataset. Hence we shall abstract the two anchors over variables, say X and Y, so that an instantiation of these variables allows the axiom to apply in more general cases. Abstracting *Ali Al-Timimi* and *Virginia* in (3.1) over variables X and Y, we can rewrite the axiom as folows:

(3.8) X was sentenced to life in prison, in $Y \Rightarrow X$ is imprisoned in Y

A problem may arise when the abstraction leads to overgeneral rules. The following example illustrates this case:

RTE-1 Test (1944)

T:	African presidents opened a two-day summit in Accra, Ghana, on Ivory Coast, where northern rebels and fiercely loyalist southerners continue to split the cocoa-rich country, ignoring a never-implemented 2003 power-sharing deal.
H:	Accra is located in Ghana.

For Example 1944 above we can pick *Accra* and *Ghana* as anchors and perform abstraction, resulting in:

(3.9) $X, Y \Rightarrow X$ is located in Y

While Axiom (3.9) is good for stating knowledge about locations, this is not always the case with more general instantiations. For example, if we instantiate X and Y with the appositive "Jessica Litman, a law professor", then of course it is not reasonable

to conclude that Jessica Litman is located in a law professor. This means we need to restrict the type of X and Y, for instance by specifying them to locations, as to avoid too general instantiations.

Furthermore, we shall try to filter out uninformative axioms. Consider the last three axioms produced from Example 339 as shown in (3.5), (3.6), and (3.7). In these axioms, the consequence is exactly the same as the antecedent ($p \Rightarrow p$); this does not tell us any additional information. Hence we discard this kind of rules from the results.

3.4 Implementation

In order to automatically generate the axioms, we first need to do a syntactic analysis on the text-hypothesis pair. This phase includes parsing and part-of-speech tagging for identifying anchors and restricting them as nouns, proper names, time expression, and some other noun phrases. We opt to use readily available tools called C&C tools (Chapter 4) to implement this step. These tools provide integrated syntactic analysis including tokenisation, CCG parsing, part-of-speech tagging, and named-entity tagging. The last component is necessary as a way of specifying the category of the anchors as mentioned during the construction of Axiom (3.9).

After doing syntactic analysis, we also need to assign semantic representation for the axioms. This semantic representation should allow us to carry out two main tasks: representing the knowledge and providing a way of drawing logical inferences for recognising textual entailment. For these reasons, we adopt Discourse Representation Theory; not only does it assign detailed semantic representation for the axioms, but it also provides a translation to first order logic which accommodates logical inference.

Next step needs to be taken is finding the meaning relation or path between two anchors. We shall think of the anchors as nodes in a graph, thus the relation between them corresponds to a path in the graph. For this purpose we will implement shortest path algorithm called Breadth First Search. This algorithm together with the tools we need for syntactic and semantic analysis will be discussed in more detail in the Chapter 4.

Finally, we shall evaluate the added value of the axioms in providing background knowledge by comparing the performance of Nutcracker RTE system in inferring entailment prediction without and with the help of our background knowledge axioms.

3.5 Comparison with DIRT

This approach is parallel to that of DIRT, in that we shall focus on paraphrasing relations between two natural language expressions denoting entities. The differences between our method and DIRT are as follows:

- the inference rules of DIRT were extracted using unsupervised method, while our axioms are produced from labelled data, hence supervised
- instead of using only surface syntactic patterns, we shall perform deep semantic analysis using logical form to discover and represent the relations between two entities
- given the domain of RTE, our axioms shall express not only relations expressing similarity, but also causal relations such as X buys $Y \Rightarrow X$ owns Y. Since the text is expected to entail the hypothesis, the directionality of the axiom is straightforward.
- our axioms shall have different levels of abstraction. That is, in addition to leaving the anchors X and Y underspecified as in DIRT, we will also restrict possible instantiations of the anchors with respect to their named-entity classes.
- DIRT method results in a lot of similar paths between two entities but contain many incorrect paraphrases, hence high recall and low precision. In contrast, our axioms are extracted from labelled data of RTE Challenge corpora, therefore we expect high precision, but since there are not many datasets available, this might result in low recall.

Chapter 4

Theoretical Background

4.1 Semantic Representation

In this section we shall discuss the semantic representation used in this thesis. The text-hypothesis pairs are represented using Discourse Representation Theory (DRT) following neo-Davidsonian event analysis. These concepts are discussed in the first two following subsections. In order to arrive at this level of representation, we employ external tools for syntactic and semantic analysis, as discussed in the last subsection.

4.1.1 Discourse Representation Theory

DRT, introduced by Kamp (1981), provides semantic representations constructed in abstract structures called Discourse Representation Structures or DRSs. These structures do not only cover single sentence representations, but also coherent natural language texts. The construction rules of DRSs involves the implementation of syntax-semantics interface (Kamp and Reyle, 1993). Since this is not the topic of this thesis (such construction is done using Boxer, which will be discussed shortly in Section 4.1.3), we will not go through the DRSs construction algorithm in this section. Instead, we will only proceed briefly by providing illustrations of how DRSs are used to represent natural language sentences.

There are two components of a DRS, namely:

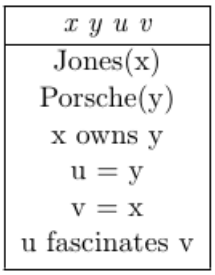


Figure 4.1: DRS for sentence (4.1)

- a set of discourse referents
- a set of DRS-conditions describing properties of discourse referents or relations between discourse referents. Since DRS is a recursive data structure, the conditions can contain other DRSs as well.

The implementation of DRT enables us to address several linguistic phenomena including anaphora and presupposition. The following example illustrates the analysis of sentences that involve pronominal anaphora, as described in Kamp and Reyle (1993). Consider the following sentences:

(4.1) Jones owns Ulysses. It fascinates him.

The resulting DRS[1] is shown in Figure 4.1. The discourse referents are shown in the top box, namely x, y, u, and v. The DRS condition $Jones(x)$ can be construed as: x stands for the individual denoted by the proper name *Jones*. The indefinite noun phrase *a Porsche* indicates that the individual must satisfy the information in the noun, hence the DRS condition $Porsche(y)$. The anaphoric pronouns *it* and *him* in the second sentence of (4.1) are resolved to the corresponding noun phrase they are referring, i.e. *a Porsche* and *Jones*. This results in DRS conditions in the form of equality.

Natural language sentences with negative polarity introduce negation operator in the DRS, while the word "or" corresponds to disjunctive operation. In addition, conditional sentences and universal quantifications are accommodated by implication operators. The construction of DRSs (Kamp and Reyle, 1993) for the donkey sentence:

(4.2) Every farmer who owns a donkey beats it

[1]In this section, we follow Kamp and Reyle (1993) abridged notation for representing DRS conditions

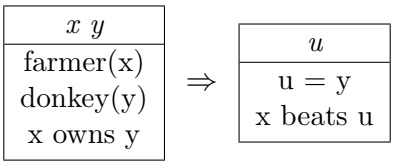


Figure 4.2: DRS for sentence (4.2)

leads to the DRSs depicted in Figure 4.2. Note that the discourse referent x is defined in the left DRS, but it is also accessed in the condition of the right DRS. The accessibility of a discourse referent is specified in a relation between DRSs called *subordination*. For example, for two DRSs $K1$ and $K2$ in the form of $K1 \Rightarrow K2$, $K1$ is said to subordinate $K2$. By this, we mean the discourse referents of $K1$ are accessible from $K2$, but not vice versa. For more information about subordination and accessibility, see Kamp and Reyle (1993).

4.1.2 Neo-Davidsonian Event Representation

In neo-Davidsonian approach of event analysis (Parsons, 1990), verbs are considered to be related to events and states. More precisely, rather than pointing out to a particular action, a verb in a sentence can be seen as an instance of a kind of event. Consider the following sentence:

(4.3) Brutus stabbed Caesar

In this view, sentence (4.3) above pretty much states the following[2]:

There is some event e such that:

- e is a stabbing
- e has Brutus as its subject
- e has Caesar as its object

Translating this into first order logic, we get:
$\exists e.stabbing(e) \wedge subject(e, Brutus) \wedge object(e, Caesar)$

[2]for now we ignore tenses

Table 4.1: Some commonly-used thematic roles and their examples

Thematic Role	Definition	Example
Agent	The volitional causer of an event	*Brutus* stabbed Caesar
Theme	The participant most directly affected by an event	Brutus stabbed *Caesar*
Goal	The destination of an object of a transfer event	He donated the money *to the hospital*
Beneficiary	The beneficiary of an event	Mary wrote a book *for John*
Instrument	An instrument used in an event	She ate *with a spoon*
Experiencer	The experiencer of an event	*Julie* felt hot

The entities denoted as subject and object from the above sentence can be viewed as event participants. Brutus plays a role as the agent doing the stabbing, while Caesar as the patient being stabbed. On this account, thematic roles are used to link an event with its participants. A list of some commonly used thematic roles (Jurafsky and Martin, 2008) is summarised in Table 4.1.

Now consider the following sentence:

(4.4) Brutus stabbed Caesar with a knife

Using these thematic roles, the neo-Davidsonian representation of sentence (4.4) above is:
$\exists e.stabbing(e) \wedge agent(e, Brutus) \wedge theme(e, Caesar) \wedge with(e, knife)$.

4.1.3 C&C tools and Boxer

Before the semantics can be formalised, we first need to took at the syntactic level. C&C Tools (Clark and Curran, 2004) provide a combined syntactic analysis of a tokenised text according to the following pipeline:

- Parts-of-speech tagging, with grammatical categories from Penn Treebank
- Named-entity tagging for recognising person, location, organisation, date, time, monetary amount, and miscellaneous entities
- Parsing based on Combinatory Categorial Grammar (CCG)

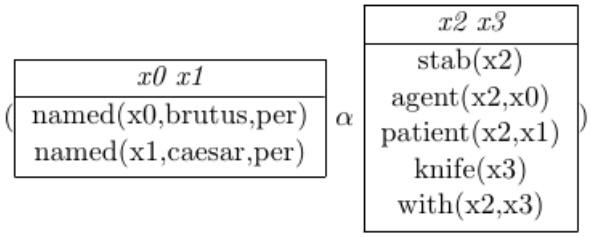


Figure 4.3: Boxer output for sentence (4.4)

The output of these processes is a CCG derivation together with the POS and named-entity tags. They become the input components for semantic analysis, which is done by Boxer. Boxer (Bos, 2008) assigns semantic representations to natural language texts by implementing Discourse Representation Theory with neo-Davidsonian event analysis. It produces DRSs which basic conditions fall into the following categories:

- time and name expressions along with their named-entity tags
- equality relations between two discourse referents
- one-place relations specifying the property of a discourse referent, introduced by nouns, verbs, adjectives, and adverbs
- two-place relations describing the relation between two discourse referents, introduced by thematic roles and prepositions

The semantic representations can be written in a number of format, e.g. Prolog, XML, and box-like structures. Figure 4.3 demonstrates Boxer output in box-like structures for sentence (4.4) in the previous subsection. The DRS on the left of α operator contains conditions that trigger presuppositions, e.g. proper names, definite descriptions, etc. The third argument of `named` relation denotes the named entity tag of the lemma, i.e. `org` for organisation, `per` for person, `loc` for location and `nam` for miscellaneous entity. By default, Boxer follows proto thematic relations with basic roles such as agent and patient. A more complicated analysis for sentence (4.5) below is shown in Figure 4.4.

(4.5) In June of 2004, an application called InfiniteCanvas (or IC) was released by student Markus Müller, attempting to provide a solution to the common problems of infinite canvas.

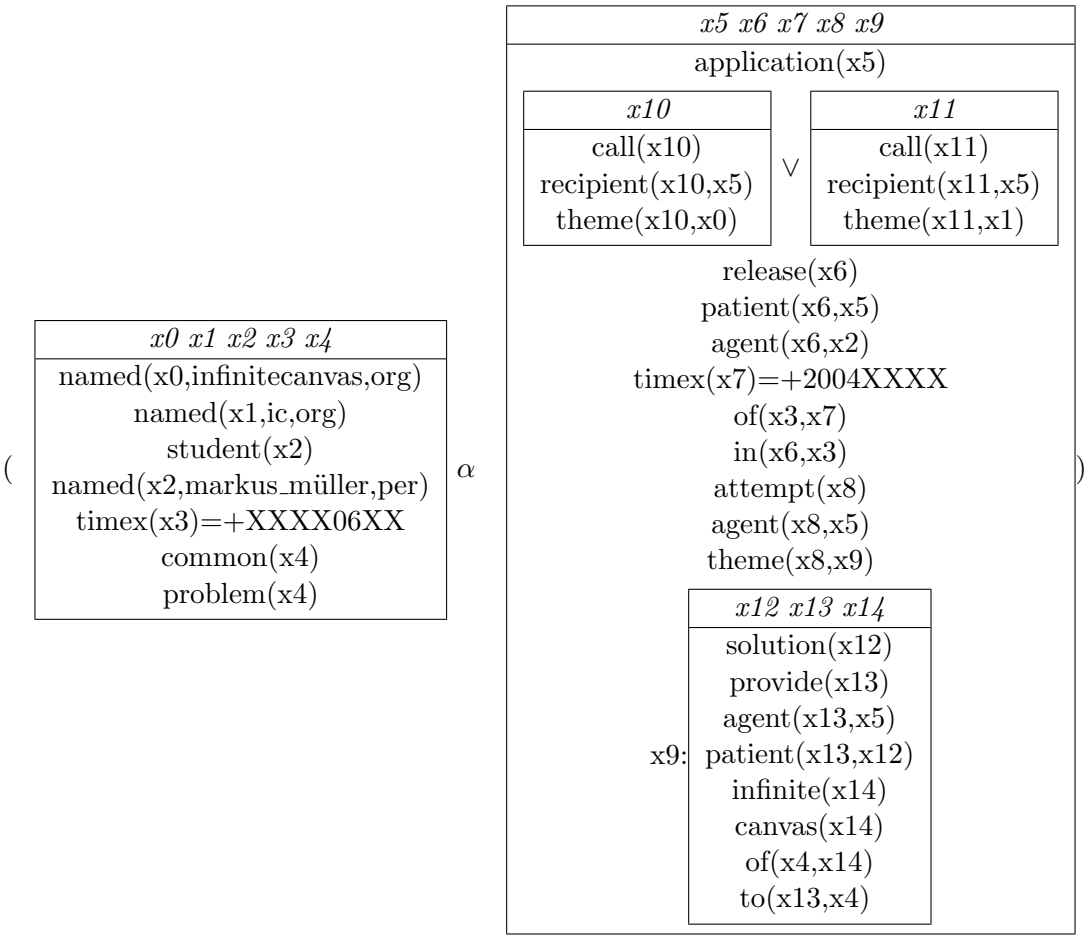


Figure 4.4: Boxer output for sentence (4.5)

4.2 RTE with Logical Inference

Bos and Markert (2005) made an approach to RTE by combining shallow and deep semantic analysis. The shallow analysis involves measuring word overlap between T and H, by taking into account the length of both text fragments as well. The deep semantic analysis departs from the DRSs for T and H produced by Boxer. These DRSs are then translated into first order logic (Kamp and Reyle, 1993). The RTE system, called Nutcracker, makes use of external theorem provers and model builders to support logical inference.

Before talking about how inference works in RTE, let us first discuss about the notion of theorem proving and model building in first order logic. A first order model can be thought of as a mathematical formalism of situations (Blackburn and Bos, 2005). Given a vocabulary, a model contains two pieces of information: a domain D specifying the entities that are talked about, and an interpretation function F specifying semantic values in D. Here is an example of a vocabulary:

$\{(LIKE, 2), (WIZARD, 1), (HARRY, 0), (RON, 0), (VERNON, 0)\}$.

In general, a first order vocabulary describes the topic of the conversation. This vocabulary, for instance, tells us that we are going to talk about a relation denoted by the symbol $LIKE$. The arity 2 shows that this relation holds between two individuals. The vocabulary also consists of a unary relation $WIZARD$, describing the property of an individual being a wizard. The symbols of arity 0, i.e. $HARRY$, RON, and $VERNON$ stand for individuals and are used to refer to Harry, Ron, and Vernon, respectively. An example model with respect to this vocabulary is as follows:

$M = (D, F)$
$D = \{d1, d2, d3\}$
$F(HARRY) = d1$
$F(RON) = d2$
$F(VERNON) = d2$
$F(WIZARD) = \{d1, d2\}$
$F(LIKE) = \{(d1, d2), (d2, d1)\}$

Model M says the following: $d1$, $d2$ and $d3$ are called Harry, Ron, and Vernon; both Harry and Ron are wizards; Harry likes Ron and Ron also likes Harry. This model also states that Vernon likes nobody and nobody likes Vernon. Given a first order formula, a model builder works by trying to build a model in which the formula is

satisfied (evaluates to true). A first order formula is called satisfiable if it is satisfied in at least one model, while a formula is called valid if it is satisfied in all models. It is computationally unfeasible, of course, to check all possible models in order to find out whether a formula is valid, since there are infinitely many models. Therefore we change from model-theoretic point of view to proof-theoretic, which investigates logical validity from a purely syntactic perspective (Blackburn and Bos, 2005). There are various techniques of theorem proving, namely natural deduction, resolution, tableau system, and so on. Nowadays we can employ the so-called theorem provers in doing automatic theorem proving, e.g. Vampire (Riazanov and Voronkov, 2002) and Gandalf (Tammet, 1997). Given a formula, a theorem prover attempts to show that the formula is valid by trying to find a proof for it.

Given an RTE pair, let T and H be the first order logic representation of the text and the hypothesis, respectively. Consider the following formula:
$\phi = T \rightarrow H$
$\psi = T \wedge H$

In Nutcracker, a theorem prover is used in finding out whether:

1. T implies H, or positive entailment, given ϕ as the input. If the theorem prover manages to find a proof for ϕ, i.e. ϕ is valid, then we can conclude that T entails H.

2. T and H is inconsistent, i.e. a case of negative entailment. In order to determine whether ψ is inconsistent (not satisfied in any model), we must prove that its negation is valid, hence the input $\neg\psi$. If the theorem prover finds a proof for $\neg\psi$, then ψ is inconsistent, which means T does not entail H.

On the other side, a model builder plays a role in determining:

1. Negative entailment, given as input $\neg\phi$. If the model builder finds a model for this formula, i.e. $\neg\phi$ is true in at least one model, then we know that ϕ is not valid. Therefore, T does not entail H.

2. Possibility of entailment, given as input ψ. If there is a model for ψ, then T and H are consistent, which can be an indication of an entailment.

These inference tasks are carried out with the help of background knowledge computed from WordNet relations with respect to T and H. Other sources of background knowl-

edge, including the axioms we shall generate in this work, can be easily included in the input for the theorem prover and model builder. Let BK be the background knowledge axioms. The inputs to the inference engines are as described before, with ϕ and ψ rewritten as follows:
$\phi = (BK \wedge T) \rightarrow H$
$\psi = BK \wedge T \wedge H$

4.3 Breadth First Search for Finding Path between Anchors

Our target axiom shall express relations between two anchors in T and H. This relation between the two anchors can be thought of as a path between two nodes in a graph. That is, we will consider each semantic representation of T and H as a graph, where the nodes are the DRS-conditions and the edges denote the connectivity between them. Since every anchor corresponds to a node, the minimal relation between two anchors can be found by looking for the shortest path between their corresponding nodes. This graph representation will be discussed in more detail in the next chapter. For now we will talk about Breadth First Search, a shortest path algorithm that we shall adopt in this work.

Breadth First Search (BFS) is a graph search strategy in which the root node is expanded first to all of its neighbouring nodes. It proceeds by expanding these neighbouring nodes into their neighbouring nodes that have not been expanded, and so on. In other words, this strategy always expands all the nodes at depth d in the search tree before those of depth $d+1$. BFS is guaranteed to find a solution (if there is any) and the first solution will always be of minimum depth in the search tree (Russell and Norvig, 2009).

Figure 4.5 shows a graph representing links between cities of Indonesia. The (partial) construction of BFS tree with respect to this graph, up to five nodes expansion, is demonstrated in Figure 4.6 with the node Bandung as the root. Note that for any node v in the tree, the path from the root to v is the shortest path.

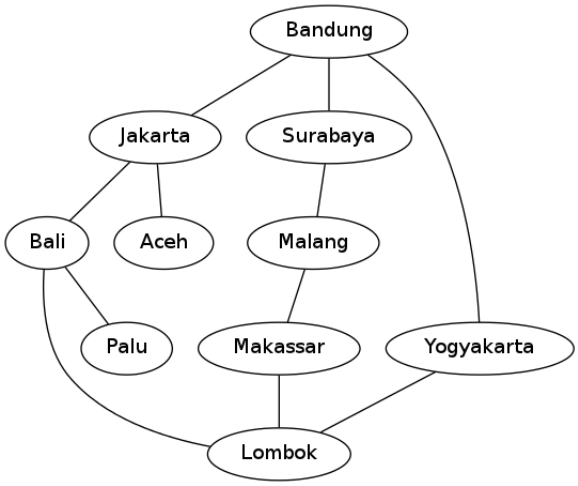


Figure 4.5: An example graph of cities

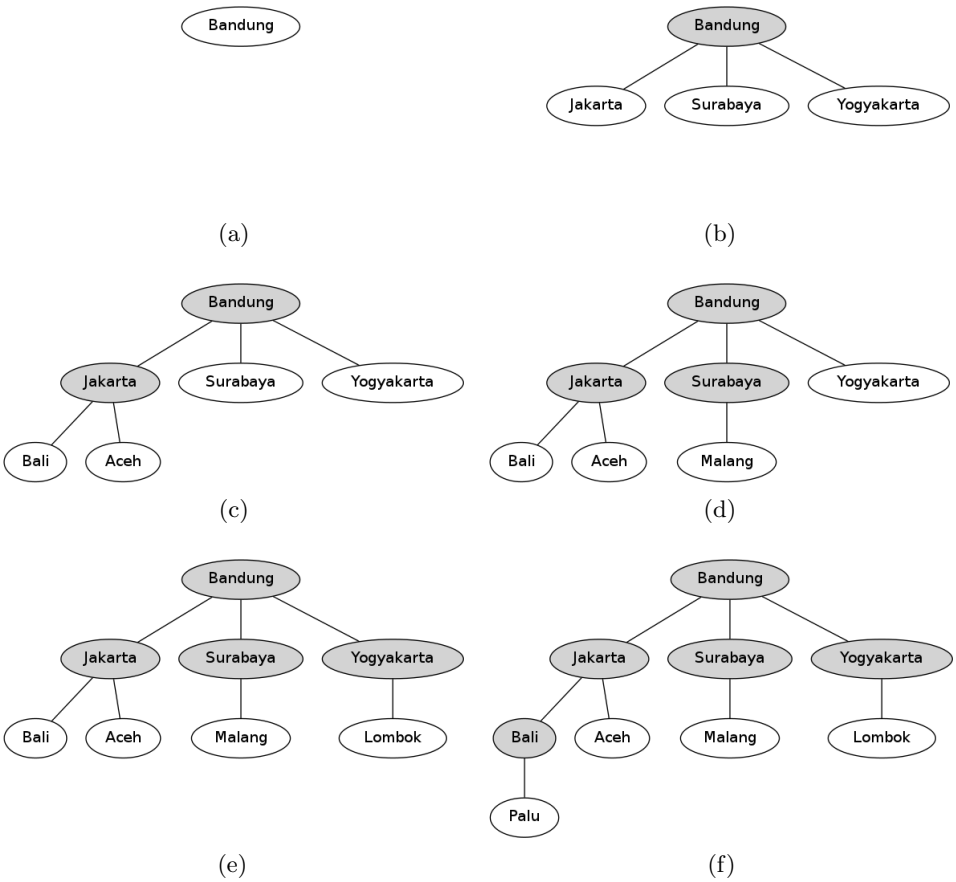


Figure 4.6: Node expansions in BFS

Chapter 5

Induction of Background Knowledge Axioms

This chapter discusses the implementation of our proposed approach to automatically inducing background knowledge axioms for recognising textual entailment. To accomplish our goal, we follow a pipeline scheme as illustrated in Figure 5.1. The architecture gives an outline of the phases in generating background knowledge. The next sections describe each phase in more detail.

5.1 Preprocessing

We begin our work by creating a semantic representation for each T and H using C&C tools and Boxer, following the pipeline described in Section 4.1.3. From the resulting DRSs, we extract the DRS-conditions to form a flat list representation which shall be used interchangeably with the term *predicates* in this chapter. Consider Example 444 below.

RTE-2 Test (444)

T:	According to the criminal complaint, Van Meeteren operated a business called Mental Health Professionals in International Falls.
H:	Van Meeteren was the manager of Mental Health Professionals.

Figure 5.2a and 5.2b illustrate the semantic representation output by Boxer for T and H. The set of the extracted predicates from these DRSs are shown in Figure 5.2c and

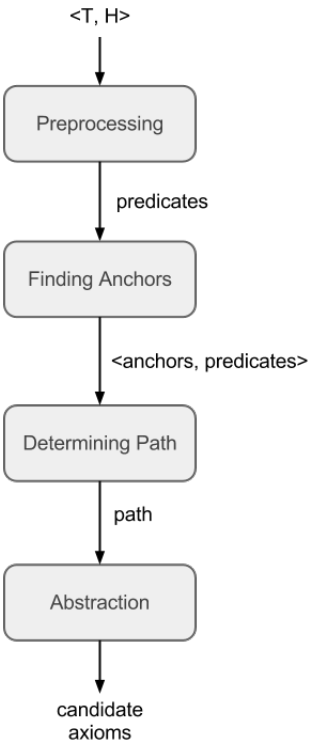
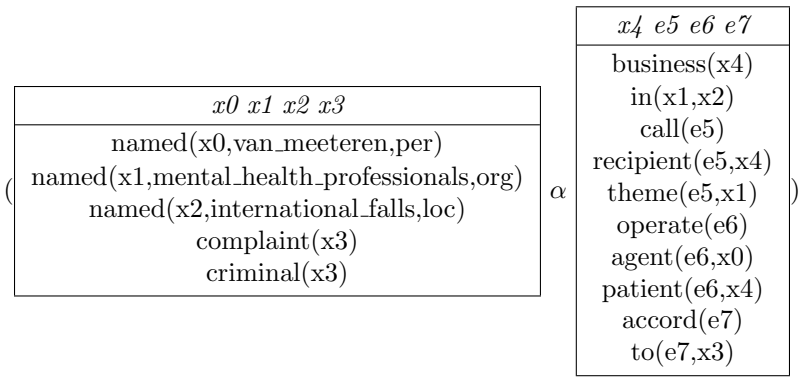


Figure 5.1: Pipeline scheme for generating background knowledge axioms

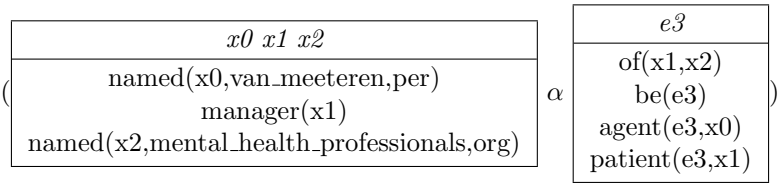
5.2d. Since we use Prolog format as Boxer output, notice the slight difference from the DRSs conditions represented in the boxes. For instance, one-place relations are represented in predicates in the form of `pred(Referent,Lemma,POS_tag,Sense)`[1]. Therefore `business(x4)` corresponds to `pred(x4,business,n,0)`, where `n` stands for noun; DRS-condition `operate(e6)` corresponds to `pred(e6,operate,v,0)`, and so on. Two-place relations are represented in `rel(Referent1,Referent2,Thematic_role,Sense)`, e.g. `in(x1,x2)` is written as `rel(x1,x2,in,0)`, etc. For convenience, we will still refer to these predicates as one-place relations and two-place relations.

The DRSs conditions are extracted regardless of the internal structure of the DRS and the scope of the discourse referents. For now we are only interested in generating inference rules in the form of T $\Rightarrow$ H, where both T and H are conjunctions of positive literals. Therefore, in this stage we simply ignore the conditions inside a negated DRS and DRSs that are connected by implication or disjunction. An example of extraction of a DRS containing a negated DRS is shown in Figure 5.3.

[1] `Sense` denotes the n-th WordNet sense picked for the corresponding lemma.



(a) DRS for T



(b) DRS for H

```
T = {named(x0,van_meeteren,per,0),
named(x1,mental_health_professionals,org,0),
named(x2,international_falls,loc,0), pred(x3,complaint,n,0),
pred(x3,criminal,a,0), pred(x4,business,n,0), rel(x1,x2,in,0),
pred(e5,call,v,0), rel(e5,x4,recipient,0), rel(e5,x1,theme,0),
pred(e6,operate,v,0), rel(e6,x0,agent,0), rel(e6,x4,patient,0),
pred(e7,accord,v,0), rel(e7,x3,to,0)}
```

(c) Predicates of T

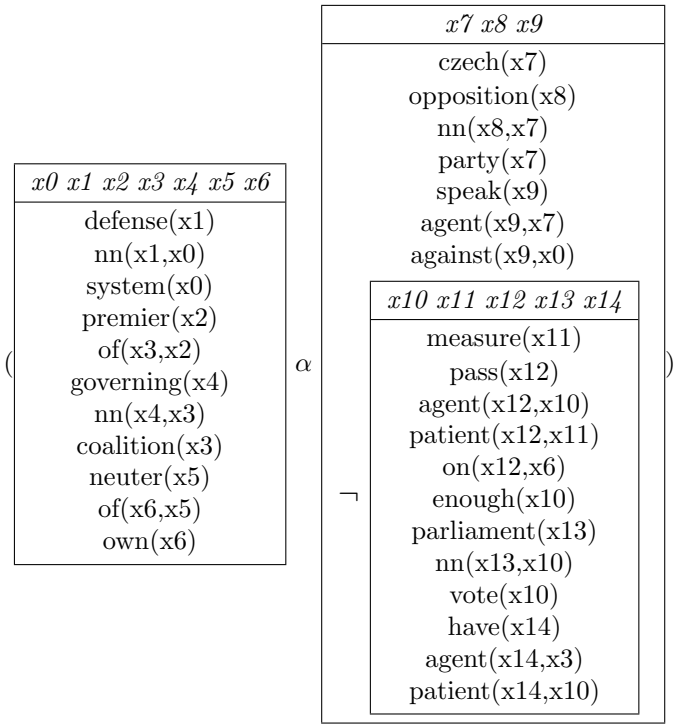
```
H = {named(x0,van_meeteren,per,0), pred(x1,manager,n,0),
named(x2,mental_health_professionals,org,0), rel(x1,x2,of,0),
pred(e3,be,v,1), rel(e3,x0,agent,0), rel(e3,x1,patient,0)}
```

(d) Predicates of H

Figure 5.2: Extraction of DRS-conditions from the semantic representation of Example 444

Czech opposition parties have spoken against the defense system, and the premier's governing coalition does not have enough parliament votes to pass measures on its own.

(a) Text



(b) Semantic representation

```
Predicates = {pred(x1,defense,n,0), rel(x1,x0,nn,0), pred(x0,system,n,0),
pred(x2,premier,n,0), rel(x3,x2,of,0), pred(x4,governing,n,0),
rel(x4,x3,nn,0), pred(x3,coalition,n,0), pred(x5,neuter,a,0),
rel(x6,x5,of,0),pred(x6,own,n,0), pred(x7,czech,a,0),
pred(x8,opposition,n,0), rel(x8,x7,nn,0), pred(x7,party,n,0),
pred(x9,speak,v,0), rel(x9,x7,agent,0), rel(x9,x0,against,0)}
```

(c) Extracted predicates

Figure 5.3: Predicates extraction from DRSs containing negation operator

5.2 Finding Anchors

Recall from Chapter 3 that our target axiom should be a form of paraphrasing relation between two entities, or the so-called anchors, in the text-hypothesis pair. Consider example RTE-2 Test 444 in the previous section. Intuitively, an ideal choice of anchors for this pair would be *Van Meeteren* and *Mental Health Professional*. We shall look at the predicate representation obtained from the previous step in order to find these anchors.

A lemma l is called an anchor if it satisfies the following requirements:

- l occurs in both T and H
- l is tagged as a noun, proper name, or time expression

Given this definition and the predicates shown in Figure 5.2c and 5.2d, we have two candidate anchors for Example 444: `van_meeteren` and `mental_health_professionals`. Our target axiom requires a pair of anchors, consequently we perform 2-combinations on the set of anchors. This leaves us with one possible anchor pairs for Example 444. Furthermore, since the axioms are between two entities, we apply the following constraint: an anchor pair must describe the property of different discourse referents in both T and H. The anchor pair of 444 satisfies this constraint. The output of this phase is a set of tuples containing predicates of T and H for each corresponding anchor pairs. Thus for anchor pair `<van_meeteren,mental_health_professionals>`, we have the following anchor predicates:

```
anchor(T): <named(x0,van_meeteren,per,0),
   named(x1,mental_health_professionals,org,0)>
anchor(H): <named(x0,van_meeteren,per,0),
   named(x2,mental_health_professionals,org,0)>
```

Synchronisation

Since the text and the hypothesis are processed separately, there are some cases where the tagger assigns mismatching POS or named-entity tags to the same lemma, as shown in the following Example 343:

RTE-2 Test (343)

T:	CERN has now grown to include 20 member states and enjoys the active participation of many other countries world-wide.
H:	CERN has 20 member states.

In T, the lemma *CERN* is tagged as an organisation, while in H it is incorrectly tagged as a location. We treat cases like these by synchronising the syntactic analyses of T and H in the CCG derivation. If there is a mismatching tag for the same lemma l in T and H, then replace the tag of l in H with its tag in T. This is based on our observation that the named-entity tags in T tend to be more accurate than those in H. Consider the following example:

RTE-2 Test (146)

T:	The first union between Sweden and Norway occurred in 1319 when the three-year-old Magnus, son of the Swedish royal Duke Eric and of the Norwegian princess Ingeborg, inherited the throne of Norway from his grandfather, Haakon V, and in the same year was elected King of Sweden, by the Convention of Oslo.
H:	Magnus was a king of Norway.

We synchronise only lemmas that have the same case sensitivity in both T and H, because surely we do not want to synchronise, for instance, cases like "king" in Example 146. By doing this synchronisation on the syntactic level, we make sure that the tags of a lemma in the semantic representation are immediately synchronised as well.

5.3 Determining Path between Anchors

5.3.1 Graph Representation

Let S be the flat semantic representation obtained from the step in Section 5.1, i.e. S is a set of predicates. For each T and H, we create an undirected graph representation $G =< Nodes, Edges >$, where $Nodes$ is S and $Edges$ is a set of edges connecting elements of S. Two predicates $P1$ and $P2$ are connected by an edge if and only if:

- at least one of $P1$ or $P2$ introduces a one-place relation (i.e. describes the property of a discourse referent)
- $P1$ and $P2$ share the same discourse referent

Figure 5.4 shows the graph representation of the hypothesis from Example 444.

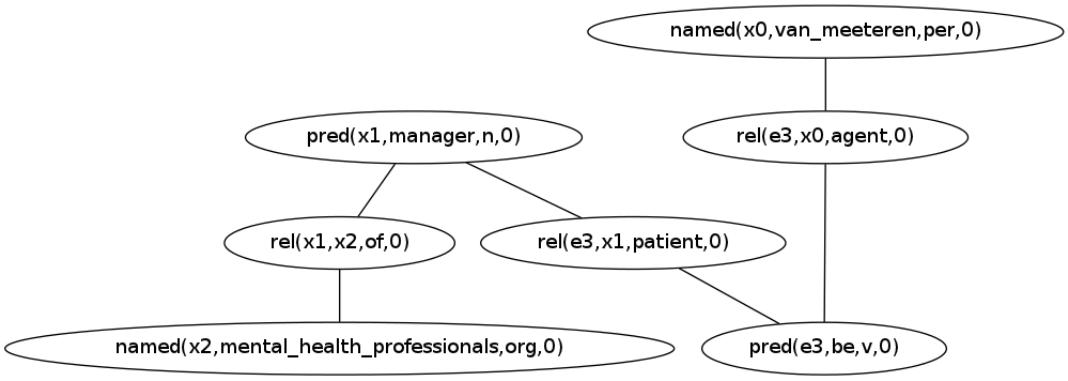


Figure 5.4: Graph representation for H in Example 444

5.3.2 Shortest Path

Now that we have anchors and the graphs expressing relations between predicates, we are ready to compute how the anchors are connected to each other. Given a graph G and a pair of anchors A and B, where both are nodes of G, we define the relation between A and B as the shortest path from A to B (or vice versa) in G. In this phase we implement Breadth First Search shortest path algorithm (Section 4.3).

The resulting paths between `van_meeteren` and `mental_health_professionals` in T from Example 444 are shown in Figure 5.5. With respect to the graph in Figure 5.4, the shortest path in H between the two anchors is:

```
path(H) = {named(x0,van_meeteren,per,0), rel(e3,x0,agent,0),
    pred(e3,be,v,0), rel(e3,x1,patient,0), pred(x1,manager,n,0),
    rel(x1,x2,of,0), named(x2,mental_health_professionals,org,0)}
```

Combining the predicates in each path with conjunction, we can roughly read the above T path as *Van Meeteren operates a business called Mental Health Professionals*, and H path as *Van Meeteren is a manager of Mental Health Professionals*. This pretty much reflects what we need for the axiom.

Choosing The Right Path

In the previous example, we found that the shortest paths express meaningful relationships between the anchors. In some cases, however, the shortest paths alone are not suitable for generating axioms. Consider Example 93 as an illustration:

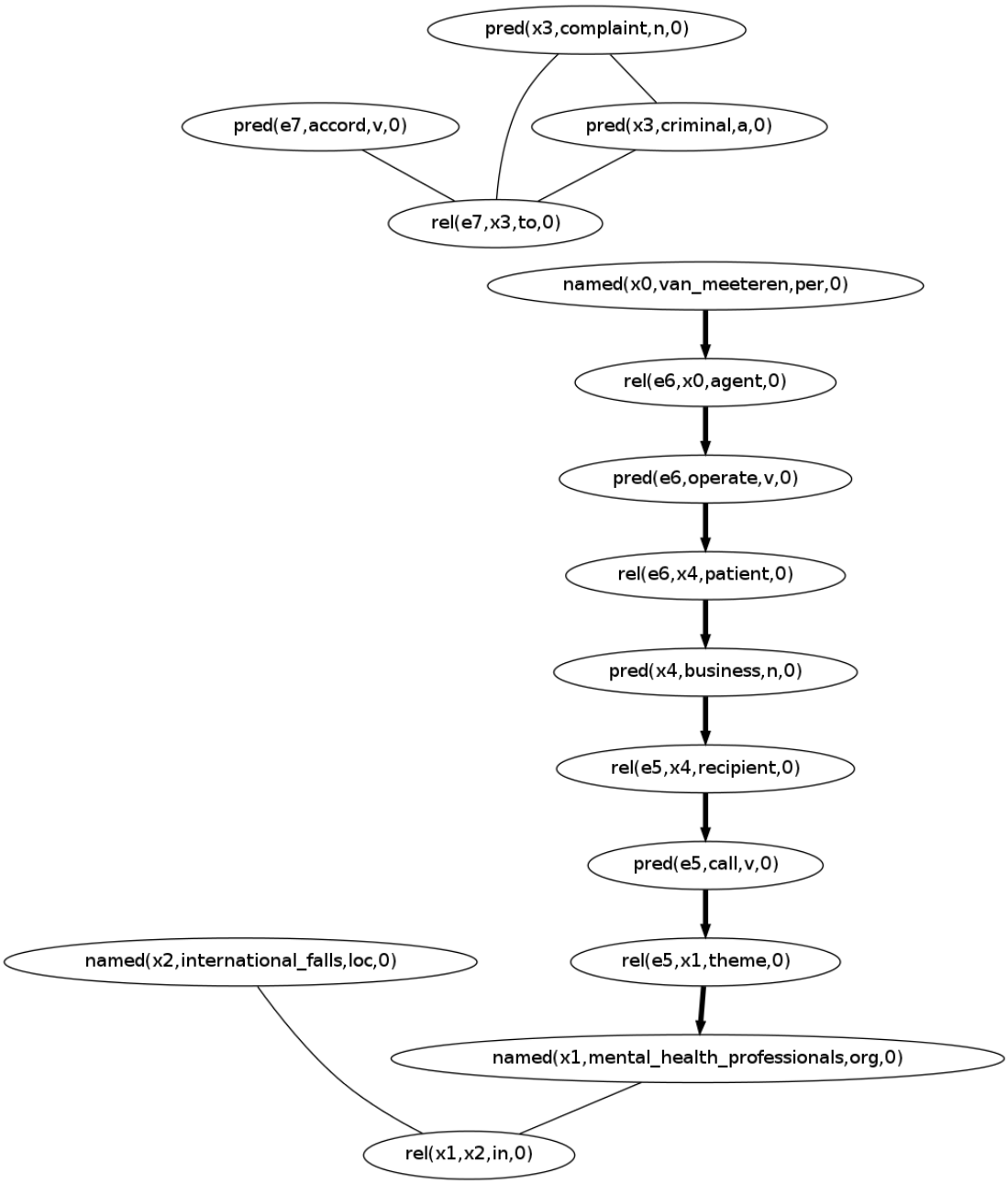


Figure 5.5: Graph representation of T from Example 444. The shortest path from anchor `van_meeteren` to `mental_health_professionals` is shown in directed edges

RTE-2 Development (93)

T: Tilda Swinton has a prominent role as the White Witch in "The Chronicles of Narnia: The Lion, The Witch and The Wardrobe", coming out in December.
H: Tilda Swinton plays the part of the White Witch.

Figure 5.6 depicts the subgraphs of graph representations for T and H. Say we want to get shortest paths between anchors `tilda_swinton` and `white_witch`. Note that for T we have two possible shortest paths, namely:

```
path1 = {named(x0,tilda_swinton,per,0), rel(x10,x0,agent,0),
    pred(x10,have,v,0), rel(x10,x8,patient,0), pred(x8,prominent,a,0),
    rel(x8,x1,as,0), named(x1,white_witch,org,0)}
path2 = {named(x0,tilda_swinton,per,0), rel(x10,x0,agent,0),
    pred(x10,have,v,0), rel(x10,x8,patient,0), pred(x8,role,n,0),
    rel(x8,x1,as,0), named(x1,white_witch,org,0)}
```

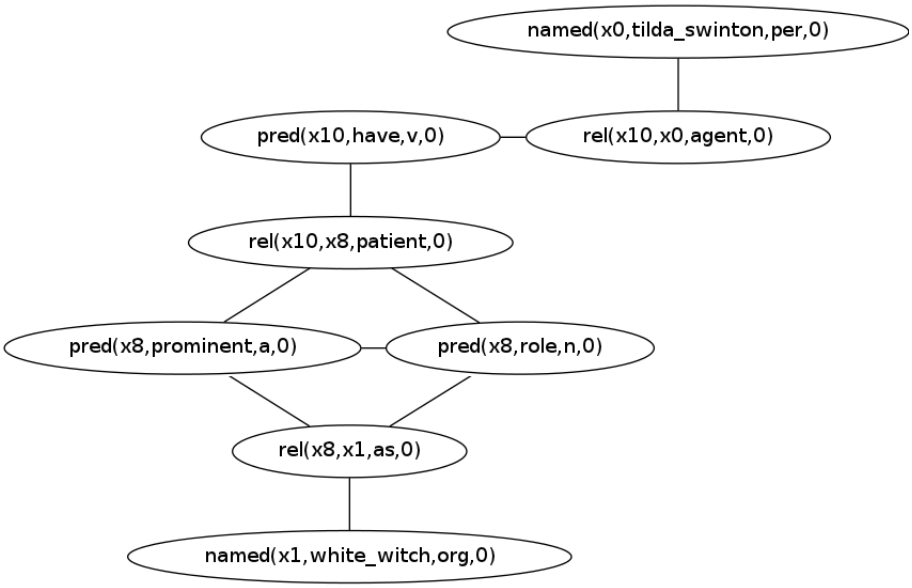
The first path is problematic because we have an adjective which is supposed to modify a noun, but the noun itself is not included in the path. Clearly this path does not lead to an informative relation between `tilda_swinton` and `white_witch`. Thus, such paths are eliminated from the candidate solutions. In addition to adjectives, this constraint is also applied to cardinalities and adverbs (in the case of adverb, the path is relative to the modified verb). The chosen paths for Example 93 are then as follows:

```
path(T) = {named(x0,tilda_swinton,per,0), rel(x10,x0,agent,0),
    pred(x10,have,v,0), rel(x10,x8,patient,0), pred(x8,role,n,0),
    rel(x8,x1,as,0), named(x1,white_witch,org,0)}
path(H) = {named(x0,tilda_swinton,per,0), rel(x3,x0,agent,0),
    pred(x3,play,v,0), rel(x3,x1,patient,0), pred(x1,part,n,0),
    rel(x1,x2,of,0), named(x0,white_witch,org,0)}
```

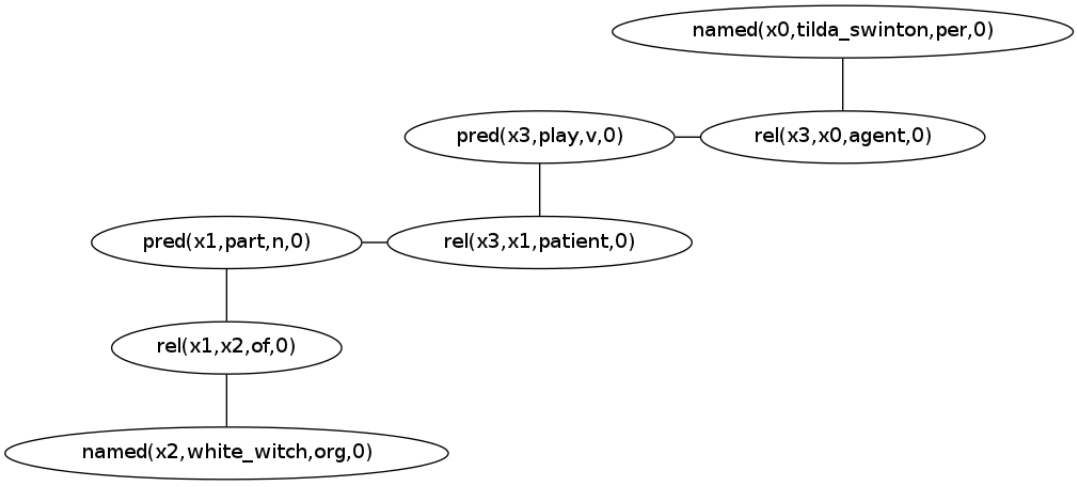
On several other examples, the shortest paths do not lead to informative axioms. Consider Example 164 below. Using the obvious anchor pair *The Eiffel Tower* and *1889*, we will get an axiom in the form of $p \Rightarrow p$, which does not contribute to valuable knowledge in any way. For this example, a proof can be found without the help of background knowledge, therefore we discard the uninformative axiom.

RTE-2 Test (164)

T: The Eiffel Tower was built in 1889 for the Universal Exposition as a monument to the scientific achievements of the 18th century.
H: The Eiffel Tower was built in 1889.



(a) Graph representation of T



(b) Graph representation of H

Figure 5.6: Graph representations of Example 93

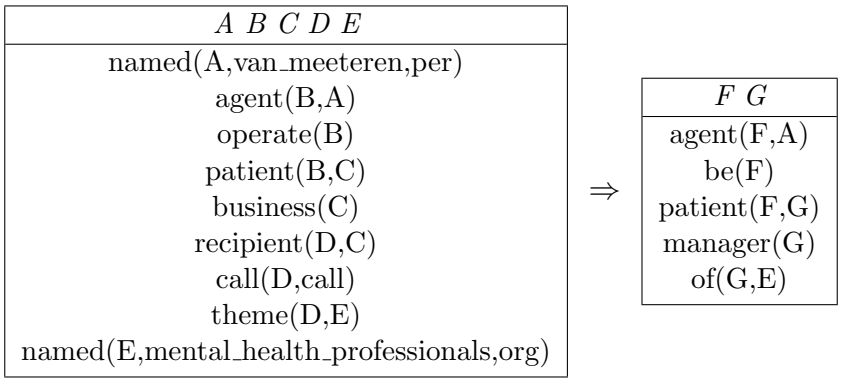


Figure 5.7: Output axiom for Example 444

5.4 Axioms Abstraction

In this step we do some adjustments on the paths obtained from the previous step in order to form an appropriate axiom. This includes renaming the discourse referents such that except for the anchors, no DRS-conditions in T and H shall have the same discourse referents. This is necessary because the raw text of T and H are processed separately and there is a possibility that different lemmas in T and H are denoted by the same discourse referent. The axiom is represented in DRSs connected by implication operator, with the antecedent being a DRS constructed from the conjunction of the DRS-conditions from the shortest path of T and the consequence from those in the shortest path of H. The axiom produced from Example 444 is shown in Figure 5.7 (since all DRS-conditions in the left box are accessible from the right one, there is no need to write the conditions for anchors *Van Meeteren* and *Mental Health Professionals* in the right box).

In the attempt of making the axiom suitable for providing general knowledge, we are now faced with two situations:

1. Keeping both anchors as they are; that is, the axiom will be exactly as shown in Figure 5.7. The downside of this option is the knowledge might become too specific, since it is unlikely to encounter many occurrences of proper names *Van Meeteren* and *Mental Health Professionals* in other RTE pairs.

2. Removing the anchors from the axiom, thus stating, with respect to Figure 5.8: *if A operates a business called E, then A is the manager of E*. However, without specifying what A and E are, we might end up with over-general axioms and problematic instantiations of A and E.

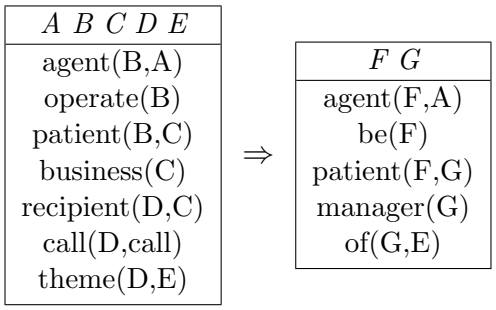


Figure 5.8: Output axiom for Example 444 with both anchors removed

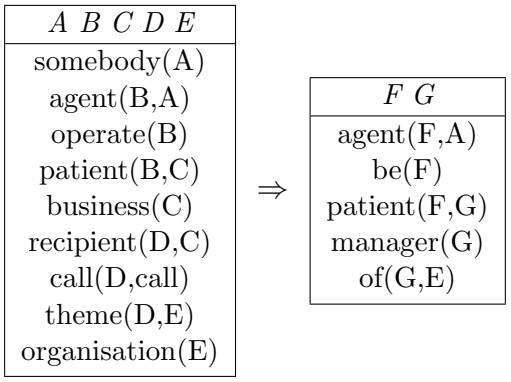


Figure 5.9: Output Axiom for Example 444 with both anchors abstracted into their named-entity categories

To seek balance between these extremes, we abstract the proper names in the axiom according to their named-entity categorisation, such as person, location, organisation, time, and cardinality. Thus the DRS-condition `named(A,van_meeteren,per)` is abstracted to `pred(A,somebody,n,1)`, while `named(E,mental_health_professionals, org)` is abstracted to `pred(E,organisation,n,1)` as demonstrated in Figure 5.9.

Chapter 6

Experiments and Results

6.1 Dataset and Evaluation Measures

We use as dataset the development and test suite from the First, Second, and Third RTE Challenge (Dagan et al., 2006; Bar Haim et al., 2006; Giampiccolo et al., 2007). In these sets, every text-hypothesis pair is further categorised into the task it was designed for, e.g. Information Extraction (IE), Question Answering (QA), Information Retrieval (IR), multi-document summarisation (SUM) or Comparable Documents (CD), Reading Comprehension (RC), Paraphrase Acquisition (PP), and Machine Translation (MT). RTE pairs from the First RTE Challenge covers all the aforementioned tasks, while those from the second and third challenge include only the first four tasks. There are some RTE pairs for which C&C tools and Boxer fail to produce syntactic or semantic analyses. We did not include these pairs in our dataset. Table 6.1 lists the number of RTE pairs we used from each set.

Table 6.1: Number of positive and negative entailment examples in RTE dataset

Challenge	Set	Positive	Negative	Total
First	Development (DEV-1)	282	282	564
	Test (TES-1)	389	390	779
Second	Development (DEV-2)	395	396	791
	Test (TES-2)	389	392	781
Third	Development (DEV-3)	405	381	786
	Test (TES-3)	407	387	794
All		2267	2228	4495

For the evaluation, we opted for leave-one-out cross validation technique. The training set consists of all the positive RTE pairs/examples from the whole dataset leaving one out. This one example is then used as the test set. We generated the axioms from the training set and selected the ones which we considered most useful. The most useful axioms from the training data are then used as the background knowledge when running Nutcracker against the test set. This experiment is done repreatedly until every example in the RTE dataset is used as the test data. The Nutcracker output of an entailment prediction can be of three features:

- word overlap: entailment is predicted by measuring word overlap between T and H,
- simple proof: entailment based on a proof found by the theorem prover,
- complex proof: entailment based on a proof found by the theorem prover with the help of background knowledge resources, be it from our automatically generated axioms, WordNet, or other sources of background knowledge.

So far most of the entailment prediction made by Nutcracker was approximated by the feature of word overlap, while the feature complex proof only has a very small coverage. We expect our axioms to provide knowledge which will improve the number of entailment prediction of feature complex proof. The helpfulness of our axioms is thus measured by the difference in coverage of complex proof found before and after the addition of the axioms to Nutcracker. In addition, we also try to identify incorrect axioms by examining the proofs found for negative entailment pairs.

6.2 Generating Axioms

Recall from Section 5.2 that sometimes there is a mismatching tags between the same lemma in T and H. This might affect axiom abstraction, i.e. if an anchor a is tagged as an organisation in T but a location in H, to which category should we abstract it? Furthermore, it also affects the applicability of the axiom for certain examples. That is, an axiom always expresses properties between two entities X and Y, where X in the antecedent is the same as X in the consequence, this is also the case for Y, and therefore the axiom will not apply for an RTE pair which contains the same lemma with different tags in T and H. For these reasons, we did synchronisation of parts-of-speech and named-entity tags as discussed in Section 5.2. Table 6.2 lists the number of pairs which lemmas are synchronised.

Table 6.2: Number of synchronised pairs on the sets

Set	Synchronisation
DEV-1	78
TES-1	107
DEV-2	104
TES-2	83
DEV-3	112
TES-3	128

Following the method described in Chapter 5, we generated axioms from the training set, choosing only one shortest path between each anchor pair of an example. An RTE pair can produce zero or more axioms. Each axiom has five different types of abstraction (Section 5.4):

- concrete axioms, with the anchors kept the same
- abstracted axioms, with the anchors abstracted into the named entity categories
- axioms with the first or left anchor removed
- axioms with the second or right anchor removed
- axioms with both anchors removed

The total numbers of distinct axioms produced from each set are shown in Table 6.3. The synchronisation proves useful in generating more axioms. For instance, without synchronisation only 328 concrete axioms were found for dataset DEV-1 (we do not present the result for unsynchronised examples here), but with synchronisation 422 concrete axioms were found for the same dataset. Table 6.4 summarises the number of RTE pairs, according to their tasks, from which axioms were extracted. Examples labelled as SUM or CD contain sentences selected from comparable news articles with high lexical overlap, therefore examples of this task tend to produce more axioms. On the other hand, we did not find many anchor pairs for examples marked with IR, hence the low number of axioms produced from examples of this task.

6.3 Useful Axioms

Our method resulted in a considerably large number of axioms which in addition to general paraphrases, also contained noise and implausible implications. Given this size,

Table 6.3: Total number of distinct axioms

Set	Total Axioms				
	Concrete	Abstracted	Left Removed	Right Removed	Both Removed
DEV-1	422	415	402	382	370
TES-1	506	482	460	449	430
DEV-2	459	455	435	434	413
TES-2	448	439	422	419	402
DEV-3	456	448	426	420	399
TES-3	470	460	434	427	402

Table 6.4: Number of axioms generated according to task

Task	Set					
	DEV-1	TES-1	DEV-2	TES-2	DEV-3	TES-3
QA	23	26	50	58	48	49
SUM/CD	30	41	61	43	55	63
IR	8	16	20	30	28	22
IE	22	41	55	62	64	67
RC	30	26	-	-	-	-
MT	18	20	-	-	-	-
PP	28	13	-	-	-	-
Total	159	183	186	193	195	201

the number of axioms triggered on each inference attempt made it a tough work for the theorem prover to find a proof. For this reason, we chose from the collection only the axioms that were considered most useful according to the number of RTE pairs from which the axioms were produced. This is based on the assumption that the same axiom that is generated by many RTE pairs occurs in many different contexts and therefore more likely to provide general knowledge. In this step, we called an axiom useful if it was produced by two or more RTE pairs. Table 6.5 lists the number of useful axioms from the whole collection based on the type of abstraction. From now on until otherwise stated, whenever we refer to axioms we mean these most useful axioms.

Table 6.5: Number of axioms produced by two or more RTE pairs

Type of Abstraction	Number
Concrete	19
Abstracted	23
Left Anchor Removed	29
Right Anchor Removed	30
Both Anchors Removed	41

Table 6.6: Examples of useful axioms and their frequency

Frequency	Axiom
19	X, Y ⇒ X is Y
7	Y, X ⇒ X is Y
4	X's Y ⇒ X has Y
4	X in (location) Y ⇒ X is located in Y
4	X, (somebody) Y ⇒ X's name is Y
3	X, widow of Y ⇒ X is the widow of Y
3	(Yoko Ono) X, widow of (John Lennon) Y ⇒ X is the widow of Y
2	(world) X's (population) Y ⇒ X Y
2	X becomes Y ⇒ X is Y
2	X founds Y ⇒ X is the founder of Y
2	(organisation) X (organisation) (Y) ⇒ X is Y
2	(location) X, (location) Y ⇒ X is located in Y
2	(earthquake) Y hits (coast) X ⇒ Y occurs on X
2	Y including (organisation) X ⇒ X is Y
2	(George W Bush) X's wife, (Laura) Y ⇒ name of X's wife is Y

We manually judged some of the axioms and list those that we considered as good in Table 6.6. The word in parentheses preceeding the variables X or Y denotes the anchor of the axioms, e.g. "(somebody) Y" corresponds to the DRS-condition `pred(Y,somebody,n,1)`, "Yoko Ono (X)" corresponds to `named(X,yoko_ono,per,0)`, etc. If a variable is not preceeded by parentheses containing a word, this means the variable stands for an anchor which is later removed from the axiom during abstraction. We also discovered some bad axioms from the result, i.e. those that we judged as incorrect, implausible, or too general, as shown in Table 6.7.

One of the RTE pairs that produced the axiom in the first row of Table 6.7 is the

Table 6.7: Examples of bad axioms and their frequency

Frequency	Axiom
2	(poll) X is conducted on the seventh of (October) Y ⇒ X is carried on the 6th of Y
2	held (today) X in (Baghdad) ⇒ buried X in Y
2	(civilians) X are killed as a result of an organisation on Y ⇒ X are killed as a result of Y's bombing
2	X's (somebody) Y ⇒ the president of X is Y
2	X of Y ⇒ X is secretary of Y
2	X's Y ⇒ X was born in Y
2	X's Y ⇒ X holds Y
2	X hits Y ⇒ X occurrs on Y

following Example 1175:

RTE-1 Test (1175)	
T:	The opinion poll was conducted on the sixth and seventh of October, and included a cross section of 861 adults with a margin of error estimated at 4%.
H:	The poll was carried out on the 6th and 7th of October.

The problem was in the predicate chosen for the shortest path between anchors *poll* and *October*. In T, the predicate *seventh* was selected, but in H where there were two possible shortest paths between the anchors, our method incorrectly selected *6th* for the path. The axiom $X\ hits\ Y \Rightarrow X\ occurs\ on\ Y$ was also judged as bad due to the over-generalised abstraction. In its original contexts, the axiom was produced with *earthquake* as X and *coast* as Y. After the abstraction, however, some instantiations of these variables, for example with category person, can lead to implausible conclusions.

6.4 Modality Axioms

In creating semantic representations, Boxer provides a detailed analysis including modality, a semantic category which allows speakers to express varying degrees of commitment to a proposition (Saeed, 2003). Consider sentence (6.1) and the DRS output by Boxer depicted in Figure 6.1.

(6.1) "Relative size and the power of the purse are certainly key factors," says Samuel L. Husk, executive director of the Council of Great City Schools.

The clause preceeding the verb *say* is represented in the DRS labelled with x7. This is to say that the clause is assumed to be true according to the knowledge of the speaker or, in this case, in a possible world represented by x7. In this work we assumed that the speakers generally tell the truth, hence the clause is true not only in the possible world x7, but also in the actual world. To accommodate this assumption, we added an axiom as represented in Figure 6.2. We did not, however, use the same assumption for verbs like *believe*, *think*, or *expect*. A total of eight similar axioms were added, involving the verbs *tell*, *reveal*, *add*, and *report*, and some conjunctions like *because*, *although*, and *when*. These shall be referred to as modality axioms.

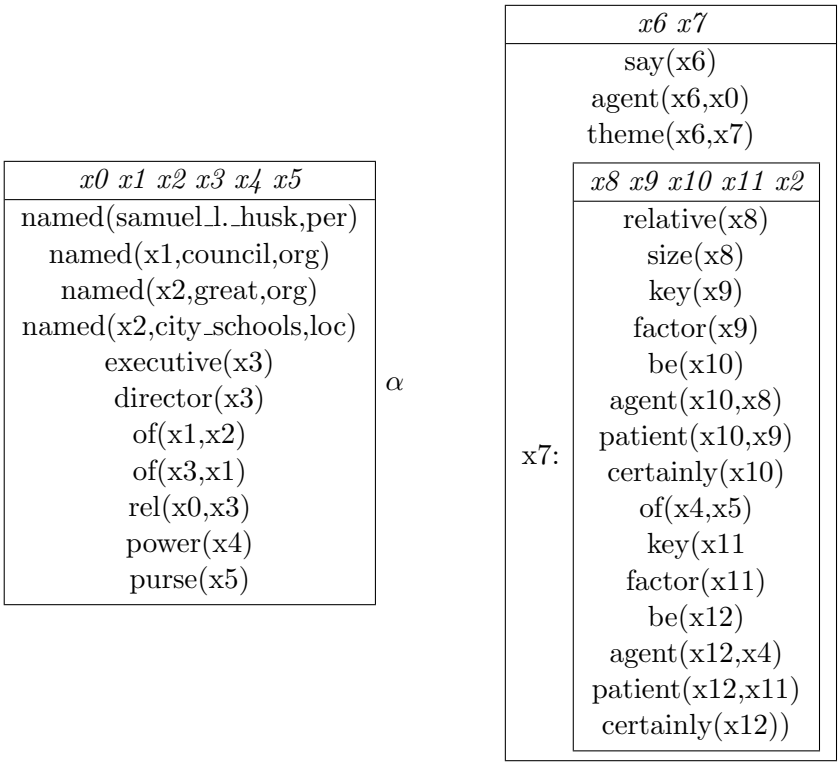


Figure 6.1: DRSs of sentence (6.1)

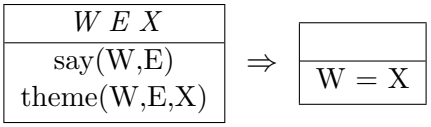


Figure 6.2: Example of modality axiom

6.5 Evaluation

The evaluation measure is the number of complex proof found by the theorem prover that leads to entailment prediction. We conducted the evaluation by running Nutcracker on all positive and negative entailment pairs in the dataset in four experiments using different sources of background knowledge as follows:

- Experiment 1: background knowledge only from WordNet
- Experiment 2: background knowledge from WordNet and manually written modality axioms
- Experiment 3: background knowledge from WordNet and the automatically generated axioms
- Experiment 4: background knowledge from WordNet, modality axioms, and the automatically generated axioms.

One of the challenges in the evaluation is the scarcity of useful axioms in the dataset. In order to address the limitation of training data and provide variability, we adopted leave-one-out cross validation technique, which can be thought of as k-fold cross validation with $k = 2267$, i.e. the number of all positive entailment examples in our dataset. The evaluation on positive examples was therefore conducted as follows:

1. The dataset was split into 2266 pairs for training set and 1 pair for test set
2. From the collection of useful axioms, we selected those produced from the pairs in the training set and ran them on the test set
3. The previous steps were repeated until every pair in the dataset was used as the test data

Evaluation on the negative entailment pairs was done separately. Since all the axioms were produced from the positive examples, we can evaluate all the useful axioms by simply running them on the negative examples. Should we get a complex proof for a negative entailment pair, there is a possibility that the axioms contain incorrect knowledge. All the experiments in the evaluation were run under the following settings:

- theorem prover: Vampire (Riazanov and Voronkov, 2002)

Table 6.8: Number of complex proofs found in the experiments. Correct proofs are produced from positive entailment pairs (column Pos). Incorrect proofs are produced from negative entailment pairs (column Neg).

Set	Number of Complex Proofs							
	Experiment 1		Experiment 2		Experiment 3		Experiment 4	
	Pos	Neg	Pos	Neg	Pos	Neg	Pos	Neg
DEV-1	3	1	4	1	8	1	9	1
TES-1	3	0	4	0	7	2	8	2
DEV-2	1	0	3	1	9	0	11	1
TES-2	3	0	4	0	5	1	6	1
DEV-3	0	0	2	0	5	0	6	0
TES-3	1	0	1	1	7	0	6	1
Total	11	1	18	3	**41**	4	**46**	6

- model builder: Paradox (Claessen and Sörensson, 2003)
- time limit for each inference attempt: 300 seconds

Table 6.8 summarises the result of the four experiments on positive and negative entailment pairs. Experiments with automatically generated axioms result in a significantly higher number of correct entailment prediction based on the complex proof feature. As expected, the precision for this class is high: 0.911 for Experiment 3 and 0.884 for Experiment 4. These experiments especially yield higher recall for complex proof feature compared to that of Experiment 1, in which recall is very low since there are only 11 complex proofs found.

We notice, however, that the addition of many axioms as background knowledge does not always have benefits if they are triggered but do not contribute to finding a proof, as is the case of positive examples in TES-3 where Experiment 3 outperforms Experiment 4. With more axioms triggered in the inference attempt on Experiment 4, the time needed for the theorem prover to find a proof on a particular example exceeded the time limit. Nevertheless, we found that given sufficient amount of time, the theorem prover managed to find a proof for this example.

An example run on pair 611 from the development set of RTE-2 using WordNet and the automatically generated axioms is shown in Figure 6.3. We can see a number of axioms are triggered by the noun *somebody*, hence providing background knowledge when looking the proof.

```
INFO: Background knowledge: 142 axioms
INFO: Text:
The book contains short stories by the famous Bulgarian writer, Nikolai Haitov.
INFO: Hypothesis:
Nikolai Haitov is a writer.
INFO: Annotation:
entailment
...
INFO: paradox found result for t (consistent, domain size: 1)
INFO: paradox found result for h (consistent, domain size: 1)
INFO: paradox found result for th (consistent, domain size: 1)
INFO: paradox found result for tth (informative, domain size: 1)
INFO: paradox found result for kt (consistent, domain size: 1)
INFO: paradox found result for kh (consistent, domain size: 1)
INFO: added axiom 29 triggered by n1somebody
INFO: added axiom 34 triggered by n1somebody
INFO: added axiom 35 triggered by n1somebody
INFO: added axiom 36 triggered by n1somebody
INFO: added axiom 38 triggered by n1somebody
INFO: added axiom 39 triggered by n1somebody
INFO: added axiom 40 triggered by n1somebody
INFO: added axiom 42 triggered by n1somebody
INFO: added axiom 48 triggered by n1somebody
INFO: added axiom 60 triggered by n1somebody
INFO: added axiom 62 triggered by n1somebody
INFO: added axiom 66 triggered by n1somebody
INFO: added axiom 70 triggered by n1somebody
INFO: added axiom 77 triggered by n1somebody
INFO: added axiom 78 triggered by n1somebody
INFO: added axiom 85 triggered by n1somebody
INFO: added axiom 92 triggered by n1somebody
INFO: added axiom 93 triggered by n1somebody
INFO: added axiom 95 triggered by n1somebody
INFO: added axiom 96 triggered by n1somebody
INFO: added axiom 100 triggered by n1somebody
INFO: added axiom 101 triggered by n1somebody
INFO: paradox found result for kth (consistent, domain size: 1)
INFO: vampire found result for ktkth (uninformative, domain size: 0)
INFO: prediction: entailed (complex proof)
```

Figure 6.3: Example of Nutcracker run using WordNet and the automatically generated axioms

6.6 Error Analysis

Cases of missing axioms could happen due to the difference in syntactic or semantic analysis between T and H, as occurred in Example 75 below.

RTE-2 Test (75)

T:	Three days after PeopleSoft bought JD Edwards in June 2003, Oracle began its offer for PeopleSoft.
H:	JD Edwards belongs to PeopleSoft.

While the lemma *PeopleSoft* was tagged as an organisation in T, it was incorrectly analysed as a verb in H. Our synchronisation step only attempted at synchronising lemmas that were analysed as nouns, thus *PeopleSoft* failed to be selected as an anchor. Now consider Example 97:

RTE-2 Test (97)

T:	The world will never forget the epic flight of Charles Lindbergh across the Atlantic from New York to Paris in May 1927, a feat still regarded as one of the greatest in aviation history.
H:	Lindbergh began his flight from New York in 1927.

So far our approach only selected as anchors overlapping lemmas of exact match. In Example 97 where we had `charles_lindbergh` in T and `lindbergh` in H, we did not manage to identify this as an anchor.

In addition, we also discovered cases where our automatically generated axioms contributed to incorrect predictions based on complex proof feature. If a complex proof is found for a negative example, the axioms triggered during the inference attempt must contain incorrect knowledge and therefore we have to identify which one of them is guilty for finding the proof. In order to do this, we investigated the axioms one at a time as follows. Let T be the negative entailment pair for which an incorrect proof is found; and a_1, a_2, ..., a_n be the axioms triggered in the inference of T. Run Nutcracker on T with only a_1 as the axiom. If it finds a proof with only a1 providing the knowledge, then we can conclude that a_1 is incorrect. Otherwise, try with a_2; and so on and so forth until each of the n axioms is observed. From a total of 2228 negative examples, six were found with incorrect proofs. The first case is shown in Example 2084:

RTE-1 Development (2084)

T:	Microsoft Israel was founded in 1989 and became one of the first Microsoft branches outside the USA.
H:	Microsoft was established in 1989.

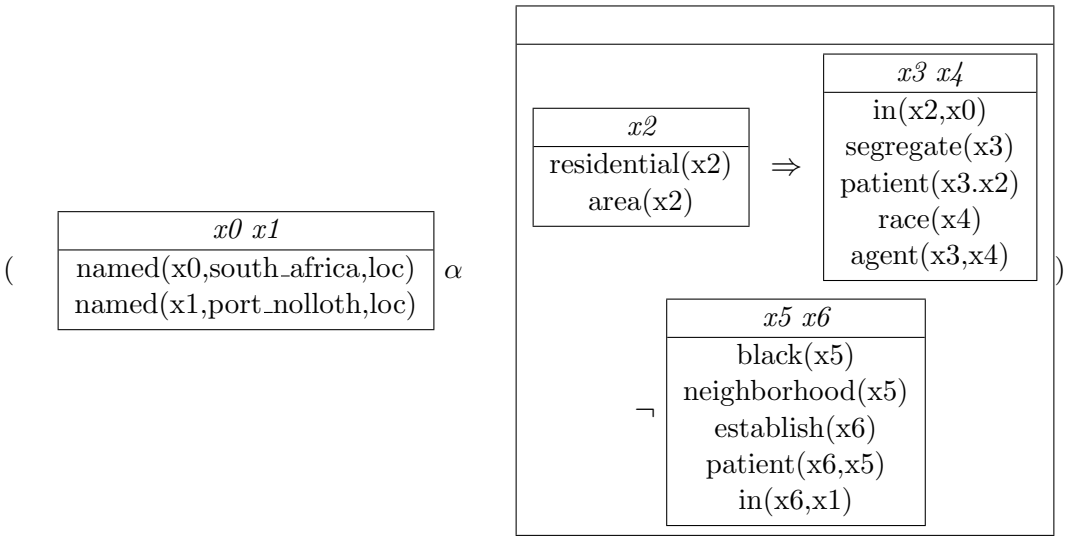


Figure 6.4: Semantic representation of T from Example 1662

In Example 2084, the finding of complex proof was not a result of our automatically generated axioms, but rather because the semantic analysis matched *Microsoft Israel* and *Microsoft* together, supported by WordNet synonymy relation between the verbs *found* and *establish*.

Another case of incorrect proof is found in the following Example 1662:

RTE-1 Test (1662)

T:	All residential areas in South Africa are segregated by race and no black neighborhoods have been established in Port Nolloth.
H:	All residential areas are located in South Africa.

The axiom responsible for the proof in Example 1662 is *X in (location) Y* ⇒ *X is located in Y*. However, we observe that this axiom is applied incorrectly because of an error in the semantic analysis of T, as illustrated in Figure 6.4. In order to represent the clause "All residential areas in South Africa are segregated by race", the DRS-condition `in(x2,x0)` is supposed to be placed in the DRS acting as the antecedent of the implication, but instead Boxer placed this condition in the DRS acting as the consequence.

We also found an incorrect proof for the following example:

RTE-1 Test (2092)

T:	Interestingly, the current Mayor of Berlin, Klaus Wowereit, is openly gay, as is the Mayor of Paris, Bertrand Delanoe.
H:	Klaus Wowereit is the Mayor of Paris.

A B C		*D*
rel(B,A) location(B) of(C,B)	⇒	agent(D,A) be(D) patient(D,C)

Figure 6.5: Axiom triggered during the inference attempt for Example 2092

The axiom that produced the proof for Example 2092 is represented in Figure 6.5. In the inference attempt, A was instantiated to *Klaus Wowereit*, B corresponded to *Berlin*, while C was instantiated to *Mayor*. The consequence of the axiom only states that A is C, i.e. *Klaus Wowereit is the Mayor*, without taking *Paris* into account, thus led the theorem prover to finding a proof. Note that even if the axiom expresses the property of being a Mayor of Paris, the abstraction of *Paris* into location, which is the same as *Berlin*, can still lead to problematic application. Thus we consider this axiom as to contain incorrect knowledge.

The fourth example for which an incorrect proof was found is as follows:

RTE-2 Test (367)

T:	The most ardent advocates of this panacea are US secretary of state Colin Powell, European Union foreign affairs executive Javier Solana, the Israeli foreign minister Shimon Peres, Arafat's close financial adviser Muhammed Rashid and Ariel Sharon's new chef de bureau Dov Weisglass.
H:	Javier Solana is the Israeli foreign minister.

We managed to find five axioms responsible for finding a proof for Example 367; all basically say the same thing if not for the different types of abstraction. Given our most frequently produced axiom (Table 6.6) as represented in Figure 6.6a, in the inference attempt X was instantiated with *Javier Solana* and Y with *Israeli foreign minister*, thus:

Javier Solana, the Israeli foreign minister ⇒ Javier Solana is the Israeli foreign minister.

The fault, however, is not with the axiom, but in the semantic analysis which falsely analysed *Javier Solana* and *the Israeli foreign minister* as an apposition. After eliminating the previous axiom, a complex proof is still found for this example due to the axiom illustrated in Figure 6.6b. Notice that this axiom is a more specific case of the one in 6.6a, with the anchors abstracted instead of removed. The term *leader* was triggered because it was the hypernym of *minister* via WordNet concepts. The three other

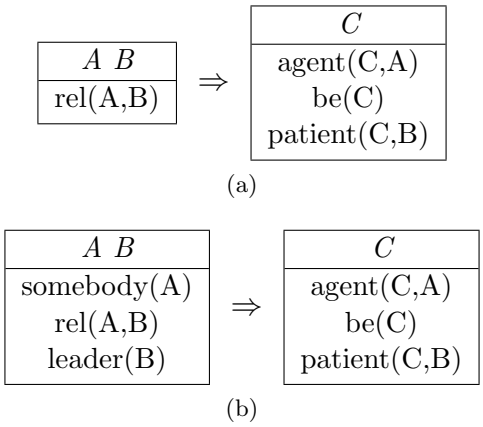


Figure 6.6: Axiom triggered during the inference attempt for Example 367

axioms are simply variants of these two axioms using different abstractions.

The last two incorrect proofs were found for the following examples:

RTE-2 Development (110)	
T:	Drew Walker, NHS Tayside's public health director, said: "It is important to stress that this is not a confirmed case of rabies."
H:	A case of rabies was confirmed.

RTE-3 Test (797)	
T:	Last year the US saw exports of goods and services rise 12.8% to $1.44 trillion, while its imports gained 10.5% to $2.2 trillion, said the Commerce Department.
H:	US exports rose 10.5%.

In Example 110 and Example 797, incorrect proofs were found because of the modality axioms depicted in Figure 6.2. Again, we discovered that the axiom itself did not *cause* the incorrect proof, but rather *allowed* the inference engines to find a proof due to a gap in the semantic analysis.

Chapter 7

Conclusion and Future Work

7.1 Conclusion

We have presented an approach to automatically generating background knowledge axioms for recognising textual entailment. The broad domain of unformalised world knowledge normally makes the task of automatising knowledge extraction a difficult one; nevertheless in this thesis we managed to make a starting point in producing plausible background knowledge axioms from available RTE Challenge corpora. The implementation of deep semantic analysis using Discourse Representation Structures and word overlap as an important feature in deciding anchors allow us to discover meaningful relationships between two entities in the text and the hypothesis.

Our proposed approach manages to produce axioms which, based on manual observation, express valuable knowledge, such as properties of appositions. Moreover, the different types of abstraction specifies the applicability of the axioms on certain cases. In contrast to existing work like DIRT that generates inference rules with low precision and high recall, our method yields axioms with high precision and low recall. Evaluation with Nutcracker RTE system shows that these axioms, together with WordNet and some modality axioms, provide background knowledge for logical inference which helps the theorem prover find a proof for positive entailment pairs.

7.2 Future Work

Much still needs to be done in order to improve our results. For instance, we expected to identify bad axioms based on the proof found for negative examples; this method was not satisfying for the following reasons. First, not many of the bad axioms contribute to finding a proof for negative examples. Second, incorrect proofs do not always indicate bad axioms, since our error analysis shows that some of them are caused by the mistake in semantic analysis. Therefore we need to come up with a better method in filtering out bad axioms. We also made an assumption that the most useful axioms are those produced by more than one RTE pairs, but in fact there are quite a lot of good axioms that do not get selected because they are produced by only one RTE pair. A better method in selecting good axioms would be necessary. Another issue that we need to address is redundancy. There are axioms that are more specific than others, for example *(somebody) X, (leader)* $Y \Rightarrow X$ *is* Y is more specific than X, $Y \Rightarrow X$ *is* Y. The problem lies in the fact that if a proof can be found with the more general axiom alone, then the knowledge expressed by the more specific one will not be useful in the inference. Rather than adding the workload of the theorem prover with these redundancy, it is better to filter out these more specific axioms. Finally, in the future we expect to produce axioms which antecedent and consequence are not only of conjunction of DRS-conditions, but also include other operators like negation, disjunction, and implication.

Bibliography

R. Bar Haim, I. Dagan, B. Dolan, L. Ferro, D. Giampiccolo, B. Magnini, and I. Szpektor. The second pascal recognising textual entailment challenge. 2006.

K. Barker and S. Szpakowicz. Semi-automatic recognition of noun modifier relationships. In *Proceedings of the 17th international conference on Computational linguistics-Volume 1*, pages 96–102. Association for Computational Linguistics, 1998.

R. Barzilay and K.R. McKeown. Extracting paraphrases from a parallel corpus. In *Proceedings of the 39th Annual Meeting on Association for Computational Linguistics*, pages 50–57. Association for Computational Linguistics, 2001.

R. Bhagat, P. Pantel, E. Hovy, and M. Rey. Ledir: An unsupervised algorithm for learning directionality of inference rules. In *Proceedings of EMNLP-CoNLL*, pages 161–170, 2007.

P. Blackburn and J. Bos. *Representation and inference for natural language: A first course in computational semantics*. Center for the Study of Language and Information, 2005.

J. Bos and K. Markert. Recognising textual entailment with logical inference. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pages 628–635. Association for Computational Linguistics, 2005.

Johan Bos. Wide-coverage semantic analysis with boxer. In Johan Bos and Rodolfo Delmonte, editors, *Semantics in Text Processing. STEP 2008 Conference Proceedings*, Research in Computational Semantics, pages 277–286. College Publications, 2008.

K. Claessen and N. Sörensson. New techniques that improve mace-style finite model finding. In *Proceedings of the CADE-19 Workshop: Model Computation-Principles, Algorithms, Applications*, pages 11–27, 2003.

P. Clark and P. Harrison. Recognizing textual entailment with logical inference. In *Proceedings of 2008 Text Analysis Conference (TAC'08), Gaithsburg, Maryland*. Citeseer, 2008.

P. Clark and P. Harrison. Large-scale extraction and use of knowledge from text. In *Proceedings of the fifth international conference on Knowledge capture*, pages 153–160. ACM, 2009.

S. Clark and J.R. Curran. Parsing the wsj using ccg and log-linear models. In *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, page 103. Association for Computational Linguistics, 2004.

I. Dagan, O. Glickman, and B. Magnini. The pascal recognising textual entailment challenge. *Machine Learning Challenges. Evaluating Predictive Uncertainty, Visual Object Classification, and Recognising Tectual Entailment*, pages 177–190, 2006.

G. Dinu and R. Wang. Inference rules and their application to recognizing textual entailment. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, pages 211–219. Association for Computational Linguistics, 2009.

C. Fellbaum. Wordnet. *Theory and Applications of Ontology: Computer Applications*, pages 231–243, 2010.

D. Giampiccolo, B. Magnini, I. Dagan, and B. Dolan. The third pascal recognizing textual entailment challenge. In *Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing*, pages 1–9. Association for Computational Linguistics, 2007.

J. Gordon and L.K. Schubert. Discovering commonsense entailment rules implicit in sentences. In *Proceedings of the TextInfer 2011 Workshop on Textual Entailment*, pages 59–63. Association for Computational Linguistics, 2011.

M.A. Hearst. Automatic acquisition of hyponyms from large text corpora. In *Proceedings of the 14th conference on Computational linguistics-Volume 2*, pages 539–545. Association for Computational Linguistics, 1992.

A. Ibrahim, B. Katz, and J. Lin. Extracting structural paraphrases from aligned monolingual corpora. In *Proceedings of the second international workshop on Paraphrasing-Volume 16*, pages 57–64. Association for Computational Linguistics, 2003.

Daniel Jurafsky and James H. Martin. *Speech and Language Processing (2nd Edition) (Prentice Hall Series in Artificial Intelligence)*. Prentice Hall, 2 edition, 2008. ISBN 0131873210.

H. Kamp. A theory of truth and semantic representation. *Formal Semantics*, pages 189–222, 1981.

H. Kamp and U. Reyle. From discourse to logic. introduction to modeltheoretic semantics of natural language, formal language and discourse representation, 1993.

S. Kim and T. Baldwin. Automatic interpretation of noun compounds using wordnet similarity. *Natural Language Processing–IJCNLP 2005*, pages 945–956, 2005.

S.N. Kim and T. Baldwin. Interpreting semantic relations in noun compounds via verb semantics. In *Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions*, pages 491–498, 2006.

D. Lin and P. Pantel. Discovery of inference rules for question-answering. *Natural Language Engineering*, 7(4):343–360, 2001.

E. Marsi, E. Krahmer, and W. Bosma. Dependency-based paraphrasing for recognizing textual entailment. In *Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing*, pages 83–88. Association for Computational Linguistics, 2007.

EC Marsi, EJ Krahmer, WE Bosma, and M. Theune. Normalized alignment of dependency trees for detecting textual entailment. 2006.

B. Pang, K. Knight, and D. Marcu. Syntax-based alignment of multiple translations: Extracting paraphrases and generating new sentences. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*, pages 102–109. Association for Computational Linguistics, 2003.

P. Pantel and D. Lin. Discovering word senses from text. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 613–619. ACM, 2002.

P. Pantel, R. Bhagat, B. Coppola, T. Chklovski, and E. Hovy. Isp: Learning inferential selectional preferences. In *Proceedings of Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-HLT)*, pages 564–571, 2007.

T. Parsons. *Events in the Semantics of English.* MIT Pr., 1990.

T. Pedersen, S. Patwardhan, and J. Michelizzi. Wordnet:: Similarity: measuring the relatedness of concepts. In *Demonstration Papers at HLT-NAACL 2004*, pages 38–41. Association for Computational Linguistics, 2004.

A. Peñas and E. Ovchinnikova. Unsupervised acquisition of axioms to paraphrase noun compounds and genitives. *Computational Linguistics and Intelligent Text Processing*, pages 388–401, 2012.

P.S. Resnik. Selection and information: a class-based approach to lexical relationships. *IRCS Technical Reports Series*, page 200, 1993.

A. Riazanov and A. Voronkov. The design and implementation of vampire. *AI communications*, 15(2):91–110, 2002.

S. Russell and P. Norvig. Artificial intelligence: A modern approach. 2009.

J.I. Saeed. Semantics (introducing linguistics). *Semantics-introducing linguistics*, 2003.

L. Schubert. Can we derive general world knowledge from texts? In *Proceedings of the second international conference on Human Language Technology Research*, pages 94–97. Morgan Kaufmann Publishers Inc., 2002.

T. Tammet. Gandalf. *Journal of Automated Reasoning*, 18(2):199–204, 1997.

M. Tatu and D. Moldovan. A logic-based semantic approach to recognizing textual entailment. In *Proceedings of the COLING/ACL on Main conference poster sessions*, pages 819–826. Association for Computational Linguistics, 2006.

E.M. Voorhees. The trec-8 question answering track report. In *Proceedings of TREC*, volume 8, pages 77–82, 1999.