SAARLAND UNIVERSITY

Textual Entailment Recognition

A Data-Driven Approach

Rui Wang

A Thesis presented for the degree of
Master of Science in Language Science and Technology

Saarbrücken, 17.09.2007

Author:
Rui Wang
LT-Lab, DFKI
Stuhlsatzenhausweg 3
66123 Saarbrücken
Germany
wang.rui@dfki.de

Thesis Advisors:
PD Dr. Günter Neumann
Prof. Dr. Hans Uszkoreit

Submitted to:
Department of Computational Linguistics & Phonetics
Universität des Saarlandes

Abstract

In this thesis, we present our work on *Recognizing Textual Entailment* (RTE). On the broad view, we have utilized three approaches: the main approach and two backup strategies. In the main approach, we have proposed a novel feature representation extracted from the dependency structure and then applied kernel-based machine learning techniques based on the entailment patterns. One backup strategy is based on local dependency relations and the other one is a simple bag-of-words method. In practice, we have taken part in the RTE-3 Challenge using our system and achieved 66.9% of accuracy on the test set, which is among the top-5 of all the results from 26 research groups. Further experiments have been performed on the RTE-2 data set (63.6% of accuracy, would score the 4th rank) and other extra data we have collected. Notice that we have only used the output from the dependency parsers without any external knowledge bases or other resources. The whole RTE-centered framework we have established has not only explored approaches tackling the problem itself, but has also tested the RTE system on other natural language processing applications, such as binary relation extraction and answer validation. In addition, the graphic user interface can also assist the annotators and developers.

Some parts of Chapter III, Chapter IV, and experiments on the RTE-2 data set and the extra data in Chapter V have been published in *Proceedings of the Twenty-Second Conference on Artificial Intelligence* (AAAI-07) (Wang and Neumann, 2007a); some parts of Chapter III, Chapter IV, and our participation of the RTE-3 Challenge in Chapter V have been published in *Proceedings of ACL-PASCAL Workshop on Textual Entailment and Paraphrasing* (Wang and Neumann, 2007b); some parts of Chapter III, Chapter IV, and the main parts of Chapter VI will be published in *Working Notes of the AVE task of CLEF2007* (Wang and Neumann, 2007c).

Declaration

The work in this thesis is based on research carried out at the LT-Lab Research Group, in the Language Technology Lab, Saarbrücken, Germany. No part of this thesis has been submitted elsewhere for any other degree or qualification and it is all my own work unless referenced to the contrary in the text.

Acknowledgements

Starting to write this part suggests the end of this thesis. Though writing it down only occupied one month time or so, this thesis does contain one-year research on the topic of *textual entailment*, about which none of my friends outside the community knew before my explanations. Anyway, I am proud of this work, so I would like to show the contributions of people other than me here.

My parents, **WANG Dingzhu** and **ZHU Qunhuan**, deserve the first place, since without them I am not sure whether I would still be here writing the acknowledgements. My father has taught me how to THINK independently; and my mother has told me how to LIVE enjoyably. They are such great parents who have given me a joyful childhood, a warm family, and "an extremely robust backup"! "**爸爸妈妈，谢谢你们！**"

A huge **DANKE** to my supervisor **PD Dr. Günter Neumann**, whose name would have been put in the author list by me if it were allowed. He has contributed A LOT to this thesis. All the brainstorming, discussions, debates, etc. will be a good memory of mine, not to mention those interesting topics, clever ideas, funny tricks, and silly mistakes. I would like to cite some of his famous words here: "*I cannot control you, but I recommend you to work!*", "*My job is to keep you busy.*", "*Yes, I know you're busy now. But could you please do this ... task?*", ... more details cf. *Neumann and Wang (to appear)*.

Another huge **DANKE** to my advisor during this whole master time, **Prof. Dr. Hans Uszkoreit**, who has given me a lot of help, not only for the thesis, but also the whole two years abroad. Though we did not meet frequently, I can still remember those heuristic questions and discussions, which always lead me into a deeper thought of the problems. Apart from the research part, he has taken a great care of my life here as well, helping me with finding the Hiwi job in DFKI, advising me on my future plan, inviting me to his Christmas party, etc.

I would like to say "**非常感谢！**" to my former supervisor in China, **Prof. Dr. YAO Tianfang**. It is he that recommended me to come here to join in this nice master program; and it is he that gave me much advice on how to do scientific writing. I want to say "**Danke!**" to **Prof. Dr. Manfred Pinkal** for giving us an interesting seminar on exactly the same topic as this thesis, which I enjoyed a lot. A "**Danke!**" to **Aljoscha Burchardt** and a "**Grazie!**" to **Marco Pennacchiotti** for all the discussions and jokes on RTE.

A sweet "**Ευχαριστίες!**" to **Konstantina Garoufi**. She is such a great friend with whom I can share both happiness and sadness. Inside the thesis, we have discovered a lot of nice examples of RTE; outside the thesis, chocolates, dinners, and ice-creams are the main topics. A big "**谢谢！**" to **ZHANG Yajing** for all the coffee breaks (especially the delicious cookies) we had together. A special "**感谢**" to **CHENG Xiwen** for caring about my living during the thesis time. "**¡Gracias!**" to my "boss-mate" **Alejandro Figueroa**, who always has innovative ideas and passion for work.

Two big "**Xiexie!**"s to **Trevor Benjamin** and **Lisa Macwan**, who have helped me with my poor English. As the first readers of my thesis, they have made great efforts to understand what I was talking about. I am really sorry for spoiling their holidays and weekends, since even at the last moment, they were still proofreading some of the chapters! Two "**多谢！**"s to **ZHANG Yi** and **SHEN Dan**, a "**Thank you!**" to **Jennifer Moore**, and a "**Dank je!**" to **Antske Fokkens** for their reading my previous conference papers, which form up the main parts of this thesis.

Life has things other than the thesis. Many thanks to my jogging friends: four "**Danke!**"s to **Michael Roth**, **Christian Sänger**, **Teresa Herrmann**, and **Anna Mündelein**; a "**Спасибо!**" to **Olga Kukina**, a "**Shukria!**" to **Danish Nadeem**, and a "**谢谢！**" to **DING Wei**. More thanks to my dinner friends: a "**Duozai！**" to **Shirley Siu**, two "**谢谢！**"s to **YE Min** and **FU Yu**. A special "**Merci!**" to our schwenker chef, **Pierre Lison**, for brining us the evil barbeque parties.

Last but not least, I would like to thank all the friends I met and all the audience of my presentations in AAAI 2007, ACL-PASCAL Workshop (RTE-3), and TaCoS 2007. If you have not seen your name, please tell me and I will thank you face to face.

This work is partially supported by a research grant from BMBF to the DFKI project HyLaP (FKZ: 01 IW F02) and the EC-funded project QALL-ME. I would like to thank all the colleagues in DFKI for contributing their work relevant to this thesis.

Table of Contents

List of Figures

List of Tables

Chapter I

Introduction

In this chapter, we will briefly give an overview of the work done in this thesis. Our task, *Recognizing Textual Entailment* (RTE), is motivated from both the demand for a more accurate semantic model of natural languages, and from active application needs like *Information Extraction* (IE), *Question Answering* (QA), etc. The main contributions include 1) achieving textual entailment with a high accuracy based on dependency parsing and *Machine Learning* (ML) techniques, 2) building an extensible experimental architecture for both applications and further research, 3) experimenting in a large scale evaluation of our system, and 4) utilizing our RTE system for concrete *Natural Language Processing* (NLP) applications. After this, we will give the organization of the rest of the chapters.

1.1 Overview

Although in logics, entailment has formal definition, in pragmatics, this concept describes a simply a particular relationship between two sentences or sets of sentences: if the first is true, the other is also true. The following examples show this,

S1: A horse is an animal.
S2: A horse has four legs and a tail.
S3: A stallion is male.
S4: A stallion is at least one year old.

Example 1

If we define a horse using the first two sentences and a stallion using the rest and "*a stallion is a horse*", then every stallion must satisfy,

S5: A stallion is an animal.
S6: A stallion has four legs and a tail.

Example 1 (continued)

Notice that this does not include the vice versa supposition, that every horse is male or at least one year old, which implies entailment relationship is **directional**.

The RTE task (Dagan and Glickman, 2004) is a concrete NLP task based on this concept,

whose definition is also a relationship between two plain texts, *Text* (**T**) and *Hypothesis* (**H**). If the meaning of **H** can be inferred from the meaning of **T**, we say **T** entails **H**; otherwise, **T** does not entail **H**. This task can be viewed as a binary classification task or a probabilistic function mapping the pair **T**-**H** to a value between 0 (not entail) and 1 (fully entail). Before coming back to the formal definition (1.2.3), we will answer the question "*Why do we need Textual Entailment?*" in the first place.

1.2 Motivations

The reasons for realizing textual entailment computationally can fall into the following two categories: 1) providing the computer with the ability of doing inferences in order to achieve a better understanding of natural languages, or 2) using RTE for other NLP tasks to improve the current performances. Zanzotto and Moschitti (2006) have expressed the similar opinions. Both of the two motivations can be observed in concrete applications.

1.2.1 Internal Goals

On the one hand, natural languages are full of ambiguity, which is not strictly logical; and on the other hand, variations of the same meaning are also necessary to make the languages live. The corresponding application for the first aspect is *Word Sense Disambiguation* (WSD), and the application for the second aspect is *Paraphrase Acquisition*. It has been shown that RTE can improve both of them, which is just the internal motivation of RTE to investigate languages.

When we encounter the following two sentences,

*S1: John was walking along the **bank** of the Saar River.*
*S2: Mary wants to open a new account in that **bank**.*

Example 2

We can infer from *S1* in Example 2 that the "*bank*" has the meaning of "*the slope immediately bordering a stream course along which the water normally runs*"[1], and in *S2* the "*bank*" means "*an institution for receiving, lending, exchanging, and safeguarding money and, in some cases, issuing notes and transacting other financial business*"[2].

On the contrary, suppose we have some vague meaning in mind that *John feels that Mary is*

[1] http://dictionary.reference.com/browse/bank
[2] The same as the above.

nice. Based on this, we can have several expressions,

S1: John likes Mary.
S2: John likes Mary very much.
S3: John loves Mary.
S4: John knows Mary very well.
S5: John wants to make friends with Mary.
…

Example 3

Observing these sentences in Example 3, we may find two kinds of relationships between each pair of sentences: 1) entailment relation and 2) paraphrasing relation. For instance, *S2* entails *S1*, but not the opposite, which is a directional relation; *S2* and *S3* more or less express the same meaning (if we consider "*love*" means "*like very much*"), and have the bidirectional paraphrasing relation. Notice that the paraphrasing can be substituted by entailment relations in both directions, i.e. *S2* entails *S3* and *S3* entails *S2* as well.

Taking a closer look at the entailment relationship, we may find several different cases of entailment. The case of *S2* entailing *S1* is a *syntactic* entailment; *S3* entails *S1* is a *semantic* entailment, because lexical semantics of "*love*" and "*like*" are needed; Cases such as S1 entails S4 or S5 entails S1 is *implicature*, since we may need some background knowledge or to perform some inferences to acquire the relationship.

One important point that should be mentioned here is the difference between the strict entailment and RTE. The classical definition of entailment is given by Chierchia and McConnell-Ginet (2000), and is *a text t entails a hypothesis h if h is true in* ***every*** *circumstance (possible world) in which t is true*. However, in real NLP applications, this is not easy to achieve. Instead, we lower the standard into *t entails h if humans agree that most likely h is true if t is given*, such as in the following example,

S1: John is devouring his dinner.
S2: John was starving to death.

Example 4

According to the strict entailment definition, in Example 4, *S2* cannot be entailed by *S1*. There could be other reasons why John devours his dinner, not necessarily that he was starving. For instance, he has to finish his dinner quickly in order to catch the next bus to the university. However, **most likely**, he was starving to death, thus, *S1* entails *S2*.

1.2.2 External Goals

Apart from the better understanding of natural languages, RTE is also designed to discover a generic solution for several current NLP applications (Dagan and Glickman, 2004), including *Information Extraction* (IE), *Question Answering* (QA), *Information Retrieval* (IR), *Summarization* (SUM), *Paraphrase Acquisition*, etc. We would like to discover an inner connection among all these applications and take a united view of them.

Among these tasks, some of them can benefit directly from the success of RTE and some others indirectly. For example, one of the subtasks of IE, *Relation Extraction*, can be transformed into a RTE problem. Given a text and a relation between *Named Entities* (NEs), whether this text contains this relation or not is just the same as whether the entailment relation exists between the text and one hypothesis constructed from the relation and corresponding NEs. For instance, the text is:

S1: Wei, who is born in a small town of China, behaves well in this university.

Example 5

If the given relation is the "*birthplace*" relation, and the NE recognizer correctly knows "*Wei*" is a person and "*China*" is a location, the task will be changed into judging the entailment relation between the *S1* in Example 5 (as **T**) and the following sentence (as **H**):

S2: Wei is born in China.

Example 5 (continued)

Since the entailment relation is true, the extraction of birthplace relation in this text has finished at the same time: "*birthplace<Wei, China>*".

Answer Validation plays an important part in QA, which tests whether the extracted answer is correct to the given question according to the relevant document containing this answer. The document, or context of the answer, forms the **T**, and the question together with the answer forms the **H**. For example,

Question: Who is the president of USA?
Document: George W. Bush, president of the United States, met with British Prime Minister in London last Saturday. They...

Example 6

The correct answer should be "*George W. Bush*". The combination of the question and the

answer will be:

Hypothesis: George W. Bush is the president of USA.

Example 6 (continued)

If the entailment relation exists between the document (i.e. **T**) and the hypothesis, the answer will be validated; otherwise, the answer may be wrong.

Recall that the entailment relation is directional. If the relation between the two parts becomes bidirectional, actually, they will be paraphrased, like the following:

S1: John is a good student.
S2: John behaves well at school.

Example 7

In Example 7, since *S1* entails *S2* and *S2* entails *S1*, they are paraphrases to each other. As long as we obtain paraphrases, it will assist *Natural Language Generation* (NLG) by making language variations or help *Machine Translation* (MT) by checking language properness. It can also be applied in IR for query extension and in SUM for avoiding redundancy. Furthermore, to some extent, it represents semantic equivalence between sentences or paragraphs.

1.2.3 Descriptions of RTE

Through the above examples, RTE shows its ability of investigating natural languages in depth and tackling concrete NLP tasks in a generic way. This section will describe the RTE task in detail based on these observations.

The RTE task was proposed by Dagan et al. (2006) and refined by Bar-Haim et al. (2006). It is defined as *recognizing, given two text fragments, whether the meaning of one text can be inferred (entailed) from the other*. The entailment relationship is a directional one from *Text* – **T** to *Hypothesis* – **H**.

We can use the following function to represent the task,

$$f: < T, H > \rightarrow \{YES, NO\}$$

Or more precisely,

$$f(< T, H >) = \begin{cases} YES, & T\ entails\ H \\ NO, & T\ does\ not\ entail\ H \end{cases}$$

The input of the function is an ordered pair of two texts, normally **T** contains one or more sentences and **H** contains one sentence.

The task can be viewed as a semantic judgment simulating human understanding or a binary classification task from the machine learning point of view. During this process, background knowledge is allowed to add in various ways. The upper bound for this is that ***H*** *cannot be entailed solely by background knowledge*. The RTE-2 Challenge has listed some criteria for preparing the data sets from real systems (Bar-Haim et al., 2006),

- *Entailment is a directional relation.*
- *The hypothesis must be fully entailed by the text.*
- *Cases in which inference is very probable (but not completely certain) are judged as YES.*
- *Our definition of entailment allows presupposition of common knowledge.*

The following examples are collected according to the criteria above,

Dataset=RTE2-dev	*Id=12*	*Task=IE*	*Entailment=NO*
Text: He met U.S. President, George W. Bush, in Washington and British Prime Minister, Tony Blair, in London.			
Hypothesis: Washington is part of London.			

Example 8

Dataset=RTE2-dev	*Id=110*	*Task=IR*	*Entailment=NO*
Text: Drew Walker, NHS Tayside's public health director, said: "It is important to stress that this is not a confirmed case of rabies."			
Hypothesis: A case of rabies was confirmed.			

Example 9

Dataset=RTE2-dev	*Id=436*	*Task=QA*	*Entailment=YES*
Text: Edward VIII shocked the world in 1936 when he gave up his throne to marry an American divorcee, Wallis Simpson.			
Hypothesis: King Edward VIII abdictated in 1936.			

Example 10

Dataset=RTE2-dev	*Id=783*	*Task=SUM*	*Entailment=YES*
Text: Bowles will take the title HRH Duchess of Cornwall, changing it to the Princess Consort, when Charles becomes King.			
Hypothesis: Bowles will take the title of Princess Consort.			

Example 11

Each **T-H** pair contains a *Text* and a *Hypothesis*, and four attributes[3]. Where the *Dataset* indicates where this example belongs to; *Id* is the index number in that dataset; *Task* specifies the original source of this **T-H** pair; *Entailment* is the answer to this pair, which will not appear in the unannotated data.

1.3 Contributions

The main contributions of the thesis are the followings:

Approaches

We have developed an approach using a novel sentence representation extracted from the dependency structure and then applying subsequence kernel methods for machine learning. Two backup strategies have also been used to deal with those cases that cannot be covered by our main approach. **The combination of different approaches** shows advantages in the experiment results.

Performances

We have participated in the RTE-3 Challenge and achieved **66.9%** of accuracy on the RTE-3 Test data, which is among the top-5 of all the submissions from 26 groups. The performance of our method on the RTE-2 Test data is **63.6%** of accuracy, which would score the 4th place of the RTE-2 Challenge. Since we have only used the output from the dependency parser(s) with no external knowledge bases or extra training data, the results have

[3] The RTE-3 Data Set has one more attribute called *Length*, which can be either "*long*" or "*short*", specifying the length of the whole pair.

set up quite a high starting point for further research.

Applications

We applied our RTE system to two NLP applications, binary relation extraction and answer validation. For the second application, we have utilized our system on the data from the *Answer Validation Exercise* (AVE) task, and have **outperformed all the results in English** of last year's submissions.

The System

The architecture of the whole system ***TERA*** (Textual Entailment Recognition and Applications) is both **flexible** to apply new modules in and **generic** for all RTE-based applications. We have also developed a *Graphic User Interface* (GUI) to perform experiments more conveniently.

The Published Works

The main approaches and experiments on the RTE-2 data were published by Wang and Neumann (2007a); the participation of the RTE-3 Challenge and extended experiments were published by Wang and Neumann (2007b); and the participation of the *Answer Validation Exercise* of the *Cross Language Evaluation Forum* 2007 (AVE@CLEF2007) was published by Wang and Neumann (2007c).

1.4 Organization of the Thesis

In Chapter II, we will present related works done by other research groups in the field of RTE. We will discuss in detail on the techniques and linguistic representations applied, features selected and evaluated, and resources and corpora used as well. After each aspect, our approach will also be compared with the others.

In Chapter III, we will elaborate our approaches to RTE in turn. Roughly speaking, our approaches have different domains of locality: a *Bag-of-Words* (BoW) method, an approach based on local dependency relations, and a *Subsequence Kernel* method capturing long distance relations. Although RTE is a heavily semantics-based task, we will mainly focus on bridging the gap between **T** and **H** using only dependency parsing in order to maintain the robustness. After the extended coverage gained through this analysis, external knowledge

bases of the additional lexical semantics (i.e. WordNet) or inference rules could be considered for future work.

In Chapter IV, after a brief introduction of data collection and the tools and techniques used, we will show the implementation of our experimental system by emphasizing the combination of different operators for different tasks. On a large scale, our system consists of the following processing phases: BoW similarity calculation, preprocessing (parsing the plain texts), triple similarity calculation, *Topic Word* (TW) pair detection, *Tree Skeleton* (TS) extraction, *Spine* generalization and merging, *Entailment Pattern* (EP) generation, and kernel-based machine learning. As well as this, we have also developed a GUI for both human annotation and experiment design.

In Chapter V, we will set up several experiments to evaluate our system and discuss the results with concrete examples. Starting with data collection and experimental settings, we will then compare the effects of applying different dependency parsers and approaches of different domains of locality, followed by discussion of each. The comparison with other groups will also be given, regarding both the results and the techniques applied.

In Chapter VI, we apply our RTE system to two concrete applications: binary relation extraction and answer validation. Both of them can achieve quite satisfying results using the RTE techniques. This helps us both to evaluate our system in a better way and to explore potential applications for it.

In Chapter VII, we summarize and discuss the basic results of the thesis and outline some important future directions.

Chapter II

Current Approaches

In this chapter, we will give an introduction of current approaches for RTE. Firstly, we will do an overview, mainly focusing on the RTE-2 Challenge (Bar-Haim et al., 2006), also mentioning some papers from RTE-1 (Ido Dagan et al., 2006). Then, we will elaborate on these according to different techniques and different representations applied, followed by some available resources used. In the summary for each subchapter, we will do a comparison on all the approaches discussed. Afterwards, a brief description of new trends in the recent RTE-3 Challenge (Giampiccolo et al., 2007) will be presented and then the summarization of this chapter.

2.1 Overview

Currently the approaches people apply to the RTE task can be viewed in several ways: a large group of people focus on *Machine Learning* (ML) methods and feature selection, either intra-pair or cross-pair; representations at various levels of *Natural Language Processing* (NLP), e.g. syntax, semantics, are considered; another fashion is to transform natural language texts into logical forms and perform inferences on them; nearly all the methods/systems can be assisted by external knowledge bases, e.g. WordNet[4] (Miller, 1995), FrameNet[5] (Baker et al., 1998). Some research groups concentrate on one of the dimensions, while many others try different combinations of the different techniques and resources.

The overview paper of RTE-2 (Bar-Haim et al., 2006) has a table (Table 2 in that paper) showing both the results and different approaches or resources used by all the participants. We will compare all the techniques applied in detail by breaking down the large table into small ones focusing on different aspects.

2.2 Machine Learning and Feature Selection

Almost all the people in this field have applied some ML methods. More often, they extract features from different representations and feed them into a classifier, e.g. *Decision Tree* (DT), *Naïve Bayes* (NB), *Support Vector Machine* (SVM), etc. The results also vary not only

[4] http://wordnet.princeton.edu/
[5] http://framenet.icsi.berkeley.edu/

according to the feature and classifier selection, but also training data. Roughly speaking, they can be classified into two groups: one is intra-pair feature-based learning; the other is cross-pair feature-based learning. Since RTE is a task to test the existence of the entailment relationship between two text fragments (i.e. *Text* – **T** and *Hypothesis* – **H**), traditional feature space is based on the relation between **T** and **H** (i.e. intra-pair features), however, others also try to discover features between **T**-**H** pairs (i.e. cross-pair features). Several learning methods are applied by different groups. DT and SVM are the most popular ones. In the rest of this subchapter, we will see the features and ML methods in turn, and in the next subchapter (2.3), we will go into details about the different linguistic representations.

2.2.1 Intra-pair Features

Intra-pair Features here mean the features extracted from comparing **T** with **H**. Intuitively, if we view the sentences as groups of tokens, overlapping tokens is a good indicator. Some particular linguistic phenomena are also helpful, like negation words, temporal expressions. Furthermore, features can be extracted based on syntactic structures or semantic representations, or even logical forms.

If we treat the sentence as a bag of words (BoW), the absolute number of overlapping words between **T** and **H**, or the ratio of the absolute number to the total number of words in **T** or **H**, could be considered as features (Adams, 2006; Bos and Markert, 2006; Hickl et al., 2006; Inkpen et al., 2006; Kozareva and Montoyo, 2006; Newman et al., 2006; Nielsen et al., 2006; Schilder and McInnes, 2006; Vanderwende et al., 2006). Bos and Markert (2006) combine a shallow method and a method based on logical inference, the former of which is mainly based on overlapping words. Hickl et al. (2006) uses abundant features at various processing levels. Matching between words in **T** and **H** is detected and helps the alignment classifier in the later stage. Inkpen et al. (2006) includes features like the number of stop words in common, content words in common, nouns and verbs in common, skip bigrams (pair of words in sentence order that allow arbitrary gaps) in common, skip bigrams containing only verbs and nouns in common, etc, and most of the features are in both absolute and normalized form. Actually, her experiments show the importance of these lexical features compared to with features from other deep analyses. Kozareva and Montoyo (2006) check the *Longest Common Subsequence* between **T** and **H**, which is n-gram overlapping.

Newman et al. (2006) proposes two systems, the first of which utilizes several similarity metrics, including the ROUGE (Recall-Oriented Understudy for Gisting Evaluation) (Lin and Hovy, 2004), Cosine, and other WordNet-based similarities for nouns. For verbs, the

VerbOcean semantic network (Chklovski and Pantel, 2004; Chklovski and Pantel, 2005) is applied. In fact, most of the groups using lexical features apply similarities between words based on WordNet or other lexical resources (Adams, 2006; Inkpen et al., 2006; Newman et al., 2006; Schilder and McInnes, 2006; etc). According to Inkpen et al. (2006)'s experiments, using WordNet doesn't improve the final results much. Nielsen et al. (2006) utilizes extra corpora (e.g. English Gigaword (Graff, 2003), the Reuters corpus (Lewis et al., 2004)) to count the document co-occurrence. We will see these resources in detail in (2.4).

Vanderwende et al. (2006) uses word alignment as the main system's backup strategy, which includes exact match between words in **T** and **H**, and linguistic cues from lexical resources as well. In our system, we also implement a baseline system using BoW method, which applies some partial match at substring level instead of using external lexical resources. Since the BoW methods have proved to be simple but robust baseline systems (Bos and Markert, 2006; Newman et al., 2006; Vanderwende et al., 2006), we take our BoW baseline system as one of our backup strategies as well.

Adams (2006) uses *Lexical Editing Distance* as a feature. Basically, it counts the number of unmapped tokens after alignment and then scales to the length of **H**, which is the *insertion* operator from the editing perspective. Similarly, according to Kouylekov and Magnini (2006), **T** entails **H** if there is a sequence of transformations applied to **T** such that **H** can be obtained with an overall cost below a certain threshold. The difference is that they calculate the editing distance based on dependency trees, which contains *insertion*, *deletion*, and *substitution*. They also report that the best settings correspond to the substitution and deletion functions are given in (Kouleykov and Magnini, 2005). The cost of deletion is always 0 and substitution is 0, if two words are similar enough according to WordNet, and infinite in all other cases. Schilder and McInnes (2006) use their *Tree Editing Distance* as one feature and apply another approximate tree similarity metric proposed by (Augsten et al., 2005). For the RTE task, usually **T** is longer than **H**, and the former contains noisy information which is not so relevant to the entailment relationship detection (maybe this is the reason why deletion has the cost 0). Therefore, in our method, we extract a part of the dependency tree as our feature space instead of directly taking the whole tree into consideration. The tree representations will be discussed in detail in 2.3.2 and 2.3.3.

As long as the direct matching between words/tokens, some linguistic features are considered as well. Negation is widely used (De Marneffe et al., 2006a; Hickl et al., 2006; Inkpen et al., 2006; Kozareva and Montoyo, 2006; Newman et al., 2006; Vanderwende et al., 2006), since in some cases, it will reverse the result. Kozareva and Montoyo (2006) checks

whether **T** or **H** contains negations. De Marneffe et al. (2006) check the existence of simple negation words (e.g. *not*), downward-monotone quantifiers (e.g. *no*, *few*), restricting prepositions (e.g. *without*, *except*), and superlatives (e.g. *tallest*) in both **T** and **H**. Inkpen et al. (2006) and Vanderwende et al. (2006) both use negations as a mismatch feature. In particular, the latter group implements a system which can quite precisely (81% of accuracy) predict false entailment cases. Their system, MENT (Microsoft Entailment), takes as its premise that it is easier for a syntactic system to predict false entailments, following the observation in Vanderwende and Dolan (2005). In addition, Newman et al. (2006) utilize VerbOcean taxonomy to find out verb negations.

Antonym (or polarity of words) plays an important role as well. De Marneffe et al. (2006) and Inkpen et al. (2006) look for antonym pairs appearing between **T** and **H**. Vanderwende et al. (2006) use this feature for the false entailment detection and Newman et al. (2006) get this feature from VerbOcean again.

Named-Entity (NE) has proved to be an important feature. Herrera et al. (2006) have done some complementary experiments to show the improvement by adding NE as features into the former system. Numbers (or numeric expressions) appear in **T** and **H** are compared either directly (De Marneffe et al., 2006a) or after being changed into values (Vanderwende et al., 2006). The latter also discovers the country name or demonym (inhabitants in a place) and uses unaligned NE as a feature. In addition, De Marneffe et al. (2006) extract Date/Time (i.e. temporal expressions), as do Hickl et al. (2006), who normalizes the entities before comparison. Furthermore, Hickl et al. (2006) check NE coreference and NE aliasing using their own NE recognizer.

Other linguistic features (e.g. modality, quantifier) are also applied in some methods as features. De Marneffe et al. (2006) consider various linguistic features, such as adjunct features (e.g. *during the embargo*), modality features (e.g. *must*, *maybe*, *not necessary*), factivity features (e.g. verbs like *try*, *manage*), quantifier features (e.g. *every*, *some*, *all*), etc. Vanderwende et al. (2006) add conditional and counter-factual mismatch (e.g. *if*) as a feature as well.

Many features are extracted from syntactic structures, semantic roles, or logical forms. Hickl et al. (2006) preprocess the texts with lexical analysis, syntactic parsing, coreference resolution, and semantic analysis and then extract features based on the output, including dependency features, paraphrase features, and semantic features. Nicholson et al. (2006) utilize *Basic Element* (BE, Hovy et al., 2005) as the feature space, which is a tuple representing important syntactic chunks in the dependency parse of a sentence. BE comes

from the summarization community, which is an automatic overlap metric that offers a more precise calculation of coverage by only considering matching units containing information bearing words.

Burchardt and Frank (2006) perform deep analysis on input texts and label semantic frames and roles on the parsing tree. Then, four features are considered: number of predicate matches, number of frame matches, number of role matches, and match graph size relative to **H** graph size. Bos and Markert (2006) do semantic analysis and logic inferences, and use the results as deep features combined with shallow features (lexical features) in ML classifier. Vanderwende et al. (2006) extract features like argument mismatch on top of a logical form generated. There semantic representations and logical forms will be seen more in (2.3.4).

Other groups (Herrera et al., 2006; Inkpen et al., 2006; Kouylekov and Magnini, 2006; Newman et al., 2006; Nielsen et al., 2006) extract many syntactic or dependency features based on dependency structures computed by dependency parses, e.g. Minipar (Lin, 1998a), which will be discussed in detail in (2.3.3).

Notice that, using the *Task* attribute of each **T**-**H** pair as a feature is mentioned by Adams (2006) and Newman et al. (2006). In fact, all the groups unconsciously include this feature, if they have the breakdown of experiment results separated by columns of different tasks.

2.2.2 Cross-Pair Features

In the RTE-2 Challenge, there is only one group (Zanzotto and Moschitti, 2006) extract cross-pair features, which means calculating similarity not between **T** and **H** but between different **T**-**H** pairs based on syntactic and lexical information. In detail, they use similarity functions between words or more complex expressions to locate anchors, which are used to connect **T** and **H** as a whole. Afterwards, a kernel based on cross-pair features is applied to calculate the similarity. They approximately take the sum of the similarity between two **T**s and the similarity between two **H**s as the similarity between the two pairs. The kernel is implemented via *Tree Kernel* (Moschitti, 2004) on top of the parsing tree generated by the Charniak Parser (Charniak, 2000). Their system got the 3rd place of the RTE-2 Challenge.

Their work motivates us to investigate kernel-based methods. From a linguistic perspective textual entailment is a relationship between two text fragments, while from the characteristics of classification task, **T**-**H** pairs belonging to the same class (either entailed or non-entailed) may share some common features. Instead of a tree kernel, we apply the subsequence kernel

which has less computational complexity, after extracting a part of the parsing tree as our feature representation (see more in 2.3.3).

2.2.3 Learning Methods

After the feature extraction and selection, choosing a learning method (i.e. classifier) is the next step. Several ML methods have been considered, such as *Decision Trees* (DTs), *Support Vector Machines* (SVMs), *Maximum Entropy* (ME), etc. Some groups compare different ML classifiers as well using the same feature set. Among the ML tools, the Weka (Witten and Frank, 2005) tool is mostly used.

DTs are widely used by groups selecting many linguistic features. Adams (2006) feeds all the extracted features into a J48 DT for training and evaluation. Bos and Markert (2006) also use both shallow and deep features to derive a DT model. While Nicholson et al. (2006) and Newman et al. (2006) apply different DT models, C4.5 (Quinlan, 1993) and C5 (Quinlan, 2002) respectively. Additionally, Burchardt and Frank (2006) utilize *LogitBoost* for the second submission, which performs additive logistic regression using the classifier *DecisionStump* (A decision stump is a decision tree with only one node).

SVM is a kernel-based ML method, which can implicitly represent all the features via transforming them into a vector. Herrera et al. (2006) put both lexical relations and NE-based features into the classifier and emphasize the importance of the latter. Kouylekov and Magnini (2006) have different settings for the Insertion operator of tree editing distance calculation. Thus, they construct five baseline systems and combine them as features into a SMO classifier. Here, SMO is John Platt's sequential minimal optimization algorithm for training a support vector classifier. Schilder and McInnes (2006) have two approaches for word-based similarity and two approaches for tree editing distance; therefore, they set up several combinations of these approaches for a SVM classifier.

Zanzotto et al. (2006) have a different feature space for the kernel-based classifier. As mentioned before (2.2.2), they extract a feature representation concerning the similarity between different **T**-**H** pairs via a syntactic parser and then apply *Tree Kernel* (Moschitti, 2004) to it. The tree kernel function checks whether two trees have sub-trees in common and all possible sub-trees are encoded in a long vector. This is implemented in SVM-light-TK[6] (Moschitti, 2004).

[6] http://ai-nlp.info.uniroma2.it/moschitti/TK1.2-software/Tree-Kernel.htm

An ME model (Berger et al, 1996) is trained by Vanderwende et al. (2006) to learn weights for all the features in the first submission. To help prevent over-fitting, the model uses a Gaussian prior over the weights and this prior is tuned to maximize development set accuracy. They show an improvement of approximately 2.5% over the method used for the second submission.

De Marneffe et al. (2006) compare the results for their system using two ways of setting the feature weights: one describes experiments with weights set by hand (but the threshold set to give balanced true/false answers on the test set); while the other describes experiments in which feature weights, including the threshold, are trained by a logistic regression classifier. The latter is shown better, especially for IR pairs.

Inkpen et al. (2006) have tried four ML classifiers in the experiments: DT, NB, *k-Nearest Neighbor* (kNN), and SVM. Among them, SVM outperforms the others, and the other three are at the same level but behave differently for pairs from different tasks. With the size of the training data, they find that if the data from RTE-1 are added for training, the result does not improve, while if only the RTE-2 development data are used, the result improves along with the size of training data. Kozareva and Montoyo (2006), who have compared kNN, DT, ME, and SVM, also find SVM is the best solution. However, Hickl et al. (2006) find, among ME, SVM, and DT C5.0 (Quinlan, 2003), DT C5.0 achieves the best result for their extracted features. Concerning the learning curve, their system performs with 10% better accuracy after enlarging the training data from only the RTE-2 development set (800 **T**-**H** pairs) to 200,000 **T**-**H** pairs. Hickl et al. (2006) report that the extra data automatically collected from the Web, help them to achieve the best result of RTE-2 Challenge. The details of collecting the data will be explained in (2.4).

Nielsen et al. (2006) do experiments with a lot of ML classifiers with tuned parameters presented as well. They also divide the data into two groups: one is the pairs from SUM task; the other is the pairs from IE, IR, and QA. For the first group, they have tried DecisionTable (with parameters X=2, -I), LogitBoost, SMO, etc. For the other group, SimpleLogistic (H=300), SMO (N=1), SMO (N=1, C=0.73), VotedPerceptron, etc, are used. Then, after obtaining the results from these classifiers, they use two strategies to decide the final results. One is to make decisions based on the average probability of the classifiers. Where the output of classifiers is almost strictly 0 and 1 probability estimates (e.g. SMO and VotedPerceptron), Nielsen et al. (2006) normalized these estimates to be consistent with the classifiers' accuracy on training set cross-validation. The second strategy is to make decisions based on the majority vote among component classifiers, breaking any ties with the average probability

estimate.

2.2.4 Comparison

The following table shows the different features selected by different groups,

Groups / Features	Word Overlap	N-Gram	Editing Distance	Negation Antonym	NE	Cross Pair	KB
Adams	X		X				X
Bos and Markert	X						X
Burchardt and Frank (run2)	X						X
De Marneffe et al.				X	X		X
Herrera et al.					X		X
Hickl et al.	X	X		X	X		X
Inkpen et al.	X	X		X			X
Kouylekov and Magnini (run2)			X				X
Kozareva and Montoyo	X	X		X	X		
Newman et al.	X	X		X			X
Nicholson et al.	X						X
Nielsen et al.	X	X					X
Schilder and McInnes	X		X				X
Vanderwende et al. (run1)	X			X	X		X
Zanzotto et al.	X					X	X
Our Approach	**X**					**X**	

Table 1 Feature Selection

Word Overlap is used by most of the groups, normally assisted by external knowledge bases, like WordNet. Notice that, this table does not include features extract from syntactic, semantic, or logical representations, which will be shown in the next subchapter (2.3).

We also can draw a table of different learning methods applied by different groups as follows,

Classifiers / Groups	DT(J48, C4.5, C5.0) LogitBoost	SVM	ME	kNN	Others
Adams	X				
Bos and Markert	X				
Burchardt and Frank (run2)	X				
De Marneffe et al.					X
Herrera et al.		X			
Hickl et al.	B	X	X		
Inkpen et al.	X	B		X	X
Kouylekov and Magnini (run2)		X			
Kozareva and Montoyo	X	B	X	X	
Newman et al.	X				
Nicholson et al.	X				
Nielsen et al.	X	X			X
Schilder and McInnes		X			
Vanderwende et al. (run1)			X		
Zanzotto et al.		X			
Our Approach		**X**			

Table 2 Machine Learning Methods

In the above table, "B" stands for the best among all the methods they have applied. We can see from the table DTs and SVMs are the two most used classifiers, which are also reported as the best ones compared with other methods.

At last, we want to mention here that ML methods encounter a problem of sparse features. Burchardt and Frank (2006) report that they have many high-frequency features that measure similarity (e.g. predicate and frame overlap), but only few and low-frequency features that identify dissimilarity, such as mismatching modalities. Therefore, the learners have a tendency to reject too little: 29.5% false positives as opposed to 12.75% false negatives. Inkpen et al. (2006) also mention that mismatch features do not help much, perhaps because they are only found in few pairs. Consequently, rule-based linguistic triggers might be a better solution. We will see some research in the next subchapter, which mainly discusses different linguistic representations used for the RTE task.

2.3 Linguistic Representations

The RTE task is defined as detecting whether an entailment relationship exists between two text fragments, but the processing level is not restricted. Consequently, various linguistic representations are involved in solving the problem, from BoW representation at the lexical level to deep semantic logical representations. In this section, several non-ML approaches will

be discussed in detail together with some interesting representations also used in ML methods.

2.3.1 Bag-of-Words/N-Gram

Most of the approaches take BoW representation as one option, at least a baseline system. Here, we will show some groups only use BoW representation or n-gram matching to recognize the entailment relationship.

Adams (2006) begins with a BoW similarity overlap measure, derived from a combination of WordNet lexical chains to form a mapping of terms in **H** to **T**, then followed by looking for negations not found in the mapping, and for the lexical edit distance of the mapping. He achieves an accuracy of 62.6%, scoring 4th place in RTE-2. The high performance of this approach sets a very high baseline score.

Kozareva and Montoyo (2006) and Clarke (2006) step further to detect n-gram overlapping between **T** and **H**. The former checks longest common subsequence and skip gram overlapping. The latter replaces each word with a bag of document identifiers representing the contexts that the word occurs in. However, their results are not as good as Adams (2006)'s.

In addition, Marsi et al. (2006) have taken the alignment algorithm described in Marsi and Krahmer (2005), which itself is based on an alignment algorithm of Meyers et al. (1996) developed specifically for machine translation.

2.3.2 Constitute Structure/Phrase Structure

Usually two kinds of parsers are used to preprocess the plain texts: constitute parsers and dependency parsers. Correspondingly, there are two kinds of parsing results, constitute tree structure and dependency tree structure. We will see the former firstly and the latter in the next section (2.3.3).

De Marneffe et al. (2006) use the Stanford Parser (Klein and Manning, 2003) to parse the input texts and extract constitute structure features like the subject and object of the (main) verb. Rus (2006)'s solution has two phases: the first one is to map **T** and **H** into graph structure; and the second one is to perform a subsumption operation between **T**-graph and **H**-graph. He parses the texts with the Charniak Parser (Charniak, 2000) to achieve the first step of mapping and applies *isomorphism* (Skiena, 1998) from graph theory to check the

subsumption relation between the two derived graphs. Furthermore, he also uses dependency parsing and compares the two results, which will be shown in next section (2.3.3).

Zanzotto et al. (2006), who extract cross-pair features and utilize a tree kernel method, set their feature representation based on the constitute parse tree generated by the Charniak Parser as well. Their method chooses overlapping words as anchors to relate the parsing trees of **T** and **H**, and then the tree kernel is applied to represent them separately. Notice that one of the characteristics of constitute tree is that all the grammatical constitutes are on the upper part of the tree, namely all the non-leaf nodes. Due to this, the tree kernel will in some sense weight constitute structure features more heavily. However, using tree kernels also comes with caveats: first, encoding all possible parsing subtrees may also include much noisy information which is not so relevant to the entailment relationship detection; second, it has high computational complexity. For the latter, Moschitti (2006) proposes an algorithm to compute tree kernels in linear average runtime.

Our approach favors structure features as well. The main differences are: a) we use dependency parsing trees as our starting point instead of constitute trees; b) before applying kernel methods, we extract out a flat sequence structure from the parse tree, a feature representation we feel is more relevant. This last difference both excludes noisy information and greatly reduces the computational complexity a lot.

2.3.3 Dependency Structure

Dependency structure is widely used by many groups, since it can provide us with more information than shallow parsing techniques with quite good robustness and runtime. Basically, a dependency parsing tree contains nodes (i.e. tokens/words) and dependency relations between nodes. Some approaches simply treat it as a graph and calculate similarity between two graphs based solely on their nodes, while some others put more emphasis on the dependency relations themselves.

Marsi et al. (2006) only consider nodes in the dependency tree. Their tree alignment algorithm (Marsi and Krahmer (2005), adapted from Meyers et al. (1996)), calculates the match between each node in a dependency tree against each node in another dependency tree. They define the matching score for each pair of nodes as depending not only on the similarity of the nodes, but also recursively the scores of the best matching pairs of their descendants. For an efficient implementation, dynamic programming is used to build up a score matrix, which guarantees that each score will be calculated only once. To their surprise, they found

that it is not beneficial to take dependency relation labels into account during node-matching.

Katrenko and Adriaans (2006) propose an approach employing embedded consitute subtrees. For graph matching, they have used a method proposed by Zaki (2005). They set the support level to 100% (requesting all nodes in the resulting subtree to be present in two trees to be matched) and searched for the maximal subtree only. Rus (2006) treats the entailment relationship between texts as a graph comparison problem as well. They use the same algorithm as they did for constitute tree (2.3.2), which checks whether the two graphs have subsumption relation in-between.

In contrast to those calculating lexical editing distance on top of BoW representation, some groups compute tree editing distance based on the dependency tree representation. Kouylekov and Magnini (2006) have implemented the tree edit distance algorithm described in Zhang and Shasha (1990), which contains three basic operators, *insertion*, *deletion*, and *substitution*. *Insertion* is defined as the insertion of a node from the dependency tree of **H** into the dependency tree of **T**; *deletion* is the removal of a node from the dependency tree of **T**, together with all its attached children; and *substitution* is the change of the label of a node in the source tree (the dependency tree of **T**) into a label of a node of the target tree (the dependency tree of **H**). *Substitution* is allowed only if the two nodes share the same *part-of-speech* (POS). In case of substitution the relation attached to the substituted node is changed with the relation of the new node. Schilder and McInnes (2006) include another approximate tree edit distance algorithm proposed by Augsten et al. (2005) as well as Zhang and Shasha's. The newer algorithm makes differences in the actual tree structure more pronounced and it is computationally far less expensive.

Some particular dependency relations are of great importance, such as the subject relation and object relation of the verb. Newman et al. (2006) only import these two nodes as separate features. Nicholson et al. (2006) checks whether the two verbs of the *Basic Element* are in the same cluster in Lin's dependency-based thesaurus (Lin, 1998c) as well as the two arguments of the verb, namely the subject and the object.

Inkpen et al. (2006) post-process the dependency tree into a list of dependency tuples containing the relation, the head, and the modifier and then use the derived representation to check the dependency pair overlap between **T** and **H**. They also handle *negation* and *number*. Their dependency tuple is quite similar to the triple representation introduced by Wang and Neumann (2007a), which is in the form of <parent node, relation, child node>. We use the overlapping ratio of the triples between **T** and **H** as our second backup strategy (the first one is the lexical similarity based on BoW representation), which actually expresses the local

dependency relation.

The dependency path, containing both the nodes and the relations, has been considered a good representation to capture long dependency relations. Herrera et al. (2006) perform mapping between **T** and **H**, which is the one designed for the previous system (Herrera et al., 2005). This matching technique is inspired by Lin's proposal (Lin, 2001), whose initial idea is to search **T**'s tree for all the branches starting at any leaf of **H**'s tree. Hence, a matching branch of **H** is defined as one all of whose nodes show a lexical entailment with the nodes from a branch of the corresponding **T**. Nielsen et al. (2006) have also considered dependency path which contains not only subject and object relation, but also preposition complement relation and other relations.

Our feature representation arising from the dependency tree is called the *Tree Skeleton*, which contains two dependency paths from the two arguments to the common predicate. This representation is used instead of the complete parsing tree. It excludes irrelevant (or unimportant) information of **T** and preserves both the nodes and dependency relations in-between; As well, this derived representation can be easily transformed into a flat sequence structure, which will greatly reduce the computational complexity for the kernel function in the later stage, in contrast with the original tree structure.

2.3.4 Semantic Representation/Logical Forms

Among all the cases of entailment, there are some which cannot be solved by lexical semantics or syntactic analysis. Some examples are presupposition, implicature, etc. Therefore, semantic role labeling, deep grammars, axioms and theorem provers have been used by some groups of researchers. Roughly speaking, those using these semantic techniques can be divided into two groups: one group obtains semantic representation or logical forms in order to compare **T** and **H** using more information; the other group performs logic inference on top of the derived logical forms.

Burchardt and Frank (2006) use a probabilistic LFG grammar for English developed at Parc (Riezler et al., 2002) to perform linguistic analysis and combine two probabilistic systems for semantic frame and role annotation: Fred and Rosy (Erk and Pado, 2006) and a rule-based system for frame assignment, called Detour (Burchardt et al., 2005). The resulting structures are converted to a *Frame Exchange Format*, a flat predicate representation comprising syntactic and semantic analysis. Accordingly, their matching graph contains more semantic information than constitute or dependency trees. However, the advantages of using

deep analysis have not been shown in the final results.

Delmonte et al. (2006) have two subsystems: one is a sequence of rule-based linguistic analyzers; the other is a module of measuring the similarity of input structures. The output representation of the first subsystem is a flat list of fully indexed *Augmented Head-Dependent Structures* (AHDS) with grammatical relations and semantic roles labels. The second subsystem use axiomatic linguistic rules to check the similarity between AHDS of **T** and AHDS of **H**, mainly at two levels: head level (e.g. intersective modifiers and adjuncts, quantifiers) and propositional level (e.g. modality, negation, conditionality). Ferrández et al.'s system (2006) has the same architecture, while their logic form, similar to Moldovan's logic form (Moldovan and Rus, 2001), is based on the format defined in the eXtendedWordNet (Harabagiu et al., 1999). Regarding the performance, Delmonte et al. (2006) report that about 40% of the errors are due to the bad parses and the other 60% come from insufficient semantic information. In fact, neither of the two groups shows promising results or improvements after applying logic forms to the RTE task, compared with rather shallow techniques.

Another means of utilizing semantic or logical representation is to use theorem prover provided with pre- defined or learned axioms. Bos and Markert (2006), in one of their systems, parse the texts with a CCG parser (Bos, 2005) using a first-order fragment of DRS language as their semantic representation: *Discourse Representation Theory* (DRT, Kamp and Reyle, 1993). DRT conveys argument structure with a neo-Davidsonian analysis and includes a recursive DRS structure to cover negation, disjunction, and implication. Third-person personal pronouns are resolved to named entities, and proper names and definite descriptions are treated as anaphoric too. They employ the theorem prover Vampire 7 (Riazanov and Voronkov, 2002) and two model builders, Paradox 1.3 (Claessen and Sörensson, 2003) and Mace 2.0 (McCune, 1998). Whereas the final figures again suggest that logical inference does not help much, closer inspection of the results revealed that for some of the subtasks logical inference did play a significant role in performance. This actually suggests that there are several categories of entailment relation, not all of which needs deep semantics.

Tatu et al. (2006) have shown a promising accuracy using a logic proving system. As the 2nd place of the RTE-2 Challenge, they have proposed a knowledge representation model on top of their logic prover. Their system, COGEX (Moldovan et al., 2003) is a natural language prover originating from OTTER (McCune, 1994), whose success seems to be attributed to knowledge acquisition in a large scale. Their system is equipped with a large quantity of world knowledge axioms, linguistic axioms, temporal axioms, etc, coming from external

knowledge base like WordNet, SUMO[7] (Niles and Pease, 2003), etc, and manually designed rules as well. Therefore, this kind of knowledge-based approach has its caveats that manually designed logical rules usually require a high amount of specialized human expertise in different NLP areas. Moreover, it cannot solve the errors in syntactic and semantic analysis, which is less robust than shallow approaches such as similarity calculation.

2.3.5 Multiple Representations

Some research groups have applied integrated NLP systems, usually consisting of components at different processing levels, such as POS taggers, NE recognizers, syntactic parsers, semantic analyzers, or even anaphora resolvers.

The best team of the RTE-2 Challenge (Hickl et al., 2006) extracts a large number of features from diversified levels via their GROUNDHOG system. Their CiceroLite NE recognition system can identify more than 150 different NE classes. Temporal expressions, (including dates and times) and spatial expressions (including names of most political and geographic locations) are then sent to their TASER temporal and spatial normalization system (Lehmann et al., 2005), which maps these expressions to their ISO 9000 equivalents. Syntactic parsing is performed using their own implementation of the Collins Parser (Collins, 1996), while semantic parsing is performed using an ME-based semantic role labeling system trained on the predicate-argument annotations found in PropBank (Palmer et al., 2005). They also use a combination of heuristics and lexica from CiceroLite to identify coreferential named entities and to perform name aliasing for all of the entities found in each **T**-**H** pair.

Vanderwende et al. (2006) have used another system called NLPwin, which is a robust system for natural language parsing and generation. Logical forms can be generated by NLPwin and the system has been successfully used in such diverse applications as summarization, machine translation, and many others (Leskovec et al., 2005; Quirk et al., 2005).

Litkowski (2006) builds his RTE system on a more generic NLP platform, named as *Knowledge Management System* (KMS). KMS is a graphical interface that enables users to create repositories of files (of several file types) and to perform a variety of tasks against the files. The tasks include question answering, summarization, information extraction, document exploration, semantic category analysis, and ontology creation.

[7] http://www.ontologyportal.org/

From the perspective of system architecture, our system is relatively small-scale. However, we have implemented in a flexible way that modules can be conveniently added into the whole system as one component of the processing pipeline. The meta-class for each component is also well-defined, which can be extended with a voting function. Currently, we have integrated two dependency parsers (i.e. Minipar and the Stanford Parser).

2.3.6 Comparison

The following table is the comparison of different representations applied by different groups,

Representations / Groups	Only BoW N-Gram	Constitute/ Phrase Str	Dependency Str	Semantic Logical	Logic Inference
Adams	X				
Bos and Markert				X	X
Burchardt and Frank		X		X	
Clarke	X				
de Marneffe et al.		X			
Delmonte et al.				X	
Ferrández et al.				X	
Herrera et al.			X		
Hickl et al.			X	X	
Inkpen et al.			X		
Katrenko and Adriaans			X		
Kouylekov and Magnini			X		
Kozareva and Montoyo	X				
Litkowski		X			
Marsi et al.			X		
Newman et al.			X		
Nicholson et al.			X		
Nielsen et al.			X		
Rus		X	X		
Schilder and McInnes			X		
Tatu et al.				X	X
Vanderwende et al.				X	
Zanzotto et al.		X			
Our Approach			**X**		

Table 3 Comparison of Representations

From the table we can see that dependency structure is widely used by many groups. Although several groups have generated logical forms, only two groups have performed logic inferences, while others have calculated similarity based on those forms. Three groups have

utilized only BoW or n-gram representation, and most of the others have taken this as a baseline. Compared with others, we are one of the groups who have only used the dependency structure, which is quite practical and efficient.

2.4 Corpora and External Knowledge Base

Among all the participators of the RTE-2 Challenge, only Hickl et al. (2006) have constructed extra **T-H** pairs for training the system. They claim this has improved the result by 10%. The only entailment corpora for other groups are the RTE-2 data set, since the RTE-1 data set is a bit different[8]. Other kinds of corpora are used by some groups to calculate the relevance of words via co-occurrence in a large set of data. As for external Knowledge Bases (KBs), WordNet is the mostly widely used, and others like FrameNet and VerbOcean are also considered by some groups.

Hickl et al. (2006) have created a corpus of approximately 101,000 **T-H** pairs using the headline and the first sentence from newswire documents, following the idea proposed in Burger and Ferro (2005). In order to increase the likelihood of including only positive examples, pairs that did not share an entity (or an NP) in common were filtered, as were pairs that discussed sports results or stock prices. In a sample of 2500 pairs selected at random, 2296 (91.8%) were judged by human annotators as positive. As for the negative examples, two approaches were applied: one is to extract pairs of sequential sentences that included mentions of the same named entity from a large newswire corpus (more than 98,000 from nearly 700,000 documents; human annotators deemed 97.5% (2438/2500) of these examples to be negative examples); the other is to extract pairs of sentences linked by discourse connectives such as *even though*, *although*, *otherwise*, *in contrast* and *but* (approximately 21,000). In an analysis of 1000 of these examples, annotators judged 942 (94.2%) to be negative for textual entailment.

However, our experience shows that except for the positive cases of SUM, **T-H** pairs are normally not very easy to collect automatically. Sometimes multi-annotator agreement is difficult to reach as well. We cannot achieve such high agreement as Hickl et al. (2006) have claimed. Moreover, enlarging the training set does not seem to be necessarily helpful for improving the performance. According to our experiments, if we use the "proper" data (here means RTE-2 data), the result will be increased along with the size of the training set, but if we collect extra data, the performance does not improve, even after double the training set.

[8] The RTE-1 dataset is constructed from 7 NLP tasks, *Information Retrieval* (IR), *Comparable Documents* (CD), *Reading Comprehension* (RC), *Question Answering* (QA), *Information Extraction* (IE), *Machine Translation* (MT), and *Paraphrase Acquisition* (PP) (Dagan et al., 2006).

The problem could be either due to the feature representation of our methods or the diversity of heterogeneous data. Further discussions will be given in 5.3.4.

Other than entailment corpora, Nielsen et al. (2006) have utilized some plain text corpora to obtain document co-occurrence counts of word pairs, as was done by Turney (2001) and Glickman et al. (2005). In all, they have used three publicly available corpora totaling 7.4M articles and 2.6B indexed terms, including English Gigaword (Graff, 2003), Reuters corpus (Lewis et al., 2004) Volume 1, and the three volume TIPSTER corpus[9]. Newman et al. (2006) use a *Latent Semantic Indexing* (Deerwester et al., 1990) measure, which attempts to calculate similarity beyond vocabulary overlap by identifying latent relationships between words through the analysis of co-occurrence statistics in an auxiliary news corpus.

Concerning the external KBs, WordNet is usually used for detecting synonyms, hyponyms, etc. On top of the WordNet taxonomy, some similarity measures are also widely used. Ferrández et al. (2006) have utilized WordNet relations as well as Lin's measure (Lin, 1998b). Newman et al. (2006) have included one WordNet-based measure, called Hirst-St-Onge (Budanitsky and Hirst, 2001). One of Schilder and McInnes' similarity measures (2006) is from Wu and Palmer (1994), which measures the depth of two concepts in WordNet and the depth of their Least Common Subsumer. In addition, both eXtendedWordNet[10] (Harabagiu et al., 1999) is applied in Tatu et al.'s system (2006) as KBs, and SUMO is applied.

VerbOcean (Chklovski and Pantel, 2004; Chklovski and Pantel, 2005) helps Newman et al. (2006) and Nicholson et al. (2006) to detect verb negation, verb synonymy, verb antonym, etc. Burchardt and Frank (2006) and Hickl et al. (2006) perform semantic role labeling with the help of FrameNet and PropBank respectively.

At last, Kouylekov and Magnini (2006) have applied paraphrasing rules from TEASE (Szpektor et al., 2004) and DIRT (Lin and Pantel, 2001) to enhance their system.

It also appears that using RTE systems, we can learn entailment rules (or patterns) as well. We consider this area as one of our future work. And perhaps instead of using external KBs, maybe it is a good idea to construct a KB using textual entailment.

2.5 The RTE-3 Challenge

After the techniques and resources of the RTE-2 participants, we would like to discuss some

[9] http://www.ldc.upenn.edu/Catalog/CatalogEntry.jsp?catalogId=LDC93T3A
[10] http://xwn.hlt.utdallas.edu/

new trends of the recent RTE-3 Challenge in this subchapter. From a broad view, there are also two categories of approaches: machine learning plus feature selection and inferences based on logic or other representations.

The first category of approaches using machine learning techniques has selected features at the lexical-syntactic level or used transformation rules on top of the dependency representation, which continues some works from the RTE-2 Challenge to some extent. "Light" baseline systems have been provided by several groups. We set up a bag-of-words similarity measure between **T** and **H** (Wang and Neumann, 2007b). Malakasiotis and Androutpoulos (2007) provided a baseline using POS tagging and then applying string-based measures to estimate the similarity. The extension of transformation-based approaches toward probabilistic settings is an interesting direction investigated by some systems (e.g. Harmeling (2007)).

The second category has moved toward deep approaches with a general consolidation of approaches based on the syntactic structure of **T** and **H**. Though there is an evident increase of systems using some form of logical inference, most of the approaches have shown results still at the state of art (e.g. *Natural Logic* introduced by Chambers et al. (2007)). Tatu and Moldovan (2007) have carried out on a sophisticated analysis of named entities, in particular person names, distinguishing first names from last names. Their work has provided us with a good example of the integration of different semantic resources aimed at specific semantic phenomena.

External KBs are also considered, which is similar to the RTE-2 Challenge. Lexical databases (mostly WordNet and DIRT) are still widely used. Extended WordNet is also a common resource (e.g. Iftene and Balahur-Dobrescu (2007)). Verb-oriented resources are also largely present in several systems, including FrameNet (e.g. Burchardt et al. (2007)), VerbNet[11] (Bobrow et al., 2007) and PropBank (e.g. Adams et al. (2007)). Compared with the previous RTE-2, the use of a large training corpus has shown less impact on the performance on the RTE-3 data set (Hickl and Bensley, 2007). This problem has been briefly discussed a bit relating to our experiments (Wang and Neumann, 2007a) and more discussions will be given in 5.3.4.

In addition, since some longer texts were included in the RTE-3 data set (see a detailed description of the data set in 5.1), a number of participating systems have addressed anaphora resolution (e.g. Delmonte et al. (2007), Bar-Haim et al. (2007), Iftene and Balahur-Dobrescu

[11] http://verbs.colorado.edu/~mpalmer/projects/verbnet.html

(2007)).

2.6 Summary

In this chapter, we have taken a view of the current approaches applied in the RTE area. A large number of research groups are using ML classifiers fed with various kinds of features, both statistical and linguistic, from BoW representation to deep semantic representation or logical forms. Different linguistic representations are considered by different approaches, from BoW, n-gram to constitute parsing tree and dependency parsing tree, and semantic representation and logical forms as well. Finally, corpora and external knowledge bases are used to assist the RTE systems.

Chapter III

A Hybrid Approach

In this chapter, we will present our hybrid approach to RTE. After an overview of relevant techniques applied in the concrete examples, we will discuss about three approaches in detail, which have different domains of locality and different feature space. We will start from a *Bag-of-Words* (BoW) approach, then introduce an approach of local dependency relation, and at last elaborate on our kernel-based approach, which is based on a new representation of sentence. In the end, we will do a summary.

3.1 Overview

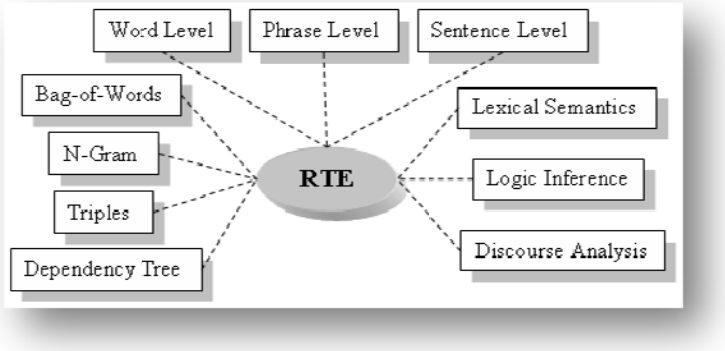

Figure 1 Overview of techniques relevant to RTE

Figure 1, as a brief review of the state of the arts, shows the different processing techniques and depths, which are also discussed in Chapter II, applied to the RTE task. Essentially, RTE is to check the relationship between two text fragments, which also suggests we discover a way to compare them.

Let us take a look at the following examples,

Dataset=RTE-2 Dev Id=13 Task=IE Entailment=YES
*Text: Sunday's earthquake was felt in the **southern Indian city of Madras** on the mainland, as well as other parts of south India. The Naval meteorological office in Port Blair said it was the second biggest aftershock after the Dec. 26 earthquake.*
*Hypothesis: The city of **Madras** is located in **Southern India**.*

Example 12

Dataset=RTE-2 Dev Id=133 Task=SUM Entailment=NO
Text: Verizon Communications Inc. ***said*** *on Monday it would* ***buy*** *long-distance telephone company MCI Communications Inc. in a deal worth $6.75 billion, giving Verizon a foothold in the market for serving large corporations.*
Hypothesis: Verizon Communications Inc.'s $6.7 billion ***takeover*** *of long-distance provider MCI Inc. transformed the telephone industry.*

Example 13

These two examples come from the RTE-2 Dev dataset. The first pair (id=13) belongs to syntactic entailment. The most relevant knowledge here is "[LN1] *city of* [LN2]" entails "[LN2] *is located in* [LN1]", although **T** focuses on the earthquake event. The last pair (id=534) is a similar case with different structures in **T**. On the other hand, the third pair (id=133) requires not only an understanding of concepts like "*buy*" and "*takeover*", but also to understand the usage of "*said*", which is a case of semantic entailment. These aspects motivate us to explore specialized entailment strategies for different NLP tasks. In other words, we want to discover the potential connections between entailment relations belonging to different linguistic layers for different applications.

In this thesis work, we have used approaches of different domains of locality to deal with different cases. In practice, we consider three: One is to treat the plain texts in a BoW fashion; the second one is to keep solely the local dependency relations between tokens; and the last one is to use a new representation of the sentence to capture the long dependency relation, and then apply subsequence kernels to implicitly collect all the features. The next subchapter will start from the most straightforward one.

3.2 A Bag-of-Words Approach

Roughly thinking, the more overlapping words there are between **T** and **H**, the more possible that **T** entails **H**, and vice versa. Like the following two examples,

Dataset=RTE-2 Dev Id=47 Task=IR Entailment=YES
Text: ***Women*** *form half the population and 54% of the voters in the country, yet* ***are very poorly represented in parliament****.*
Hypothesis: ***Women are poorly represented in parliament****.*

Example 14

Dataset=RTE-2 Dev Id=140 Task=IR Entailment=NO
Text: ***Aspirin****, an inexpensive drug helps protect survivors of heart attack and stroke from subsequent heart attacks and death, and even helps reduce the number of deaths that occur within the first hours following a heart attack.*
Hypothesis: People experienced adverse effects while taking ***aspirin****.*

Example 15

In the first example, every word in **H** also appears in **T**, and the meaning of **H** is just what one part of **T** conveys; while in the second example, "aspirin" is the only overlapping word between **T** and **H**, and **T** and **H** convey irrelevant information. Therefore, it seems that the number of overlapping words can tell us whether **T** entails **H**.

However, our rough assumption is not always true. See the following two examples,

Dataset = RTE2-dev Id=103 Task=IR Entailment=YES
Text: This paper describes American alcohol use, the temperance movement, Prohibition, and the War on ***Drugs*** *and explains how* ***legalizing drugs*** *would reduce crime and public health problems.*
Hypothesis: ***Drug legalization*** *has benefits.*

Example 16

Dataset=RTE2-dev Id=35 Task=IE Entailment=NO
Text: Meanwhile, in an exclusive interview with a TIME journalist, the first one-on-one session given to a Western print publication since his election as president of Iran earlier this year, ***Ahmadinejad attacked*** *the "threat" to bring the issue of Iran's nuclear activity to* ***the UN Security Council*** *by the US, France, Britain and Germany.*
Hypothesis: ***Ahmadinejad attacked the UN Security Council****.*

Example 17

In the first **T-H** pair, only half of the words appear both in **T** and **H**, but the entailment relationship holds. Here, "*reduce crime and public health problems*" entails "*has benefits*". In the second one, every word of **H** can be found in **T**, but fortunately, the answer is "*NO*". "*Ahmadinejad*" didn't attack "*the UN Security Council*" but "*the 'threat' to bring the issue ... to the UN Security Council*". The first one is rather difficult for the moment; therefore we will start handling the second one in the next subchapter (3.3).

Before that, one point should be mentioned here. BoW method has its advantages. There are some examples as follows seem very difficult to manage even for human beings, but using BoW method can predict the answer correctly without any "deep thinking",

Dataset=RTE2-dev	*Id=513*	*Task=IE*	*Entailment=YES*
*Text: These acoustic methods are now expected to be useful for the long-range **remote sensing** of schools of fish as well as for distant **ocean** bottom characterizations.*			
*Hypothesis: **Ocean remote sensing** is developed.*			

Example 18

Dataset=RTE2-test	*Id=155*	*Task=QAEntailment=YES*
*Text: The **EZLN** differs from most **revolutionary groups** by having stopped military action after the initial uprising in the first two weeks of 1994.*		
*Hypothesis: **EZLN** is a **revolutionary group**.*		

Example 19

If the system really needs to obtain the answer after understanding both the **T** and **H**, as human beings do, it must have a deep semantic parser to know "some methods are expected to be useful for some advanced technique" entails "some not so advanced technique has already been developed." Therefore, here, the acoustic methods are expected to be useful for **long-range** remote sensing of schools of fish, implies that remote sensing is developed. Furthermore, the second usage of the methods, ocean bottom characterizations, implies the remote sensing is also in the ocean. Altogether, **ocean remote sensing** is developed. Example 19 is another difficult example, which asks the systems to know "*EZLN is a revolutionary group*", if it can be compared with other "*revolutionary groups*".

How does our BoW method deal with these pair? It is straightforward. It does not care about the relationship between "*ocean*" and "*remote sensing*", which are just overlapping words between **T** and **H**; neither does it care about the implicature between expected usage for some technique and the development state of that technique. It views these pairs as "most of the words are the same." More examples are given below,

Dataset=RTE3-dev	*Id=216*	*Task=IR*	*Entailment=YES*	*Length=short*
*Text: Anti-**nuclear** protesters on Wednesday **delayed** the progress of a shipment of radioactive **waste** toward a dump **in** northern **Germany**. The train stopped for the fourth time since crossing into Germany as it neared the northern town of Lueneburg.*				
*Hypothesis: **Nuclear waste** transport **delayed in Germany**.*				

Example 20

Dataset=RTE3-dev	*Id=730*	*Task=SUM*	*Entailment=YES*	*Length=short*
Text: ***The IAEA board*** *in February referred* ***Iran*** *to the Security Council, suggesting* ***it*** *had breached the Nuclear Nonproliferation Treaty and* ***might be trying to make nuclear weapons****.*				
Hypothesis: ***Iran might be trying to make nuclear weapons*** *according to* ***the IAEA board****.*				

Example 21

In the first example, all the words in **H** are distributed in the first sentence of **T**, which is difficult to resume the same relationships between these words, but the method still works well. The second example is even more difficult, because anaphora resolution is needed to know the subject of "*might be trying to make nuclear weapons*" is "*Iran*" in **T**. It can also be handled by the BoW method simply due to the high word overlapping ratio between **T** and **H**.

As a summary, the examples in this subchapter give us some basic ideas to handle the RTE task. It seems that word overlapping itself is not enough to cover all the cases, though it can predict some **T-H** pairs. However, notice that, word overlapping calculation can be done on any pair, which means every pair has a word overlapping ratio between **H** and **T**. This characteristic makes this Bag-of-Word approach extremely robust, ignoring the accuracy. That's why we use it as a backup strategy, which will deal with all the pairs cannot be solved by our main approach.

3.3 An Approach of Local Dependency Relation

Since language is not a bag of words, we need some way to represent the relationship between words. Let us consider the unsolved Example 17 again (repeated as follows),

Dataset=RTE2-dev	*Id=35*	*Task=IE*	*Entailment=NO*
Text: Meanwhile, in an exclusive interview with a TIME journalist, the first one-on-one session given to a Western print publication since his election as president of Iran earlier this year, ***Ahmadinejad attacked*** *the "threat" to bring the issue of Iran's nuclear activity to* ***the UN Security Council*** *by the US, France, Britain and Germany.*			
Hypothesis: ***Ahmadinejad attacked the UN Security Council****.*			

Example 17 (again)

Though **T** and **H** have these words (i.e. "*Ahmadinejad*", "*attacked*", and "*the UN Security Council*") in common, the relations between words are different. In **H**, the object of "*attacked*" is "*the UN Security Council*", while in **T**, there is no direct relation between them. In detail, in **T**, the object of "*attacked*" is "*the 'threat' ...*" and "*the UN security Council*" is the object of "*bring*", and furthermore, there is some relation between "*attacked*" and "*bring*", which is

infinitive. If we could know the relations between the words, this pair would be much easier to handle.

3.3.1 Dependency Structure

Dependency Structure is the parsing tree of a sentence using *Dependency Grammar* (DG), which consists of a set of dependency relationships. A dependency relationship (Hays, 1964; Hudson 1984; Mel'čuk, 1988) is an asymmetric binary relationship between one token (i.e. parent node or head) and another token (i.e. son node or modifier). The dependency structure is a connected tree of all the tokens of the sentence, where each parent node can have several son nodes but each son node can only have one parent node.

The following figure is an example of the dependency structure of the sentence "*I ate two big fish.*"

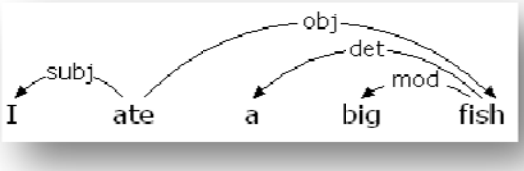

Figure 2 an Example of Dependency Grammar

In the dependency structure, each word (or token) is attached to another word (or token) except for the main verb (i.e. predicate) of the sentence, which is the root node of the dependency tree. The edge between each pair of words is a directional relationship, namely dependency relation, which specifies the relationship between the two words. In the figure, "*subj*" stands for subject-of relation, "*obj*" stands for object-of relation, "*det*" stands for determiner-of relation, and "*mod*" stands for modifier-of relation. There are more dependency relations than these four, but this example gives us a basic idea of dependency structure.

The motivation to use dependency structure is that on the one hand, dependency structure provides more information than shallow parsing techniques; and on the other hand, it is more robust and fast than deep parsing (Lin, 1998a). Compared with syntactic structure (i.e. constitute parsing tree), dependency structure captures the relationship between words other than the buildup process of the sentence. In fact, dependency structure can be generated from constitute structure (de Marneffe et al., 2006b).

3.3.2 Local Dependency Relation

Provided with the dependency structure, the unsolved Example 17 in (3.2) can be resolved. The following (local) dependency relations (together with two nodes) from **T** are relevant to the words in **H**,

…
attack –subj → Ahmadinejad
attack –obj → threat
attack –comp → bring
…
Council –det → the
Council –nn → UN
Council –nn → Security
bring –prep_to → Council
…

Table 4 Local Dependency Relation Set of Example 17 (partial)

Notice the highlighted ones that the object of "*attack*" is not the "*Council*". Consequently, if we check all the dependency relations in common between **T** and **H** as well as the overlapping words, this pair will be predicted as non-entailment.

In fact, dependency relation checking can be viewed as a stricter test compared with BoW methods, because checking the overlapping words is anyway the prerequisite of this step. It helps us to filter out those pairs which are seemingly positive cases but actually negative ones. In practice, we apply some partial matching techniques to make it more flexible (4.2.4).

Furthermore, the dependency relation can sometimes detect interesting errors made by the BoW method,

Dataset=RTE2-dev Id=300 Task=QA Entailment=NO
Text: Despite Bjork making her first ***live*** *performance in two years, the crowd of 10,000 people was only half of what the hall in the Tokyo suburb of Makuhari could hold.*
Hypothesis: 10,000 people ***live*** *in Tokyo.*

Example 22

In this example, the "*live*" in **T** is totally different and irrelevant to the "*live*" in **H**. Our BoW method fails in this example, because all the words in **H** appear in **T** on the surface. But with dependency relations, it is clear that no connections between "*live*" and "*people*" or "*live*" and "*Tokyo*" in **T**.

However, if we take a closer look at this method based on local dependency relations, there is still some space to improve. See the problem shown in the following two examples,

Dataset=RTE3-dev *Id=55* *Task=IE* *Entailment=YES* *Length=short*
*Text: Bosnia's leading Muslim daily Dnevni Avaz writes excitedly about "a sensational discovery" of "**the first European pyramid**" in the central town of Visoko, **just north of Sarajevo**.*
*Hypothesis: **Europe's first pyramid** has been discovered **near Sarajevo**.*

Example 23

Dataset=RTE3-dev *Id=739* *Task=SUM* *Entailment=NO* *Length=short*
*Text: In an interview this week, the president said that Vice President **Dick Cheney** and **Defense Secretary Donald Rumsfeld** will be a valued part of his administration until his presidency comes to an end.*
*Hypothesis: **Dick Cheney** is Vice President of the **Defense Secretary Donald Rumsfeld**.*

Example 24

The first example is a positive case of entailment, but has very few overlapping dependency relations. For instance, in **T**, "*European*" is the modifier of "*pyramid*", but in **H**, the "*Europe*" is the possessor of "*pyramid*". In fact, in order to obtain the final answer, several steps should be taken: first of all, "*the first European pyramid*" is a paraphrase of "*Europe's first pyramid*"; then something is "*just north of Sarajevo*" implies that it is "*near Sarajevo*"; and at last, "*writes ... about 'a ... discovery'*" entails that (the pyramid) "*has been discovered*". Hence, we need to firstly indicate the common entities mentioned in both **T** and **H**, but not necessarily the same mentions, and then check the relationships between (or among) them contain the entailment relation or not.

In the second example, the overlapping dependency relations are also very few, thus our previous method does not work either. Let us try the new findings based on the first example: two steps, common entity indication and relationship checking. The first step is straightforward in this example, but the second one is not. But we know that the relationships between entities are implicitly expressed via "*and*" and "*is Vice President of*". Therefore, one preliminary "rule" could be drawn that "coordinates cannot have belong-to relationship in-between" (because it is a negative case).

Notice that, our new (preliminary) method is to discover some common features shared by all the positive (or negative) entailment cases, instead of looking for common features shared by **T** and **H**. This is very different, because we have just changed from **intra-pair** features

into **cross-pair** features. One more example will make this clearer,

Dataset=Rte3-dev Id=34 Task=IE Entailment=YES Length=short
*Text: Parviz Davudi was representing Iran at a meeting of the Shanghai Co-operation Organisation (SCO), the fledgling association that **binds** Russia, China and four former Soviet republics of central Asia **together** to fight terrorism.*
*Hypothesis: China **is a member of** SCO.*

Example 25

This is a positive case with a similar structure in **H**. Compared to the previous negative example, both **H**s have the structure of "*is N of*", whereas the two **T**s are different. The two entities in previous **T** are coordinates, but in this **T** "*China*" and "*SCO*" have a belong-to relation. This is a cross-pair comparison. In addition, this example also has a low overlapping ratio of dependency relations, which means the method based local dependency relation cannot differentiate these three examples. But we still keep this method as the other backup strategy (one is the BoW method), which shows some advantages in some entailment cases. We also use it as one of our baseline systems for comparison (see 5.2).

As a short conclusion before going on, we have found out three shortcomings of our method based on local dependency relation: the first one is incapable of dealing with various mentions of one same entity (i.e. paraphrasing or entailment relation between nouns); the second one is too strict with the dependency relations, which cannot capture the real semantic relation between two entities (i.e. paraphrasing or entailment relation between verb predicates); and the last one is that intra-pair features cannot always indicate the entailment relation (i.e. the actual feature space of classification task). Therefore, how to solve these problems and represent the seemingly complex "rules" will be the main topic of next subchapter.

3.4 A Kernel-based Approach

The examples above have shown the limitation of the method based on local dependency relation concerning the RTE task. In this subchapter, we will elaborate on our kernel-based approach. Before going into the details, let us skip out to see the whole picture of the entailment problem again.

The RTE task is a binary classification on **T**-**H** pairs, two groups of sentence(s). The two classes are positive, when the entailment holds in the pair, and negative, when it does not. In nature, the feature space should be the similarity or dissimilarity among all the **T**-**H** pairs, not

the similarity or dissimilarity between **T** and **H**. Therefore, we should combine **T** and **H** together, but not separate them.

Another point needs to be mentioned here is that the entailment relation is a one-directional relationship between two text fragments. Unlike the paraphrase relation, which is a two-directional relationship between two text fragments, we also need to show the directionality in some way.

With these two reflections, our method starts from analyzing **H** instead of **T**. We extract some key information contained in **H**, namely nouns and verbs, and then use them as anchors to locate the corresponding parts of **T**. These are all done on top of the dependency structure; therefore, two partial structures are obtained separately as a result. After that, we merge these two partial structures into one pattern, and take it as our feature representation. In particular, we exclude all the common parts out and leave the dissimilar parts. To represent the features, we then apply the subsequence kernel. In all, the underlying assumption is that *the dissimilarity between parts of **T** (indicated by **H**'s key information) and **H** is the indicator for entailment relation from **T** to **H***.

3.4.1 From H to T

As mentioned before, for the RTE task, **H** is a kind of target which we need to verify each piece of information contained can be obtained from **T** (assisted with common knowledge). If we start from **T**, we may choose the wrong way, but if we can successfully trace back from **H** to **T**, the relevant parts will be identified out. Notice that this process has the opposite direction of the entailment relation.

Some practical reasons are also considered here. Based on some basic statistics on the RTE-2 data set (Garoufi, 2007), we find that few cases have a longer **H** than **T**, that is, in most of the **T**-**H** pairs, **T** is longer than **H**. Therefore, **H** is easier to process and extract important parts out.

The following two examples are extreme cases,

Dataset=RTE2-dev	*Id=13*	*Task=IE*	*Entailment=YES*
*Text: Sunday's earthquake was felt in the **southern Indian city of Madras** on the mainland, as well as other parts of south India. The Naval meteorological office in Port Blair said it was the second biggest aftershock after the Dec. 26 earthquake.*			
Hypothesis: The city of Madras is located in Southern India.			

Example 26

Dataset=RTE3-dev	*Id=410*	*Task=QA*	*Entailment=YES*	*Length=long*
*Text: Bush and his wife, Laura, departed the White House on Thursday afternoon to spend four days at the wooded presidential retreat of rustic cabins in the Maryland mountains. Along with the first lady's mother, Jenna Welch, the weekend gathering includes the president's parents, former President **George H.W. Bush and his wife, Barbara**; his sister Doro Koch and her husband, Bobby; and his brother, Marvin, and his wife, Margaret.*				
Hypothesis: The name of George H.W. Bush's wife is Barbara.				

Example 27

In both cases, the relevant parts of **T** to the entailment detection are very small parts of the long texts. If we read the whole texts, the **T** in the first example is about an earthquake event, and the **T** in the second example is about Bush's family gathering. Therefore, in order to know whether the entailment holds, we do not necessarily know all the information provided in **T**, not even the main topic of **T**. If we start analyzing **T**, we may either obtain irrelevant information or exclude seemingly unimportant but actually relevant information. It is misleading, at least, inefficient.

3.4.2 Topic Words

In this section, we will talk about which parts are important and how to extract them out. Since we want to know the information or content of **H**, we are more interested in the contents words. We know from the previous examples (3.3.2) that the mentions of the same entity could of great variety, but the content words (rather than function words) or stems of those words are usually kept.

As the first step, we identify all the nouns in **H** and mark them as topic words. Then we look into **T** and locate the corresponding topic words. In order to avoid the loss of strict word matching, we apply some partial matching techniques at the substring level, such as checking whether most of the letters are overlapping instead of all the letters, the full name compared to acronym, etc. This step can be replaced or improved in various ways. *Lexical Edit Distance* method (Adams, 2006) could be one option to calculate the word similarity. External lexical

resources (e.g. WordNet) could also be considered, like many research groups do (2.4). In this thesis work, we did not apply any external knowledge base, which is planned to be a future work.

Then we will have a set of topic word pairs, like the following example,

Dataset=RTE3-dev	*Id=390*	*Task=IR*	*Entailment=YES*	*Length=short*
Text: ***Typhoon Xangsane*** *lashed the* ***Philippine*** *capital on Thursday, grounding flights, halting vessels and closing schools and markets after triggering fatal flash floods in the centre of the country.*				
Hypothesis: A ***typhoon*** *batters the* ***Philippines****.*				
Topic Word Pair Set: {<"Typhoon Xangsane", "typhoon">, <"Philippine", "Philippines">}				

Example 28

As well as the nouns, verbs are certainly very important. For a simple sentence containing only one verb, it is easy to identify it as the main verb of the sentence. But usually a longer sentence will have more than one verb, not saying that T can consist of more than one sentence. In fact, the verb we need is the predicate of the topic words, if we consider the nouns as arguments of the unknown predicate. Based on the dependency structure of the sentence, this is not too difficult to reach. But before describing the method, let us firstly take a closer look at the domain of the predicate argument structure, which will show the necessity of using dependency structure.

3.4.3 Overlapping Domains

A domain here means a certain range of words. If we take the previous example again, a domain could be the whole sentence containing the topic words, or the clause, or even smaller grammatical units. This is based on the plain text level. In this example, if we take the clause containing the topic words as our domain, the overlapping domains of **T** and **H** will be "*Typhoon Xangsane lashed the Philippine capital on Thursday*" and "*A typhoon batters the Philippines*". We can already see some improvement, if we extract the verb based on these overlapping domains. However, we will still have problems with parentheses or appositions like the first sentence of **T** in the following example (the words in bold are topic words),

Dataset=RTE3-dev	*Id=776*	*Task=SUM*	*Entailment=NO*	*Length=long*
Text: ***Yunus****, who was nominated for the peace prize at least twice before, is the first person from Bangladesh, a country of 147 million, to win a* ***Nobel Prize****. His Nobel Prize is a rare bright light in a country struggling to defeat Islamic terrorists, and which is chronically inundated by floods and battered by storms that blow in from the Bay of Bengal. The country, formerly known as East Pakistan, is bordered by the Bay of Bengal, India and Burma.*				
Hypothesis: ***Yunus*** *is the first person to win a* ***Nobel Prize****.*				

Example 29

Even if we can identify the verb properly, it is still unnecessary to include the information like "*on Thursday*" for entailment detection in the previous example, because "*Thursday*" does not appear in **H**, thus, it is not a topic word.

How can we restrict our domain into the most relevant and adequate one? We use dependency structure. Because of the characteristics of the dependency structure (3.3.1), it can be also viewed as a tree, namely dependency tree. The following figure shows the dependency tree of the **H** in the following example,

Dataset=RTE2-dev	*Id=61*	*Task=IE*	*Entailment=YES*
Text: Although they were born on different planets, Oscar-winning actor Nicolas Cage's new son and Superman have something in common - both were named Kal-el.			
Hypothesis: Nicolas Cage's son is called Kal-el.			

Example 30

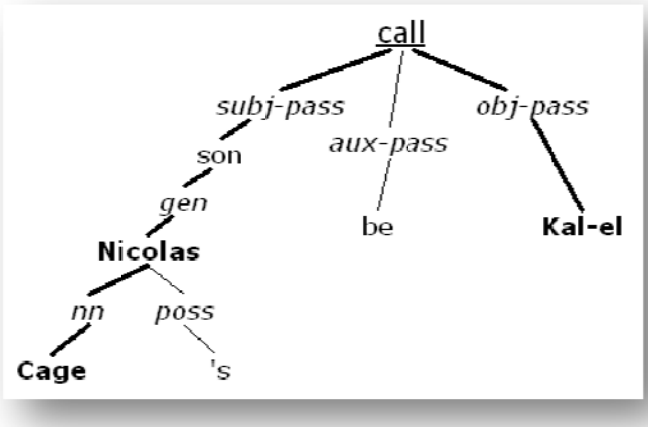

Figure 3 an Example of Dependency Tree

In the above figure, symbols in italics are dependency relations, words in bold are topic words, and the underlined word is the verb. The subtree connected with thick lines is the domain we are looking for. This kind of extended predicate argument structure (Gildea and

Palmer, 2002) consists of one or more dependency relation path from topic words (i.e. nouns) to the predicate (i.e. the verb). This domain ensures us 1) to include the most relevant information in **H** and **T** (i.e. all the topic words) and 2) to exclude all the irrelevant information in **H** (e.g. "*is*") and **T** (e.g. "*on Thursday*" in Example 28). In addition, it also solves the problem raised by parentheses and appositions, because this is the dependency structure of the sentence, which captures the syntactic/semantic relationship between words instead of the adjacency of words.

Furthermore, if we are given the topic words (i.e. nouns), the verb is not difficult to identify. With the help of dependency tree, we could easily reach the verb[12] via tracing up from the topic words on the tree. In the meanwhile, no matter how many topic words are there, we could always locate the lowest verb higher than all of them, i.e. the nearest common parent node of all the topic word nodes. Imagine that if we add "*Simon says*" before the **H** of Example 30, the verb "*say*" will be higher than "*call*" in the dependency tree. Notice that this is also important, because some verbs will reverse the polarity of the embedded statement. This thesis work does not cover this issue, which is also an option of future work. Some detailed study has been done by Nairn et al. (2006).

As a short summary, actually, what we have extracted out is a richer form of predicate argument structure without semantic role labeling. The richness lies on the dependency relation paths between the topic words and the verb, which is considered the source of our feature representation. In the next section, we will talk about how to obtain the feature representation on top of the overlapping domains of **T** and **H**.

3.4.4 Dissimilarity and Closed-Class Symbols

In retrospect, the identification of the topic words and the construction of the topic word pair set can be considered a preliminary classification on the **T**-**H** pairs. Roughly speaking, at least, it has classified all the cases into two groups: **T**-**H** pairs which have corresponding topic word pairs, and **T**-**H** pairs which do not have. Further, more subclasses can be derived in the first class, like two pairs group, three pairs group, etc. Since this step is checking whether **T** and **H** are about the same topics, it is a similarity checking between **T** and **H**. Thus, we can assume that, the first group contains more likely the positive cases than the negative cases. As an initialization of the later steps, we set the default Boolean value of entailment for each pair in

[12] In fact, it is not necessarily to be a verb. It could be a noun, a connective, or even a dependency relation, but our method can be also applied to those cases. Therefore, for convenience, we will keep using "the verb" to indicate this common parent node in this chapter and it will be substituted by a formal term with definition in the next chapter.

the first group as **positive** (i.e. "YES").

Consequently, the more overlapping information between **T** and **H** does not make sense to change the default value. Then we could focus on solely the dissimilarity based on the overlapping domains we have obtained from the previous steps. They are the essential parts which may reverse the answer to "NO". Let us look back at Example 24 again (repeated as follows),

Dataset=RTE3-dev *Id=739* *Task=SUM* *Entailment=NO* *Length=short*
Text: In an interview this week, the president said that Vice President ***Dick Cheney*** *and* ***Defense Secretary Donald Rumsfeld*** *will be a valued part of his administration until his presidency comes to an end.*
Hypothesis: ***Dick Cheney*** *is Vice President of the* ***Defense Secretary Donald Rumsfeld****.*

Example 24

If we perform the previous steps on this **T-H** pair, we will have the overlapping domain of **H** is almost the whole dependency tree except the determiner, "*the*", and the overlapping domain of **T** is the dependency tree for "*Dick Cheney and Defense Secretary Donald Rumsfeld*". Now, the topic words are not necessary any longer, because they have already finished their task – to set a default value of "YES". Therefore, after excluding the similar parts, we will have the remainder of the overlapping domain of **H** is the dependency tree of "*is Vice President of*", and the corresponding remainder of **T** is the dependency tree of "*and*"[13]. This dissimilar part will help us predict the answer (i.e. "NO") according to our assumption at the beginning of this subchapter. If it is a positive case, we hope the dissimilarity will appear "differently". The previous Example 25 is repeated as below,

Dataset=Rte3-dev *Id=34* *Task=IE* *Entailment=YES* *Length=short*
Text: Parviz Davudi was representing Iran at a meeting of the Shanghai Co-operation Organisation (SCO), the fledgling association that ***binds*** *Russia, China and four former Soviet republics of central Asia* ***together*** *to fight terrorism.*
Hypothesis: China ***is a member of*** *SCO.*

Example 25

In this example, the dissimilarity between the two overlapping domains will be the dependency trees of "*binds ... together*" and "*is a member of*". It is obviously different from the Example 24's, because they use different expressions to deliver totally different information. Therefore, we have two questions to answer: 1) How to generalize these

[13] Note that this dependency tree is not a one-node tree, but a tree contains a root node (i.e. "*and*") and two child nodes "cc", which stands for the dependency relation between the connective and the coordinate. The same as "*is Vice President of*", which also keeps the dependency relations with the topic words excluded out.

dissimilarities and 2) how to extract features after the generalization. We will answer the first question in this section and leave the second one for the next section.

Fortunately, we have part-of-speech (POS) on hand, which is very easily to obtain via shallow parsing techniques. The POS taggers of all the words nicely generalize the expressions into a new form which can be expressed in a symbol set of a very limited size. Together with the dependency relation names, the dependency tree in all can be changed into a generalized form using a finite number of symbols, named ***Closed-Class Symbols*** (CCSs).

After the generalization, we can say that the remaining part is our feature space, named as Pattern of **T** and Pattern of **H**, and together as ***Entailment Pattern*** (EP) afterwards. The coming problem is how to represent the features or how to extract them, which we will see in the next section.

3.4.5 Subsequence Kernels

If all the cases are like Example 24 and Example 25, life will be much easier. Firstly, they have the same pattern of **H** after generalization. Secondly, the pattern of **T** is also simple and meaningful. Consequently, we can lightly generate a rule based on the EP. However, there are some complex cases, such as the following example,

Dataset=RTE3-dev	*Id=693*	*Task=SUM*	*Entailment=YES*	*Length=short*
Text: More than 6,400 migratory birds and other animals were killed in Nevada by drinking water in the cyanide-laced ponds produced by gold mining operations.				
Hypothesis: Animals have died by the thousands from drinking at cyanide-laced holding ponds.				

Example 31

We can obtain the EP using the previous steps we have discussed before as follows,

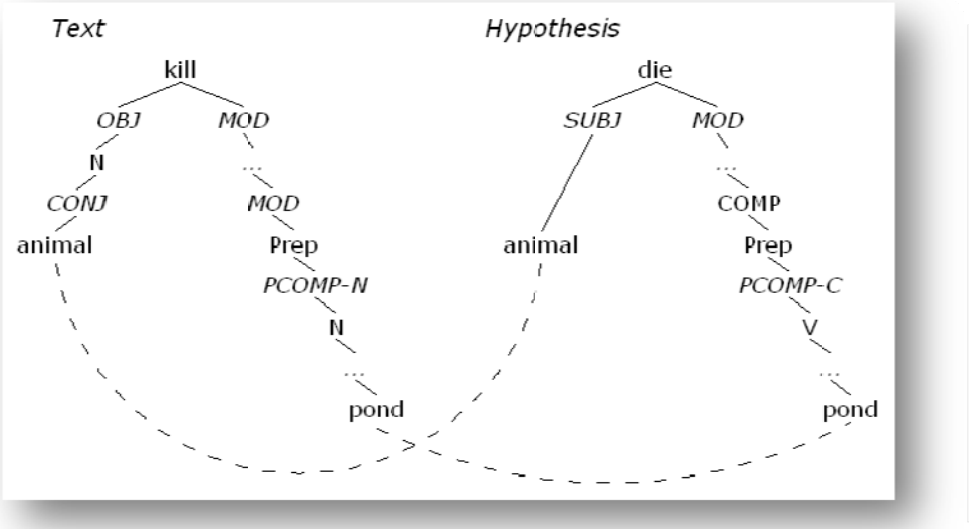

Figure 4 an Example of the EP

Though they are parts of the dependency structures, they still look quite complex. It is difficult to extract some general rules from them; it is also not elegant to use it directly as a rule, because this rule could appear only once in the whole data set. In addition, if we try to make a more generalized form from EP, the simple cases (e.g. Example 24 and Example 25) might be over-generalized into a trivial form.

When we do not explicitly know how the features look like, we ask help for kernel-based methods (Vapnik, 1995). There are some available methods, called *Convolution Kernels* (Haussler, 1999), for tackling with complex structures rather than numeric vectors. In essence, these kernel-based methods enumerate all the possible parts of the structure to avoid knowing which part is more important than the others. These methods, such as the *Tree Kernel* (Moschitti, 2004), encode all the possible subtrees in a Boolean value vector to represent the tree structure. However, it is designed for constitute parsing tree; therefore, it could not work on dependency trees. There is another group of kernels utilized on linear representations, like *String Kernels* (Lodhi et al., 2001), *Subsequence Kernels* (Bunescu and Mooney, 2006), etc. Especially, Bunescu and Mooney (2006) applied their method in the relation extraction task, which is quite relevant task to RTE.

We will firstly answer the question how to apply subsequence kernel methods to our current EP. Although we have successfully represented the dissimilarities using a set of CCSs (i.e. the EP), as a whole, they are still tree structures. After that, we will show the advantages of applying this method.

If we look at the EP again (Figure 4), in fact, the trees contained in the EP are not usual trees. Each pattern (i.e. each tree) contains a common parent node and several "branches" or paths from the parent node to all the topic word nodes. In fact, this characteristic comes from the construction of the pattern. Since we identify the topic words first and then trace up in the dependency tree to find the common parent node, the pattern can be broken down into several paths naturally, just taking the opposite way of the process. This characteristic can ensure the application of subsequence kernel methods, as our pattern is a kind of sublinear representation.

The subsequence kernel methods are basically enumerating all the possible subsequences contained in a sequence. Therefore, each element of the feature vector will be a Boolean value indicating whether a particular subsequence is contained or not. Furthermore, which particular subsequence of CCSs is important or meaningful is not necessary to consider anymore, because all the possibilities are encoded in the feature vector and the machine learning procedure will assign different weights for each subsequence based on the training data set. That is the most advantage of using subsequence kernel methods.

At the end of this section, we will discuss a practical problem. As mentioned before, **T-H** pairs could have different numbers of topic word pairs. The following two examples contain one topic word pair and four topic word pairs respectively,

Dataset=RTE2-dev Id=307 Task=IR Entailment=YES
*Text: Napkins, invitations and plain old paper **cost** more than they did a month ago.*
*Hypothesis: The **cost** of paper is rising.*
Topic Word Pair Set: {<"cost", "cost">}

Example 32

Dataset=RTE3-dev Id=768 Task=SUM Entailment=NO Length=short
*Text: The **FDA** would not say in which **states** the **pills** had been sold, but instead recommended that customers determine whether products they bought are being recalled by checking the store **list** on the FDA Web site, and the batch list. The batch numbers appear on the container's label.*
*Hypothesis: The **FDA** provided a **list** of **states** in which the **pills** have been sold.*
Topic Word Pair Set: {<"FDA", "FDA">, <"list", "list">, <"states", "states">, <"pills", "pills">}

Example 33

In practice, we have only dealt with **T-H** pairs containing two topic word pairs; because the

structure is clearer and meaningful (there are a dominant number of two-argument predicates). But in principle, our method can be applied to all the other cases, which is considered our future work as well.

3.4.6 Parent Node and its Adjacent Dependency Relations

This section is about some complementary features extracted from the parent node and its adjacent dependency relations. As well as all the implicit features encoded in the subsequence kernel, explicit features around the parent node also attract our attention. The following example shows the great importance of them,

Dataset=RTE3-dev	*Id=1*	*Task=IE*	*Entailment=YES*	*Length=short*
Text: The sale was made to pay Yukos' US$ 27.5 billion tax bill, Yuganskneftegaz was originally sold for US$ 9.4 billion to a little known company Baikalfinansgroup which was later ***bought*** *by the Russian state-owned oil company Rosneft .*				
Hypothesis: Baikalfinansgroup was ***sold*** *to Rosneft.*				

Example 34

The parent nodes of the patterns in this example are "*bought*" and "*sold*", which are antonyms. Since the generalization step (3.4.4) will exclude all the word information and only keep the POS tags, this lexical semantic information will not be preserved. Unfortunately, this will directly generate the wrong answer.

In the meanwhile, the directionality of actions makes life even more difficult. Some relationships between two participants have no direction, that is, the two participants can be interchanged freely. However, the two verbs in this example do not belong to such cases. If we say "A sells something to B", it means "B buys something from A". Therefore, we should take care of the order of the participants in such actions or event.

Let us get back to our approach. The second problem has been solved via adding another feature called ***Verb Relation Consistency*** to check whether the same participants play the same roles in both **T** and **H**. The first problem has not been accounted for in the current version of our system. It is not easy to know the relationship between two verbs, without using any external semantic resources. Now we just make a dummy feature called ***Verb Consistency*** and leave it for the future work.

3.5 Summary

In this chapter, we have talked about our hybrid approach consisting of a kernel-based method plus two backup strategies, a BoW method and a method based on local dependency relation. We have shown both the advantages and disadvantages of the BoW method, which lead us to use the dependency structure. The method based on local dependency relation can solve some problems left behind by the BoW method, but it also has its limitations, which motivate us to use a more effective method on top of cross-pair features. At last, our main approach is described step by step with many examples and also some optional future directions are pointed out during the process.

Chapter IV

The *TERA* System

In this chapter, we will discuss in detail the system implementation. Based on the observations on the data and methods proposed in the previous chapter, our system, ***TERA*** (Textual Entailment Recognition and Applications), is implemented in Java 1.5. We will start with external tools or packages used in our system; then focus on the architecture and all the components; and finally the summarization of this chapter will be presented.

4.1 Used Tools

This section will mainly focus on the external tools applied in our system. Since Java has a good plug-in feature, we can conveniently add packages, even software into our system. Currently, we have utilized two dependency parsers, a machine learning tool, and other free resources like sentence boundary detector, XML file reader and writer, etc.

4.1.1 Dependency Parsers: Minipar and Stanford Parser

There are many available dependency parsers for the English language, such as Minipar[14]; other parsers like the Stanford Parser[15] can also provide us with a dependency structure. We will introduce them one by one and describe how we will use them.

Minipar

Minipar is a descendent of PRINCIPAR (Lin, 1993a; Lin, 1994), which is an efficient, broad-coverage, principle-based English dependency parser. Minipar adopted some of the ideas in the Minimalist Program (Chomsky, 1995), such as bare phrase structure and economy principles. The input of Minipar is a sentence, and the output is the dependency structure of the sentence. For example,

[14] http://www.cs.ualberta.ca/~lindek/minipar.htm
[15] http://www-nlp.stanford.edu/downloads/lex-parser.shtml

Input: *I ate a big fish.*

Output:

```
(
E0      (()     fin C   *       )
1       (I      ~ N     2       s       (gov eat))
2       (ate    eat V   E0      i       (gov fin))
E2      (()     I N     2       subj    (gov eat)       (antecedent 1))
3       (a      ~ Det   5       det     (gov fish))
4       (big    ~ A     5       mod     (gov fish))
5       (fish   ~ N     2       obj     (gov eat))
6       (.      ~ U     *       punc)
)
```

Example 35

The first column is the id of each token; the second column is the word form; the third column is the lemma of the word; the fourth column is the governor of the current token (i.e. its parent node); the fifth column is the dependency relation between the current token and its governor; the sixth column specifies the token which the current token governs; and the seventh column is for grammatical usage, which only grammatical functional nodes have. Note that not all the tokens have all the 7 columns; for instance, the root of the dependency tree has no governor. In practical use, we have transformed the output format into the triple representation (4.2.3).

An evaluation of Minipar with the SUSANNE corpus shows that Minipar achieves about 88% precision and 80% recall with respect to dependency relationships, which runs on a Pentium II 300 with 128MB memory, and parses about 300 words per second.

Stanford Parser

The Stanford Parser (Klein and Manning, 2003) is a probabilistic natural language parser, which is a both highly optimized PCFG (Johnson, 1998) and lexicalized dependency parser and a lexicalized PCFG parser. Varying from rule-based parsers, probabilistic parsers use knowledge of language gained from hand-parsed sentences to try to produce the most likely analysis of new sentences. The input of the Stanford Parser is also a sentence, but the output has both the phrase structure and the dependency structure of the sentence, because the dependency structure can be generated from the phrase structure parse (de Marneffe et al., 2006a). In this thesis, we will only use the POS tags and the dependency structure of the Stanford Parser output. Here is an example,

Input: *I ate a big fish*

Output:
POS tags: *I/PRP ate/VBD a/DT big/JJ fish/NN ./.*
Typed Dependencies:
nsubj(ate-2, I-1)
det(fish-5, a-3)
amod(fish-5, big-4)
dobj(ate-2, fish-5)

Example 36

Notice that both the POS tags and dependency relation names used in the Stanford Parser output are different from those of the Minipar output, therefore, in implementation we have used some rules to adapt these names into the ones used in Minipar. See all the transformation rules in Appendix. The transformation from the output to the triple representation will be the same as the Minipar output.

The current version of the parser requires Java 5 (JDK1.5). The parser also requires plenty of memory (about 100Mb to run as a PCFG parser on sentences up to 40 words in length). Its performance is 86.36% (F1).

To briefly conclude, we have chosen these two parsers, in order to see whether a rule-based parser and a statistical parser will have varying results for the same task (more details in 5.3.2).

4.1.2 Machine Learning Tool: Weka

Weka[16] (Witten and Frank, 1999) is a collection of machine learning algorithms for data mining tasks. The algorithms can either be applied directly to a dataset or called from users' own Java code. Weka contains tools for data pre-processing, classification, regression, clustering, association rules, and visualization. It is also well-suited for developing new machine learning schemes. The SMO classifier (Platt, 1998) is used for all the machine learning procedures, since 1) it has been shown as one of the effective methods by other research groups (2.2.3) and 2) it is quite stable according to some preliminary tests done by us. We have also used its Java API for integration the machine learning process into the whole system, as well as the working environment provided by Weka, as shown below,

[16] http://www.cs.waikato.ac.nz/ml/weka/

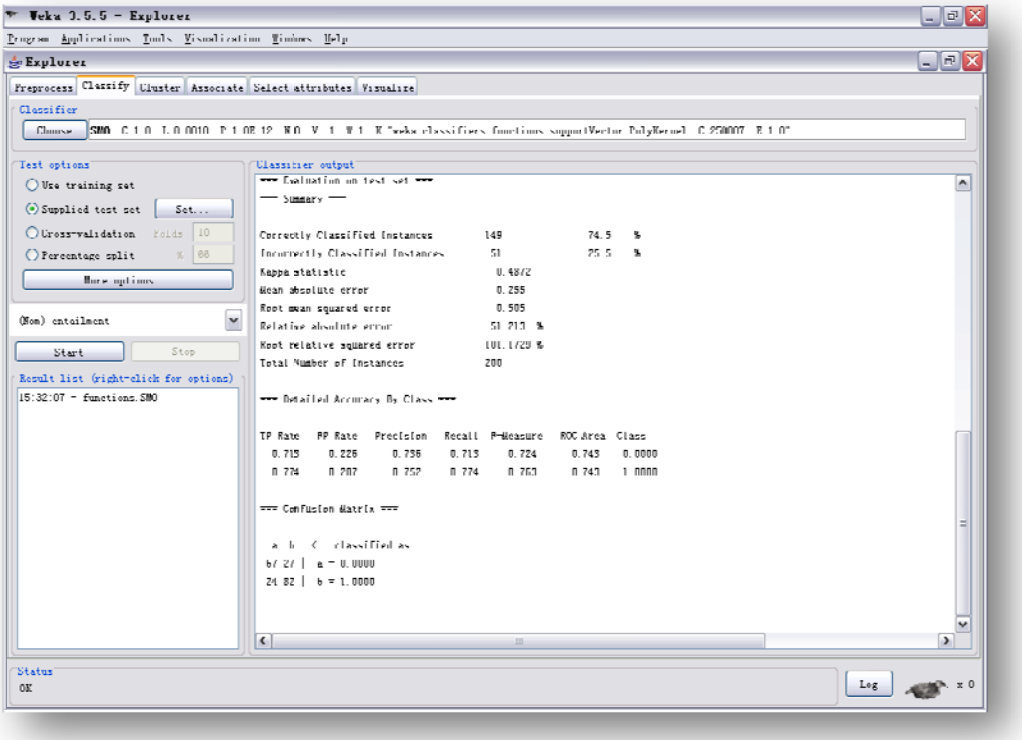

Figure 5 A Snapshot of the Weka Tool

4.1.3 Other Tools

Some other tools are also applied in ***TERA***, such as OpenNLP[17] and JDOM[18].

OpenNLP is hosts a variety of java-based NLP tools which perform sentence detection, tokenization, pos-tagging, chunking and parsing, named-entity detection, and coreference using the OpenNLP Maxent[19] machine learning package. In our system, we have only utilized it as a sentence boundary detector, since the input for both Minipar and the Stanford Parser is a single sentence.

JDOM provides a solution for using XML from Java. It interoperates well with existing standards such as the Simple API for XML (SAX) and the *Document Object Model* (DOM), and it provides a robust, light-weight means of reading and writing XML data without the complex and memory-consumptive options that current API offerings provide.

[17] http://opennlp.sourceforge.net/
[18] http://www.jdom.org/
[19] http://maxent.sourceforge.net/

4.2 System Description

Our system, ***TERA***, is a RTE-centered NLP application framework. Our goal involves using the RTE subsystem as a core engine and applying it to several applications, such as *Relation Extraction*, *Answer Validation*, *Paraphrasing Acquisition*, etc. The following figure is the whole architecture of ***TERA***,

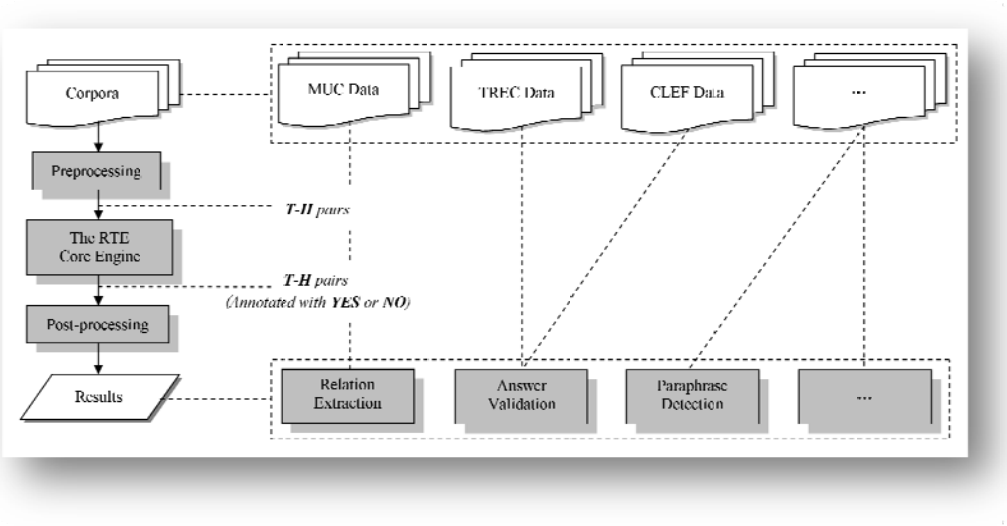

Figure 6 Framework of the *TERA* System

Figure 6 gives out a generic architecture of the entailment-based applications. Usually, **T**-**H** pairs are constructed from different corpora by the preprocessor, and then sent to the RTE core engine (see 4.2.1), which will be further discussed immediately afterwards. After obtaining the **T**-**H** pairs with annotation of "*YES*" or "*NO*", we will do some post-processing to adapt the results into various applications. For example, for the Answer Validation task, **H**s will be constructed using questions and their answers, and the corresponding **T**s will be the supporting or irrelevant documents (or snippets). After knowing whether the entailment relation exists in each **T**-**H** pair or not, the result (usually "*validated*" or "*rejected*") for each question-answer pair will be generated with no difficulty.

In this chapter, we will mainly focus on the RTE core engine, and leave the other parts to Chapter VI. In the coming section, we will begin with the architecture of the RTE subsystem.

4.2.1 Overview of the RTE Core Engine

Our RTE subsystem has a module-oriented organization. Each of the components is responsible for a step of the whole workflow. The following figure shows the whole

architecture of the RTE subsystem,

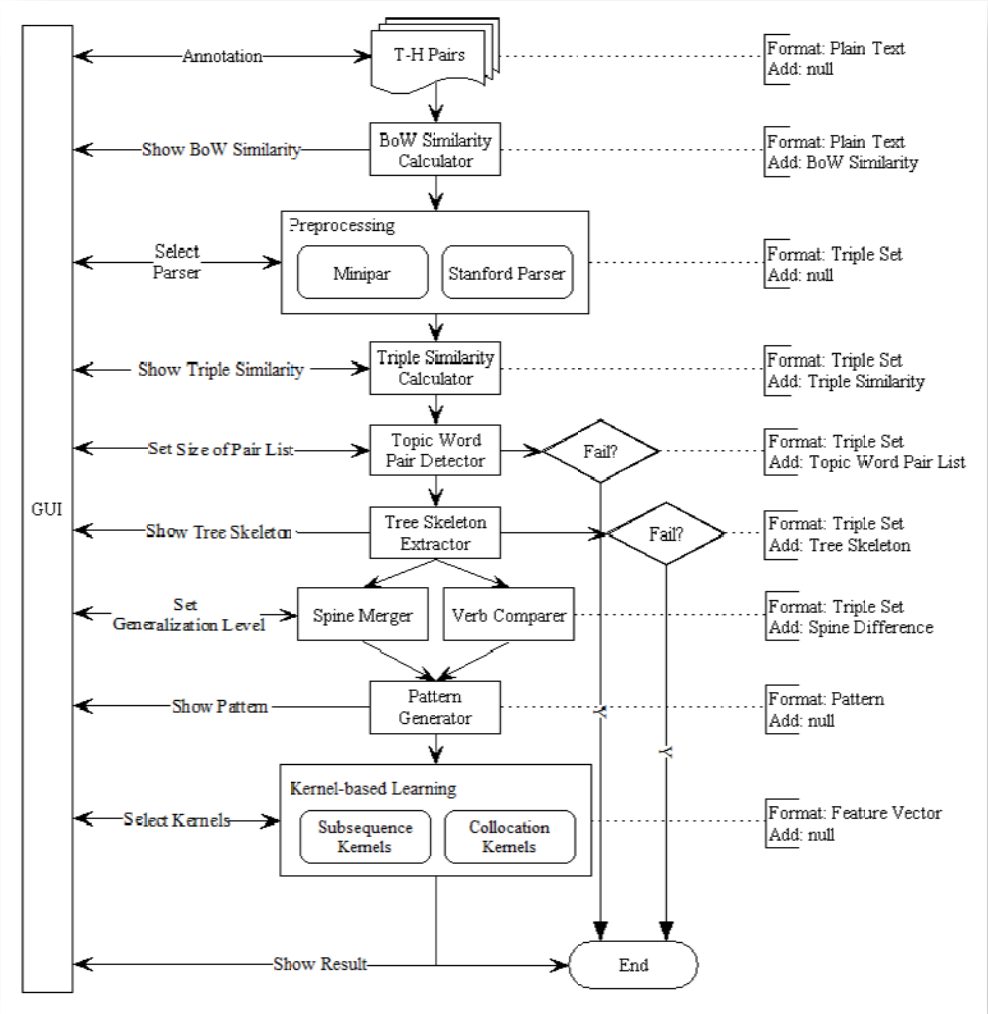

Figure 7 Architecture of the RTE Subsystem

The input of the RTE subsystem is **T-H** pairs in XML format and the output is those pairs with "*YES*" or "*NO*" annotations. It consists of several components to obtain *Entailment Patterns* (EPs) which defines our feature space, plus a *Machine Learning* (ML) module which applies *Subsequence Kernels* (SKs) to perform the binary classification (i.e. "*Yes*" or "*NO*") on the EPs. These components for acquiring the EPs are used in both training and testing phrases. Regarding the ML process, the input data for training are annotated data, while for testing are unannotated data.

Our approach will firstly calculate the BoW similarity between **T** and **H** as one of our backup strategies using a straightforward word overlapping method (4.2.2). Secondly, it will apply dependency parsers to parse both **T** and **H** and obtain the dependency structures. Based

on the local dependency relations, another similarity score will be calculated as the other backup strategy (4.2.3).

After that, our main approach will start. It is based on the hypothesis that *some particular differences between **T** and **H** will reverse the entailment relationship*. As we mentioned in 3.4.4, when judging the entailment relation, we initially assume that the relationship holds for every **T**-**H** pair (using the default value "*YES*"). Then, the following steps are performed one by one:

1) Generate *Topic Word* (TW) Pair List using **T** and **H** (4.2.4): We discover the TWs in **H**, and locate the corresponding words or phrases in **T** using some partial matching techniques.

2) Extract Tree Skeletons of **T** and **H** (4.2.5): We begin with the TWs found in the former step, and trace up along the dependency trees of **T** and **H** respectively, in order to find the lowest common parent nodes. On the dependency tree of **H**, this common parent node is usually a verb, whereas for **T**, it is not necessary to be the case. It could be a noun, a preposition, or even a dependency relation. We then define the lowest TWs as *Foot Nodes* (FNs), and the common parent nodes as *Root Nodes* (RNs). If there are only two FNs, the left-most one will be called *Left FN* and the right-most one *Right FN*. As a whole, the subtree without inner yield is named as *Tree Skeleton* (TS).

3) Generalize the Spines and merge them in each **T**-**H** pair (4.2.6): The dependency path from a FN to the RN (both exclusive) is called a *Spine*. We will generalize all the spines by ignoring the lemmas (i.e. preserving the POS tags) and substituting some dependency relation names. The remaining symbols form the *Closed-Class Symbol* (CCS) set. The merging process is performed via excluding the longest common prefixes and the longest common suffixes. The remained parts are named as *Spine Difference* (SD). If there are only two Spines (i.e. two FNs), there will be a *Left SD* (LSD) and a *Right SD* (RSD).

4) Acquire an entailment pattern for each **T**-**H** pair (4.2.7): An *Entailment Pattern* (EP) is a quadruple in the form of <*LSD, RSD, VC, VRC*>. The first two parts are obtained in the previous steps; VC stands for *Verb Consistency*, and VRC stands for *Verb Relation Consistency*. The last two parts are introduced in 3.4.6, which check whether the two RNs (from **T** and **H** respectively) and their adjacent dependency relations are consistent or not.

5) Apply subsequence kernels to the acquired EPs and perform the machine learning (4.2.8): the subsequence kernel method is used to represent LSD and RSD. Together with two trivial kernels checking VC and VRC, we combine these kernels into one and perform the machine learning process. We also consider another kernel called *Collocation Kernel* which checks the co-occurrence of subsequences in LSD and RSD between **T** and **H**. The binary classification is performed using the trained model.

In conclusion, the RTE subsystem consists of a main approach and two backup strategies. The main approach extracts parts of the dependency structures to define a new representation for feature extraction and then applies kernel methods to perform ML; and the two backup strategies will deal with the cases which cannot be handled by the main approach. One of them is a simple BoW method; and the other is calculating similarity using local dependency relations. The system makes use of these three approaches to process different entailment cases in practice.

In addition, at the end of this subchapter, we briefly present the *Graphic User Interface* (GUI) of our RTE subsystem, which makes the experimentation more convenient and friendlier to the users as well.

4.2.2 Bag-of-Words Similarity Calculation

The BoW similarity is calculated in a straightforward way. Since this score is used as one of our backup strategies, the robustness is more important than other criteria. As mentioned in 3.2, BoW methods work fairly well on some cases, especially some "difficult" examples from the human's point of view. While the results show the advantages of this method, they also set up a high baseline in the experimentation (5.2).

In implementation, we just split the input string into tokens according to the spaces in-between without using any other preprocessing techniques. After that, the number of overlapping tokens between **T** and **H** is counted, and so as the total number of tokens in **H**. The BoW similarity score is defined as,

$$the\ BoW\ Similarity\ Score = \frac{number\ of\ overlapping\ tokens\ in\ \boldsymbol{T}\ and\ \boldsymbol{H}}{number\ of\ tokens\ in\ \boldsymbol{H}}$$

Equation 1 Definition of the BoW Similarity

Here is an example describing how it works,

Dataset=RTE2-dev Id=625 Task=IE Entailment=NO
*Text: The loss offered a minor moral victory for **Liverpool**, as they scored only the second goal this season against **Chelsea** in league play.*
*Hypothesis: **Liverpool** beat **Chelsea**.*
Number of Overlapping Tokens: 2;
*Number of Tokens in **H**: 3*
*The BoW Similarity Score: **0.6667***

Example 37

As well as the BoW similarity, we also consider another similarity metrics based on local dependency relations, which will be introduced in the next section.

4.2.3 Preprocessing and Triple Similarity Calculation

In 3.3.1, we have introduced the dependency structure. In practice, we have utilized two dependency parsers, Minipar and the Stanford Parser (4.1.1). Though they vary in POS tags and dependency relation tags, usually, the output dependency structures (if both are correct) will be the same. The following figure shows the two dependency trees of the same input sentence computed by the two parsers respectively,

I give Tom a nice book of mine.

Example 38

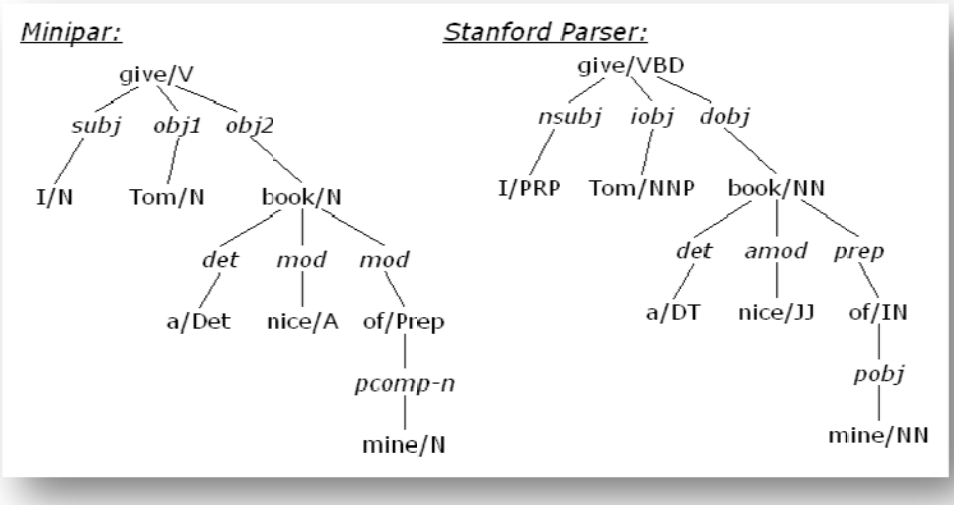

Figure 8 Comparison between the outputs of Minipar and Stanford Parsers

In Figure 8, we can see that the two parsers have different naming systems for POS tags

and dependency relations. In the practical system, a subset of Minipar's tags has been used. Since we need to do generalization of both the POS tags and dependency relation tags in the later stage (4.2.6), we will put the unification of the different tags in that stage as well. For convenience, all the examples in the rest of the thesis will use Minipar's tags and **T** and **H** will denote either the original texts or the dependency structures.

In order to calculate the overlapping ratio of the local dependency relations, we will introduce a new form of the dependency structure now, namely *Triple Representation*. See the triple representation of the dependency structure of Example 38 as follows,

give:V subj I:N
give:V obj1 Tom:N
give:V obj2 book:N
book:N det a:Det
book:N mod nice:A
book:N mod of:Prep
of:Prep pcomp-n mine:N

Table 5 Triple Representation of the Dependency Structure of Example 38

Basically, each triple consists of three elements in the format of <*Node1, Relation, Node2*>, where Node1 represents the head (i.e. governor or parent node), node2 the modifier (i.e. child node), and Relation the dependency relation. The inner structure of each node consists of the lemma and the POS tag, separated by a colon. Since this form breaks down the whole dependency parse tree into a set of triples, the local dependency relations are captured by this node-edge-node representation.

On top of this triple representation, we construct another similarity function which operates on two triple sets and determines how many triples of **H** are contained in **T**. The core assumption here is that the higher the number of matching triple elements, the more similar both sets and the more likely it is that **T** entails **H**. The similarity checker of two triples makes use of an approximate matching function,

```
TRIPLE-MATCH (<Tn1, Tr, Tn2>, <Hn1, Hr, Hn2>):
  if (Tn1 = Hn1 & Tr = Hr & Tn2 = Hn2):
    return FullMatch;
  elseif (Tn1 = Hn1 & Tr = Hr):
    return LeftMatch
  elseif (Tr = Hr & Tn2 = Hn2):
    return RightMatch
  elseif (Tn1 = Hn1 & Tn2 = Hn2):
    return ArgsMatch
```

Equation 2 Triple Matching Function

Note that in all cases a successful match between two nodes requires that they share the same lemma and the same POS tag. The triple matching function is applied to the series of triples of **T** and **H**, ignoring sentence boundaries. The motivation for returning the different matching cases is to perform a partial match instead of an exact one. Different cases (i.e. ignoring either the parent node or the child node, or the relation between nodes) might provide different indications for the similarity between **T** and **H**. Consequently, the similarity function can be defined more precisely based on the sum of the matched triple elements of **H** divided by the cardinality of **H** needed for normalization,

$$\begin{aligned} &the\ Triple\ Similarity\ Score \\ &= \frac{1}{Card(\boldsymbol{H})} \times (a_1 \times NumOfFullMatch + a_2 \times NumOfLeftMatch \\ &+ a_3 \times NumOfRightMatch + a_4 \times NumOfArgsMatch) \end{aligned}$$

Equation 3 Definition of the Triple Similarity Score

Inside the equation, *Card(**H**)* is the number of triples in **H**; a_1 to a_4 are the different weights for the different matching cases.

Normalizing the sum of matching elements by the cardinality of **H** guarantees that the Triple Similarity Score will be in the closed interval [0, 1]. A value of 0 means that **H** has nothing in common with **T**, a value of 1 means that **H** is completely covered by **T**, and a value in-between means that **H** is partially covered by **T**.

The weights (i.e. a_1 to a_4) learned from the corpus imply that the different "amount of missing linguistic information" influences the entailment relation differently.

If a_1=1, a_2=a_3=0.5, and a_4=0.75, and the Triple Similarity Score for the following example will be calculated,

Dataset=RTE3-dev Id=73 Task=IE Entailment=YES Length=long
Text: On October 1 2001, EU and other countries introduced the option for domestic animal owners to apply for Pet passports under the Pets Travel Scheme (PETS for short), for pets returning from abroad to the United Kingdom. This replaced the old system of 6 months compulsory quarantine for all domestic pets.
Hypothesis: In 2001, the EU introduced a passport for pets.
***FullMatch**:< fin:C punc ,:U>, <fin:C i introduce:V>, <introduce:V subj EU:N>, <for:Prep pcomp-n pet:N> → 4*
***LeftMatch**:<fin:C mod In:Prep>, <introduce:V obj passport:N>, <passport:N mod for:Prep> → 3*
***RightMatch**: null → 0*
***ArgsMatch**: null → 0*
*The Triple Similarity Score: (1×4+0.5×3+0+0)/10=**5.5***

Example 39

From the discussions in 3.3.2, we also find that the method based on local dependency relations (i.e. the Triple Similarity Score) has its shortcomings. From the next section, we will start introducing our main approach.

4.2.4 Topic Word Pair Detection

This is the first process of the main approach, which consists of two steps: 1) extracting Topic Words (TWs) from **H**; and 2) locating the corresponding TWs in **T**.

For the first step, we take the assumption that *most of the meaning of a sentence is conveyed via content words instead of function words*. Furthermore, the nouns or noun phrases are usually the topics of the sentence. For the second step, if the entailment relationship holds between **T** and **H**, they will at minimum share some common topics or semantically relevant[20] topics. These are the motivations for this process.

Given a sentence after POS tagging, it is not difficult to identify all the nouns; the TWs of **H** will thus be easily extracted. The difficulty lies in how to correspond the TWs to those ones in **T**. In Chapter III, we have seen some examples (e.g. Example 28) with various expressions referring to the same entities. More examples are illustrated as follows,

[20] Here semantic relevance refers to *Hyponymy* and *Hypernymy* relationships.

Dataset=RTE3-dev	*Id=390*	*Task=IR*	*Entailment=YES*	*Length=short*
Text: ***Typhoon Xangsane*** *lashed the* ***Philippine*** *capital on Thursday, grounding flights, halting vessels and closing schools and markets after triggering fatal flash floods in the centre of the country.*				
Hypothesis: A ***typhoon*** *batters the* ***Philippines****.*				
Topic Word Pairs: {<"Typhoon_Xangsane", "typhoon">, <"Philippine", "Philippines">}				

Example 28 (again)

Dataset=RTE3-dev	*Id=35*	*Task=IE*	*Entailment=YES*	*Length=short*
Text: A leading human rights group on Wednesday identified Poland and Romania as the likely locations in eastern Europe of secret prisons where al-Qaeda suspects are interrogated by the ***Central Intelligence Agency****.*				
Hypothesis: ***CIA*** *secret prisons were located in Eastern Europe.*				
Topic Word Pairs: {<"Central Intelligence Agency", "CIA">, <"prisons", "prisons">, <"eastern Europe", "Eastern Europe">}				

Example 40

To handle these problems, we have applied several partial matching techniques at the substring level, such as partially inclusive matching, acronym, etc. In Example 28, "*typhoon*" in **H** will be related to "*Typhoon Xangsane*" in **T**, as the former word is contained in the latter phrase. In order to relate "*Philippines*" in **H** with "*Philippine*" in **T**, we have used a criterion for a successful matching that most of the letters are overlapping instead of all of them. In practice, we have set a threshold for the overlapping ratio of 60% and the lengths of both words of 5 letters. Example 40 raises another problem. In **T**, the organization appears in the full name "*Central Intelligence Agency*", while in **H**, the abbreviation (i.e. acronym in this case) "*CIA*" is used. We have grouped all the initial letters of the words contained in a noun phrase and make them into upper case, so as to help to locate the corresponding TW. This also works well with "*European Union*" referring to "*EU*", "*United States*" referring to "*US*", etc.

After several matching strategies have been applied to detect the TW pairs, different numbers of TW pairs may be extracted from different **T**-**H** pairs. If there is no TW pair, this **T**-**H** pair will be delegated directly to the backup strategies; otherwise, it will be passed to the next stage of the main approach.

4.2.5 Tree Skeleton Extraction

After obtaining the TW pair(s) for each **T**-**H** pair, we will mark them on the dependency parse trees of both **T** and **H**. Then, we will trace up from all the TWs to reach the lowest common

parent node on the trees of **T** and **H** respectively. The following figure illustrates the result of this process,

Dataset=RTE2-dev *Id=61* *Task=IE* *Entailment=YES*
Text: Although they were born on different planets, Oscar-winning actor Nicolas Cage's new son and Superman have something in common - both were named Kal-el.
Hypothesis: Nicolas Cage's son is called Kal-el.

Example 41

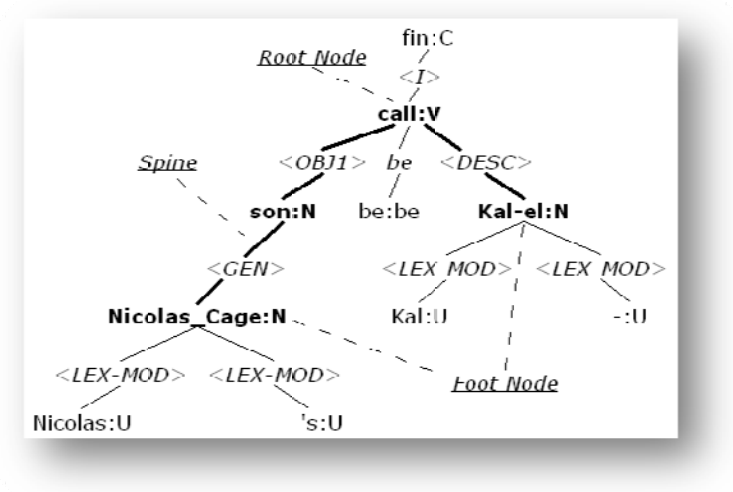

Figure 9 Tree Skeleton of H of Example 41

We name the subtree (i.e. the part in bold in Figure 9) ignoring the inner yield as the *Tree Skeleton* (TS); the lowest common parent node is named the *Root Node* (RN); each lowest TW is named a *Foot Node* (FN); and the dependency path from each FN to the RN (both exclusive) is called a *Spine*.

Notice that in most cases, the RN of **H** is a verb, but the RN of **T** might not be, such as in the following example (words in bold are TWs),

Dataset=RTE2-dev *Id=534* *Task=IE* *Entailment=NO*
Text: The main library at 101 E. Franklin St. changes its solo and group exhibitions monthly in the Gellman Room, the Second Floor Gallery, the ***Dooley Foyer*** *and the* ***Dooley Hall****.*
Hypothesis: ***Dooley Foyer*** *is located in* ***Dooley Hall****.*

Example 42

In Example 42, the RN of **H** is "*located*", while the RN of **T** is "*and*". Since our algorithm still works well on this case, it will not be a problem whether the RN of **T** is a verb or not. But we do need to take a closer look at the numbers of spines.

In the previous Example 41, there are two spines in **H**, while the following example has only one spine,

Dataset=RTE2-dev *Id=307* *Task=IR* *Entailment=YES*
*Text: Napkins, invitations and plain old **paper cost** more than they did a month ago.*
*Hypothesis: The **cost** of **paper** is rising.*
Topic Word Pair: {<"paper", "paper" >}

Example 43

There is only one pair of TWs in Example 43, because the current version of our algorithm cannot relate the "*cost*" in **H**, which is a noun, with the "*cost*" in **T**, which is a verb. Therefore, there is no RN (i.e. lowest common parent node) for both **T** and **H** in this pair. So far, this kind of **T**-**H** pairs have not been covered by our main approach, which will be delegated to the backup strategies. However, if several TWs belong to different sentences (i.e. each sentence has one TW), a dummy parent node will be added to connect the dependency trees, thus the spines.

Not only can too few spines cause problems, but also too many spines. If there are more than two spines, such as in the following example,

Dataset=RTE2-dev *Id=133* *Task=SUM* *Entailment=NO*
*Text: **Verizon Communications Inc.** **said** on Monday it would buy long-distance **telephone** company **MCI Communications Inc.** in a deal worth **$6.75 billion**, giving Verizon a foothold in the market for serving large corporations.*
*Hypothesis: **Verizon Communications Inc.**'s **$6.7 billion** takeover of long-distance provider **MCI Inc.** **transformed** the **telephone** industry.*

Example 44

In Example 44, there are four spines under the RNs, "*said*" and "*transformed*". Since in the later stages (4.2.8) we will use subsequence kernels to represent parts of the TSs, the cases like this example will also not covered currently. Principally, cases with more spines could be solved with some redundancy in representations, but we will leave it for future work (7.2).

To briefly summarize, the prerequisites for the current version of our main approach is: 1) TW pairs are found; 2) Two but only two spines are contained in the TSs of both **T** and **H**. According to our experimental results, among all the 800 **T**-**H** pairs of the RTE-2 test set, we successfully extracted TSs in 296 pairs, i.e., 37% of the test, and for RTE-3 test set, the percentage is 36% (5.3).

4.2.6 Spine Generalization and Merging

Now we have two TSs, one of **T** and the other of **H**. Before acquiring dissimilarity between them, some generalization is necessary for making the comparison less strict. In essence, each spine is a sequence of dependency relations and words (with their POS tags), thus, the generalization will also have aspects: 1) generalizing the dependency relations; and 2) generalizing the words.

The next two examples illustrate the motivation of the first aspect,

Dataset=RTE3-test Id=110 Task=IE Entailment=YES Length=short
Text: Leloir was promptly given the Premio de la Sociedad Científica Argentina, one of few to receive such a prize in a country in which he was a foreigner.
Hypothesis: Leloir won the Premio de la Sociedad Científica Argentina.
*Some Dependency Relations in **T**: give:V **<OBJ1>** Leloir:N; give:V **<OBJ2>** Sociedad_Científica_Argentina:N; …*
*Some Dependency Relations in **H**: win:V **<SUBJ>** Leloir:N; win:V **<OBJ>** Sociedad_Científica_Argentina:N; …*

Example 45

Dateset=RTE3-dev Id=542 Task=QA Entailment=YES Length=short
Text: Even while accepting the Russian plan, IMF Managing Director Michel Camdessus noted that the efficiency of Russia's State Taxation Service "is declining rapidly."
Hypothesis: Michel Camdessus is managing director of IMF.
*Some Dependency Relations in **T**: Michel_Camdessus:N **<PERSON>** Managing_Director:N; IMF:N **<NN>** Managing_Director:N; …*

Example 46

Firstly, the passive voice is parsed by the dependency parser, so we do not need to worry about that. Secondly, in **T** of Example 45, "*Leloir*" is the indirect object of "*give*" and "*Sociedad Científica Argentina*" is the direct object, which suggests that he got the prize. Therefore, the ditransitive relation can be divided into three binary relations, the subject with the direct object, the indirect object with the direct object, and the subject with the indirect object. In any of the three cases, we can make it accordant with **H**. Consequently, we will have the generalization rules when we encounter "*<OBJ1>*" and "*<OBJ2>*" relations: "*<OBJ1>* → *<SUBJ>*" and "*<OBJ2>* → *<OBJ>*".

In Example 46, both "*Michel Camdessus*" and "*Managing Director*" refer to the same

person, and "*IMF*" and "*Managing Director*" could also be concatenated. Consequently, we will group these nodes connected via dependency relations like "<*PERSON*>", "<*NN*>", etc. Nodes have the "<*CONJ*>" relation in-between usually share some common characteristics, and thus, could also be put together.

Not only have we changed some tags, but we have also deleted some dependency tags which at least for our approach are not so relevant to the RTE task. See Appendix for all the generalization rules.

For the second aspect of the generalization, we will simply exclude the lemmas of all the words, preserving solely the POS tags. Consequently, the generalized TS of the **H** of Example 41 is as follows in the form of "*Left Foot Node # Left Spine # Root Node # Right Spine # Right Foot Node*", in which # is a separator,

Nicolas_Cage:N # <GEN> N <SUBJ> # call:V # <OBJ> # Kal-el:N

Example 47

We could also perform this step on the TS of the **T** of Example 41 as below,

Nicolas_Cage_actor:N # <GEN> N <SUBJ> V <I> # name:V # <OBJ> # Kal-el:N

Example 48

Notice that all the symbols contained in spines come from a set of a limited size. This symbol set consists of some dependency relations and POS tags. We call this set *Closed-Class Symbol* (CCS) set. Some of the CCSs are listed below (the whole set is in Appendix),

Types	Examples
Dependency Relations Tags	*<GEN>, <SUBJ>, <OBJ>, …*
POS Tags	*N, V, Prep, …*

Table 6 Examples of CCSs

In the next step, we will merge the two TSs via obtaining the dissimilarity of the spines: 1) excluding the longest common prefixes for left spines; and 2) excluding the longest common suffixes for right spines. The remaining infixes are named as the *Left Spine Difference* (LSD) and the *Right Spine Difference* (RSD). Then the LSD and RSD for Example 41 (i.e. the merging of Example 47 and Example 48) in the form of "***T**-part ## **H**-part*", where ## is a separator, will be,

LSD: *(null)*[21] *## V <I>*
RSD: *(null) ## (null)*

Example 49

According to the CCS set before, both LSD and RSD have two parts, consisting of sequences of CCSs or empty sequences. These two will be the main part of our feature space and the rest part is the features about RNs and their adjacent dependency relations. We will continue this in the next section.

4.2.7 Pattern Generation

Apart from the LSD and RSD, the RNs also play important roles in the entailment detection. Verbs like "*buy*" and "*sell*" can reverse the meaning of the sentences, and thus, may change the entailment relation, when one appears in **T** and the other in **H**. In addition, the adjacent dependency relations of RNs are also of great importance, if the verbs convey directional actions, as we mentioned in 3.4.6.

Applying these observations, we will have two extra features, one is *Verb Consistency*[22] (VC) and the other is *Verb Relation Consistency* (VRC). The former checks whether the two RNs are consistent; and the latter checks whether the adjacent relations are consistent. Currently, we have kept all the verbs consistent and for adjacent dependency relations focused solely on "<*SUBJ*>" and "<*OBJ*>" relations. Other dependency relations will be consistent with both "<*SUBJ*>" and "<*OBJ*>". In Example 41, namely the comparison between Example 47 and Example 48, on the left, "<*I*>" and "<*SUBJ*>" are consistent, and on the right, "<*OBJ*>" and "<*OBJ*>" are the same. However, if we have the following example,

Dataset=RTE3-dev Id=660 Task=SUM Entailment=NO Length=short
Text: Most of the mines are in arid areas and animals searching for water are ***attracted*** *to the cyanide-laced holding ponds that are an integral part of the mining operations.*
Hypothesis: Animals have ***died*** *by the thousands from drinking at cyanide-laced holding ponds.*
Adjacent Dependency Relations in ***T****: ... to:Prep <MOD> # attract:V # <OBJ> animal:N ...*
Adjacent Dependency Relations in ***H****: ... by:Prep <MOD> # die:V # <SUBJ> animal:N ...*

Example 50

In Example 50, on the right side, the adjacent dependency relations are "<*OBJ*>" and "<*SUBJ*>", which are inconsistent. Therefore, in this example, the VRC is not consistent.

[21] "(null)" represents an empty sequence.
[22] Note that the RN is not necessary to be a verb and the name is just for convenience.

When the adjacent relations are neither "<*SUBJ*>" nor "<*OBJ*>", the VRC will be neither consistent nor inconsistent, resulting in a third value.

With all these acquired features, our *Entailment Pattern*[23] (EP) will be composed of four elements, which is a quadruple of "<*LSD, RSD, VC, VRC*>". In particular, LSD and RSD are either empty or CCS sequences; VC is a Boolean value, where *true* means that the two RNs are consistent and *false* otherwise; VRC has a ternary value, where *1* means that both relations are consistent, *-1* means at least one pair of corresponding relations is inconsistent and *0* means otherwise. Therefore, the EP of Example 41 is,

LSD: *(null) ## V <I>*
RSD: *(null) ## (null)*
VC: *1*
VRC: *1*

Example 51

4.2.8 Kernel-based Machine Learning

We will perform machine learning on top of the EPs. For the LSD and RSD, we will utilize a *Subsequence Kernel* (Bunescu and Mooney, 2006) and a *Collocation Kernel* (Wang and Neumann, 2007a) to represent the features implicitly; for VC and VRC, we will just use two trivial kernels to check the consistency; and finally, we will combine them linearly by a composite kernel giving the four basic kernels different weights.

The basic idea of the subsequence kernel is encoding all the possible subsequences of a sequence into a vector. For instance, if we are given a sequence as follows,

<GEN> N <SUBJ>

Example 52

The feature vector will be checking whether the following possible subsequences appear in a sequence,

<"<GEN>", "N", "<SUBJ>", "<GEN> N", "N <SUBJ>", "<GEN> N <SUBJ>">

Example 53

Thus, for the sequence of Example 52, the value of its feature vector will be "*<1, 1, 1, 1, 1,*

[23] Note that the entailment pattern in this chapter is principally the same as the one in the previous chapter, but more formally defined.

1>". If the given sequence is "<*GEN*> *N*", the feature value will be "<*1, 1, 0, 1, 0, 0*>". In a generalized form, if we are given a set of CCS sequences, the size of the feature vector will be the number of all the possible subsequences, where each element represents whether a particular subsequence exists in a sequence. Then, the definition of the kernel function is as follows,

$$K_{subsequence}(< T, H >, < T', H' >) = \sum_{i=1}^{|T|}\sum_{i\prime=1}^{|T'|} K_{CCS}(CCS_i, CCS_{i\prime}) + \sum_{j=1}^{|H|}\sum_{j\prime=1}^{|H'|} K_{CCS}(CCS_j, CCS_{j\prime})$$

Equation 4 Definition of the Subsequence Kernel

where *T* and *H* refer to the sets of all the possible subsequences of LSD and RSD from **T**[24] and **H**, and *T'* and *H'* are from another pair; |*T*|, |*T'*|, |*H*|, and |*H'*| represent the cardinalities of these sets; and the function K_{CCS}*(CCS, CCS')* checks whether its two arguments are equal.

Notice that Equation 4 does not capture the relatedness between **T** and **H**, instead, it simply concatenate the two parts. Therefore, the following collocation kernel has been considered,

$$K_{collocation}(< T, H >, < T', H' >) = \sum_{i=1}^{|T|}\sum_{i\prime=1}^{|T'|}\sum_{j=1}^{|H|}\sum_{j\prime=1}^{|H'|} K_{CCS}(CCS_i, CCS_{i\prime}) \cdot K_{CCS}(CCS_j, CCS_{j\prime})$$

Equation 5 Definition of the Collocation Kernel

In the collocation kernel, all the elements in the feature vector will be various possible combinations of the subsequences of **T** and **H**.

Consequently, together with the other two kernels, K_{VC} and K_{VRC}, the final composite kernel is,

$$K_{composite} = \alpha K_{subsequence} + \beta K_{collocation} + \gamma K_{VC} + \delta K_{VRC}$$

Equation 6 Definition of the Composite Kernel

where γ and δ are learned from the training corpus, and $\alpha=\beta=1$.

In practice, to further reduce the computational complexity, we have set the maximum length of the subsequence in the subsequence kernel as *5* and in the collocation kernel as *3*.

[24] In this section, **T** and **H** refer to the spines instead of the original texts or dependency structures.

In the testing phase, the previous steps (from 4.2.2 to 4.2.7) will be the same as the training phase. The only difference is that the input will be unannotated data, that is to say, there is no value for the *entailment* attribute. The trained model will be applied to each entailment pattern in order to predict whether it is a positive one or a negative one, and furthermore predict whether the entailment relationship exists in the corresponding **T-H** pairs.

4.2.9 Graphic User Interface

The previous sections in this subchapter have been talking about the process of the approaches, while this section will present a snapshot of the GUI of our system,

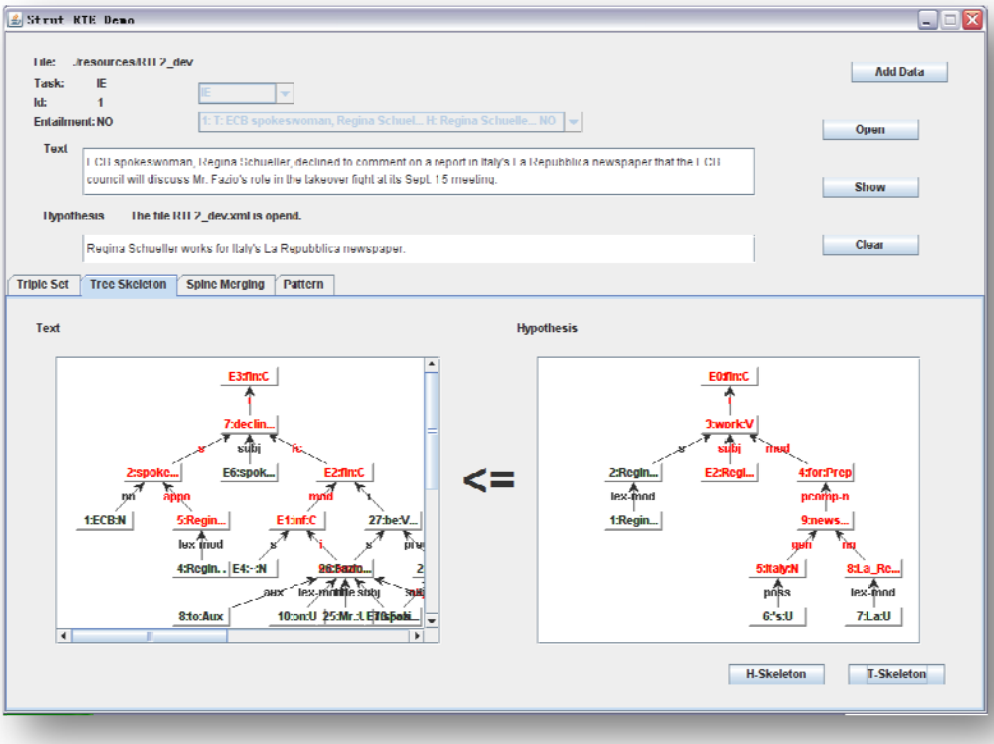

Figure 10 A Snapshot of the GUI

After loading the data, the GUI can show all the main steps of our system procedure. There are four main representations (i.e. four tabs in the lower part of the GUI) after choosing one **T-H** pair: 1) triple sets (4.2.3); 2) dependency trees (4.2.4); 3) tree skeletons (4.2.5, 4.2.6); and 4) entailment patterns (4.2.7). There are more snapshots in the Appendix.

In addition, the button in the top right corner will invoke another GUI for data annotation as follows,

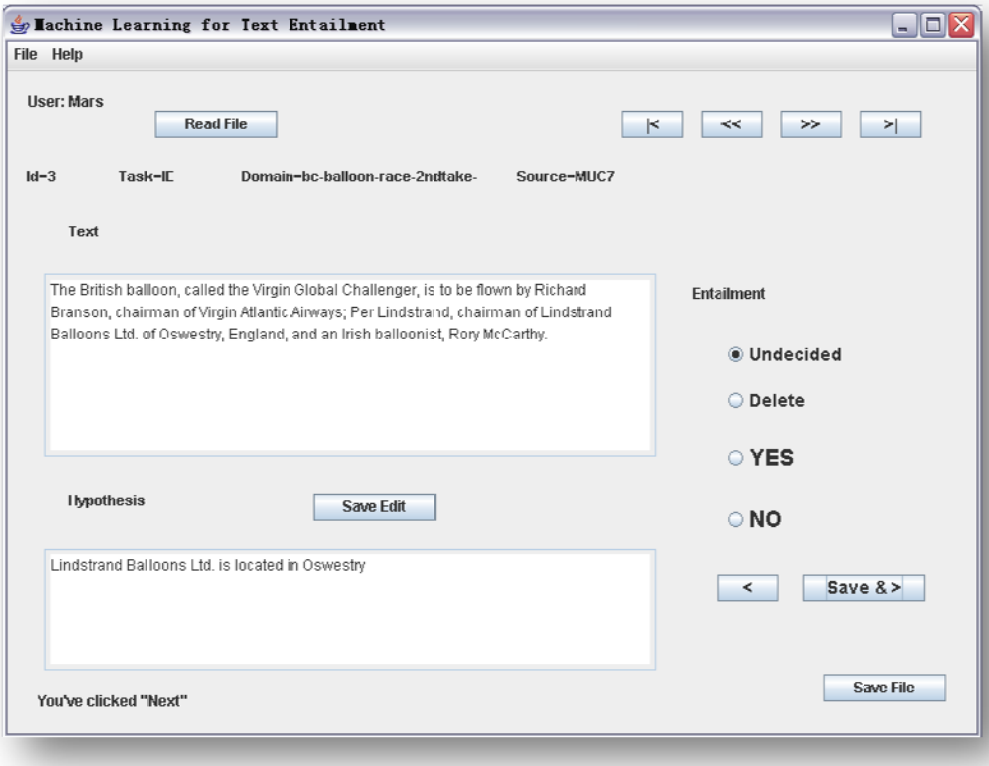

Figure 11 A Snapshot of the GUI for Data Annotation

For each unannotated **T-H** pair, the annotator can choose "*YES*", "*NO*", or "*Undecided*"; and if the pair is not an appropriate one, it can also be marked as "*Delete*". The annotation file will be automatically saved with separate file names for individual annotators.

4.3 Summary

In this chapter, we have presented the implementation of the approaches described in the previous chapter. At the beginning, we introduced the external tools used in our system, such as dependency parsers, machine learning tools, etc. Then, we elaborated on our system step by step, from the backup strategies to the main approach. We used some concrete examples to show the problems in practice and how to solve them as well. In the end, some snapshots of the GUI of our system and the annotation tool were shown.

Chapter V

System Evaluation

In this chapter, we will present our evaluation of the RTE system, which is the core part of ***TERA***, and leave concrete NLP applications for the next chapter. Firstly, we will describe the data from the RTE challenges and how we have collected our extra data. Secondly, the results will be shown, after the specification of the experiments and parameter settings. Thirdly, error analysis will be performed, followed by discussions on interesting examples. Fourthly, we will compare our system with related works done by other research groups. In the end, we will summarize this chapter.

5.1 Data Preparation

The data used for our evaluation can be grouped into two parts: 1) data from the RTE challenges[25]; and 2) extra data automatically collected and partially annotated.

The RTE-2 data set (Bar-Haim et al., 2006) is composed of a training set and a test set. Both of them contain 800 **T-H** pairs, each pair consisting of a Text – a text paragraph which has one or two sentences and a Hypothesis – one sentence. Each pair also has several attributes: *id number*, *task*, and *entailment*. The attribute *task* can be "*Information Extraction* (IE)", "*Information Retrieval* (IR)", "*Question Answering* (QA)", or "*Summarization* (SUM)" and each task occupies 25% of the whole data set. The attribute *entailment* can be either "*YES*" (i.e. positive cases) or "*NO*" (i.e. negative cases) in the training set (each occupying 50%) and no such attribute in the test set. The annotated version of the test set is also released after the challenge and has the same positive and negative distribution. Therefore, in all, there are 1600 annotated **T-H** pairs in the RTE-2 data set.

The RTE-3 data set (Giampiccolo et al., 2007) has a similar specification. The main difference is that RTE-3 data have one more attribute for each **T-H** pair, which is called *length*. This attribute indicates the length of the texts, which can be either "*long*" or "*short*". A pair will be marked as "*long*" if this pair exceeds 270 bytes. Another difference is the distribution of attribute values, which is slightly unbalanced compared with the RTE-2 data set. The distributions of all the **T-H** pairs in both the RTE-2 and the RTE-3 data sets are shown as below,

[25] We did not use RTE-1 Data Set, because it is a bit different from the data sets of the other two challenges.

(Dev/Test)		IE	IR	QA	SUM	Long	Short	ALL
RTE-2 Data Set	YES	100/100	100/100	100/100	100/100			400/400
	NO	100/100	100/100	100/100	100/100			400/400
	ALL	200/200	200/200	200/200	200/200			800/800
RTE-3 Data Set	YES	109/105	89/87	107/106	107/112	78/58	334/352	412/410
	NO	91/95	111/113	93/94	93/88	57/59	331/331	388/390
	ALL	200/200	200/200	200/200	200/200	135/117	665/683	800/800

Figure 12 Distribution of the RTE-2 and the RTE-3 Data Set

As well as the data provided by the RTE challenges, we have also constructed some extra data according to the description of data collection from Bar-Haim et al. (2006) and Hickl et al. (2006). In detail,

For IE Pairs

We have two data sources: 1) documents with *Named-Entity* (NE) annotation from MUC-6[26]; and 2) parsed texts from BinRel (Roth and Yih, 2004).

From the first data source, we have taken out all the NEs and grouped close ones into pairs. For example, we have the following document and NE lists,

*The union has hired a number of professional consultants in its battle with the company, including **Ray Rogers** of **Corporate Campaign Inc.**, the **New York** labor consultant who developed the strategy at **Geo. A. Hormel & Co.**'s **Austin**, **Minn.**, meatpacking plant last year. That campaign, which included a strike, faltered when the company hired new workers and the **International Meatpacking Union** wrested control of the local union from **Rogers'** supporters.*
Person Names (PNs): "Ray Rogers", "Rogers"
Location Names (LNs): "New York", "Austin", "Minn."
Organization Names (ONs): "Corporate Campaign Inc.", "Geo. A. Hormel & Co.", "International Meatpacking Union"

Example 54

The possible NE pairs are,

[26] http://cs.nyu.edu/cs/faculty/grishman/muc6.html

<PN, LN>: *<"Ray Rogers", "New York">, <"Rogers", "Austin">, ...*
<PN, ON>: *<"Ray Rogers", "Corporate Campaign Inc.">,<"Rogers", "Geo. A. Hormel & Co.">, <"Ray Rogers", "International Meatpacking Union">, ...*
<ON, LN>: *<"Corporate Campaign Inc.", "New York">,<"Geo. A. Hormel & Co.", "Austin">,<"International Meatpacking Union", "Minn.">, ...*

Example 54 (continued)

Using these NE pairs, we have added binary relations in-between and constructed **H**s. For the same example, the possible **H**s are,

H1: *"Ray Rogers* ***is born in*** *New York."*
H2: *"Ray Rogers* ***is working for*** *International Meatpacking Union."*
H3: *"Corporate Campaign Inc.* ***is located in*** *Austin."*
...

Example 54 (continued)

With these **H**s, **T** is naturally the originally document. Therefore, we will obtain several **T**-**H** pairs on top of the document shown in Example 54. After that, human annotation is needed to judge whether the NE relations truly exist. If they do, the **T**-**H** pair will be marked as a positive case; otherwise, as a negative case.

The second data source contains three parsed corpora with NEs and NE relations listed after each sentence: 1) the *kill* relation corpus; 2) the *birthplace* relation corpus; and 3) the negative corpus (i.e. there are NEs but no these two kinds of relations in-between). The following **T**-**H** pairs will be easily constructed using this information,

*Relation: kill Entailment=**YES***
Text: Today's Highlight in History: On the Ides of March, 44 B.C., Roman Emperor Julius Caesar was assassinated by a group of nobles that included Brutus and Cassius.
Hypothesis: Brutus ***killed***[27] *Roman Emperor Julius Caesar.*

Example 55

*Relation: birthplace Entailment=**YES***
Text: Dole is at an organizational disadvantage in the South but has had his wife, Elizabeth, a native of North Carolina, working the region for him.
Hypothesis: Elizabeth ***is born in*** *North Carolina.*

Example 56

[27] In practice, when we generated the Hs, we have made variations in surface realization.

Relation: birthplace *Entailment=**NO***
Text: Mrs. Thatcher promised Haughey that Tom King, her minister for Northern Ireland, would make a statement soon about the "organizational implications" of the affair, the British spokesman said.
*Hypothesis: Tom King **is born in** Northern Ireland.*

Example 57

Therefore, as a whole, we have constructed 784 IE pairs (248 pairs from MUC-6 and 536 pairs from BinRel), with equal numbers of positive and negative cases.

For QA Pairs

We have built QA pairs using the data from TREC2003[28]. The questions and corresponding answers have been used for constructing **H**s and the supporting documents for **T**s. For instance, we have the following questions, their answers, and the supporting documents,

Question: How did George Washington die?
Answer: throat infection.
Document: Washington died from a throat infection at age 67, almost three years after leaving the presidency.

Example 58

Question: What country made the Statue of Liberty?
Answer: France.
Document: In 1885, Statue of Liberty arrives in New York City from France.

Example 59

We can combine the questions and the answers of both Example 58 and Example 59 into statements, "*George Washington died of throat infection*" and "*France made the Statue of Liberty*". These two sentences will be the **H**s and the corresponding documents will be the **T**s. Regarding these two examples, we can easily decide that Example 58 is a positive case and Example 59 is a negative case.

In brief, a positive case satisfies both that the answer is the correct answer to the given question and that supporting document contains this answer; while a negative case could be lack of at least one conditions. In all, we have built 214 QA pairs.

[28] http://trec.nist.gov/pubs/trec12/t12_proceedings.html

Others

We have only collected extra **T**-**H** pairs on two tasks, IE and QA. The reasons why we did so are: 1) pairs for other two tasks are difficult to collect (see details afterwards); 2) our system has achieved better results on IE and QA pairs (5.3.1 and 5.3.2); 3) IE and QA are the most straightforward applications of the RTE system (see Chapter VI); and 4) IE pairs were supposed to be the most difficult ones (Bar-Haim et al., 2006).

IR pairs are a bit similar to QA pairs, since both of them use the relevant documents as **T**s. Compared with combining the question and the answer, more work needs to be done manually adapting a query into a natural language sentence, namely an **H**. For SUM pairs, the positive cases are easily obtained (Burger and Ferro, 2005), but the "good" negative cases are difficult to decide.

The annotation tool mentioned in 4.2.9 has been utilized for annotating the data.

5.2 Experiment Settings

In experimentation, we have compared three approaches, two backup systems (BoW and Triple Set Matcher – TSM) as baseline systems, and our main approach, the subsequence kernel method plus backup strategies (SK+BS).

Several experiments have been set up: 1) 10-fold cross-validation on the **whole** RTE-2 data set using *Task* as a feature (Exp1AT); 2) training on the RTE-2 Dev Set and testing on the Test Set (Exp1BT); 3) 10-fold cross-validation on the RTE-3 **Dev** Set using *Task* or *Length* as a feature (Exp2AT or Exp2AL); 4) training on the RTE-3 Dev Set and testing on the Test Set using *Task* or *Length* as a feature (Exp2BT or Exp2BL); 5) 10-fold cross-validation on the cases covered by our main approach from the extra IE and QA pairs (Exp3AT).

The following table shows the combinations of systems and data for different experiments,

Systems	Data	RTE-2 Data	RTE-3 Data		Extra
		Task	*Task*	*Length*	*IE+QA*
BoW	10-cv	Exp1AT	Exp2AT	Exp2AL	Exp3AT
	Train-Test	Exp1BT	Exp2BT	Exp2BL	
TSM	10-cv	Exp1AT	Exp2AT	Exp2AL	Exp3AT
	Train-Test	Exp1BT	Exp2BT	Exp2BL	
SK+BS	10-cv	Exp1AT	Exp2AT	Exp2AL	Exp3AT
	Train-Test	Exp1BT	Exp2BT	Exp2BL	

Table 7 Different Experiment Settings

The evaluation metrics is ***Accuracy***, that is, the percentage of correctly predicted **T-H** pairs.

5.3 Experiment Results

In this section, the results of our experiments will be presented, by the order of experiments on RTE-2 data (5.3.1), RTE-3 data (5.3.2), and the extra data (5.3.3). At last, we will talk about the relationship between different sizes of the training data with the system performances. The coming section will focus on the error analysis and discussions about the results.

As mentioned in the previous chapter, the kernel-based machine learning is performed via the classifier SMO from the WEKA toolkit (Witten and Frank, 1999). In addition, on the RTE-3 data set, we have tested two dependency parsers, Minipar (Mi) and the Stanford Parser (SP), while on the RTE-2 data set, only Minipar.

5.3.1 On RTE-2 Data Set

We have performed two groups of experiments on the RTE-2 data set, one is the 10-fold cross validation on the whole data set (Exp1AT) and the other is training on the Dev Set and testing on the Test Set (Exp1BT). The results are shown as below,

Exp1AT: 10-fold cross validation on the whole RTE-2 Data Set					
Systems	**IE**	**IR**	**QA**	**SUM**	**ALL**
BoW	50.0%[29]	58.8%	58.8%	**74.0%**	60.4%
TSM	50.8%	57.0%	62.0%	70.8%	60.2%
SK+BS[30]	**61.2%**	58.8%	63.8%	74.0%	**64.5%**

Table 8 Results of Exp1AT

Exp1BT: Training on the RTE-2 Dev Set and Testing on the Test Set					
Systems	**IE**	**IR**	**QA**	**SUM**	**ALL**
BoW	**50.0%**	56.0%	60%	**66.5%**	58.1%
TSM	**50.0%**	53.0%	64.5%	65.0%	58.1%
SK+BS[31]	**62.0%**	61.5%	64.5%	66.5%	**63.6%**

Table 9 Results of Exp1BT

From both Table 8 and Table 9, the main approach SK+BS have achieved the highest improvement over the baseline systems for IE pairs. This suggests that the kernel method seems to be more appropriate if the underlying task conveys a more "relational nature." The improvements for the other tasks are not so convincing as compared to the more "shallow" methods realized via BoW and TSM. Such as for SUM pairs, the BoW method has already obtained fairly good results.

Nevertheless, the overall result obtained in Table 9 would have been among the top-4 of the RTE-2 challenge. Note that we do not exploit any additional knowledge source besides the dependency trees computed by Minipar. In order to take a closer look at the performance of our main approach, we will show more detailed results in the following table,

Experiments	**IE**	**IR**	**QA**	**SUM**	**ALL**
Exp2AT: Coverage	**63.0%**	18.3%	**36.3%**	16.3%	33.5%
Exp2AT: Accuracy	64.0%	67.1%	66.2%	73.9%	**66.2%**
Exp2BT: Coverage	**64.0%**	23.5%	**44.0%**	17.0%	37.0%
Exp2BT: Accuracy	66.9%	70.2%	58.0%	64.7%	**64.5%**

Table 10 Cases Covered by the Main Approach

For the IE and QA pairs, the method SK+BS obtained the highest coverage. However, for IR and SUM pairs, although it achieves good accuracy, the number of covered cases is low, and hence the backup systems will deal with most of the cases for IR and SUM. As a whole,

[29] The accuracy is actually 47.6%. Since the random guess will achieve 50%, we will take this as the least value for comparison.
[30] We have chosen BoW as backup strategy for IR and SUM pairs and TSM for IE and QA pairs according to the baselines' results.
[31] Same as 6.

the main approach can cover about 1/3 of the whole data set and achieve better accuracy than baseline systems. Further discussions on the results will be in the next subchapter (5.4).

5.3.2 On RTE-3 Data Set

Similar to the experiments on the RTE-2 data set, we have also performed 1) 10-fold cross validation on the RTE-3 Dev Set[32] (Exp2AT and Exp2AL), and 2) training on the Dev Set and testing on the Test Set (Exp2BT and Exp2BL). The difference here is that the RTE-3 data set contains one more attribute, *Length* (Exp2AL and Exp2BL). Therefore, it has been considered another dimension to group the data, as well as the attribute *Task* (Exp2AT and Exp2BT). Additionally, another difference will be that two dependency parsers (Minipar – Mi and the Stanford Parser – SP) are used.

Consequently, the following two tables will first show the results of Exp2AT and Exp2AL,

Exp2AT: 10-fold cross validation on the RTE-3 Dev Set					
Systems	**IE**	**IR**	**QA**	**SUM**	**ALL**
BoW	54.5%	70.0%	76.5%	**68.5%**	67.4%
TSM	53.5%	60.0%	68.0%	62.5%	61.0%
SK+BS (Mi)	**63.0%**	74.0%	79.0%	68.5%	**71.1%**
SK+BS (SP)	**60.5%**	70.0%	81.5%	68.5%	**70.1%**

Table 11 Results of Exp2AT

Exp2AL: 10-fold cross validation on the RTE-3 Dev Set			
Systems	**Long**	**Short**	**ALL**
BoW	69.6%	67.1%	67.5%
TSM	66.7%	60.5%	61.5%
SK+BS (Mi)	73.3%	71.0%	**71.4%**

Table 12 Results of Exp2AL

In Table 11, the performance on IE pairs has been improved greatly compared with the baseline systems, and others have less improvement. One interesting point here is that the BoW method has achieved quite good result, i.e. 67.4%; the other interesting point is that the two dependency parsers varied in their performances on pairs from different tasks. According to the results, using Minipar will get better results on IE and QA pairs, while the Stanford Parser seems to be more suitable for QA pairs, and on SUM pairs, they have just achieved the same result. As a whole, using Minipar has 1% better accuracy than using the Stanford Parser.

[32] When I did the experiment, the annotated Test Set was still not available yet.

By breaking down the data using *Length* (Table 12), we have achieved better results than using *Task*. Notice that all the three approaches have performed better on the long **T-H** pairs than the short ones, which is interesting because the longer ones were supposed to be more difficult when discourse analysis might be needed.

We have submitted two runs for the RTE-3 Challenge, which are shown in the following table,

Exp2BT&Exp2BL: Training on the RTE-3 Dev Set and Testing on the Test Set					
Systems	**IE**	**IR**	**QA**	**SUM**	**ALL**
BoW	54.5%	66.5%	76.5%	56.0%	63.4%
TSM	54.5%	62.5%	66.0%	54.5%	59.4%
SK+BS (Mi+SP+*Task*) – run1	**59.5%**	70.5%	75.5%	60.5%	65.5%
SK+BS (Mi+*Length*) – run2	**58.5%**	70.5%	79.5%	59.0%	**66.9%**

Table 13 Results of Exp2BT and Exp2BL

The results in Table 13 are quite consistent with the previous results from the 10-fold cross validation: the main approaches (i.e. the two runs of submission) have achieved good results on IE pairs, while for other pairs, the improvement is not so great, compared with the baseline systems, especially with the BoW method.

Finally, we have been among the top-5 results of the RTE-3 Challenge (Giampiccolo et al., 2007) with the second run of submission. We have also done the detailed statistics on the performance of our main approach. The following table is about the first run of submission, and the second one is more or less the same,

Experiments	**IE**	**IR**	**QA**	**SUM**	**ALL**
Exp2AT: Coverage	**53.0%**	19.0%	23.5%	31.5%	31.8%
Exp2AT: Accuracy	67.9%	78.9%	**91.5%**	71.4%	**74.8%**
Exp2BT: Coverage	**58.5%**	16.0%	27.5%	42.0%	36.0%
Exp2BT: Accuracy	57.2%	81.5%	**90.9%**	65.5%	**68.8%**

Table 14 Cases Covered by the Main Approach

For IE pairs, we have the best coverage, then for SUM pairs and QA pairs, and for IR pairs we have the lowest coverage. The accuracy is promising. Notice that for QA pairs, the accuracy of the covered cases is more than 90%, which helps the average accuracy to be around 70%. It seems that for IE pairs, the accuracy still needs to be improved, while for other pairs, the coverage is the main bottleneck (see more in 5.4).

5.3.3 On Extra Data Set

In order to get a deeper view of our method, we evaluated our systems using additional data. The results of the experiments achieved so far suggest that our method works well for the IE and QA tasks. Therefore, we have collected additional data from relevant sources (MUC-6, BinRel, and TREC2003) so as to test how our method performs for larger training sets.

In all, we have 998 **T-H** pairs (half positive cases and half negative cases), and our main approach can cover about 62.4% of them (515 IE pairs and 108 QA pairs). The following table displays the results on the covered cases,

Exp3AT: 10-fold cross validation on the covered cases of the Extra Data			
Systems	**IE** (*MUC-6*, *BinRel*)	**QA** (*TREC2003*)	**Overall**
SK - Coverage	515/784 (65.7%)	108/214 (50.5%)	623/998 (62.4%)
BoW	62.9%	61.4%	62.3%
TSM	64.9%	62.3%	63.8%
SK - Accuracy	76.3%	65.7%	74.5%

Table 15 Results of Exp3AT

SK shows a fairly acceptable coverage on data from these two tasks (65.7% and 50.5%). It has improved nearly 12% accuracy for IE pairs compared with the baseline systems, while for QA pairs about 3.4%. Before we discuss the gains and losses in detail, we will first take a look at the impact of the training corpus size.

5.3.4 Impact of the Training Corpus Size

It was reported by Hickl et al. (2006) that enlarging the training corpus can improve the results by about 10%, while Hickl and Bensley (2007) showed that there was no obvious improvement even 100,000 pairs were used for training.

We have also applied the extra data to the training phase, but the performance did not increase much, sometimes even decrease (Wang and Neumann, 2007b). One possible reason could be the fairly low coverage of our main approach. That is to say, most of the cases are delegated to the backup strategies, where the training effect will not be so visible because of the limited feature space. Another reason could be the differences between heterogeneous and homogenous data. Since the entailment phenomena can be divided into several cases belonging to different linguistic levels, it is difficult to collect a larger data set which has the

similar distributions as the RTE data set. Even the RTE-2 and the RTE-3 data sets differ from each other. More discussions will be continued in the next subchapter.

5.4 Error Analysis and Discussions

As a whole, developing task specific entailment operators is a promising direction. As we mentioned at the beginning of the thesis, the RTE task is neither a one-level nor a one-case task. The experimental results uncovered differences among pairs of different tasks with respect to coverage and accuracy. The following figure is based on the results of Table 10,

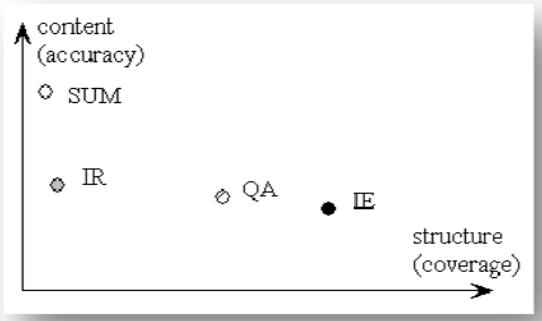

Figure 13 Pairs Distribution (according to Table 10)

Figure 13 visualizes the differences between pairs from different tasks. In brief, IE pairs have the best coverage, indicating that the tree skeletons can be extracted among most of the pairs; SUM pairs have the best accuracy, but lowest coverage, denoting that lexical semantics may play a more important role. We will discuss these issues one by one, the coverage of the main approach (5.4.1), the accuracy of the main approach (5.4.2), and the backup strategies (5.4.3).

5.4.1 Coverage of the Main Approach

In retrospect, there are two things relevant to this aspect: 1) how to find the right topic word pairs; and 2) the resulting number of the spines.

Though we have applied some partial matching techniques at the substring level to make correspondence between pairs like "*Philippine*" and "*Philippines*", the following two examples are still difficult to catch,

Dataset=RTE2-dev	*Id=186*	*Task=IE*	*Entailment=NO*
*Text: An **Afghan** interpreter, employed by the United States, was also wounded.*			
*Hypothesis: An interpreter worked for **Afghanistan**.*			

Example 60

Dataset=RTE3-dev	*Id=90*	*Task=IE*	*Entailment=NO*	*Length=short*
*Text: As an active member of the National Guard, he was called to duty in 1941. Although Kennon did not see active combat, he did not return home from **World War II** until May of 1945.*				
*Hypothesis: Kennon did not participate in **WWII**.*				

Example 61

In Example 60, though "*Afghan*" is contained in "*Afghanistan*", the ratio between the lengths of the two words is below 60%. If we were to lower the threshold, words such as "*Austria*" and "*Australia*" would be identified as a topic word pair. Example 61 is a tougher example. If we were to apply the normal rule of acronym, the acronym of "*World War II*" would be "*WWI*", but not "*WWII*".

The topic words which are restricted to solely nouns will also cause a problem, such as in the following example,

Dataset=RTE2-dev	*Id=13*	*Task=IE*	*Entailment=YES*
*Text: Sunday's earthquake was felt in the **southern Indian** city of Madras on the mainland, as well as other parts of south India. The Naval meteorological office in Port Blair said it was the second biggest aftershock after the Dec. 26 earthquake.*			
*Hypothesis: The city of Madras is located in **Southern India**.*			

Example 62

Though "*southern Indian*" and "*Southern India*" are very similar from the appearance, they have different POS tags and the first one is an adjective, which is not covered by our current definition of topic words.

Moreover, hypo- and hyper-nyms are also not considered. Since we do not have any external lexical knowledge base, this kind of matching is difficult to realize, as below,

Dataset=RTE3-test	*Id=201*	*Task=IR*	*Entailment=YES*	*Length=short*
*Text: Berlin has a new landmark. Among the cranes which still dominate the skyline of Europe's newest capital now stands a **chancellery**, where the head of government Gerhard Schroeder will live and the German cabinet will hold its regular meetings.*				
*Hypothesis: New **buildings** have been erected in Berlin.*				

Example 63

If the system knows "*chancellery*" is a kind of "*building(s)*", it can capture this topic word pair; otherwise, only one topic word pair is found, "*Berlin*" and "*Berlin*".

Furthermore, sometimes, discovering the same Named-Entities (NEs) mentioned in **T** and **H** is not trivial, especially the abbreviations of person names and various kinds of temporal expressions, such as in the following examples,

Dataset=RTE2-dev	*Id=232*	*Task=IE*	*Entailment=NO*
*Text: Thanks to the recent acquisition of **J.D. Edwards**, Oracle will soon be able to run JDE apps on its Fusion Middleware platform, too.*			
*Hypothesis: **J.D.E.** is the owner of Oracle.*			

Example 64

Dataset=RTE2-dev	*Id=439*	*Task=IE*	*Entailment=NO*
*Text: San Salvador, **Jan. 13, '90** (Acan-Efe) -The bodies of Hector Oqueli and Gilda Flores, who had been kidnapped yesterday, were found in Cuilapa, Guatemala, near the border with El Salvador, the relatives of one of the victims have reported.*			
*Hypothesis: Gilda Flores was kidnapped on the **13th of January 1990**.*			

Example 65

In Example 64, "*J.D.E*" is the abbreviation of "*J.D. Edwards*". There are similar cases like making correspondence between "*J.F.K.*", "*J.F. Kennedy*", and "*John F. Kennedy*", and so on. Example 65 has raised a problem of another type of NEs – temporal expressions. "*Jan. 13, '90*" in **T** refers to the exactly same date as "*13th of January 1990*" in **H** does, though they differ a lot literally.

From the previous examples, we can see the necessity of improving the matching strategy, in order to obtain more common topics shared by **T** and **H**. On the contrary, we have to tackle the cases with too many spines[33] as follows,

[33] Notice that more topic word pairs do not necessarily mean more spines, since more than one topic word can be contained in one spine.

Dataset=RTE3-test	*Id=750*	*Task=SUM*	*Entailment=YES*	*Length=short*
*Text: The British **government** has indicated its readiness to allow Argentine **companies** to take part in the development of **oilfields** in the Falkland islands' territorial waters.*				
*Hypothesis: The British **government** is ready to allow Argentine **companies** to participate in the development of **oilfields**.*				

Example 66

The words in bold in Example 66 are all foot nodes, indicating that the number of spines is three. In fact, this complex event involves three participants, the British "*government*", the Argentine "*companies*", and the development of "*oldfields*". Notice that our current version of the tree skeleton can only allow two spines, so that the whole structure can be represented in a sequence. We could also consider extending the tree skeleton to three spines or more. However, the dependency path from the common parent node of "*government*" and "*oilfield*" (i.e. "*participate*") to the common parent node of the three topic words (i.e. "*allow*") will be calculated twice. This may increase the computational complexity if there are many spines. In spite of this, the extension of the tree skeleton has a great potential for future work, aiming to enlarge the coverage of the main approach.

5.4.2 Accuracy of the Main Approach

Compared with the coverage, the accuracy of the main approach has achieved fairly good results (5.3), especially for IE (Table 15) and QA (Table 14) pairs. Looking into the errors, we have found two aspects are of great importance: 1) the structure of the tree skeleton; and 2) linguistic patterns.

Regarding the first aspect, there are two parts uncovered by the current version of the tree skeleton: one is the *modifiers* of the topic words and the other is the *verbs higher* than the root node on the dependency tree. For instance, our approach cannot correctly predict the following example,

Dataset=RTE2-dev	*Id=701*	*Task=IE*	*Entailment=NO*
*Text: FMLN guerrilla units ambushed the 1st company of **military** detachment no. 2 Jr. Battalion at la Pena Canton, Villa Victoria Jurisdiction.*			
*Hypothesis: FMLN guerrilla units attacked a **commercial** company.*			

Example 67

In **T** of this example, the "*company*" is a "*military*" one, but in **H**, the "*company*" is a "*commercial*" one. In a dependency tree, the modifier is below the noun it modifies.

Therefore, in our algorithm (4.2.5), the tree skeleton starts from the foot nodes (i.e. nouns) to the root node, excluding all the nodes lower than the foot nodes. One possible solution could be adding matching between words with different POS tags, as mentioned before (Example 62); another solution could be including the modifiers into the tree skeleton structure, making the spines longer than before.

As well as the prolonging the spine, the verbs higher than the root node are another missing part of the tree skeleton.

Dataset=RTE2-dev *Id=133* *Task=SUM* *Entailment=NO*
Text: Verizon Communications Inc. ***said*** *on Monday* ***it would*** *buy long-distance telephone company MCI Communications Inc. in a deal worth $6.75 billion, giving Verizon a foothold in the market for serving large corporations.*
Hypothesis: Verizon Communications Inc.'s $6.7 billion takeover of long-distance provider MCI Inc. transformed the telephone industry.

Example 68

Dataset=RTE3-dev *Id=759* *Task=SUM* *Entailment=YES* *Length=short*
Text: CVS will ***stop*** *selling its own brand of 500-milligram acetaminophen caplets and pull bottles from store shelves nation wide, spokesman Mike DeAngelis said.*
Hypothesis: CVS will ***not*** *sell its own brand of 500-milligram acetaminophen caplets any longer.*

Example 69

Example 68 is a very difficult example. Not only the verb in **T** "*buy*" should be corresponded with the noun in **H** "*takeover*", but also "*said*" and "*would*" are the trick for obtaining the correct answer. Since our tree skeleton will stop at the lowest common parent node (i.e. the root node), all the verbs on the higher part of the dependency tree will be ignored. Nairn et al. (2006) have done more about this: Verbs like "*forget*", "*refuse*", "*attempt*", and so on, are classified and analyzed, because they may change the polarity of the embedded statements.

Example 69 is an interesting **T**-**H** pair. In **T**, a higher verb "*stop*" negates the whole statement; and in **H**, the negation word "*not*" is directly used, which has the same effect. Consequently, this pair has resulted in a positive case, correctly guessed by our system. Another truly solved example of the negation will be presented later. Before that, the last point which needs to be mentioned of the tree skeleton structure is about the root node.

Inside the tree skeleton, the root node has not been carefully dealt with. Without the help of

any lexical knowledge base of verbs, such as the VerbOcean (Chklovski and Pantel, 2004), the relations between two verbs are not easily captured. However, most of the cases such as the following example can be handled,

Dataset=RTE3-test	*Id=246*	*Task=IR*	*Entailment=YES*	*Length=short*
*Text: Overall the accident rate worldwide for commercial aviation has been **falling** fairly dramatically especially during the period between 1950 and 1970, largely due to the introduction of new technology during this period.*				
*Hypothesis: Airplane accidents are **decreasing**.*				

Example 70

Apart from this kind of *similar* relation between "*falling*" and "*decreasing*" in Example 70, there are also other relations, such as the antonymous relation between "*sell*" and "*buy*". We will consider either using a verb resource or learning the relations from corpora.

The second aspect of the accuracy is about linguistic phenomena. From a broad view, our approach has used subsequence kernels to implicitly represent the features extracted solely from the output of the dependency parser. After analyzing all the gains, we have found some patterns related to some particular linguistic phenomena.

The following example is about the negation again,

Dataset=RTE2-dev	*Id=77*	*Task=QA*	*Entailment=NO*
*Text: It is totally idiotic to call Christo and Jeanne-Claude the "wrapping artists." So many works were **not** wrapping, for instance the Iron Curtain by Christo, 1962.*			
Hypothesis: The Iron Curtain was wrapped by Christo.			

Example 71

In **T** of Example 71, there is a negation word "*not*" before "*wrapping*", but in **H**, there are no such words. As well as the negation, some other patterns can also be found, especially for IE and QA pairs, in which our method has achieved better results. Let us recall the example mentioned in Chapter IV as follows,

Dataset=RTE2-dev	*Id=534*	*Task=IE*	*Entailment=NO*
*Text: The main library at 101 E. Franklin St. changes its solo and group exhibitions monthly in the Gellman Room, the Second Floor Gallery, the Dooley Foyer **and** the Dooley Hall.*			
*Hypothesis: Dooley Foyer **is located in** Dooley Hall.*			

Example 42 (again)

In **T**, "*Dooley Foyer*" and "*Dooley Hall*" are coordination, conveyed by the conjunction

"*and*"; in **H**, the relation between these two places is "*located in*". Thus, an informal pattern could be like "[LN[34]1] *and* [LN2]" does not entail "[LN1] *is located in* [LN2]". A positive case is shown as below,

Dataset=RTE3-test	*Id=40*	*Task=IE*	*Entailment=YES*	*Length=short*
*Text: Robinson's garden style can be seen today at Gravetye Manor**,** West Sussex, England, though it is more manicured than it was in Robinson's time.*				
*Hypothesis: Gravetye Manor **is located in** West Sussex.*				

Example 72

If the two place names are connected via a comma (i.e. ","), the first place belongs the second one. A candidate pattern will be like "[LN1]*,* [LN2]" entails "[LN1] *is located in* [LN2]". In fact, comma delivers various meanings in different context. The following comma represents another relationship between a person and an organization,

Dataset=RTE3-dev	*Id=37*	*Task=IE*	*Entailment=YES*	*Length=short*
*Text: Colarusso**,** the Dover police captain, said authorities are interested in whether their suspect made a cell phone call while he was in the Dover woman's home.*				
*Hypothesis: Colarusso **works for** Dover police.*				

Example 73

In Example 73, the "*works for*" relation between the person "*Colarusso*" and the organization "*Dover police*" is also conveyed via the comma in **T**. Consequently, "[PN]*,* [ON]" entails "[PN] *works for* [ON]". Furthermore, the "*works for*" relation has more relevant patterns,

Dataset=RTE2-dev	*Id=186*	*Task=IE*	*Entailment=NO*
Text: An Afghan interpreter, employed by the United States, was also wounded.			
*Hypothesis: An interpreter **worked for** Afghanistan.*			
Pattern: "[Country Name] [Profession]" entails "[Profession] worked for [Country Name]"			

Example 74

[34] LN stands for Location Name. We assume that the NEs have been recognized. And in the rest of this chapter, PN stands for Person Name, and ON stands for Organization Name.

Dataset=RTE2-dev *Id=712* *Task=IE* *Entailment=YES*
*Text: "I think we've already seen the effect on oil and gas prices," said economist Kathleen Camilli **of** New York-based Camilli Economics.*
*Hypothesis: Kathleen Camilli **works for** Camilli Economics.*
Pattern: "[PN] of [ON]" entails "[PN] works for [ON]"

Example 75

Though all of these examples can be solved by our main approach, going into details about these *Closed-Class Words* involved patterns seems to be a great potential for the future research. More work could be done such as 1) obtaining frequent subsequences to form patterns, 2) defining the patterns more formally, and 3) grouping patterns according to different dimensions (e.g. *Task*). Since the current RTE results have not been impressive after applying lexical knowledge base like WordNet (as we mentioned in 2.4), the closed-class words are worth considering.

5.4.3 Backup Strategies

Apart from the main approach, improving the accuracy of the backup strategies without losing their robustness will increase the results as well. For the BoW method, it is useful to add NE overlapping checking to our current simple method. See the following,

Dataset=RTE2-dev *Id=313* *Task=QA* *Entailment=NO*
*Text: Mohandas Karamchand Gandhi never received the Nobel Peace Prize, though he was nominated for it five times between **1937** and **1948**.*
*Hypothesis: Mohandas received the Nobel Prize in **1989**.*

Example 76

According to the definition of the overlapping word ratio in 4.2.2, the BoW similarity will be quite high (about 0.714). However, the entailment relation does not hold, because the year is not correct. If we add one more feature about the overlapping ratio of NEs, the result of the BoW method may be improved. Similarly, this feature could also be added to the triple set matcher, so as to solve the following problem,

Dataset=RTE3-test *Id=533* *Task=QA* *Entailment=NO* *Length=short*
*Text: In the last two years in the Darfur region more than **70,000** people have been **killed**, and over **2 million** displaced.*
*Hypothesis: More than **200,000** people have been **killed** in the air attack in the Darfur region.*

Example 77

Assuming that "*2 million*" and "*200,000*" could be successfully matched via the NE recognizer, the BoW method will get a fairly high score, while triple set matcher will help us the know which figure is the correct one for "*killed*" using the local dependency relations.

As well as the NEs, some linguistic trigger words can be very helpful, such as the negation in Example 71. In all, the linguistic phenomena analyzed above may also improve the backup strategies to some extent.

5.5 Comparison with Others

In the previous parts of this chapter, we have shown the performances of our system; and in this subchapter, some comparison will be done with others regarding the techniques applied.

We have utilized the system description table of Bar-Haim et al. (2006) to compare our system with the best two systems of the RTE-2 Challenge in the following table,

Systems	Lx[35]	Ng	Sy	Se	LI	Co	ML	BK	ED
Hickl et al. 2006	X	X	X	X		X	X		**X**
Tatu et al. 2006	X				**X**			X	
Ours	X	X					X		

Table 16 Applied Techniques and Resources

The best system (Hickl et al., 2006) applies both shallow and deep techniques, especially in acquiring extra entailment corpora. The second best system (Tatu et al., 2006) contains many manually designed logical inference rules and background knowledge. On the contrary, we exploit no additional knowledge sources besides the dependency trees computed by the parsers, nor any extra training corpora. The comparable results we have obtained have already shown the advantages of our system, which has set up a good starting point for solving the RTE problem. In the meanwhile, the RTE system has served as a core engine of the whole framework of ***TERA***, whose applications will be presented in the next chapter.

5.6 Summary

As a summary, we have presented the evaluation on our RTE system, which is the core part of ***TERA***. We first described the data sets from RTE-2 and RTE-3 Challenges, followed by the

[35] Following the notation in (Bar-Haim et al., 2006): Lx: Lexical Relation DB; Ng: N-Gram / Subsequence overlap; Sy: Syntactic Matching / Alignment; Se: Semantic Role Labeling; LI: Logical Inference; Co: Corpus/Web; ML: ML Classification; BK: Paraphrase Technology / Background Knowledge; ED: Acquisition of Entailment Corpora.

collection of extra data of IE and QA pairs. After the introduction to the experiment settings, the results of several experiments were shown. Different experiments varied in data sets, evaluation methods, data distributions, etc. Finally, a detailed error analysis and rich discussions were given with abundant examples, pointing out both the problems and possible solutions.

Chapter VI
Applications

This chapter will focus on the applications of our RTE system, so as to show the rest of ***TERA***. We will start with the related work, which is using the RTE technique for improving other NLP applications. Then, two applications will be discussed in detail: *Binary Relation Extraction* and *Answer Validation*. Inside each application, we will describe how we cast the original task into an RTE problem, show the experiment results, and present the discussions on both gains and losses. The summary of this chapter will be given at the end.

6.1 Related Work

One of the original goals of the RTE task was to discover a generic approach to tackle different NLP applications, e.g. IE, IR, QA, and SUM, and also where the data have come from. Therefore, a lot of research has been done to make use of the RTE techniques.

Romano et al. (2006) have proposed a paraphrase-based approach for relation extraction. Since paraphrase is a bidirectional entailment relation, they have used the TEASE algorithm (Szpektor et al., 2004) to acquire entailment relations from the Web for a given input template in an unsupervised way. Traditionally, this is performed in a supervised manner, requiring many examples for each relation with semantic variations. Therefore, they have shown a high potential for unsupervised paraphrase acquisition.

De Salvo Braz et al. (2005) and Harabagiu and Hickl (2006) have utilized RTE for question answering systems. A typical QA system contains three steps: question analysis, document retrieval, and answer extraction. Harabagiu and Hickl's entailment system (2006) has been applied at the second and third steps, for filtering and re-ranking. They have tested entailment between the question and candidate retrieved document, that is, included entailment score in document ranking; they have also tested entailment between the question and the answer to filter and re-rank the candidate answers; furthermore, the entailment between the question and *Automatically Generated Questions* (AGQ) created from candidate documents has also been tested to check whether the answers can match the entailed AGOs. Finally, the accuracy has been increased by 20% after applying RTE.

Many researchers have used RTE for answer validation in the *Answer Validation Exercise*

(AVE)[36] (Peñas et al., 2006). This task is a subtask of the QA Track[37] in *Cross Language Evaluation Forum* (CLEF)[38], which asks systems to validate the answers of QA systems participating at CLEF. The RTE techniques have been used to detect whether the entailment holds between the supporting document and the hypothesis generated from the question and the answer. Most of the groups use lexical or syntactic overlapping as features for machine learning; other groups derive the logic forms of natural language texts and perform proving.

We have achieved fairly high results on both the RTE-2 data set and the RTE-3 data set (5.3), especially on IE and QA pairs. Therefore, one of our motivations is to improve these applications by using RTE, and the other is to test our RTE system in other concrete NLP tasks. We will first briefly show the application on binary relation extraction (6.2), and then describe how we took part in the AVE@CLEF2007 task (1.2.3).

6.2 Binary Relation Extraction

Since the annotated binary relation corpus BinRel (Roth and Yih, 2004) has already been introduced in 5.1, the application here will be straightforward.

We have used the original texts as **T**s, and combined NEs contained using either *kill* relation or *birthplace* relation. In detail, a positive *kill* **T-H** pair will be an existing *kill* relation between two NEs, which are both *Person Names* (PNs); a negative one will be two PNs with no *kill* relation in-between. Similarly, a positive *birthplace* example will be a true relation between a PN and a *Location Name* (LN), where the person was born in that place; a negative one will be no such relations between a PN and a LN. Notice that both of these two relations are directional.

In practice, 918 *kill* pairs (268 positive cases) and 849 *birthplace* pairs (199 positive cases) have been constructed from the corpus. The results are shown in the following table,

Systems	*kill* Relation	*birthplace* Relation
BoW	72.0%	75.0%
TSM	70.3%	76.4%
SK+TSM	**84.1%**	**86.5%**

Table 17 Results of Binary Relation Extraction

The results are quite high, because the task itself is simplified. In real relation extraction

[36] http://nlp.uned.es/QA/ave/
[37] http://clef-qa.itc.it/
[38] http://www.clef-campaign.org/

systems, or IE systems, this operator (i.e. detect whether a particular binary relation exists) can be utilized iteratively. In addition, the error analysis has been included in 5.4, since a subset of these **T**-**H** pairs have been used to enlarge the training corpus. Therefore, we will directly go into the next application – *Answer Validation*.

6.3 Answer Validation

We have taken part in the AVE@CLEF2007 task. We have first changed the question and the answer into Hypothesis (**H**) and view the document as Text (**T**), in order to cast the AVE task into a RTE problem. Then, we have used our RTE system to tell us whether the entailment relation holds between the documents (i.e. **T**s) and question-answer pairs (i.e. **H**s). Finally, we have adapted the results for the AVE task.

The architecture of our AVE system is shown in the following figure, which consists of the preprocessing part, the RTE component, and the post-processing part,

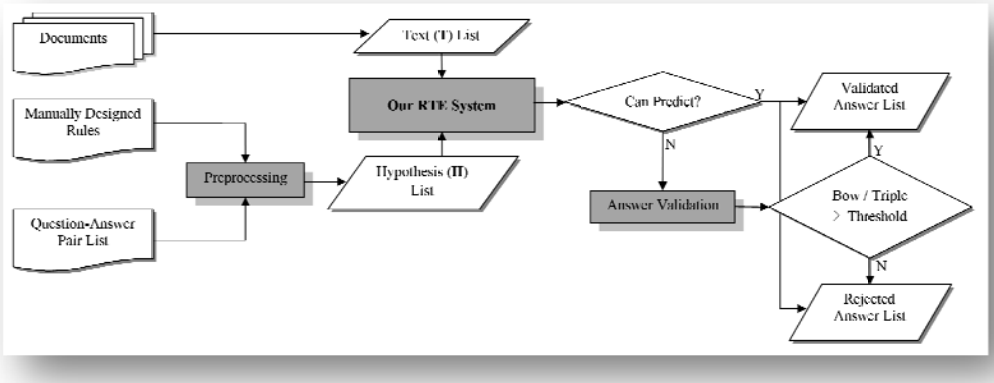

Figure 14 Architecture of Using RTE for Answer Validation

Our AVE system uses our RTE system as a core component, and includes preprocessing and post-processing modules. The preprocessing module mainly adapts questions, their corresponding answers, and supporting documents into **T**-**H** pairs, assisted by some manually designed patterns. The post-processing module (i.e. the "*Answer Validation*" in Figure 1) will validate each answer and select the most proper one based on the output of the RTE system.

We will see the details of both preprocessing and post-processing modules in the coming section (6.3.1) and the results of our two runs of submission in the next section (6.3.2) followed by a discussion on error sources (6.3.3). We will summarize our work in the next subchapter (6.4).

6.3.1 Task Casting

The given input of the AVE task is a list of questions, their corresponding answers and the documents containing these answers. Usually, we need to validate several answers for each question. For instance, the question is,

Question (id=178): In which country was Edouard Balladur born?

Example 78

The QA system gives out several candidate answers to this question, as follows,

Answer1 (id=178_1): Frances
Answer2 (id=178_3): 12% jobless rate
Answer3 (id=178_5): 7
…

Example 78 (continued)

Each answer will have one supporting document where the answer comes from, like this,

Document1(id=178_1):Paris, Wednesday CONSERVATIVE Prime Minister Edouard Balladur, defeated in France's presidential election, resigned today clearing the way for President-elect Jacques Chirac to form his own new government. Balladur's move was a formality since outgoing President Francois Mitterrand hands over power next week to Chirac, the conservative Paris mayor who won last Sunday's run-off election... (parts)
…

Example 78 (continued)

The assumption here is that if the answer is relevant to the question, the document which contains the answer should entail the statement derived by combining the question and the answer. We will first focus on the combination of the question and the answer in order to fit the input of the RTE system and then talk about how to make use of the output of the system.

To combine the question and the answer into a statement, we need some language patterns. Normally, we have different types of questions, such as *Who*-questions asking about persons, *What*-questions asking about definitions, etc. Therefore, we manually construct some language patterns for the input questions. For the example given above (id=178), we will apply the following pattern,

Pattern (id=178): Edouard Balladur was born in <Answer>.

Example 78 (continued)

Consequently, we substitute the "*<Answer>*" by each candidate answer to form **H**s – hypotheses. Since the supporting documents are naturally the **T**s – texts, the **T**-**H** pairs are built up accordingly,

Id: 178_1
Entailment: Unknown
Text: Paris, Wednesday CONSERVATIVE Prime Minister Edouard Balladur, defeated in France's presidential election, resigned today clearing the way for President-elect Jacques Chirac to form his own new government… (parts)
Hypothesis: Edouard Balladur was born in Frances.

Example 78 (continued)

These **T**-**H** pairs can be the input for any the generic RTE system.

After using our RTE system, several things can be obtained: 1) for some of the **T**-**H** pairs, we directly know whether the entailment holds; 2) every **T**-**H** pair has a triple similarity score; 3) every **T**-**H** pair has a BoW similarity score. If the **T**-**H** pairs are covered by our main approach, we will directly use the answers; if not, we will use a threshold to decide the answer based on the two similarity scores. In practice, the threshold is learned from the training corpus and the two similarity scores are used in different runs of submission.

For the adaption back to the AVE task, the "*YES*" entailment cases will be validated answers and the "*NO*" entailment cases will be rejected answers. In addition, the selected answers (i.e. the best answers) will naturally be the pairs covered by our main approach or (if not,) with the highest similarity scores.

6.3.2 Experiment Results

The AVE 2007 task asks the system to judge whether an answer extracted from a document is a valid answer to the given question. The result can be either "*VALIDATED*" or "*REJECTED*", which mean it's a valid answer or not respectively. Furthermore, among all the "*VALIDATED*" answers to each question, one best answer will be marked as "*SELECTED*", but if there is no "*VALIDATED*" answers, there will be no "*SELECTED*" answer, either.

The AVE training data contain 200 questions, 1121 answers and 1121 supporting documents, among which there are 130 validated answers and 991 rejected answers. The AVE

testing data contain 67 questions, 202 answers and supporting documents, among which there are 21 validated answers, 174 rejected answers, and 7 unknown answers according to the gold standard. Notice that both the two data sets are unbalanced and the evaluation metrics are the Recall and Precision of "*VALIDATAED*" cases and the accuracy of "*SELECTED*" cases.

We have submitted two runs. For both of the two runs we have used the main approach and one backup strategy. The difference is that in the first run, the BoW similarity score is the backup, while in the second run, the triple similarity score is taken. Our machine learning process is performed by using the classifier SMO from the WEKA toolkit (Witten and Frank, 1999). In the following, we will first show the table of the results and then present an error analysis in the next section (6.3.3),

Runs of Submission	Recall	Precision	F-Measure	QA Accuracy
dfki07-run1 (SK+BoW)	0.62	0.37	**0.46**	0.16
dfki07-run2 (SK+TSM)	0.71	0.44	**0.55**	0.21

Table 18 Results of Our Two Runs of Submission

Though the absolute scores are not very promising, they are still better than all the results for English from last year. The second run outperforms the first run in all respects, which shows advantages of the triple similarity score. The gold standard does not contain the "*SELECTED*" answers, thus, we will not discuss the QA accuracy for now. Instead, the error analysis will focus on the loss of recall and precision and the room for improvement in the future.

6.3.3 Discussions

Among all the errors, half of them belong to one type. For questions like "*What is the occupation of Kiri Te Kanawa?*", we have used the pattern "*The occupation of Kiri Te Kanawa is* <Answer>", which might cause problems, because "*occupation*" usually does not appear in the documents. Instead, a pattern like "*Kiri Te Kanawa is* <Answer>" might be much better. Some other errors are from the noise of web documents, on which the dependency parser could not work very well. For instance, some documents are menu items on a website, which have no syntax, thus, cannot be parsed. A truly difficult example is the following one,

Question (id=160): Which American President masterminded the Camp David Agreement?
Answer (id=160_2): Jimmy Carter.
Document (id=160_2): United States President Jimmy Carter invited both Sadat and Begin to a summit at Camp David to negotiate a final peace.

Example 79

Not only the lexical semantics of "*mastermind*" and "*negotiate*" are necessary, but also some world knowledge like the name of an agreement is usually the name of the place where people subscribe it.

The precision of our two runs are rather poor. After taking a closer look at the errors, we have found that most of the errors also belong to one type. In those answer-document pairs (e.g. id=119_2, id=125_1, id=133_1, etc.), the answers are usually very long, which consist of a large part of the documents. In some extreme cases (e.g. id=112_2, id=172_2, etc.), the answers are very long and exactly the same as the documents. Due to the characteristics of our method (i.e. using RTE for AVE), these answers will get high similarity scores, which will be wrongly validated. Our method will also make mistakes in the following example, because there are three guitarists, but we have wrongly taken the first two as one person,

Question (id=106): Who is Thom Rotella?
Answer (id=106_3): Grant Geissman.
Document (id=106_3): As founder of Positive Music Records, Navarro is responsible for launching and furthering the recording careers of saxophonists Bob Militello and Brandon Fields, ***guitarists Grant Geissman, Thom Rotella and Pat Kelley****, and keyboardists Gregg Karukas and Marcus Johnson.*

Example 80

Some other errors like trivial answers (e.g. "*one*") could be avoided by adding some rules, such as the following example,

Question: (id=182): What is the most popular snack food in the UK?
Answer (id=182_4): one
Document (id=182_4): Galaxy (chocolate) Wrapper from 150gram Galaxy bar (UK) Galaxy is a brand of milk choclate made and marketed by by the ***Mars company*** *(****one*** *of several related products punning upon the name Mars or "Master Foods" by using an astronomical name).* ***It*** *is most popular in the United Kingdom and the Middle East, where it is the leading brand in many countries.*

Example 81

On the whole, more fine-grained classification of answers could be helpful to improve the

system. Compared with the QA pairs we have collected from TREC2003 (5.1), the data of the AVE task 1) are unbalanced and 2) have ungrammatical or noisy documents (i.e. **T**s). These two features make the task tougher. Both the coverage of our main approach and the accuracy are not as good as the experiment results we have achieved before (5.3).

Based on the analysis above, the possible future directions are: 1) preprocessing the documents to clean the noisy web data; 2) improving the patterns or learning them automatically; 3) utilizing question analysis tools to acquire more useful information.

6.4 Summary

In conclusion, we have described two applications of our RTE system. After a brief introduction to using RTE for relation extraction and question answering, we have applied our system on concrete task, i.e. binary relation extraction and answer validation. For the first one, we tested on the binary relation corpus; and for the second one, we took part in AVE@CLEF2007. One the one hand, it is an effective way to improve the answer validation task; on the other hand, it is also a promising application for our developed RTE system. The results have shown the advantages of our method.

Chapter VII

Conclusion and Future Work

This chapter will summarize this thesis. We have proposed a RTE-based framework – ***TERA***, which provided us with both a standalone RTE system and the ability to use it for other NLP applications. Regarding the RTE system, we have constructed a new feature representation extracted from the dependency structure and applied the subsequence kernel method for machine learning. Various experiments have been performed for evaluation and the errors have been discussed as well. We have also shown two concrete applications based on the RTE system – binary relation extraction and answer validation. Plenty of possible future work has emerged after analyzing both the gains and losses, which will be pointed out in the second part of this chapter.

7.1 Conclusion

RTE aims to, on the one hand, explore phenomena concerning knowledge representation and understanding; and on the other hand, to discover a generic approach to several NLP applications. In this thesis, we have analyzed different cases of entailment using read data and applied different strategies for them respectively. The experiment results have shown that this is a reasonable solution for tackling this problem. As well as the evaluation on its own, it has also been tested on concrete applications. The main conclusions are as follows,

Different strategies for different cases is a reasonable solution for the RTE task, which can be done neither at one linguistic processing level nor using one simple technique. According to the *task*, IE pairs have clearer predicate argument structures; SUM pairs usually include more participants in the events. According to the techniques applied, some cases only need to check the words overlapping; while some other cases need to deal with semantics and inferences. To some extent, our combination of the main approach and backup strategies has pre-classified all the entailment cases and provided separate solutions.

Quite promising results are achieved on the real data. Though our approach has only utilized the output of the dependency parser(s) with no external knowledge bases or larger training corpus, we have been among the top-4 results on the RTE-2 test set (63.6% of accuracy tested on our own) and top-5 results on the RTE-3 test set (66.9% of accuracy in the RTE-3 Challenge). This approach has in fact set up a high-standard starting point for further

improvement.

Linguistic patterns and entailment rules can be generated based on the results. After analyzing both our gains and losses, we have found interesting patterns relevant to particular linguistic phenomena and entailment rules, though they are not formal enough currently. However, they have already shown the potential for future research.

A generic RTE-based application framework is possible to build up. In 4.2, we have shown the architecture of ***TERA***, which is centered by an RTE system as a core engine. Several applications can be handled if both the input and output formats have been adapted to the RTE system, which are basically the **T**-**H** pairs.

Two NLP applications have successfully utilized our RTE system: binary relation extraction and answer validation. The first one is comparably simple. After casting the task into an RTE problem, the system can predict whether a binary relation exists between two given NEs, which can be used as a basic operator in the future relation extraction or IE systems. Answer validation can be viewed as an RTE problem as well, after combining the question and the answer into a statement. The results of our two submissions to AVE2007@CLEF (0.46 and 0.55 of f-measure) have outperformed all the results from last year in the English language.

7.2 Future Work

There are a lot of unexplored issues remaining, which can be summarized in the following five points: 1) applying lexical semantic resources; 2) extending the tree skeleton structure; 3) obtaining rewriting rules for fragments of the dependency structure; 4) testing on languages other than English; and 5) exploring more applications for RTE.

Lexical semantics could be added in. There are two ways to achieve this, using external knowledge bases, such as WordNet, FrameNet, VerbOcean, etc., and learning automatically from the corpora. Lexical resources of nouns could be used for matching the topic words between **T** and **H**, such as hyponym, hypernym, synonym, antonym, etc. Resources of verbs could be considered in the verb consistency checking, in other words, the comparison between frames. According to our experiment results and error analysis in 5.4, the question of how to use functional words for assisting RTE has proved to be a promising research direction.

The extension of the tree skeleton is necessary to enlarge the coverage of our main

approach. Three possible ways have been proposed: adding modifiers into the structure, adding the verbs higher than the root node in the dependency structure, and increasing the number of spines. The modifiers are necessary because they restrict the entities, such as "*a commercial company*" and "*a military company*"; the verbs higher in the dependency tree can change the polarity of the embedded statements, which may also reverse the result. A tree skeleton with more than two spines can represent events involving more than two participants. In addition, the computational complexity needs to be considered after the extension as well.

More formal definition of entailment rules is also a promising research direction. According to our error analysis and discussions, some particular linguistic phenomena have been captured by some patterns, and entailment rules can be generated from some fragments of the dependency structure. A unified formal framework of such patterns or rules could help us to both further understand the entailment relation itself and to use it for other NLP applications.

Making the RTE system multilingual is another of our plans. Since our approach has only used the output of the dependency parsers and afterwards been language independent, we can easily adapt it into languages other than English on top of dependency parsers for those languages.

Many other NLP applications can also apply the RTE techniques in. Paraphrase acquisition has already been proved to be one option by other researchers; unsupervised relation extraction from the web has also used RTE to acquire semantic variations; text summaries can be evaluated using RTE; text-based inferences could be achieved using entailment relation as a unit operator; and so on. To sum up, RTE as a basic operation or module can be applied in various fields.

Bibliography

(Adams, 2006) Rod Adams. 2006. Textual Entailment Through Extended Lexical Overlap. In *Proceedings of the Second PASCAL Challenges Workshop on Recognising Textual Entailment*, Venice, Italy.

(Adams et al., 2007) Rod Adams, Gabriel Nicolae, Cristina Nicolae and Sanda Harabagiu. 2007. Textual Entailment Through Extended Lexical Overlap and Lexico-Semantic Matching. In *Proceedings of the Workshop on Textual Entailment and Paraphrasing*, pages 119–124, Prague, June 2007.

(Augsten et al., 2005) Nikolaus Augsten, Michael Böhlen, and Johann Gamper. 2005. Approximate matching of hierarchical data using pq-grams. In *Proceedings of the 31st international conference on Very large data bases*, Pages: 301 - 312. Trondheim, Norway.

(Baker et al., 1998) Collin F. Baker, Charles J. Fillmore, and John B. Lowe. 1998. The Berkeley FrameNet project. In *Proceedings of COLING-ACL*, Montreal, Canada.

(Bar-Haim et al., 2006) Roy Bar-Haim, Ido Dagan, Bill Dolan, Lisa Ferro, Danilo Giampiccolo, Bernardo Magnini and Idan Szpektor. 2006. The Second PASCAL Recognising Textual Entailment Challenge. In *Proceedings of the Second PASCAL Challenges Workshop on Recognising Textual Entailment*, Venice, Italy.

(Bar-Haim et al., 2007) Roy Bar-Haim, Ido Dagan, Iddo Greental, Idan Szpektor and Moshe Friedman. 2007. Semantic Inference at the Lexical-Syntactic Level for Textual Entailment Recognition. In *Proceedings of the Workshop on Textual Entailment and Paraphrasing*, pages 131–136, Prague, June 2007.

(Berger et al., 1996) Adam L. Berger , Vincent J. Della Pietra , Stephen A. Della Pietra, A maximum entropy approach to natural language processing, *Computational Linguistics*, v.22 n.1, p.39-71, March 1996.

(Bobrow et al., 2007) Daniel Bobrow, Dick Crouch, Tracy Halloway King, Cleo Condoravdi, Lauri Karttunen, Rowan Nairn, Valeria de Paiva and Annie Zaenen. 2007. Precision-focused Textual Inference. In *Proceedings of the Workshop on Textual Entailment and Paraphrasing*, pages 16–21, Prague, June 2007.

(Bos, 2005) Johan Bos. 2005. Towards wide-coverage semantic interpretation. In *Proceedings of Sixth International Workshop on Computational Semantics IWCS-6*, pages 42-53.

(Bos and Markert, 2006) Johan Bos and Katja Markert. 2006. When logical inference helps determining textual entailment (and when it doesn't). In *Proceedings of the Second PASCAL Challenges Workshop on Recognising Textual Entailment*, Venice, Italy.

(Budanitsky and Hirst, 2001) Alexander Budanitsky and Graeme Hirst. 2001. Semantic distance in WordNet: An experimental, application-oriented evaluation of five measures. In *Proceedings of Workshop on WordNet and Other Lexical Resources, Second meeting of the North American Chapter of the Association for Computational Linguistics*.

(Bunescu and Mooney, 2006) Bunescu, R. and Mooney, R. 2006. Subsequence Kernels for Relation Extraction. In *Proc. of the 19th Conference on Neural Information Processing Systems*.

(Burchardt et al., 2005) Aljoscha Burchardt, Katrin Erk, and Anette Frank. 2005. A WordNet Detour to FrameNet. In B. Fisseni, H.-C. Schmitz, B. Schröder, and P.Wagner, editors, *Sprachtechnologie, mobile Kommunikation und linguistische Resourcen*, volume 8 of Computer Studies in Language and Speech, pages 408 - 421. Peter Lang, Frankfurt am Main.

(Burchardt and Frank, 2006) Aljoscha Burchardt and Anette Frank. 2006. Approaching Textual Entailment with LFG and FrameNet Frames. In *Proceedings of the Second PASCAL Challenges Workshop on Recognising Textual Entailment*, Venice, Italy.

(Burchardt et al., 2007) Aljoscha Burchardt, Nils Reiter, Stefan Thater and Anette Frank. 2007. A Semantic Approach To Textual Entailment: System Evaluation and Task Analysis. In *Proceedings of the Workshop on Textual Entailment and Paraphrasing*, pages 10–15, Prague, June 2007.

(Burger and Ferro, 2005) John Burger and Lisa Ferro. 2005. Generating an entailment corpus from news headlines. In *Proceedings of the ACL Workshop on Empirical Modeling of Semantic Equivalence and Entailme*nt, pages 49–54, Ann Arbor, Michigan, June.

(Chambers et al., 2007) Nathanael Chambers, Daniel Cer, Trond Grenager, David Hall, Chloe Kiddon, Bill MacCartney, Marie-Catherine de Marneffe, Daniel Ramage, Eric Yeh and Christopher D. Manning. 2007. Learning Alignments and Leveraging Natural Logic. In

Proceedings of the Workshop on Textual Entailment and Paraphrasing, pages 165–170, Prague, June 2007.

(Charniak, 2000) Eugene Charniak. 2000. A maximum-entropy-inspired parser. In *Proceedings of the first conference on North American chapter of the Association for Computational Linguistics*. Seattle, Washington.

(Chierchia and McConnell-Ginet, 2000) Gennaro Chierchia and Sally McConnell-Ginet. 2000. Meaning and Grammar: An Introduction to Semantics, 2nd Edition. *MIT Press*, March 2000.

(Chklovski and Pantel, 2004) Timothy Chklovski and Patrick Pantel. 2004.VerbOcean: Mining the Web for Fine-Grained Semantic Verb Relations. In *Proceedings of Conference on Empirical Methods in Natural Language Processing (EMNLP-04)*. Barcelona, Spain.

(Chklovski and Pantel, 2005) Chklovski, T. and Pantel, P. 2005. Global Path-based Refinement of Noisy Graphs Applied to Verb Semantics. In *Proceedings of The Second International Joint Conference on Natural Language Processing (IJCNLP-05)*, Jeju Island, South Korea, October 11-13, 2005.

(Chomsky, 1995) Noam Chomsky. 1995. Minimalist Program. *MIT Press*, 1995.

(Claessen and Sörensson, 2003) K. Claessen and N. Sörensson. 2003. New techniques that improve mace-style model finding. In *Model Computation – Principles, Algorithms, Applications (Cade-19 Workshop)*, Miami, Florida, USA.

(Clarke, 2006) Daoud Clarke. 2006. Meaning as Context and Subsequence Analysis for Entailment. In *Proceedings of the Second PASCAL Challenges Workshop on Recognising Textual Entailment*, Venice, Italy.

(Collins, 1996) Michael Collins. 1996. A New Statistical Parser Based on Bigram Lexical Dependencies. In *Proceedings of the 34th Annual Meeting of the ACL*, Santa Cruz.

(Dagan and Glickman, 2004) I. Dagan and O. Glickman. 2004. Probabilistic textual entailment: Generic applied modeling of language variability. In *Learning Methods for Text Understanding and Mining*. Grenoble, France, 2004.

(Dagan et al., 2006) Ido Dagan, Oren Glickman, and Bernardo Magnini. 2006. The PASCAL Recognising Textual Entailment Challenge. In Quiñonero-Candela et al., editors, *MLCW*

2005, LNAI Volume 3944, pages 177-190. Springer-Verlag.

(de Marneffe et al., 2006a) Marie-Catherine de Marneffe, Bill MacCartney, Trond Grenager, Daniel Cer, Anna Rafferty, and Christopher D. Manning. 2006a. Learning to distinguish valid textual entailments. In *Proceedings of the Second PASCAL Challenges Workshop on Recognising Textual Entailment*, Venice, Italy.

(de Marneffe et al., 2006b) Marie-Catherine de Marneffe, Bill MacCartney and Christopher D. Manning. 2006. Generating Typed Dependency Parses from Phrase Structure Parses. In *LREC 2006*.

(de Salvo Braz et al., 2005) Rodrigo de Salvo Braz, Roxana Girju, Vasin Punyakanok, Dan Roth, and Mark Sammons. 2005. Knowledge Representation for Semantic Entailment and Question-Answering. *IJCAI'05: Workshop on Knowledge and Reasoning for Question Answering*.

(Deerwester et al., 1990) Scott Deerwester, Susan Dumais, George Furna, Thomas Landauer and Richard Harshman. 1990. Indexing by Latent Semantic Analysis. *Journal of the American Society for Information Science*.

(Delmonte et al., 2006) Rodolfo Delmonte, Antonella Bristot, Marco Aldo Piccolino, Boniforti, and Sara Tonelli. 2006. Coping with semantic uncertainty with VENSES. In *Proceedings of the Second PASCAL Challenges Workshop on Recognising Textual Entailment*, Venice, Italy.

(Delmonte et al., 2007) Rodolfo Delmonte, Antonella Bristot, Marco Aldo Piccolino Boniforti and Sara Tonelli. 2007. Entailment and Anaphora Resolution in RTE3. In *Proceedings of the Workshop on Textual Entailment and Paraphrasing*, pages 48–53, Prague, June 2007.

(Erk and Pado, 2006) Katrin Erk and Sebastian Pado. 2006. Shalmaneser - a toolchain for shallow semantic parsing. In *Proceedings of LREC-2006*, Genoa, Italy.

(Ferrández et al., 2006) Ó. Ferrández, R. M. Terol, R. Munõz, P. Martínez-Barco, and M. Palomar. 2006. An approach based on Logic Forms and WordNet relationships to Textual Entailment performance. In *Proceedings of the Second PASCAL Challenges Workshop on Recognising Textual Entailment*, Venice, Italy.

(Garoufi, 2007) Konstantina Garoufi. 2007. Towards a Better Understanding of Applied

Textual Entailment. *Master Thesis* in Universität des Saarlandes.

(Giampiccolo et al., 2007) Danilo Giampiccolo, Bernardo Magnini, Ido Dagan, and Bill Dolan. 2007. The Third PASCAL Recognizing Textual Entailment Challenge. In *Proceedings of the Workshop on Textual Entailment and Paraphrasing*, pages 1–9, Prague, June 2007.

(Gildea and Palmer, 2002) Daniel Gildea and Martha Palmer. 2002. The Necessity of Parsing for Predicate Argument Recognition. In *Proceedings of the 40th Meeting of the Association for Computational Linguistics (ACL 2002)*:239-246, Philadelphia, PA.

(Glickman et al., 2005) Glickman, Oren and Dagan, Ido, and Koppel, Moshe. 2005. Web Based Probabilistic Textual Entailment. In *Proceedings of the PASCAL Recognizing Textual Entailment Challenge Workshop*.

(Graff, 2003) David Graff. 2003. English Gigaword. http://www.ldc.upenn.edu/Catalog/CatalogEntry.jsp?catalogId=LDC2003T05.

(Harmeling, 2007) Stefan Harmeling. 2007. An Extensible Probabilistic Transformation-based Approach to the Third Recognizing Textual Entailment Challenge. In *Proceedings of the Workshop on Textual Entailment and Paraphrasing,* pages 137–142, Prague, June 2007.

(Harabagiu and Hickl, 2006) Sanda Harabagiu and Andrew Hickl. 2006. Methods for Using Textual Entailment in Open-Domain Question Answering. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the ACL*, pages 905–912, Sydney, July 2006.

(Harabagiu et al., 1999) S. Harabagiu, G.A. Miller, and D.I. Moldovan. 1999. Wordnet 2 - a morphologically and semantically enhanced resource. In *Proceedings of ACL-SIGLEX99: Standardizing Lexical Resources*, pages 1–8, Maryland, June.

(Haussler, 1999) Haussler, D. 1999. Convolution Kernels on Discrete Structures. *Technical report, University of Santa Cruz.*

(Hays, 1994) David Hays. 1964. Dependency Theory: a formalism and some observations. *Language*, 40:511-525, 1964.

(Herrera et al., 2006) Jesús Herrera, Anselmo Peñas, Álvaro Rodrigo, and Felisa Verdejo. 2006. UNED at PASCAL RTE-2 Challenge. In *Proceedings of the Second PASCAL*

Challenges Workshop on Recognising Textual Entailment, Venice, Italy.

(Herrera et al., 2005) Jesus Herrera, Anselmo Peñas and Felisa Verdejo. Textual Entailment Recognition Based on Dependency Analysis and WordNet In *Proceedings of PASCAL Workshop on Recognizing Textual Entailment*, Southampton, UK, 2005.

(Hickl and Bensley, 2007) Andrew Hickl and Jeremy Bensley. 2007. A Discourse Commitment-Based Framework for Recognizing Textual Entailment. In *Proceedings of the Workshop on Textual Entailment and Paraphrasing*, pages 171–176, Prague, June 2007.

(Hickl et al., 2006) Andrew Hickl, Jeremy Bensley, John Williams, Kirk Roberts, Bryan Rink, and Ying Shi. 2006. Recognizing Textual Entailment with LCC's GROUNDHOG System. In *Proceedings of the Second PASCAL Challenges Workshop on Recognising Textual Entailment*, Venice, Italy.

(Hovy et al., 2005) Eduard Hovy, Chin-Yew Lin, and Liang Zhou. 2005. Evaluating duc 2005 using basic elements. In *Proceedings of the Fifth Document Understanding Conference (DUC)*, Vancouver, Canada.

(Hudson, 1984) Richard Hudson. 1984. Word Grammar. *Basil Blackwell Publishers Limited*., Oxford, England, 1984.

(Iftene and Balahur-Dobrescu, 2007) Adrian Iftene and Alexandra Balahur-Dobrescu. 2007. Hypothesis Transformation and Semantic Variability Rules Used in Recognizing Textual Entailment. In *Proceedings of the Workshop on Textual Entailment and Paraphrasing*, pages 125–130, Prague, June 2007.

(Inkpen et al., 2006) Diana Inkpen, Darren Kipp, and Vivi Nastase. 2006. Machine Learning Experiments for Textual Entailment. In *Proceedings of the Second PASCAL Challenges Workshop on Recognising Textual Entailment*, Venice, Italy.

(Johnson, 1998) Mark Johnson. 1998. PCFG models of linguistic tree representations. *Computational Linguistics*, v.24 n.4, p.613-632, December 1998

(Kamp and Reyle, 1993) Kamp, Hand and Uwe Reyle. 1993. From Discourse to Logic: Introduction to Model Theoretic Semantics of Natural Language, Formal Logic and Discourse Representation Theory. Kluwer Academic Publishers, London, Boston,

Dordrecht. *Studies in Linguistics and Philosophy*, Volume 42.

(Katrenko and Adriaans, 2006) Sophia Katrenko and Pieter Adriaans. 2006. Using Maximal Embedded Syntactic Subtrees for Textual Entailment Recognition. In *Proceedings of the Second PASCAL Challenges Workshop on Recognising Textual Entailment*, Venice, Italy.

(Klein and Manning, 2003) Dan Klein and Christopher D. Manning. 2003. Accurate Unlexicalized Parsing. In *Proceedings of the 41st Meeting of the Association for Computational Linguistics*, pp. 423-430.

(Kouylekov and Magnini, 2006) Milen Kouylekov and Bernardo Magnini. 2006. Tree Edit Distance for Recognizing Textual Entailment: Estimating the Cost of Insertion. In *Proceedings of the Second PASCAL Challenges Workshop on Recognising Textual Entailment*, Venice, Italy.

(Kozareva and Montoyo, 2006) Zornitsa Kozareva and Andrés Montoyo. 2006. MLEnt: The Machine Learning Entailment System of the University of Alicante. In *Proceedings of the Second PASCAL Challenges Workshop on Recognising Textual Entailment*, Venice, Italy.

(Lehmann et al., 2005) John Lehmann, Paul Aarseth, Luke Nezda, Murat Deligonul, and Andrew Hickl. 2005. TASER: A Temporal and Spatial Expression Recognition and Normalization System. In *Proceedings of the 2005 Automatic Content Extraction Conference*. Gaithersburg, MD.

(Leskovec et al., 2005) Jure Leskovec, Natasa Milic-Frayling, and Marko Grobelnik. 2005. Impact of Linguistic Analysis on the Semantic Graph Coverage and Learning of Document Extracts. In Proceedings of AAAI 2005, Pittsburgh, PA.

(Lewis et al., 2004) D. Lewis, Y. Yang, T. Rose, and F. Li. RCV1: A new benchmark collection for text categorization research. *JMLR*, 5:361–397, 2004.

(Lin and Hovy, 2004) Chin-Yew Lin and Ed Hovy. 2004. Automatic Evaluation of Summaries using n-gram co–occurence statistics. In *Proc. Document Understanding Conference (DUC)*, National Institute of Standards and Technology.

(Lin, 1993a) D. Lin, 1993a. Principle-based Parsing without Overgeneration. In *Proceedings of ACL-93*, pp.112--120, Columbus, Ohio.

(Lin, 1994) D. Lin, 1994. PRINCIPAR---An Efficient, broad-coverage, principle-based

parser. In *Proceedings of COLING-94*. pp.42--488, Kyoto, Japan.

(Lin, 1998a) D. Lin, 1998a. Dependency-based Evaluation of MINIPAR. In *Workshop on the Evaluation of Parsing Systems*, Granada, Spain, May, 1998.

(Lin, 1998b) D. Lin, 1998b. An Information-Theoretic Definition of Similarity. In *Proceedings of International Conference on Machine Learning*, Madison, Wisconsin, July, 1998.

(Lin, 1998c) Dekang Lin, 1998c. Automatic Retrieval and Clustering of Similar Words. *COLING-ACL98*, Montreal, Canada, August, 1998.

(Lin, 2001) D. Lin. 2001. LaTaT: Language and Text Analysis Tools. In *Proceedings of Human Language Technology Conference 2001*. pp.222--227. 2001.

(Lin and Pantel, 2001) Dekang Lin and Patrick Pantel. 2001. Dirt - discovery of inference rules from text. In *Proc. of the 7th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD-2001)*, pages 323–328.

(Litkowski, 2006) Ken Litkowski. 2006. Componential Analysis for Recognizing Textual Entailment. In *Proceedings of the Second PASCAL Challenges Workshop on Recognising Textual Entailment*, Venice, Italy.

(Lodhi et al., 2001) Lodhi, H., Christianini, N., Shawe-Taylor, J., and Watkins, C. 2001. Text Classification using String Kernels. In *Advances in Neural Information Processing Systems 13*, MIT Press.

(Malakasiotis and Androutpoulos, 2007) Prodromos Malakasiotis and Ion Androutsopoulos. 2007. Learning Textual Entailment using SVMs and String Similarity Measures. In *Proceedings of the Workshop on Textual Entailment and Paraphrasing*, pages 42–47, Prague, June 2007.

(Marsi et al., 2006) Erwin Marsi, Emiel Krahmer, Wauter Bosma, and Mariët Theune. 2006. Normalized alignment of dependency trees for detecting textual entailment. In *Proceedings of the Second PASCAL Challenges Workshop on Recognising Textual Entailment*, Venice, Italy.

(McCune, 1994) William W. McCune, 1994. OTTER 3.0 Reference Manual and Guide.

(McCune, 1998) W. McCune. 1998. Automatic Proofs and Counterexamples for Some Ortholattice Identities. *Information Processing Letters*, 65(6):285-291.

(Mel'čuk, 1988) I. A. Mel'čuk. 1988. Dependency Syntax: theory and practice. *State University of New York Press*, Albany, NY.

(Meyers et al., 1996) Adam Meyers, Roman Yangarber, and Ralph Grisham. 1996. Alignment of shared forests for bilingual corpora. In *Proceedings of 16th International Conference on Computational Linguistics (COLING-96)*, pages 460–465, Copenhagen, Denmark.

(Miller, 1995) G. A. Miller. 1995. WordNet: A Lexical Databases for English. *Communications of the ACM*, pages 39–41, November.

(Moldovan et al., 2003) Dan Moldovan, Christine Clark, Sanda Harabagiu, and Steve Maiorano. 2003. COGEX A Logic Prover for Question Answering. In *Proceedings of the HLT/NAACL*, Edmonton, Canada, May.

(Moldovan and Rus, 2001) Dan Moldovan and Vasile Rus. 2001. Logic form transformation of wordnet and its applicability to question-answering. In *Proceedings of 39th Annual Meeting of the Association for Computational Linguistics*, Toulouse, France, July.

(Moschitti, 2004) Alessandro Moschitti. A study on Convolution Kernels for Shallow Semantic Parsing. In *Proceedings of the 42-th Conference on Association for Computational Linguistic (ACL-2004)*, Barcelona, Spain, 2004.

(Moschitti, 2006) Alessandro Moschitti. 2006. Making tree kernels practical for natural language learning. In *Proceedings of the Eleventh International Conference on European Association for Computational Linguistics*, Trento, Italy, 2006.

(Nairn et al., 2006) Rowan Nairn, Cleo Condoravdi, and Lauri Karttunen. 2006. Computing Relative Polarity for Textual Inference. In *Proceedings of ICoS-5*.

(Newman et al., 2006) Eamonn Newman, John Dunnion, and Joe Carthy. 2006. Constructing a Decision Tree Classifier using Lexical and Syntactic Features. In *Proceedings of the Second PASCAL Challenges Workshop on Recognising Textual Entailment*, Venice, Italy.

(Nicholson et al., 2006) Jeremy Nicholson, Nicola Stokes, and Timothy Baldwin. 2006. Detecting Entailment Using an Extended Implementation of the Basic Elements Overlap Metric. In *Proceedings of the Second PASCAL Challenges Workshop on Recognising*

Textual Entailment, Venice, Italy.

(Nielsen et al., 2006) Rodney D. Nielsen, Wayne Ward, and James H. Martin. 2006. Toward Dependency Path based Entailment. In *Proceedings of the Second PASCAL Challenges Workshop on Recognising Textual Entailment*, Venice, Italy.

(Niles and Pease, 2003) Ian Niles and Adam Pease. 2003. Linking lexicons and ontologies: Mapping wordnet to the suggested upper merged ontology. In H.R. Arabnia, editor, *IKE*. CSREA Press.

(Palmer et al., 2005) Martha Palmer, Dan Gildea, and Paul Kingsbury. 2005. The Proposition Bank: A Corpus Annotated with Semantic Roles. *Computational Linguistics* 31:1.

(Peñas et al., 2006) Peñas, A., Rodrigo, Á., Sama, V., and Verdejo, F. 2006. Overview of the Answer Validation Exercise 2006. In the *AVE 2006 Working Notes*.

(Platt, 1998) J. Platt. 1998. Machines using Sequential Minimal Optimization. In B. Schoelkopf and C. Burges and A. Smola, editors, *Advances in Kernel Methods - Support Vector Learning*, 1998.

(Quinlan, 1993) J.R. Quinlan, C4.5: Programs for Machine Learning. Morgan Kaufmann, 1993.

(Quinlan, 2002) Quinlan, R. 2002. C5.0, version 5.1. C5.0. http://www.rulequest.com.

(Quirk et al., 2005) Chris Quirk, Arul Menezes, and Colin Cherry. 2005. Dependency Treelet Translation: Syntactically Informed Phrasal SMT. In *Proceedings of ACL 2005*.

(Riazanov and Voronkov, 2002) A. Riazanov and A. Voronkov. 2002. The Design and Implementation of Vampire. *AI Communications*, 15(2-3).

(Riezler et al., 2002) Stefan Riezler, Tracy H. King, Ronald M. Kaplan, Richard Crouch, John T. III Maxwell, and Mark Johnson. 2002. Parsing the Wall Street Journal using a Lexical-Functional Grammar and Discriminative Estimation Techniques. In *Proceedings of ACL'02*, Philadelphia, PA.

(Romano et al., 2006) Lorenza Romano, Milen Kouylekov, Idan Szpektor, Ido Dagan, and Alberto Lavelli. 2006. Investigating a generic paraphrase-based approach for relation extraction. In *Proc. of EACL-06*.

(Roth and Yih, 2004) Roth, D. and Yih, W. 2004. A linear programming formulation for global inference in natural language tasks. In *Proc. of the 8th Conference on Computational Natural Language Learning*, pp1-8.

(Rus, 2006) Vasile Rus. 2006. Two Related Lexico-Syntactic Approaches to Entailment. In *Proceedings of the Second PASCAL Challenges Workshop on Recognising Textual Entailment*, Venice, Italy.

(Schilder and McInnes, 2006) Frank Schilder and Bridget Thomson McInnes. 2006. Word and tree-based similarities for textual entailment. In *Proceedings of the Second PASCAL Challenges Workshop on Recognising Textual Entailment*, Venice, Italy.

(Skiena, 1998) S. Skiena. 1998. The Algorithm Design Manual. *Springer-Verlag*, 1998.

(Szpektor et al., 2004) Idan Szpektor, Hristo Tanev, Ido Dagan and Bonaventura Coppola. 2004. Scaling Web-based Acquisition of Entailment Relations. In *Proceedings of EMNLP 2004*.

(Tatu et al., 2006) Marta Tatu, Brandon Iles, John Slavick, Adrian Novischi, and Dan Moldovan. 2006. COGEX at the Second Recognizing Textual Entailment Challenge. In *Proceedings of the Second PASCAL Challenges Workshop on Recognising Textual Entailment*, Venice, Italy.

(Tatu and Moldovan, 2007) Marta Tatu and Dan Moldovan. 2007. COGEX at RTE 3. In *Proceedings of the Workshop on Textual Entailment and Paraphrasing*, pages 22–27, Prague, June 2007.

(Turney, 2001) Turney, Peter D. (2001). Mining the web for synonyms: PMI-IR versus LSA on TOEFL. In *Proceedings of the Twelfth European Conference on Machine Learning (ECML-2001)*, 491–502.

(Vanderwende and Dolan, 2005) Vanderwende, L., Dolan,W.B. 2005. What syntax can contribute in the entailment task. In *PASCAL Challenges Workshop on Recognizing Textual Entailment*, Southampton, United Kingdom, Springer-Verlag (2005) 205–216.

(Vanderwende et al., 2006) Lucy Vanderwende, Arul Menezes, and Rion Snow. 2006. Microsoft Research at RTE-2: Syntactic Contributions in the Entailment Task: an implementation. In *Proceedings of the Second PASCAL Challenges Workshop on*

Recognising Textual Entailment, Venice, Italy.

(Vapnik, 1995) V. Vapnik. 1995. The Nature of Statistical Learning Theory. *Springer*, N.Y., 1995.

(Wang and Neumann, 2007a) Rui Wang and Günter Neumann. 2007a. Recognizing textual entailment using a subsequence kernel method. In *Proceedings of the Twenty-Second Conference on Artificial Intelligence (AAAI 2007)*, pp. 937–942, Vancouver, Canada.

(Wang and Neumann, 2007b) Rui Wang and Günter Neumann. 2007b. Recognizing Textual Entailment Using Sentence Similarity based on Dependency Tree Skeletons. In *Proceedings of the Workshop on Textual Entailment and Paraphrasing*, pages 36–41, Prague, June 2007.

(Wang and Neumann, 2007c) Rui Wang and Günter Neumann. 2007c. Working Notes in *Answer Validation Exercise of CLEF2007*.

(Witten and Frank, 1999) Witten, I. H. and Frank, E. Weka: Practical Machine Learning Tools and Techniques with Java Implemen-tations. Morgan Kaufmann, 1999.

(Wu and Palmer, 1994) Zhibiao Wu and Martha Palmer. 1994. Verb semantics and lexical selection. In *Proceedings of the 32nd. Annual Meeting of the Association for Computational Linguistics*, pages 133-138. New Mexico State University, Las Cruces, New Mexico.

(Zaki, 2005) Mohammed J. Zaki. 2005. Efficiently Mining Frequent Trees in a Forest: Algorithms and Applications. In *IEEE Transaction on Knowledge and Data Engineering, special issue on Mining Biological Data*.

(Zanzotto and Moschitti, 2006) Fabio Massimo Zanzotto and Alessandro Moschitti. 2006. Automatic learning of textual entailments with cross-pair similarities. In *Proceedings of the Joint 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics (COLING-ACL)*, Sydney, Australia, 2006.

(Zanzotto et al., 2006) F.M. Zanzotto, A. Moschitti, M. Pennacchiotti, and M.T. Pazienza. 2006a. Learning textual entailment from examples. In *Proceedings of the Second PASCAL Challenges Workshop on Recognising Textual Entailment*, Venice, Italy.

(Zhang and Shasha, 1990) Kaizhong Zhang ,Dennis Shasha. Fast algorithm for the unit cost editing distance between trees. *Journal of algorithms*, vol. 11, p. 1245-1262, December.

Appendix

A.1 Generalization Rules

<SUBJ>
<BY-SUBJ> → <SUBJ>
<OBJ1> → <SUBJ>
<OBJ>
<OBJ2> → <OBJ>
<DESC> → <OBJ>
<APPO>
<CONJ> → <APPO>
<NN> → <APPO>
<ABBREV> → <APPO>
<NUM> → <APPO>
<TITLE> → <APPO>
<AMOUNT-VALUE> → <APPO>
<INSIDE> → <APPO>
<MOD>
<AMOD> → <MOD>
<PNMOD> → <MOD>
<APPO-MOD> → <MOD>
<PERSON> → <MOD>

Table 19 Generalization Rules

A.2 Closed-Class Symbol Set

Types	Symbols
Dependency Relation Tags	*<OBJ>, <SUBJ>, <GEN>, <I>, <MOD-BEFORE>*
POS Tags	*N, V, PREP, A*

Table 20 The Set of Closed-Class Symbols

A.3 Output Format Adaption of the Stanford Parser

Dependency Tags
If contains "mod" → <MOD>
If contains "subj" → <SUBJ>
If contains "obj" → <OBJ>
If contains "prep" → <PREP>
If contains "aux" → <AUX>
If contains "comp" → <COMP>
If contains "appo" → <APPO>
If contains "conj" → <CONJ>
POS tags
NN/NNS/NNP/NNPS/PRP/PRP$/CD → N
VB/VBD/VBG/VBN/VBP/VBZ → V
DT → Det
JJ/JJS/JJR → A
RB/RBR/RBS → ADV
TO/IN → Prep

Table 21 Format Adaption Rules for the Stanford Parser

A.4 Screenshots of the GUI

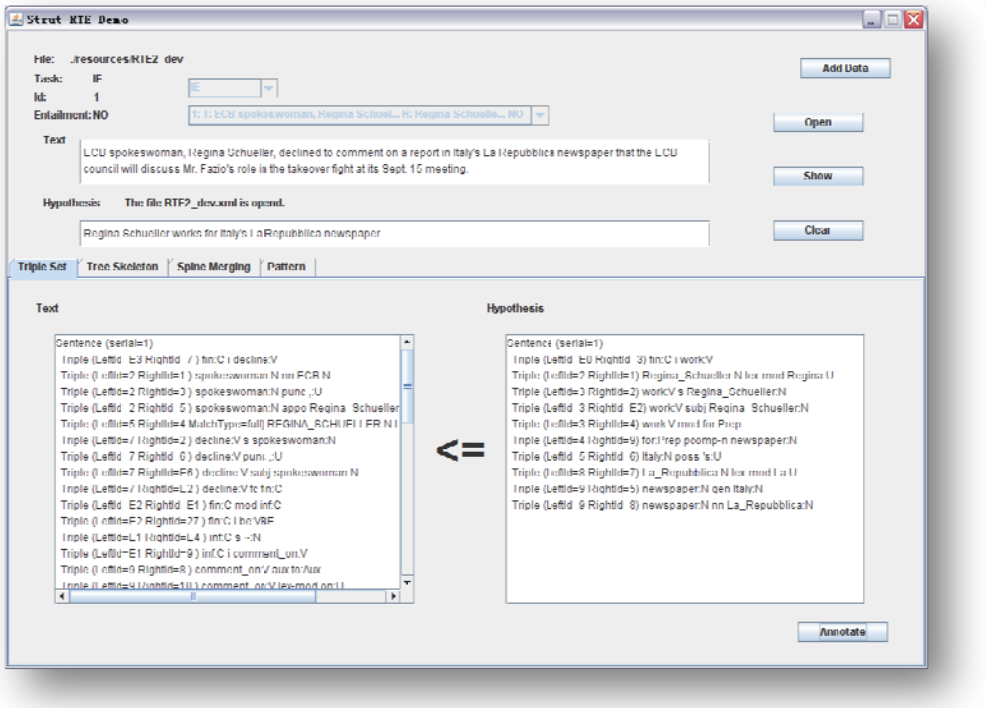

Figure 15 Triple Representation

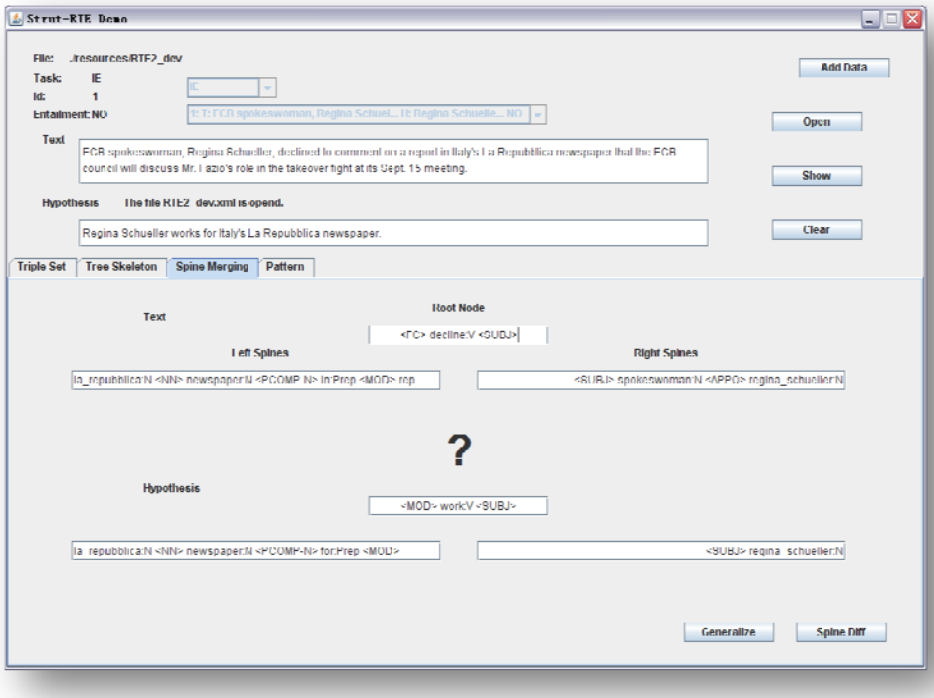

Figure 16 Tree Skeletons

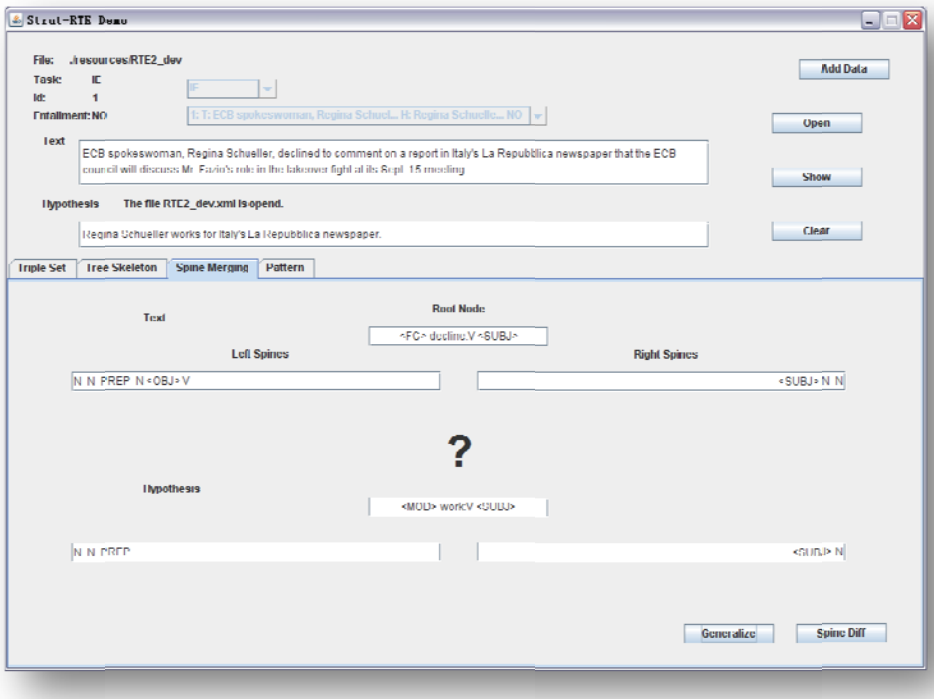

Figure 17 Spine Generalization

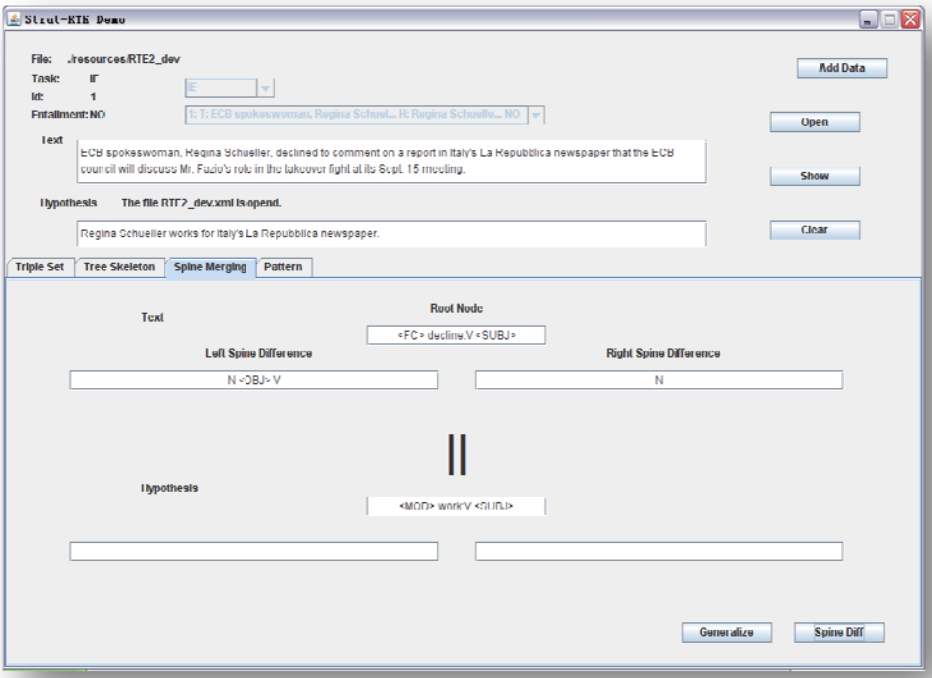

Figure 18 Spine Differences

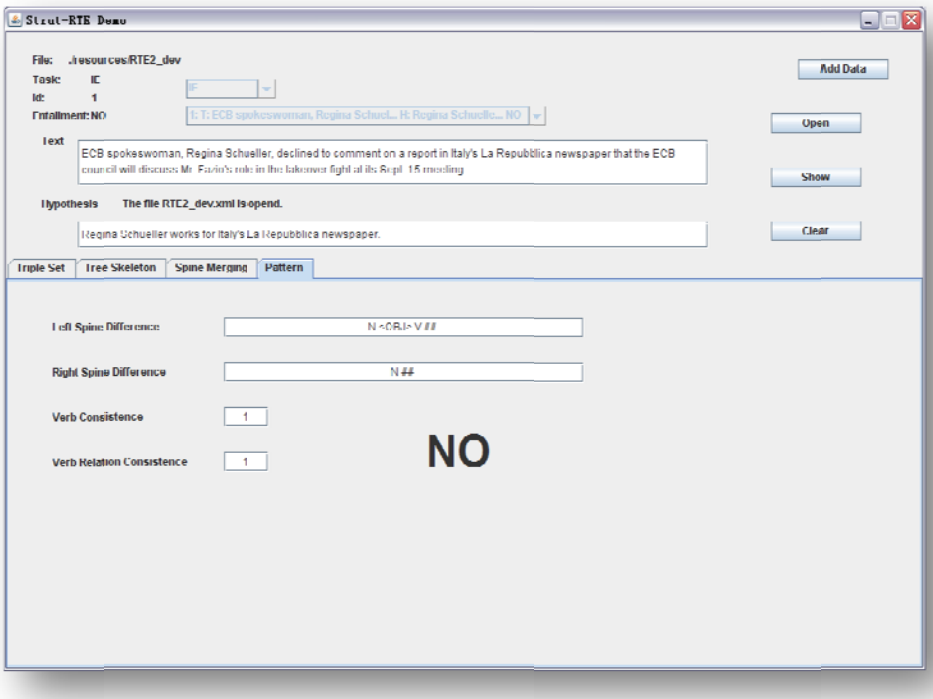

Figure 19 Entailment Pattern