APPLYING SEMANTIC ANALYSIS TO FINDING SIMILAR QUESTIONS IN COMMUNITY QUESTION ANSWERING SYSTEMS

NGUYEN LE NGUYEN

NATIONAL UNIVERSITY OF SINGAPORE

2010

APPLYING SEMANTIC ANALYSIS TO FINDING SIMILAR QUESTIONS IN COMMUNITY QUESTION ANSWERING SYSTEMS

NGUYEN LE NGUYEN

A THESIS SUBMITTED FOR THE DEGREE OF
MASTER OF SCIENCE

SCHOOL OF COMPUTING
NATIONAL UNIVERSITY OF SINGAPORE

2010

Dedication

To my parents: Thong, Lac and my sister Uyen for their love.

"Never a failure, always a lesson."

Acknowledgments

My thesis would not have been completed without the help of many people to whom I would like to express my gratitude.

First and foremost, I would like to express my heartfelt thanks to my supervisor Prof. Chua Tat Seng. For past two years, he had been guiding and helping me through serious research obstacles. Specially, during my rough time facing study disappointment, he was not only encouraging me with crucial advice, but also supporting me financially. I always remember what he was doing to give insightful comments and critical reviews of my work. Last but not least, he is very nice to his students at all times.

I would like to thank my thesis committee members Prof. Tan Chew Lim and A/P Ng Hwee Tou for their feedback of my GRP and thesis works. Furthermore, during my study in National University of Singapore (NUS), many Professors imparted me knowledge and skills, gave me good advice and help. Thanks to A/P Ng Hwee Tou for his interesting course in basic and advance Natural Language Processing, A/P Kan Min Yen, and other Professors in NUS.

To complete the description of the research atmosphere at NUS, I would like to thank my friends. Ming Zhaoyan, Wang Kai, Lu Jie, Hadi, Yi Shiren, Tran Quoc Trung and many people in Lab for Media Search (LMS) are very good and cheerful friends, who helped me to master my research and adapt the wonderful life in NUS. My research life would not have been so endeavoring without you. I wish all of you brilliant success on your chosen adventurous research path at NUS. The memories about LMS shall stay with me forever.

Finally, the greatest gratitude goes to my parents and my sister for their love and enormous support. Thank you for sharing your rich life experience and helping me in this right decision of my life. I am wonderfully blessed to have such a wonderful family.

Abstract

Research in Question Answering (QA) has been carried out for a long time from the 1960s. In the beginning, traditional QA systems were basically known as the expert systems that find the factoid answers in the fixed document collections. Recently, with the emergence of World Wide Web, automatically finding the answers to user's questions by exploiting the large-scale knowledge available on the Internet has become a reality. Instead of finding answers in a fixed document collection, QA system will search the answers in the web resources or community forums if the similar question has been asked before. However, there are many challenges in building the QA systems based on community forums (cQA). These include: (a) how to recognize the main question asked, especially on measuring the semantic similarity between the questions, and (b) how to handle the grammatical errors in forums language. Since people are more casual when they write in forums, there are many sentences in the forums that contain grammatical errors and are semantically similar but may not share any common words. Therefore, extracting semantic information is useful for supporting the task of finding similar questions in cQA systems.

In this thesis, we employ a semantic role labeling system by leveraging on grammatical relations extracted from a syntactic parser and combining it with a machine learning method to annotate the semantic information in the questions. We then utilize the similarity scores by using semantic matching to choose the similar questions. We carry out experiment based on the data sets collected from Healthcare domain in Yahoo! Answers over a 10-month period from 15/02/08 to 20/12/08. The results of our experiments show that with the use of our semantic annotation approach named GReSeA, our system outperforms the baseline Bag-Of-Word (BOW) system in terms of MAP by 2.63% and Precision at top 1 retrieval results by 12.68%. Compared with using the popular SRL system ASSERT (Prad-

han et al., 2004) on the same task of finding similar questions in Yahoo! Answer, our system using GReSeA outperforms those using ASSERT by 4.3% in terms of MAP and by 4.26% in Precision at top 1 retrieval results. Additionally, our combination system of BOW and GReSeA achieves the improvement by 2.13% (91.30% vs. 89.17%) in Precision at top 1 retrieval results when compared with the state-of-the-art Syntactic Tree Matching (Wang et al., 2009) system in finding similar questions in cQA.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In the world today, information has become the main reason that enables people to succeed in their business. However, one of the challenges is how to retrieve useful information among the huge amount of information on the web, books, and data-warehouses. Most information is phrased in natural language form which is easy for human to understand but not amendable to automated machine processing. In addition, with the explosive amount of information, it requires vast computing powers of computers to perform the analysis and retrieval. With the development of Internet, search engines such as Google, Bing (Microsoft), Yahoo, etc. have became widely used by all to look for information in our world. However, the current search engines process the information requirements based on surface keyword matching, and thus, the retrieval results are low in the quality.

With improvement in Machine Learning techniques in general and Natural Language Processing (NLP) in particular, more advanced techniques are available to tackle the problem of imprecise information retrieval. Moreover, with the success of Penn Tree Bank project, large sets of annotated corpora in English for NLP tasks such as Part Of Speech (POS), Name Entities, syntactic and semantic parsing, etc. were released. However, it is also clear that there is a reciprocal effect

between the accuracy of supporting resources such as syntactic, semantic parsing and the accuracy of search engines. In addition, with differences in domains and domain knowledge, search engines often require different adapted techniques for each domain. Thus the development of advanced search solution may require the integration of appropriate NLP components depending on the purpose of the system. In this thesis, our goal is to tackle the problem of Question Answering (QA) system in community QA systems such as Yahoo! Answer.

QA system was developed in the 1960s with the goal of automatically answering the questions posed by users in natural language. To find the correct answer, a QA system analyzes the question to extract the relevant information and generates the answers from either a pre-structured database or a collection of plain text (un-structure data), or web pages (sem-structured data).

Similar to many search engines, QA research needs to deal with many challenges. The first challenge is the wide range of question types. For example, in natural language, question types are not only limited to factoid, list, how, and why type questions, but also semantically-constrained and cross-lingual questions. The second challenge is the techniques required to retrieve the relevant documents available in generating the answers. Because of the explosion of information on the Internet in recent years, many search collections exist that may vary from small-scale local document collection in a personal computer, to large-scale Web pages in the Internet. Therefore, the QA systems require appropriate and robust techniques adapting to document collections for effective retrieval. Finally, the third challenge is in performing domain question answering, which can be divided into two groups:

- Closed-domain QA: which focuses on generating the answers under a specific domain (for example, music entertainment, health care, etc.). The advantage of working in closed-domain is that the system can exploit the domain knowledge in finding precise answers.

- Open-domain QA: that deals with questions without any limitation. Such systems often need to deal with enormous dataset to extract the correct answers.

Unlike information extraction and information retrieval, QA system requires more complex natural language processing techniques to understand the question and the document collections to generate the correct answers. On the other hand, QA system is the combination of information retrieval and information extraction.

1.1 Problem statement

Recently, there has been a significant increase in activities in QA research, which includes the integration of question answering with web search. QA systems can be divided into two main groups:

(1) Question Answering in a fixed document collection: This is also known as the traditional QA or expert systems that are tailored to specific domains to answer the factoid questions. With the traditional QA, people usually ask a factoid question in a simple form and expect to receive a correct and concise answer. Another characteristic of traditional QA systems is that one question can have multiple correct answers. However, all correct answers often present in a simple form such as an entity, or a phrase instead of a long sentence. For example, with the question "Who is Bill Gates?", traditional QA systems have these following answers: "Chairman of Microsoft", "Co-Chair of Bill & Melinda Gates Foundation", etc. In addition, traditional QA systems focusing on generating the correct answers in a fixed document collection so they can exploit the specific knowledge of the predefined information collections, including (a) the documents collected are presented as standard free text or structure document; (b) the language used in these documents is grammatical

correct writing in a clear style; and (c) the size of the document collection is fixed so techniques required for constructing data are not complicated.

In general, the current architecture of traditional QA systems typically include two modules (Roth et al., 2001):

- Question processing module with two components. (i) Question classification that classifies the type of question and answer. (ii) Question formulation that expresses a question and an answer in a machine-readable form.
- Answer processing module with two components. (i) Passage retrieval component uses search engines as a basic process to identify documents in the document set that likely contain the answers. It then selects the smaller segments of texts that contain the strings or information of the same type as the expected answers. For example, with the question "Who is Bill Gates?", the filter returns texts that contain information about "Bill Gates". (ii) Answer selection component looks for concise entities/information in the texts to determine if the answer candidates can indeed answer the question.

(2) Question Answering in community forums (cQA): Unlike traditional QA systems that generate answers by extracting from a fixed set of document collections, cQA systems reuse the answers for questions from community forums that are semantically similar to user's questions. Thus the goal of finding answers from the enormous data collections in traditional QA system is replaced by finding semantically similar questions in online forums; and then using their answers to answer user's question. In this way, cQA systems can exploit the human knowledge in users generated contents stored in online forums to find the answers.

In online forums, people usually seek solutions to problems that occurred in their real life. Therefore, the popular type of questions in cQA is the "how" type question. Furthermore, the characteristics of questions in traditional QA and cQA are different. While in traditional QA, people often ask simple questions and expect to receive simple answers. In cQA, people always submit a long question to explain their problems and they hope to receive a long answer with more discussion about their problems. Another difference between traditional QA and cQA is the relationships between questions and answers. In cQA, there are two relationships between question and answer: (a) one question has multiple answers; and (b) multiple questions refer to one answer. The reason why multiple questions have the same answer is because in many cases, different people have the same problem in their life, but they pose questions in different threads in forum. Thus, only one solution is sufficient to answer all similar problems posed by the users.

The next difference between traditional QA and cQA is about the document collections. Community forums are the places where people freely discuss about their problems so there are no standard structures and presentation styles required in forums. The languages used in the forums are often badly-formed and ungrammatical because people are more casual when they write in forums. In addition, while the size of document collections in traditional QA is fixed, the numbers of thread in community forum increase day by day. Therefore, cQA requires an adaptive technique to retrieve documents in dynamic forum collections.

In general, question answering in community forums can be considered as a specific retrieval task (Xue et al., 2008). The goal of cQA becomes that of finding relevant question-answer pairs for new user's questions. The retrieval task of cQA can also be considered as an alternative solution for the challenge of traditional QA, which focuses on extracting the correct answers. The comparison between traditional QA and cQA is summarized in Table 1.1.

	Traditional QA	Community QA
Question type	Factoid question Simple question → Simple answer	"How" type question Long question → Long answer
Answer	One question → multiple answers	One question → multiple answers Multiple questions → one answer
Language characteristic	Grammatical, clear style	Ungrammatical, Forum language
Information Collections	Standard free text and structure documents Using predefined collection documents	No standard structure required Using dynamic forum collections

Table 1.1: The comparison between traditional QA and community QA

1.2 Analysis of the research problem

Since the questions in traditional QA were written in a simple and grammatical form, many techniques such as rule based approach (Brill et al., 2002), syntactic approach (Li and Roth, 2006), logic form approach (Wong and Mooney, 2007), and semantic information approach (Kaisser and Webber, 2007; Shen and Lapata, 2007; Sun et al., 2005; Sun et al., 2006) were applied in traditional QA to process the questions. In contrast, questions in cQA were written in a badly-formed and ungrammatical language, so techniques applied for question processing are limited. Although people believe that extracting semantic information is useful to support the process of finding similar questions in cQA systems, the most promising approach used in cQA is statistical technique (Berger et al., 2000; Jeon et al., 2005; Xue et al., 2008). One of the reasons semantic analysis cannot be applied effectively in cQA is that semantic analysis may not handle the grammatical errors well in forum language. To circumvent the grammatical issues, we propose an approach to exploit the syntactic and dependency analysis that is robust to grammatical errors in cQA. In our approach, instead of using the deep features in syntactic relation, we focus on the general features extracted from full syntactic parser tree that are useful to analyzing the semantic information. For example, in Figure 1.1, the two noun phrases "the red car" and "the car" have different syntactic relations. However, in general view, these two noun phrases describe the same object "the car". Based

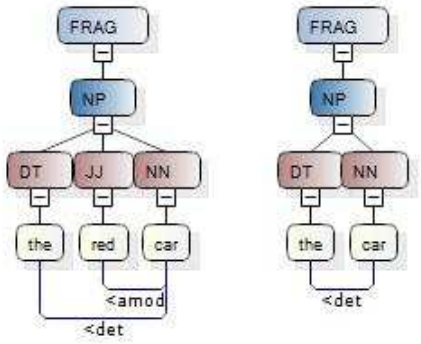

Figure 1.1: Syntactic trees of two noun phrases “the red car” and “the car”

on the general features from syntactic trees combined with dependency analysis, we recognize the relation between the word and its predicate. This relation then becomes the input feature to the next stage that uses machine learning method to classify the semantic labels. When applying to forum languages, we found that our approach using general features is effective in tackling the grammatical errors when analyzing semantic information.

To develop our system, we collect and analyze the general features extracted from two resources: PropBank data and questions in Yahoo! Answers. We then select 20 sections from Section 2 to Section 21 in the data sets released in CoNLL 2005 to train our classification model. Because we do not have the ground truth data sets to evaluate the performance of annotating semantic information, we use an indirect method by testing it on the task of finding similar questions in community forums. We apply our approach to annotate the semantic information and then utilize the similarity score to choose the similar questions. The Precision (percentage of similar questions that are correct) of finding similar questions reflects the Precision in our approach. We use the data sets containing about 0.5 million question-answer pairs from Healthcare domain in Yahoo! Answers from 15/02/08 to 20/12/08 (Wang et al., 2009) as the collection data sets. We then selected 6 subcategories including Dental, Diet&Fitness, Diseases, General Healthcare, Men’s

health, and Women's health to verify our approach in cQA. In our experiments, first, we use our proposed system to analyze the semantic information and use this semantic information to find similar questions. Second, we replace our approach by ASSERT (Pradhan et al., 2004), a popular system for semantic role labeling, and redo the same steps. Lastly, we compare the performance of the two systems with the baseline Bag-Of-Word (BOW) approach in finding similar questions.

1.3 Research contributions and significance

The main contributions of our research is two folds: (a) We develop a robust technique adapting to handle grammatical errors to analyze semantic information in forum language.(b) We conduct the experiments to apply semantic analysis to finding similar questions in cQA. Our main experiment results show that our approach is able to effectively tackle the grammatical errors in forum language and improves the performance of finding similar questions in cQA as compared to the use of ASSERT (Pradhan et al., 2004) and the baseline BOW approach.

1.4 Overview of this thesis

In chapter 2, we survey related work in traditional QA systems. Chapter 3 surveys related work in cQA systems. Chapter 4 introduces semantic role labeling and its related work. In chapter 5, we present our architecture for semantic parser to tackle the issues in forum language. Chapter 6 describes our approach to apply semantic analysis to finding similar questions in cQA systems. Finally, chapter 7 presents the conclusion and our future works.

Chapter 2

Traditional Question Answering Systems

The 1960s saw the development of the early QA systems. Two of the most famous systems in 1960s (Question-Answering-Wikipedia, 2009) are "BASEBALL" which answers questions about the US baseball league and "LUNAR" which answers questions about the geological analysis of rocks returned by the Apollo moon missions. In 1970s and 1980s, the incorporation of computational linguistic led to open-domain QA systems that contain comprehensive knowledge to answer a wide range of questions. In the late 1990s, the annual Text Retrieval Conference (TREC) has been releasing the standard corpus to evaluate QA performance, and has been used by many QA systems until present. The TREC QA includes a large number of factoid questions that varied from year to year (TREC-Overview, 2009; Dang et al., 2007). Many QA systems evaluate their performance in answering factoid questions from many topics. The best QA system achieved about 70% accuracy in 2007 for the factoid-based question (Dang et al., 2007).

The goal of the traditional QA is to directly return answers, rather than documents containing answers, in response to a natural language question. Traditional

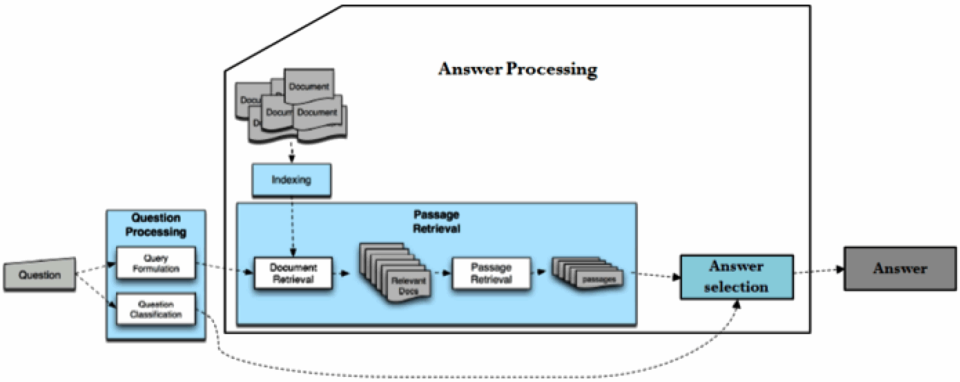

Figure 2.1: General architecture of traditional QA system

QA focuses on factoid questions. A factoid question is a fact-based question with short answer such as "Who is Bill Gates?". With one factoid question, traditional QA systems locate multiple correct answers in multiple documents. Before 2007, TREC QA task provides text document collections from newswire so that the language used in the document collections is a well-formed (Dang et al., 2007). Therefore, many techniques can be applied to improve the performance of traditional QA systems. In general, the architecture of traditional QA systems, as illustrated in Figure 2.1, includes two main modules: question processing, and answer processing (Roth et al., 2001).

2.1 Question processing

The goal of this task is to process the question so that the question is represented in a simple form with more information. Question processing is one of the useful steps to improve the accuracy of information retrieval. Specifically, question processing has two main tasks:

- *Question classification* which determines the type of the question such as Who, What, Why, When, or Where. Based on the type of the question,

traditional QA systems try to understand what kind of information is needed to extract the answer for user's questions.

- *Question formulation* which identifies various ways of expressing the main content of the questions given in natural language. The formulation task also identifies the additional keywords needed to facilitate the retrieval of main information needed.

2.2 Question classification

This is an important part to determine the type of question and find the correct answer type. A goal of question classification is to categorize questions into different semantic classes that impose constraints on potential answers. Question classification is quite different with text classification because questions are relatively short and contain less word-based information. Some common words in document classification are stop-words and there are less important for classification. Thus, stop-word is always removed in document classification. In contrast, the roles of stop-words tend to be important because they provide information such as collocation, phrase mining, etc. for question classification. The following example illustrates the difference between question before and after stop-word removal.

S1: Why do I not get fat no mater how much I eat?

S2: do get fat eat?

In this example, S2 represent the question S1 after removing stop-words. Obviously, with fewer words in sentence S2, it becomes an impossible task for QA system to classify the content of S2.

Many earlier works have suggested various approaches for classifying questions (Harabagiu et al., 2000; Ittycheriah and Roukos, 2001; Li, 2002; Li and Roth, 2002; Li and Roth, 2006; Zhang and Lee, 2003) including using rule-based models,

statistical language models, supervised machine learning, and integrated semantic parsers, etc. In 2002, Li presented an approach using language model to classify questions (Li, 2002). Although language modeling achieved the high accuracy about 81% in 693 TREC questions, it has the usual drawback with the statistical approaches to build the language model, as it requires extensive human labors to create a large amount of training samples to encode their models. Another approach proposed by Zhang et al. exploits the advantage of the syntactic structures of question (Zhang and Lee, 2003). This approach uses supervised machine learning with surface text features to classify the question. Their experiment results show that the syntactic structures of question are really useful to classify the questions. However, the drawback of this approach is that it does not exploit the advantage of semantic knowledge for question classification. To overcome these drawbacks, Li et al. presented a novel approach that uses syntactic and semantic analysis to classify the question (Li and Roth, 2006). In this way, question classification can be viewed as a case study in applying semantic information to text classification. Achieving the high accuracy of 92.5%, Li et al. demonstrated that integrating semantic information into question classification is the right way to deal with question classification.

In general, question classification task has been tackled with many effective approaches. In these approaches, the main features used in question classification include: syntactic features, semantic features, named entities, WordNet senses, class-specific related words, and similarity based categories.

2.2.1 Question formulation

In order to find the answers correctly, one important task is to understand what the question is asking for. Question formulation task is to extract the keywords from the question and represent the question in a suitable form to find answers.

The ideal formulation should impose constraints on the answer so that QA systems may identify many candidate answers to increase the system's confidence in them.

In question formulation, many approaches were suggested. Brill et al. introduced a simple approach to rewrite a question as a simple string based on manipulations (Brill et al., 2002). Instead of using a parser or POS tagger, they used a lexicon for a small percentage of rewrites. In this way, they created the rewrite rules for their system. One advantage of this approach is that the techniques are very simple. However, creating the rewrite rules is a challenge for this approach such as how many rules are needed, and how the rule set is to be evaluated, etc.

Sun et al. presented another approach to reformulate questions by using syntactic and semantic relation analysis (Sun et al., 2005; Sun et al., 2006). They used web resources to solve their problem in formulating question. They found the suitable query keywords suggested by Google and replaced it for the original query. By using semantic parser ASSERT, they parsed the candidate query into expanded terms and analyzed the relation paths based on dependency relations. Sun's approach has many advantages by exploiting the knowledge from Google and the semantic information from ASSERT. However, this approach depends on the results of ASSERT, hence the performance of their system is dependent on the accuracy of the automatic semantic parser.

Kaisser et al. used a classical semantic role labeler combined with a rule-based approach to annotate a question (Kaisser and Webber, 2007). This is because factoid questions tend to be grammatically simple so they can find the simple rules that help the question annotation process dramatically. By using resources from FrameNet and PropBank, they developed a set of abstract frame structure. By mapping the question analysis with this frame, they are able to infer the question they want. Shen et al. also used semantic roles to generate a semantic graph structure that is suitable for matching a question and a candidate answers (Shen and

Lapata, 2007). However, the main problem with these approaches is the ambiguity in determining the main verb when there is more than one verb in the question. As long questions have more than one verb, their systems will be hard to find a rule set or a structure, which can be used to extract the correct information for these questions.

Applying semantic information to question classification (Kaisser and Webber, 2007; Shen and Lapata, 2007; Sun et al., 2005; Sun et al., 2006) achieves the highest accuracies. For example, Sun's QA system obtains 71.3% accuracy in finding factoid answers in TREC-14 (Sun et al., 2005). However, the disadvantage of these approaches is that they are highly dependent on the performance of the semantic parsers. In general, semantic parsers do not work well in cases of long sentences and especially the ungrammatical sentences. In such cases, the semantic parsers tend not to return any semantic information, and hence the QA systems cannot represent the sentence with semantic information.

Wong et al., on the other hand represented a question as a query language (Wong and Mooney, 2007). For example, the question "What is the smallest state by area?" is represented as the following query form

$$answer(x1, smallest(x2, state(x1), area(x1, x2))).$$

The parser tree of this query form is shown in Figure 2.2[1].

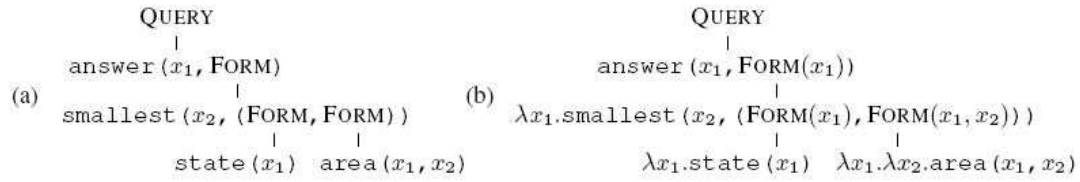

Figure 2.2: Parser tree of the query form

Similar to (Wong and Mooney, 2007) to enable QA system to understand the

[1]The figure is adapted from (Wong and Mooney, 2007)

question given in natural language, Lu et al. presented an approach to represent the meaning of a sentence with hierarchical structures (Lu et al., 2008). They suggested an algorithm for learning a generative model that is applied to map sentences to hierarchical structures of their underlying meaning. The hierarchical tree structure of sentence "How many states do not have rivers?" is shown in Figure 2.3[2].

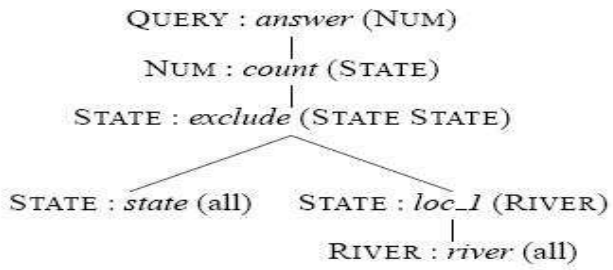

Figure 2.3: Example of meaning representation structure

Applying these approaches (Lu et al., 2008; Wong and Mooney, 2007), information about a question such as the question type and information asked was represented fully in a structure form. Because the information in both questions and answer candidates are represented fully and clearly, the process of finding answers can achieve higher accuracies. Lu's experiments show that their approach obtains the effective result of 85.2% in finding answers for Geoquery data set. Unfortunately, to retrieve the answers as the query from the database, one needs to consider is how to build the database. Since the cost of preprocessing the data is expensive, using the query structure for question answering has severe limitation about the knowledge domain.

Bendersky et al. proposed the technique to process a query through identifying the key concepts (Bendersky and Croft, 2008). They used the probabilistic model to integrate the weights of key concepts in verbose queries. Focusing on the keyword queries extracted from the verbose description of the actual information is an important step to improving the accuracy of information retrieval.

[2]The figure is adapted from (Lu et al., 2008)

2.2.2 Summary

In question processing task, many techniques were applied and achieved promising performance. In particular, applying semantic information and meaning representation are the most promising approaches. However, several drawbacks exist in these approaches. The heavy dependent on semantic parser and limitations about the domain knowledge severely limit the application of these approaches to realistic problems. In particular, applying semantic analysis in QA tracks in TREC 2007 and later faces many difficulties about the characteristics of blog language because from 2007, QA tracks in TREC collected documents not only from newswire but also from blog. Therefore, there is a need to improve the performance of semantic parsers to work well with the mix of "clean" and "noise" data.

2.3 Answer processing

As we mention above, the goal of the traditional QA is to directly return the correct and concise answers. However, finding the documents that contains relevant answers is always easier than finding the short answers. The performance of traditional QA systems is represented through the accuracy of the answers finding. Hence, answer processing is the most important task to select the correct answers from numerous candidate relevant answers.

The goal of answer processing task can be described as two main steps:

- *Passage retrieval* which has two components: (a) information retrieval that retrieves all relevant documents from the local databases or web pages; and (b) information extraction that extracts the information from the sub-set of documents retrieved. The goal of this task is to find the best paragraphs or phrases that contain the answer candidates to the questions.

- *Answer selection* which selects the correct answers from the answer candidates

through matching the information in the question and information in the answer candidates. In general, all answer candidates are re-ranking using one or more approaches and the top answer candidates are presented the best answer.

2.3.1 Passage retrieval

Specifically, the passage retrieval comprises two steps. The first step is information retrieval. The main role of this step is to retrieve a subset of entire document collections, which may contain the answers, from local directory or web. In this task, high recall is required because the QA systems do not want to miss any candidate answer. Techniques used for ranking document and information retrieval were used in this task such as Bag-of-Word (BoW), language modeling, term weighting, vector space model, and probabilistic ranking principle, etc. (Manning, 2008)

To make the QA systems more reliable in finding the answers for the real world questions, instead of seeking in the local document collections, QA systems typically also use web resources as the external supplements to find the correct answers. Two popular web resources used to help in document retrieval are http://www.answers.com (Sun et al., 2005) and Wikipedia (Kaisser, 2008). The advantages of using these resources are that they contain more context and related concepts to the query. For example, information extracted from web pages such as the title is very useful for the next step to match with information in the question.

The second step is information extraction. The goal of this step is to extract the best candidates containing the correct answers. Normally, the correct answers can be found in one or more sentences or a paragraph. However, in the long documents, sentences contain answers can be in any position of the document. Thus, information extraction requires many techniques to understand the natural language content in the documents.

One of the simplest approaches for extracting the answer candidates, employed by MITRE (Light et al., 2001), is matching the information presenting in the question with information in the documents. If the question and the sentence in the relevant document have many words overlap, then the sentence is may contain the answer. However, matching based on counting the number of words overlapping has some drawbacks. First, two sentences have many common words overlapping may not be semantically similar. Second, many different words have similar meaning in natural language, thus, matching through word overlap is not an effectively approach. Obviously, word-word matching or strict matching cannot be used for matching the semantic meaning between two sentences.

To tackle this drawback, PiQASso (Attardi et al., 2001) employed the dependency parser and used dependency relations to extract the answers from the candidate sentences. If the relations reflected in the question are matched with the candidate sentence, this sentence was selected as the answer. However, the above system selects the answer based on strict matching of dependency relations. In (Cui et al., 2005), Cui et al. analyzed the disadvantages in strict matching for matching dependency relations between questions and answers. Strict matching fails when the equivalences of semantic relationships are phrased differently. Therefore, these methods often retrieve the incorrect passages modified by the question terms. They proposed two approaches to perform fuzzy relation matching based on statistical models: mutual information and statistical translation (Cui et al., 2005).

- Mutual information: they measured relatedness of two relations by their bipartite co-occurrences in the training path except the co-occurrences of the two relations in long paths.

- Statistical translation: they used GIZA to compute the probability score between two relations.

Sun et al. suggested an approach using Google snippets as the local context

and sentence based matching to retrieve passages (Sun et al., 2006). Exploiting Google snippets improves the accuracy of passage retrieval because the snippets give more information about the passage such as the title, context of passage, position of the passage in the document, etc.

Miyao et al. proposed a framework for semantic retrieval consisting of two steps: offline processing and online retrieval (Miyao et al., 2006). In offline processing, they used semantic to annotate all sentences in a huge corpus with predicate argument structures and ontological identifiers. Each entity in real world is represented as an entry in ontology databases with pre-defined template and event expression ontology. In online processing, their system retrieves information through structure matching with pre-computed semantic annotations. The advantage of their approach is that it exploits information about the ontology and template structures built in the offline step. However, this approach requires an expensive step to build the predicate argument structures and ontological identifiers. It thus has severe limitation about the domain when applying to real data.

Ahn et al. proposed the method named Topic Indexing and Retrieval to directly retrieved answer candidates instead of retrieving passages (Ahn and Webber, 2008). The basic idea is in extracting all possible named entity answers in a textual corpus offline based on three kinds of information: textual content, ontological type, and relations. The expressions were seen as the potential answers that support the direct retrieval in their QA system. The disadvantage of Topic Indexing and Retrieval method is that this approach is effective and efficient only for questions with named entity answers.

Pizzato et al. proposed a simple technique named Question Prediction Language Model (QPLM) for QA (Pizzato et al., 2008). They investigated the use of semantic information for indexing documents and employed the vector space model (three kinds of vector: bag-of-words, partial relation, full relation) for ranking doc-

uments. Figure 2.4 and 2.5[3] illustrate the example for Pizzato's approach.

QPLM representation for "*Bill kicked the ball*":

ID	Relation
11	$Who(kick) \rightarrow Bill$
12	$What(kick) \rightarrow ball$

Inverted file representation:

Term	Document	Rel. ID	Rel. Type	Role
Bill	1	11	Who	Arg
kick	1	11	Who	Pred
		12	What	Pred
ball	1	12	What	Arg

Figure 2.4: Simplified representation of the indexing of QPLM relations

Query	Returns documents that
$*(kick) \rightarrow *$	contain the word kick
$Who(kick) \rightarrow *$	inform that someone kicks
$Who(*) \rightarrow Bill$	inform that Bill does an action
$Who(kick) \rightarrow Bill$	inform that Bill kicks

Figure 2.5: QPLM queries (anterisk symbol is used to represent a wildcard)

Similar to previous approaches that use semantic information (Kaisser and Webber, 2007; Shen and Lapata, 2007; Sun et al., 2005; Sun et al., 2006), the disadvantage of Pizzato's approach is that their system needs a good automated semantic parser. In addition, the limitations of semantic parser such as the slow speed, instability when parsing large amounts of data with long sentences and ungrammatical sentences also effect in the accuracy of this approach.

2.3.2 Answer selection

After extracting the answer candidates in the previous step, the goal of answer selection is to find the most likely correct answer. This task requires high precision

[3]The figure is adapted from (Pizzato et al., 2008)

because people believe that the QA systems, which have no answer, are better than those that provide the incorrect answers (Brill et al., 2002).

Ko et al. proposed the probabilistic graphical model for joint answer ranking (Ko et al., 2007). In their work, they used joint prediction model to estimate the correct answers. Ko et al. exploited the relationship of all candidate answers by estimating the joint probabilities of all answers instead of just the probability of an individual answer. The advantage of their approach is that joint prediction model supports probabilistic inference. However, joint prediction model requires high time complexity to calculate the joint probabilities than calculating the individual probabilities.

Ittycheriah et al. used training corpus with labeled name entities to extract the answer patterns (Ittycheriah and Roukos, 2001). They then used the answer patterns to determine the correct answers. The weight of the features extracted from training corpus was based on maximum entropy algorithm. The answer candidate that has the highest probability is chosen as the answer. Although this approach achieves an improved accuracy in TREC-11, it has some disadvantages:

- It is expensive to prepare the training corpus with labeled name entities.
- It requires an automatic name entities recognizer to label the training corpus.

2.3.3 Summary

In answer processing task, passage retrieval is the most important component because it builds a subset of document collection for generating the correct answers. Although information retrieval returns a set of relevant documents, the top-rank documents probably do not contain the answer to the question. This is because document contains a lot of information and it is not a proper unit to rank with respect to the goal of QA. In passage retrieval stage, information extraction is used to ex-

tract a set of potential answers. Therefore, many approaches explored techniques to improve the precision of information extraction. In the previous approaches, soft matching based on dependency path together with the use of semantic analysis achieves promising performance. However, these approaches are highly dependent on the performance of the semantic parser, and thus the limitations of semantic parser such as the slow speed, instability when parsing large amounts of data with long sentences and ungrammatical sentences effect in the accuracy of these approaches. More specifically, these approaches will face many challenges when used to perform QA on blog or forum documents. Therefore, improving semantic parsers to work well with blog or forum language is essential to improve the performance of the overall QA systems.

Table 2.1 summaries the approaches used in two main tasks of traditional QA to seek the correct answers. Since the requirements related to process natural language in two tasks are similarity, almost all potential approaches can be applied in both question processing module and answer processing module. In these above approaches, past research found that semantic analysis gives high accuracy and applying semantic analysis seems to be a suitable choice for developing the next generation of QA systems.

Methods	Tasks
Rules based	Question processing, Answer processing
Graph based	Question processing, Answer processing
Statistical Model	Question processing, Answer processing
Sequence patterns	Question processing, Answer processing
Query representation	Question processing, Answer processing
Syntactic analysis	Question processing, Answer processing
Semantic and Syntactic analysis	Question processing, Answer processing

Table 2.1: Summary methods using in traditional QA system

Chapter 3

Community Question Answering Systems

Before 2007 the documents used in the TREC QA tracks are collected from newswires and thus the corpus is very "clean". However, in 2007 the data released by TREC was different. Instead of releasing the "clean" data, TREC included a blog data corpus for question answering (Dang et al., 2007). The blog data corpus is the mixture of "clean" and "noisy" text. In fact, real-life data is inherently noisy because people were less careful and formal when writing in spontaneous media such as the blogs or forums. The occurrence of noisy text moved question answering to more realistic settings. In addition, blogs or forums are the place where people present their personal ideas so they can write everything with their styles. There are no restrictions in blogs and forums about the grammar, and presentation styles, etc. In contrast, technical reports or newspapers are more homogenous in styles.

Moreover, unlike traditional QA systems that focus on generating factoid answers by extracting them from a fixed document collection, cQA systems reuse answers for a question that is semantically similar to user's question in community forums to generate the answers. Thus, the goal of finding answers from the enor-

mous data collections in traditional QA is replaced by finding semantically similar questions in online forums; and then use their answers to answer user's questions. This is because community forum contains large archives of question-answer pairs, although they have been posed in different threads. Therefore, if cQA can find questions similar to user's questions, it can reuse the answers of similar questions to answer user's questions. In this way, cQA systems can exploit human knowledge in user generated contents stored in online forums to provide the answers and thus reduce the time spent in searching for answers in huge document collections.

The popular type of questions in cQA is the "how" type questions because people usually use online forums to discuss and find solutions to their problems occurring in their daily life. To help other people understand their problems, they usually submit a long question to explain what problems they faced. They then expect to obtain a long answer with more discussion about their problems. Therefore, answer in cQA requires a summarization from many knowledge domains than providing simple information in a single document. In contrast, in traditional QA, people often ask simple question and expect to receive a simple answer with concise information. Another key difference between traditional QA and cQA is the relationships between questions and answers. In cQA, there are two relationships between question and answer: (a) one question has multiple answers; and (b) multiple questions refer to the same answer. The reason why multiple questions have the same answer is because in many cases, different people have the same problem in their life, but they pose them in different ways and submit to different threads in the forums.

In traditional QA, the systems perform the fixed steps of: question classification → question formulation → passage retrieval → answer selection; to generate the correct answers. On the other hand, cQA systems aim to find similar questions, and use their answers already submitted to answer user's questions. Thus, the key

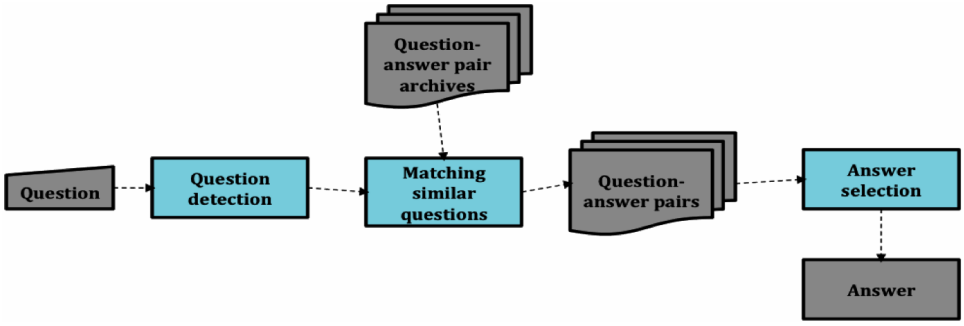

Figure 3.1: General architecture of community QA system

challenge in finding similar questions in cQA is how to measure the semantic similarity between questions posed on different structures and styles because current semantic analysis techniques may not handle ungrammatical constructs in forum language well.

Research in cQA has just started in recent years and there are not many techniques developed for cQA. To the best of our knowledge, the recent methods that have the best performance on cQA are based on statistical models (Xue et al., 2008) and syntactic tree matching (Wang et al., 2009). In particular, there is no research on applying semantic analysis to finding similar questions in cQA.

3.1 Finding similar questions

cQA systems try to detect the question-answer pairs in the forums instead of generating a correct answer. Figure 3.1 illustrates the architecture of cQA system with three main components:

- Question detection: In community forums, questions are typically relatively long that include the title and the subject fields. While the title may contain only one or two words, the subject is usually a long sentence. The goal of this task is to detect the main information asked in the thread.

- Matching similar question: This is the key step in finding similar questions. The goal of this task is in checking whether two questions are semantically similar or not.

- Answer selection: In community forum, the relationship between questions and answers is complicated. One question may have multiple answers, and multiple questions may refer to the same answer. The goal of this task is to select answers in the cQA question-answer archives after the user's question has been analyzed.

3.1.1 Question detection

The objective of question detection is to identify the main topic of the questions. One of the key challenges in forums is that the language used is often badly-formed and ungrammatical, and questions posed by user may be complex and contain lots of variations. Users always write all information in their question because they hope that the readers can understand their problems clearly. However, they do not separate which part is the main question, and which part is the verbose information. Therefore, question detection is a basic step to recognize the main topic of the question. However, this is not easy. Simple rule based methods such as question mark and 5W1H question word are not enough to recognize the questions in forum data. For example, the statistics in (Cong et al., 2008) show that 30% of questions do not end with question mark and 9% of questions end with question mark are not real question in forum data.

Shrestha and McKeown presented an approach to detect the question in email conversations by using supervised rule induction (Shrestha and McKeown, 2004). Using the transcribed SWITCHBOARD corpus annotated with DAMSL tags[1], they extracted the training examples. By using information about the class

[1]From the Johns Hopkins University LVCSR Summer Workshop 1997, available from

and feature values, they learned their rules for question detection. Their approach achieves an F1-score of 82% when tested on 300 questions in interrogative form from ACM corpus. However, the disadvantage of this approach is the inherent limitation of the rules learned. With the small rule set learned, the declarative phrases that used to detect question in test data may be missed. Therefore, question detection cannot work well in many cases.

Cong et al. proposed the classification-based technique using sequential patterns automatically (Cong et al., 2008). From both question and non-question sentences in forum data collection, they extracted the sequential patterns as the features to detect the question. An example describing the label sequential patterns (LSPs) developed in (Cong et al., 2008) is given below. For the sentence: "i want to buy an office software and wonder which software company is best", the sequential pattern "wonder which...is" would be a good pattern to characterize the question. As compared to the rule-based methods such as question mark, 5W1H question, and the previous approach (Shrestha and McKeown, 2004), the LSPs approach obtains the highest F1-score of 97.5% when testing on their dataset.

3.1.2 Matching similar question

The key challenge here is in matching user's question and the question-answer pairs in the archives of the forum site. The matching problem is challenging not only for the cQA systems but also for traditional QA systems. The simple approach of matching word by word is not satisfactory because two sentences may be semantically similar but they may not share any common words. For example[2], "Is downloading movies illegal?" and "Can I share a copy of DVD online?" have the same meaning but most lexical words used in the questions are different. Therefore, word matching cannot handle such problem. Another challenge arises because of

http://www.colorado.edu/ling/jurafsky/ws97/

[2]The data is adapted from (Jeon et al., 2005)

the language used in the forums. In traditional QA, the document collections were presented with grammatically correct writing in clear style. Hence, at least the semantic parsers can be applied to analyze the semantic information. In cQA, forum languages may contain many grammatical errors and thus the semantic parsers need to be able to handle grammatical errors when applying to cQA.

Many different types of approaches have been developed to tackle the challenge of word mismatch. One of these approaches use knowledge databases based on machine readable dictionaries (MRDs) (Burke et al., 1997). This approach uses shallow lexical semantics from WordNet to represent the knowledge of the sentences and then recognizes the similarity and matching between these sentences. Burke et al. believed that using semantic representation has many advantages such as providing the critical semantic relations between words, and requires less complexity to compute relations (Burke et al., 1997). However, the results of the experiments are not satisfactory because they did not tackle the problem of forum language characteristics when applying semantic analysis.

Another approach developed by Sneiders used template that cover the conceptual model of the database (Sneiders, 2002). A question template contains entity slots representing the main concepts or main entities in the concept model and describes the relationship between the concepts in the sentence. When the concepts were filled by the instance data in the database, the question templates become an original question. For example, "When does <performer> perform in <place>?" is a question template where <performer> and <place> are the entity slots. Figure 3.2[3] shows the relationship between the concepts in the example question.

The original question when the slots were filled with data instances is "When does Depeche Mode perform in Globen?". This approach can be described in three steps:

[3]The figure is adapted from (Sneiders, 2002)

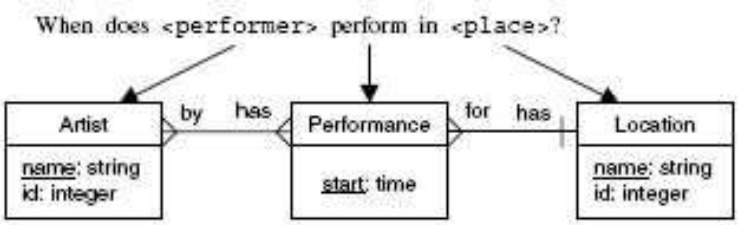

Figure 3.2: Question template bound to a piece of a conceptual model

(1) retrieve the relevant instances in user's question from database;

(2) retrieve the relevant question templates from the relevant instances that match with user's question; and

(3) retrieve the instances from the data that match with question templates and fill the instance data to create the raw data questions as a natural language form.

The advantage of this approach is that it does not require sophisticated processing for user's question. Finding similar questions becomes executing the query from database. However, the cost for processing question templates is high and it is hard to scale to large document collections with a wide variety of topics.

Berger et al. suggested the use of statistical techniques developed in information retrieval and natural language processing (Berger et al., 2000). They compared five statistical techniques to answer-finding for user's question in data collection from Ben & Jerry's customer support. Five statistical techniques were presented in Figure 3.3[4]

Believing that statistics is the most promising approach, (Jeon et al., 2005; Xue et al., 2008) tackled the word mismatch problem using word-to-word translation probabilities. There is one main difference between the models used in these approaches. While Jeon et al. used IBM translation model 1 (Jeon et al., 2005),

[4]The figure is adapted from (Berger et al., 2000)

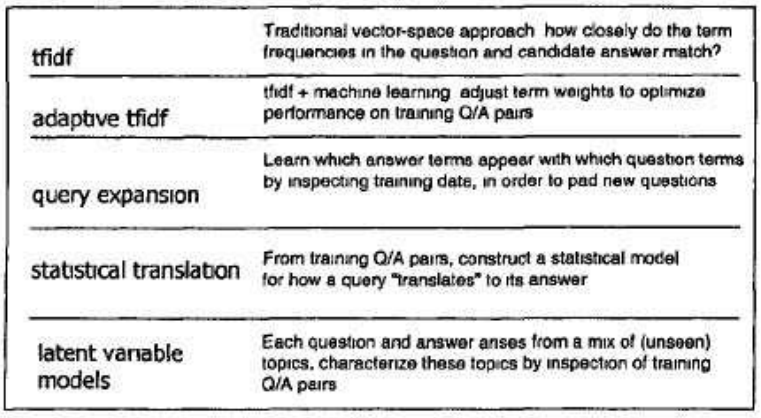

Figure 3.3: Five statistical techniques used in Berger's experiments

Xue et al. developed a mixed model of both query likelihood language model and IBM model (Xue et al., 2008). In addition, Xue et al. suggested the solution to learn good word-to-word translation probabilities based on question-answer pairs. Their experiments in Wondir[5] collection, which consists roughly of 1 million question-answer pairs, show that the combination of translation based language model and the query likelihood language model has outperformed the baseline methods such as IBM model and query likelihood language model.

The advantages of statistical approach are that the techniques used are simple and the accuracy is high at over 48% (Xue et al., 2008). However, such approaches have the limitation on the size of the training sets. Because such system builds the translation model based on statistic and thus needs a large training corpus. Unfortunately, there are no large scale collections available. The task of collecting large training corpus is difficult and expensive.

To integrate linguistic knowledge in matching similar questions, Wang et al. proposed an approach that seeks the similar question based on syntactic trees (Wang et al., 2009). In this approach, they suggested a new weighting scheme to make the similarity measure faithful and robust. Instead of using a full syntactic

[5]http://www.wondir.com

tree as an input for tree kernel, they fragment the full tree into sets of small trees and measuring the similarity score based on the sets of small trees. Furthermore, Wang employed a fuzzy matching method to incorporate semantic features. Applying syntactic tree matching in their experiments, they obtained high performance at 88.56% in data collected from Yahoo! Answer in Heathcare domain over a 10-month period from 15/02/2008 to 20/12/2008.

3.1.3 Answer selection

Answer selection aims to find answers in cQA question-answer archives after the user's question has been analyzed. There are some differences between the document collections in tradition QA and forums QA. Firstly, while document collections in traditional QA are separated documents, in forums QA, multiple questions and answers may be discussed in parallel or interwoven. Secondly, there are many kinds of relationship between question and answer such as one question has multiple answers, and multiple questions refer to the same answers.

One of the previous approaches (Huang et al., 2007) to finding answer adopts the traditional document retrieval approach where candidate answers were assumed to be isolated documents. By applying the ranking methods such as cosine similarity, query likelihood language model, KL-divergence language model, etc., the system retrieves the relevant answers from all candidate answers. However, this approach does not consider the characteristics of forums QA such as the relationships on the distances between the answers and questions posted in the same threads.

To exploit the features of forums QA, Cong et al. developed an unsupervised graph-based approach to detect answers (Cong et al., 2008). They modeled the relationship between answers as a graph based on three features: probabilities assigned by language model between one candidate answer and another candidate answers, the distance between the candidate answer and question, and the authority

of users who post the answer in the forums. Figure 3.4[6] presents the example of graph built from the candidate answers. Using the graph, they calculated the score for each candidate answer. After that, they used the ranking method to select the relevant answers.

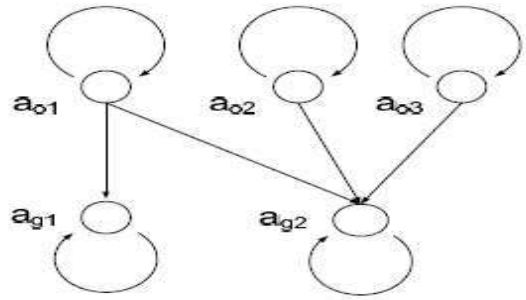

Figure 3.4: Example of graph built from the candidate answers

The advantages of Cong's approach are that: (a) it exploits the inter-relationship of all candidate answers to estimate the ranking scores; and (b) the graph-based method is complementary with supervised methods for knowledge extraction when training data is available. The main disadvantage of the system is that it requires high time complexity to build the graphs.

Liu et al. presented an approach to predict the satisfactory of answers named "Asker Satisfaction Prediction" (Liu et al., 2008). They used standard classification framework to classify whether the question asker is satisfied with the answers obtained. They used six features such as question, question-answer relationship, asker user history, answerer user history, category features, and textual features to learn the classifier. This approach exploits the power of machine learning to select the best candidate answers. However, this approach requires a training set with satisfactory judgment between the questions and answers.

[6]The figure is adapted from (Cong et al., 2008)

3.2 Summary

In cQA, the main task in answering users' questions is in finding similar questions in large archives of question-answer pairs. In particular, matching similar question is the most important step in finding similar questions that determines the performance of a cQA system. Research in utilizing the semantically similar question achieves promising performances with many approaches such as using knowledge based on MRDs, WordNet, template, statistic and especially syntactic tree matching. The high accuracy in finding similarity question based on syntactic tree shows that linguistic knowledge is really useful for working in the field related with natural language processing. Table 3.1 summaries the proposed approaches in community QA systems.

Methods	Tasks
Rules/knowledge based	Question matching, Answer selection
Graph based	Question matching, Answer selection
Statistical Model	Question matching
Syntactic tree matching	Question matching

Table 3.1: Summary of methods used in community QA systems

Although many promising approaches have been proposed in building cQA systems, there is no research on applying semantic analysis to finding semantically similar questions in cQA. This is because cQA system needs to circumvent at least two challenges: (a) handle forum language that is not well-formed; and (b) deal with discourse structures that are more informal and less reliable in forum language. Applying semantic analysis in real-life cQA systems requires semantic parsers that can handle these two challenges.

Chapter 4

Semantic Parser - Semantic Role Labeling

Semantic parsing is an important task to understand the meaning of the sentence and has been applied in many deep NLP applications, such as the information extraction, and question answering tasks. To tackle the difficulty of deep semantic parsing, previous works only focus on shallow semantic parsing which is known as Semantic Role Labeling (SRL). For each predicate in a sentence, the main goal of SRL is to recognize all the constituents of the target predicate and map these constituents into corresponding semantic roles. From 2002, many techniques based on the syntactic parser tree were applied to develop SRL systems. Generally, SRL uses syntactic parser tree as input and determines the semantic labels for arguments through two steps:

- Identify the boundaries of the arguments in the sentence given by the predicate.
- Classify the arguments into the specific semantic role.

In recent years, following the success of Proposition Bank[1] (PropBank) project and NomBank[2] project, a large set of annotated corpora in English for tasks such as POS, Chunker, and SRL were released. With the availability of these corpora, many machine learning techniques such as Support Vector Machine (SVM) (Mitsumori et al., 2005; Pradhan et al., 2004), Maximum Entropy (ME) (Liu et al., 2005), joint model (Haghighi et al., 2005), AdaBoost (Màrquez et al., 2005), etc. driven by data have been applied to SRL. Although the syntactic parser tree was added to PropBank data, all recently systems do not exploit effectively all the features from syntactic parsing for an SRL system. In these systems, the features only used to build the structure of the syntactic tree and position of the constituent following its predicate. In addition, the syntactic parser is so sensitive to small changes in the sentence that two sentences with one different word also have very different syntactic parser trees. Thus, the performance of these semantic parsers varies greatly as they depend heavily on the accuracy of the automatic syntactic parser.

4.1 Analysis of related work

From 2002, SRL has become one of the main focuses of many NLP conferences. Many SRL systems were developed to tackle this task. However, the lack of annotated corpora has limited the range of techniques applied to SRL. The first SRL system developed by Gildea et al. in 2002 builds a statistic model based on FrameNet (Gildea and Jurafsky, 2002). The features proposed in this system became the basic features for many SRL systems in recent years. The next generation of SRL system was developed by adding some features exploited from the PropBank corpora (Pradhan et al., 2004; Surdeanu et al., 2003). Most basic features used in

[1]http://www.ldc.upenn.edu/Catalog/CatalogEntry.jsp?catalogId=LDC2004T14
[2]http://nlp.cs.nyu.edu/meyers/NomBank.html

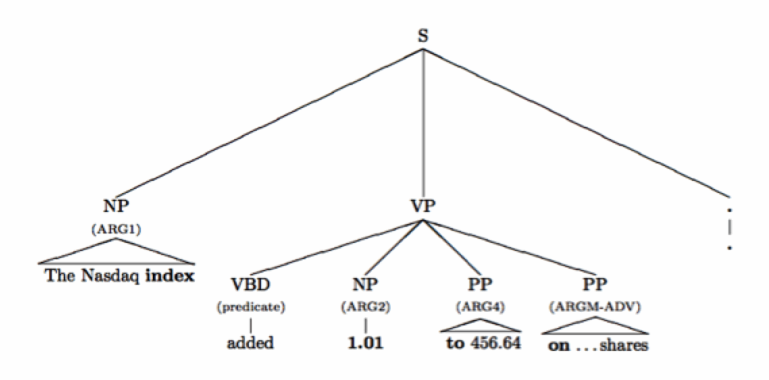

Figure 4.1: Example of semantic labeled parser tree

prior research in (Gildea and Jurafsky, 2002; Pradhan et al., 2004; Surdeanu et al., 2003) can be categorized in three types: sentence level features, argument-specific features, and argument-predicate relational features. Table 4.1 illustrates the basic features in SRL systems. Figure 4.1 is an example of the semantically labeled parse tree and Table 4.2 illustrates the basic features for NP (1.01) on Figure 4.1.

Features	Description
Sentence level features	
Predicate (Pr)	Predicate lemma in the predicate-argument structure
Voice (Vo)	Grammatical voice of the predicate, either active or passive
Subcategorization (Sc)	Grammar rule that expands the predicate's parent node in the parse tree
Argument-specific features	
Phrase type (Pt)	Syntactic category of the argument constituent
Head word (Hw)	Head word of the argument constituent
Argument-predicate relation features	
Path (Pa)	Syntactic path through the parse tree from the argument constituent to the predicate node
Position (Po)	Relative position of the argument constituent with respect to the predicate node, either left or right

Table 4.1: Basic features in current SRL system

Recently, with the availability of large human annotated corpora, many machine learning techniques were applied and have obtained better results. Various

Type of features	Value
Pr	add
Vo	Active
Sc	VP:VBD_NP_PP_PP
Pt	NP
Hw	1.01
Pa	NP↑VP_VBD
Po	Right

Table 4.2: Basic features for NP (1.01)

learning algorithms, labeling strategies, and feature design were submitted to Conference on Computational Natural Language Learning (CoNLL) 2005. In the nineteen systems participated in CoNLL 2005, most systems employed machine learning algorithms such as the Decision tree, Support vector machine, Log-linear models, AdaBoost, TBL, CRFs, IBL, etc. (Carreras and Màrquez, 2005). In addition, past research has found that SRL system with full syntactic parse gives higher accuracy than SRL system with shallow syntactic parse.

Pradhan et al. presented an approach using SVM to argument classification for semantic parsing (Pradhan et al., 2004; Pradhan et al., 2005). Beside the basic flat features exploited from syntactic parser tree introduced in (Gildea and Jurafsky, 2002) such as predicate, phrase type, parser tree path, position, verb, voice, head word, and verb sub-categorization, they added 12 new features including named entities in constituents, head word POS, verb clustering, partial path, verb sense information, head word of preposition phrases, first and last word/POS in constituent, ordinal constituent position, constituent tree distance, and constituent relative features to improve the SRL system. Figure 4.2 illustrates the effect of each feature on the two main tasks of SRL system: argument identification and argument classification, when added to the baseline features.

Although many features were used and feature based method represents

	Argument Classification	Argument Identification		
Features	*A*	*P*	*R*	F_1
Baseline	87.9	93.7	88.9	91.3
+ Named entities	**88.1**	93.3	88.9	91.0
+ Head POS	***88.6**	94.4	90.1	***92.2**
+ Verb cluster	**88.1**	94.1	89.0	**91.5**
+ Partial path	**88.2**	93.3	88.9	91.1
+ Verb sense	**88.1**	93.7	89.5	**91.5**
+ Noun head PP (only POS)	***88.6**	94.4	90.0	***92.2**
+ Noun head PP (only head)	***89.8**	94.0	89.4	**91.7**
+ Noun head PP (both)	***89.9**	94.7	90.5	***92.6**
+ First word in constituent	***89.0**	94.4	91.1	***92.7**
+ Last word in constituent	***89.4**	93.8	89.4	**91.6**
+ First POS in constituent	**88.4**	94.4	90.6	***92.5**
+ Last POS in constituent	**88.3**	93.6	89.1	91.3
+ Ordinal const. pos. concat.	87.7	93.7	89.2	**91.4**
+ Const. tree distance	**88.0**	93.7	89.5	**91.5**
+ Parent constituent	**87.9**	94.2	90.2	***92.2**
+ Parent head	85.8	94.2	90.5	***92.3**
+ Parent head POS	***88.5**	94.3	90.3	***92.3**
+ Right sibling constituent	**87.9**	94.0	89.9	**91.9**
+ Right sibling head	**87.9**	94.4	89.9	***92.1**
+ Right sibling head POS	**88.1**	94.1	89.9	**92.0**
+ Left sibling constituent	***88.6**	93.6	89.6	**91.6**
+ Left sibling head	86.9	93.9	86.1	89.9
+ Left sibling head POS	***88.8**	93.5	89.3	**91.4**
+ Temporal cue words	***88.6**	–	–	–
+ Dynamic class context	**88.4**	–	–	—

Figure 4.2: Effect of each feature on the argument classification task and argument identification task, when added to the baseline system

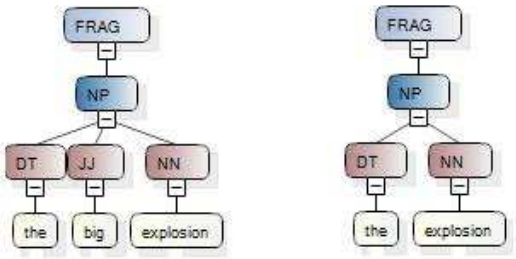

Figure 4.3: Syntactic trees of two noun phrases "the big explosion" and "the explosion"

the state-of-the-art for SRL, the key limitation of syntactic parser still exist. In particular, the syntactic parser tree is so sensitive to small changes in input sentence that the SRL systems often fail to detect a general pattern to label the semantic roles. For instance, two simple noun phrases "the big explosion" (NP → DT JJ NN) and "the explosion" (NP → DT NN) have two different syntactic parser trees; and thus the general pattern for these two syntactic trees cannot be extracted. Figure 4.3 presents the two different trees for this instance.

In SRL system, the problem of shallow semantic parsing can be viewed as a sequence of processing steps: identify and classify the semantic arguments of the predicate. In particular, there are two types from classification task separated by the unit level.

- Constituent-by-Constituent (C-by-C) classification approach: the candidate chunks are provided by the full syntactic parse of a sentence, thus, the classification is done on constituents.

- Word-by-Word (W-by-W) classification approach: the classification is done at the word-level. Hence, each word in a sentence is labeled with a tag.

Using the same features extracted from full syntactic parser, the experiment in (Pradhan et al., 2005) shown that the performance obtained by the W-by-W paradigm is lower than that obtained by the C-by-C paradigm. Table 4.3 illustrates

the comparison of C-by-C and W-by-W classifiers which presented in (Pradhan et al., 2005).

System	Precision	Recall	F1
C-by-C	80.6	67.1	73.2
W-by-W	70.7	60.5	65.2

Table 4.3: Comparison of C-by-C and W-by-W classifiers

All systems participated in CoNLL 2005 achieved the high accuracy when determining the argument structure of verb predicates (Carreras and Màrquez, 2005). However, these SRL systems cannot annotate the argument structures of noun predicates in NomBank data. Jiang and Ng presented a NomBank SRL system using Maximum Entropy to label semantic roles (Jiang and Ng, 2006). They applied techniques used in building PropBank SRL system to develop NomBank SRL systems. In addition, Jiang et al. also proposed new features to improve the accuracies of NomBank SRL system. Their paper presented the first result of statistical SRL system in NomBank data.

Alternative to feature based method; kernel methods (Collins and Duffy, 2001) were used to circumvent the limitation of syntactic tree. Instead of using the features extracted from syntactic tree, kernel methods measure the similarity between two syntactic structures. More and more kernels were used such as Tree Kernel (Moschitti, 2004), String Subsequence Kernel (Kate and Mooney, 2006), and Graph Kernel (Suzuki et al., 2003).

Moschitti presented an approach using SVM tree kernel to label semantic roles (Moschitti, 2004). Instead of exploiting the flat features, they selected portions of syntactic tree including predicate/argument sub-structures as the features for SVM. The classifier calculates the similarity score between two tree-structures based on the similarity score between their sub-structures. By using the sub-tree

structure, the small difference may not affect the kernel classifier. For example, when predicate in the sentence used in a simple past tense was replaced by predicate used in a simple present tense, the tree structure has a small difference but the sub-structures are the same. Therefore, their system outperforms systems that used flat features (Gildea and Jurafsky, 2002; Pradhan et al., 2004; Surdeanu et al., 2003). However, their approach only performs the hard matching between the sub-structures without considering the linguistic knowledge. Hence, their approach fails when handling two similar phrases as presented in Figure 4.3.

Zhang et al. proposed a system named grammar-driven convolution Tree Kernel (Zhang et al., 2007) to tackle the limitation of (Moschitti, 2004). Using the rules extracted from the training corpus, they built a set of reduced rules to construct the sets of syntactic tree and create new sub-trees. The number of the new sub-trees is larger than the original so the kernel method has more objects to measure the similarity between them. This approach obtained an effective results in corpus released in CoNLL 2005. However, as with most previous SRL systems, this approach did not exploit the features of dependency between words in the sentence.

In the previous research, almost all systems assumed that each candidate constituent is independent in the classification task. This also means that each candidate has a local score from the classification process. Unfortunately, the past research found that the SRL system that captures the interdependence among all arguments of a predicate gives the best overall accuracy for semantic argument classification (Pradhan et al., 2005). Jiang et al. proposed the use of the neighboring semantic arguments of a predicate as the semantic context features to classify the current semantic argument (Jiang et al., 2005). In addition, they integrated with the assumption that semantic arguments are processed in the linear ordering in the sentence to improve the accuracy of their system.

On the other hand, to tackle the drawback of the assumption that each

candidate constituent is independent, Haghighi et al. proposed using a joint model based on the global scoring and the non-overlapping constraint (Haghighi et al., 2005). After local scoring from classification process, the non-overlapping constraint was run, and the n-best candidates were generated and arranged as a sequence. From this sequence, a set of features, including all the features used at the local level and sequence-based features, was extracted and combined in a log-linear model to re-rank the n-best list.

4.2 Corpora

In recent years, there are two large human annotated corpora available for semantic role labeling task: FrameNet and PropBank. However, there are some differences between these two corpora. First, FrameNet annotates the predicate argument based on frame elements while PropBank annotate the argument structure of verbs. Second, while FrameNet annotates in predicate-specific roles, PropBank annotates in predicate-independent roles. Lastly, FrameNet focuses on the semantic considerations during annotation, while PropBank prefers to maintain the consistency with their syntactic alternations. Table 4.4 and Table 4.5 illustrate the instances of sentences annotated in FrameNet and PropBank.

[$_{Theme}$ The swarm] [$_{Predicate}$ went] [$_{Direction}$ away] [$_{Goal}$ to the end of the hall].

Table 4.4: Example sentence annotated in FrameNet

[$_{A0}$ He and two colleagues] [$_{Predicate}$ went] [$_{A1}$ on an overnight fishing trip].

Table 4.5: Example sentence annotated in PropBank

	Train.	Devel.	tWSJ	tBrown
Sentences	39,832	1,346	2,416	426
Tokens	950,028	32,853	56,684	7,159
Propositions	90,750	3,248	5,267	804
Verbs	3,101	860	982	351
Arguments	239,858	8,346	14,077	2,177
A0	61,440	2,081	3,563	566
A1	84,917	2,994	4,927	676
A2	19,926	673	1,110	147
A3	3,389	114	173	12
A4	2,703	65	102	15
A5	68	2	5	0
AA	14	1	0	0
AM	7	0	0	0
AM-ADV	8,210	279	506	143
AM-CAU	1,208	45	73	8
AM-DIR	1,144	36	85	53
AM-DIS	4,890	202	320	22
AM-EXT	628	28	32	5
AM-LOC	5,907	194	363	85
AM-MNR	6,358	242	344	110
AM-MOD	9,181	317	551	91
AM-NEG	3,225	104	230	50
AM-PNC	2,289	81	115	17
AM-PRD	66	3	5	1
AM-REC	14	0	2	0
AM-TMP	16,346	601	1,087	112
R-A0	4,112	146	224	25
R-A1	2,349	83	156	21
R-A2	291	5	16	0
R-A3	28	0	1	0
R-A4	7	0	1	0
R-AA	2	0	0	0
R-AM-ADV	5	0	2	0
R-AM-CAU	41	3	4	2
R-AM-DIR	1	0	0	0
R-AM-EXT	4	1	1	0
R-AM-LOC	214	9	21	4
R-AM-MNR	143	6	6	2
R-AM-PNC	12	0	0	0
R-AM-TMP	719	31	52	10

Figure 4.4: Semantic roles statistic in CoNLL 2005 dataset

In the CoNLL shared tasks 2004, 2005, the standard dataset built on PropBank with alternative format was released. In the CoNLL 2005, data are presented in table format and all the discontinuous and co-referential arguments are annotated. There are thirty-five semantic roles classified into three clusters: core arguments, adjuncts, and references (Carreras and Màrquez, 2005). The summary of semantic roles in the data released in ConNLL 2005 is presented in Figure 4.4.

4.3 Summary

Semantic role labeling is an important task to understand the meaning of the sentence and it has been applied to many deep NLP applications, such as the information extraction, question answering, etc. Research in SRL has started from 2002 and up to now had achieved promising performances. The best SRL system achieved about 79% accuracy in CoNLL 2005 (Carreras and Màrquez, 2005). In particular, Zhang's approach (Zhang et al., 2007) obtains high performance of over 91% on semantic role classification. However, in CoNLL 2005, information about dependency-based representation for syntactic and semantic dependencies is missing in the data released and thus the observation that uses the richer set of syntactic dependencies to improve SRL may be missing too. Integrating syntactic dependencies to SRL system has been started from CoNLL 2007 (Johansson and Nugues, 2007). Although dependency relations provide more invariant structures to improve SRL systems, they tend to be efficient only for short sentences and incur errors on long distance relations. Therefore, some challenges such as the dependencies derived from name entity structures, long-distance grammatical relations, etc. are called for SRL systems in CoNLL 2008. SRL systems using syntactic dependencies model are more complex than the ones used in the previous CoNLL share task (Surdeanu et al., 2008).

Recently, most SRL systems trained and tested in PropBank data collected from Wall Street Journal. Unfortunately, there is always a gap between data collected from newswire and data collected from forum. Hence, when SRL systems are applied to cQA, they face many challenges such as the need to handle forum language that is not well-formed, and discourse structures that are more informal and less reliable in forum language, etc.

Chapter 5

System Architecture

In this chapter, we present the architecture of our semantic parser system named GReSeA for **G**rammatical **Re**lations for **Se**mantic **A**nalyzer. First, we describe the overall architecture of GReSeA. Second, we analyze our observations in grammatical relations and describe how to apply them in GReSeA. Last, we describe the descriptions on key components before presenting our experiments to evaluate the accuracy of GReSeA.

5.1 Overall architecture

Before CoNLL 2008, the task of identification and disambiguation of semantic predicates is omitted in developing SRL system because information about predicates was integrated in the data released. From CoNLL 2008 shared task, the target predicates are not predefined, thus, identifying predicates became one of the main tasks. Determining predicate is a very important task because many current semantic parsers such as ASSERT (Pradhan et al., 2004) are not able to recognize support verb constructions. For example, ASSERT cannot recognize the verb frame "go" in sentence "I go to play football". Hence, missing predicates means many

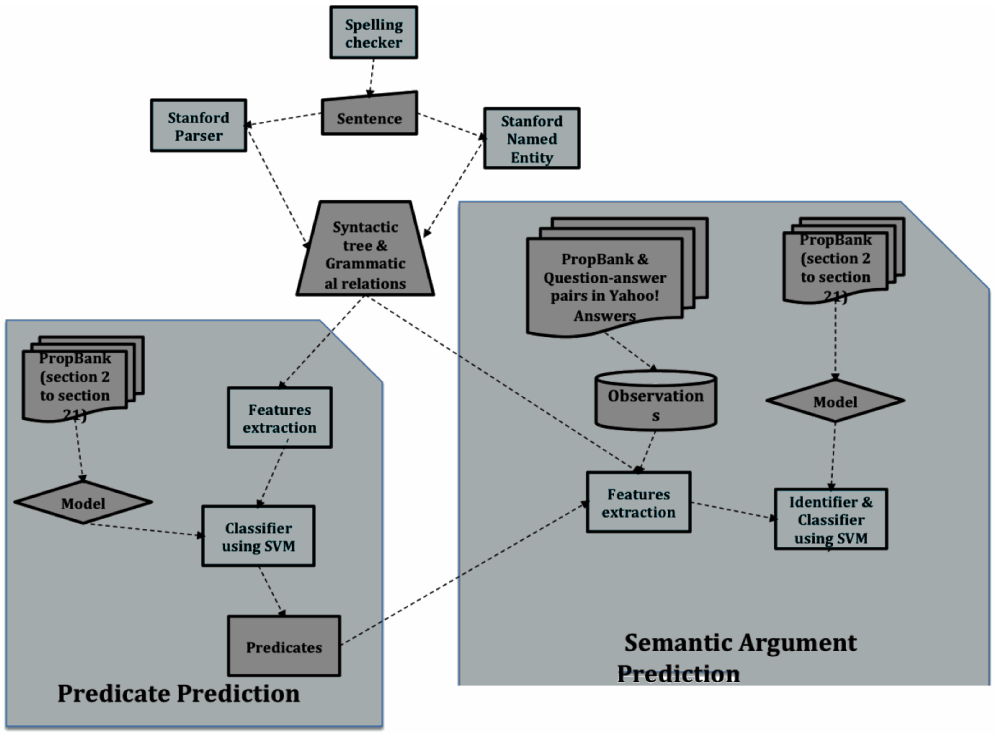

Figure 5.1: GReSeA architecture

predicate-argument structures will be missed. Many approaches were presented to identify predicate in CoNLL 2008, including Markov Logic Networks (Riedel and Meza-Ruiz, 2008), Maximum Entropy classifier (Sun et al., 2008), multiclass average Perceptron classifier (Ciaramita et al., 2008), etc. In GReSeA, we separate the SRL task into two stages, illustrated in Figure 5.1. The stages are predicate prediction and semantic argument prediction.

Initially, GReSeA receives a sentence as the input and then, Stanford Parser and Stanford Name Entity Recognition are run in the sentence. The output of preprocessing stage is a sentence consisting part-of-speech analysis, noun/verb phrase chunking, full syntactic parse, name entities, and grammatical relations. Grammatical relations are otherwise known as the dependency relations, which provide a simple description of the grammatical relationships between words in a sentence. Grammatical relations are also useful for people without linguistic expertise who

want to extract textual relations. Each grammatical relation is a binary relation that holds between a governor and a dependent. All grammatical relations used in our systems are defined in (de Marneffe and Manning, 2008). The following stages then process the sentence based on these above features.

In the predicate prediction stage, GReSeA selects some features from full syntactic parsing, which is useful to recognize the predicate candidates in the sentence. We treat the task of recognizing the predicate as the binary classification. Each token in the sentence consists of the same number of features, which are used to determine whether the token is a predicate. To train the model for binary classification, from the CoNLL 2005 data sets, we use data from Section 2 to Section 21. At the end of this stage, we have a list of tokens that has been determined as the predicates of the events described in the sentence.

The second stage is to predict the semantic arguments of the predicate. Each predicate, which recognized from the previous "predicate prediction" stage, goes through the same processes to annotate its semantic arguments. First, GReSeA extracts the features such as name entities, syntactic tree, and grammatical relations before we optimize the above features. From the analysis based on grammatical relations in data collected from CoNLL 2005 data sets and Yahoo! Answer, we derive some observations that help GReSeA optimizes the grammatical relations between the token and other tokens in the sentence; and thus, GReSeA can recognize more candidate arguments. Second, we process in two subtasks including argument identification and argument classification.

In semantic argument prediction module, instead of using the whole input sentence to classify the semantic argument, we generate a dependency tree from parse tree and use the dependency tree for classifying the semantic argument. The key idea of GReSeA starts from the assumption that in a sentence, we can remove or reduce some modified words but the grammar structure and the semantic roles

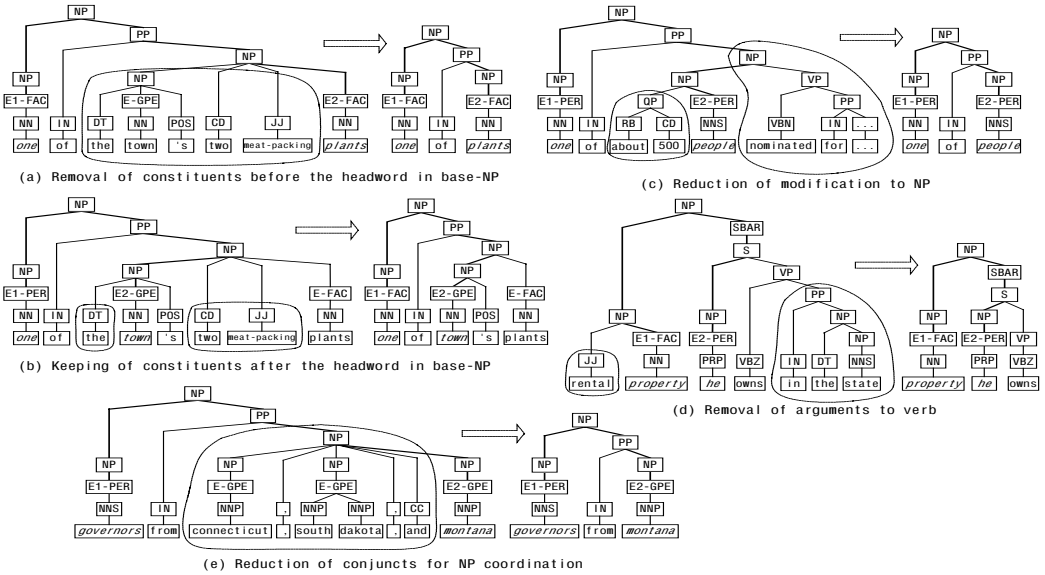

Figure 5.2: Removal and reduction of constituents using dependency relations

of the sentence remain the same (Qian et al., 2008). For instance, from the noun phrase "one of about 500 people nominated for ...", we can reduce the modification such as "500", "nominated", etc. The new noun phrase is "one of people", which has the same semantic role in a sentence. Figure 5.2[1] illustrates the instances of above assumption.

According to (Johansson and Nugues, 2008), dependency syntax has received less attention in developing the SRL system, despite the fact that dependency structures offer a more transparent encoding of predicate-argument relations. Furthermore, in terms of performance, SRL systems based on dependencies were generally found to be much worse than their constituent-based counterparts. However, the past research found that grammatical function information, which is available in grammatical relations, is more resilient to lexical problems caused by the change of domain and grammatical errors. In addition, the dependency-based SRL system is biased in finding argument heads, rather than argument text snippets. Thus, the advantages and drawbacks of SRL system will depend on the applications - for

[1]This Figure is adapted from (Qian et al., 2008)

instance, application required template-filling might need complete segments, while other applications require semantic information as vector space representation such as text categorization, similarity matching, or a reasoning application, might prefer to use the heads only.

In semantic argument classification, only some words have contributed information. The remaining words of a sentence just modify and contribute less information in the classification process. However, the occurrences of the modifier words seem to be the cause that changes the detailed structure of the syntactic tree. In GReSeA, we use grammatical relations to present the general view about the semantic role associated with the selected predicate. It means that instead of classifying all words in a sentence as W-by-W, we only select and classify some headwords in a sentence. Thus, we reduce a lot of time to process all the words in a sentence, in particular, with a long sentence.

Similar to the predicate prediction stage, we also use the same data Sections from CoNLL 2005 data sets to train our classification model. However, there is a difference between the model of argument identification and argument classification. In argument identification, we treat this task as binary classification to recognize each token in the sentence as whether or not belong to any argument. In contrast, in argument classification subtask, instead of classifying all tokens in the sentence, we select some potential tokens to classify. In addition, we do not use binary classification such as true and false in argument classification. Since GReSeA annotates a subset of 24 labels, which reduced from 35 standard labels in CoNLL 2005 data sets, we build a “One vs All” formalism, which involves training n binary classifiers for an n-class problem.

We integrate spelling checker process to correct the popular spelling errors in the sentence collected from forums. In addition, we also correct some popular abbreviations used in forum such as “4” and “for”, “g9” and “good night”, etc.

5.2 Observations based on grammatical relations

To develop GReSeA adapted to forum language, we collect and analyze the grammatical relations extracted from two diverse resources: CoNLL 2005 data sets and questions in Yahoo! Answers. We choose these two resources because we want to analyze the grammatical errors occur in both newswire (Wall Street Journal) and forums (Yahoo! Answers). We use Stanford Parser to parse 500 sentences, which is randomly selected from the two resources. The average length of these sentences is 15 words. We then manually analyze the grammatical relations between words in these sentences. Through the analysis of instances, we have derived three observations. Using these observations and our analysis, we derive rules to disambiguate syntactic ambiguity by optimizing the relations between words in the sentence to recognize the role of arguments such as subject, direct object, indirect object, preposition object, preposition object of time, and preposition object of location.

5.2.1 Observation 1

To make the complex sentence more concise, people usually reduce some elements in the sentence. For instance, for two simple sentences "PhD students research a new problem" and "PhD students publish papers about their research", there is a parallel structure to make a complex sentence such as "PhD students research a new problem and publish papers about their research". In this case, the subject of the verb "publish" was reduced and of course, the grammatical relation between "PhD students" and "publish" is also ignored. The absence of this relation is the reason why it is hard to recognize the subject for the verb "publish". However, in terms of meaning, both the verbs "research" and "publish" have the same subject. We call these two verbs the adjacent verbs. We give the definition for the adjacent

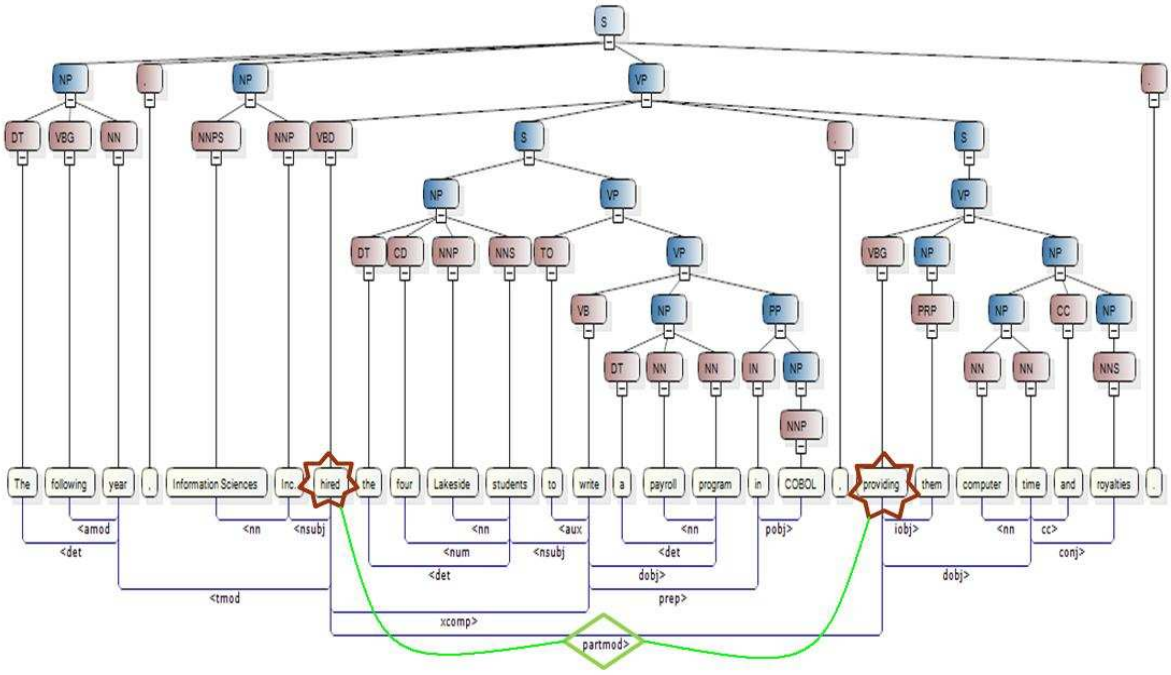

Figure 5.3: The relation of pair adjacent verbs (hired, providing)

verbs as follow.

- Two verbs are adjacent verbs if they have some relationships such as clausal complement with internal/external subject, participial modifier, and conjunction. For example, in the sentence "The following year, Information Sciences Inc. hired the four Lakeside students to write a payroll program in COBOL, providing them computer time and royalties.", the two verbs "hire" and "provide" have relationship participial modifier and thus they are adjacent verbs. Figure 5.3 illustrates the syntactic tree for the pair of adjacent verbs (hired, providing) in the above sentence.

- Two verbs are adjacent verbs if they do not have any relationship but both are the nodes in the sub clause of the syntactic tree. Figure 5.4 illustrates the syntactic tree for the pair of adjacent verbs (faces, explore) appearing in the sentence "The 1.4 billion robot spacecraft faces a six-year journey to explore

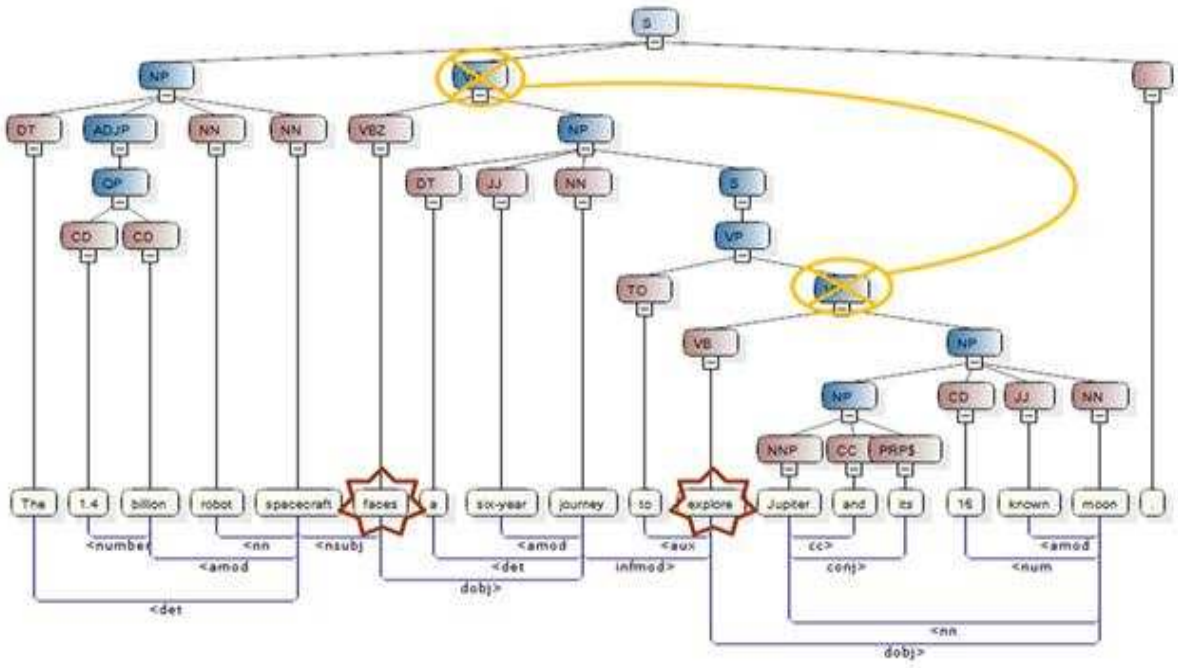

Figure 5.4: The relation of pair adjacent verbs (faces, explore)

Jupiter and its 16 known moon."

Based on this observation, we build the rules to optimize the grammatical relations for each verb. As a result, we can rewrite a complex sentence by a group of simple sentences in which one sentence has only one verb with the simplest structure of S-V-O.

5.2.2 Observation 2

In the sentence, which has many continuous preposition phrases, the result of a syntactic parser, in particular, Stanford parser usually has only one main preposition phrase connecting to the predicate. The other preposition phrases are connected to their previous preposition phrase. In our statistics, we recognize that all continuous preposition phrases are usually connected to the predicate except the preposition phrase that starts with the preposition "of". Therefore, we optimize the relation-

ship to ensure that all preposition phrases, except the preposition phrase that starts with "of", are connected to the predicate. The following examples illustrate this observation.

Example 1: "[In Tokyo] [on Monday], the U.S. currency opened for trading at 141.95 yen."

In this example, we detect only one preposition phrase "In Tokyo" has grammatical relation with predicate "opened" and the preposition phrase "on Monday" depend on the phrase "In Tokyo". Applying our rules, the relationship between "on Monday" and "In Tokyo" was transformed to the new relationship between "on Monday" and predicate "opened".

Example 2: "Loius Pasteur was born [on December 27 1822] [in Dole] [in the Jura region of France]."

In this example, our rules create the new relationship for the preposition phrases "in Dole" and "in the Jura region" with predicate "born". However, no relationship was created for preposition phrase "of France" and predicate "born". The role of preposition phrase "of France" is a modifier for the preposition phrase "in the Jura region".

5.2.3 Observation 3

Sentence with the verb "be" is very popular in forums. However, in CoNLL 2005 data sets, the annotation semantic for verb "be" is ignored. In grammatical relation, "be" is an auxiliary verb creating a direct connection between the arguments. The arguments can be a noun, a noun phrase, an adjective, or an adjective phrase. Based on the observation, we build a rule to predict the subject and object for the predicate "be".

Example: "[Theresa E. Randle] [is] [an American stage , film and television actress]."

Grammatical relations = {nsubj(actress-12, Randle-3), cop(actress-12, is-4)}

In this example, GReSeA recognize the relationship between two object "Theresa E. Randle" and "an American stage, film and television actress" by applying the rules in an auxiliary verb "is".

5.2.4 Summary

In this section, we discuss the observations that are useful for improving the accuracy of GReSeA and increasing the adaptation of GReSeA in forum language. We believe that these observations are useful because they create more effective information. For instance, "Born in 1937 in a Baltic Sea town now part of Poland, he was eight years old when World War II ended. ", the subject of verb "born" is reduced. However, using our observations, GReSeA can tackle this problem. First, we apply the third observation and thus we recognize the verb "was". Second, we apply the first observation and recognize "born" and "was" are adjacent verbs. We then create the new relation subject between two verbs "born" and "he". Therefore, GReSeA recognizes "he" to be the subject argument for the verb "born". In comparing with the semantic roles labeled by ASSERT, while GReSeA recognize two predicate-argument structures for the two verb "born" and "was", ASSERT cannot recognize any of them.

5.3 Predicate prediction

To understand the semantic meaning in the sentence under natural language, the most important thing that artificial intelligent (AI) systems require is recognizing the action that is described in the sentence. Based on the action, AI systems find the related semantic arguments. Over the last year, predicates are always provided for all systems worked in semantic parser. However, from 2008, predicate prediction

became an important task to evaluate the performance of SRL systems.

Although in most sentences in natural language, predicates usually have part-of-speech (POS) verb, there are some exceptions. Sometimes, predicates in the sentence can have another POS such as noun, adjective, etc. Table 5.1 illustrates the statistic for POS of predicates from the Section 23 of CoNLL 2005 data sets.

POS	VB*	NN*	Others	Total
Frequency	89105	709	713	90527
%	98.43	0.78	0.79	

Table 5.1: POS statistics of predicates in Section 23 of CoNLL 2005 data sets

From the statistics, we found that almost all predicates in the sentence are started with POS "VB" (VB*[2]). However, more than 1.5% of predicates in the Section 23 of CoNLL 2005 data sets are started with other POS. Obviously, the challenge for current SRL system is to recognize the remaining predicates that are not started with the POS "VB".

In GReSeA, in addition to recognizing the predicates starting with POS "VB", we focus on recognizing the remaining predicates starting with POS "NN" (NN*[3]). It means that GReSeA omits the predicates starting with other POS (Others[4]) such as "JJ", "IN", "RB", etc.

One of the simplest approaches in predicate prediction is to use heuristic rules. For instance, if a token in the sentence is started with POS "VB", this token is determined to be a predicate. Although this approach is very simple, based on the statistic in Table 5.1, we found that the accuracy obtained is over 95%.

To tackle the challenge of recognizing predicates started with POS other than "VB", we proposed using support vector machine. Firstly, we extract some

[2]POS starts with VB such as VB, VBN, VBG, VBP, VBD, VBZ
[3]POS starts with N such as NN, NNS, NNP
[4]POS starts with JJ, IN, MD, RB, CD, JJR, FW, RP

features from the syntactic tree for each token. We divide these features in two types: basic features and additional features. The details are described in Table 5.2.

Features	Description
Basic features	
Word	Token in the sentence
Name entity	Name entity of token
POS	POS of token
Lemma word	Word lemma
IsLemmaPreviousWordEqual“Be“	Is word lemma of previous word equaled “be”, either true or false
IsPOSPreviousWordStarted“VB“	Is POS of previous word started “VB”, either true or false
IsPOSNextWordStarted“VB“	Is POS of next word started “VB”, either true or false
Additional Features	
IsFirstCharacterUppercase	Is first character of token written as uppercase, either true or false
IsGovernorOfDependency	Is token stayed at governor position in a relationship between two tokens, either true or false Example: “faces” is a governor in relation nsubj(faces, spacecraft)
IsFirstCharacterPreviousWordUppercase	Is first character of previous token written as uppercase, either true or false
IsPreviousWordEqualArticle	Is previous word article “a”, “an”, “the”, either true or false
IsFirstCharacterNextWordUppercase	Is first character of next token written as uppercase, either true or false

Table 5.2: Features for predicate prediction

To classify the labels of the predicates, we use TinySVM along with YamCha, a toolkit which is mainly used for Support Vector Machine (SVM) based chunker, as the SVM training and testing software. To build up the training model, we use 20 sections, from Section 2 to Section 21, in the data sets released in CoNLL 2005. This is the standard data developed for particular purpose of evaluating SRL systems. Unfortunately, in this data, predicates of verb “be” are omitted. Hence, in GReSeA, we divide the prediction process into two small steps: (1) we use machine learning to classify the predicates in a sentence. All 3322 predicates annotated in CoNLL 2005 data sets are predicted by SVM model. (2) Based on observation 3,

GReSeA recognizes the potential predicates of verb "be". Then, we combine the two lists of predicates predicted from steps (1) and (2) with following constraints:

- predicate of verb "be" must have at least two arguments including subject and object arguments.
- predicate of verb "be" must be the main verb in the simple sentence or clause.

At the end of the predicate prediction stage, we have a list of predicates in a sentence. Based on these predicates, we extract features for the next stage of semantic argument prediction.

5.4 Semantic argument prediction

5.4.1 Selected headword classification

As we analyze in Section 4.1, most of the SRL systems are highly dependent on the full syntactic tree. Unfortunately, the full syntactic tree is sensitive to the changes in a sentence. Hence, in GReSeA, we study how to create a concise and effective tree for a relation instance by exploiting grammatical relations between word and word.

In GReSeA, instead of using the full syntactic tree as the resources to extract the features, we generate a dependency tree from the grammatical relations in parse tree. We point out that since the information based on the grammatical relations between word and word is directly encoded in the argument structure of lexical units in the sentence, it is useful to localize the semantic role associated with the selected predicate. By selecting the headword of each grammatical relation, we recognize which words in a sentence have contribution in the semantic argument detection. Then, we reduce the remaining words that have less effect in the classification. Figure 5.5 illustrates the full dependency tree restructured from

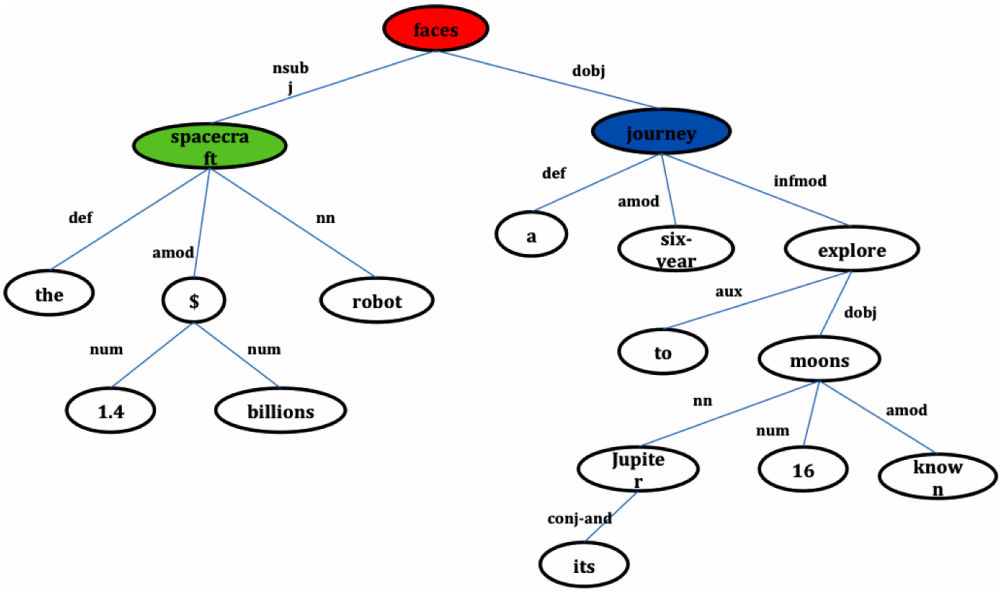

Figure 5.5: Example of full dependency tree

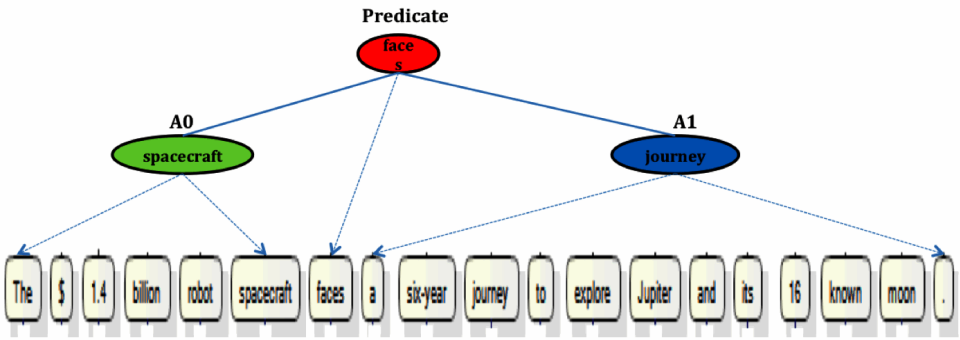

Figure 5.6: Example of reduced dependency tree

the grammatical relations for the instance "The $1.4 billions robot spacecraft faces a six-year journey to explore Jupiter and its 16 known moons.". Figure 5.6 shows the reduced dependency tree for the classification semantic argument associated with the selected predicate "faces".

After selecting the headword for the classification process, we extract the features for each selected word. We divide the features for classification into 4 types: (A) represent information about syntactic tree; (B) represent information about the grammatical relations; (C) represent the semantic meaning of word; and

(D) represent other information such as name entity, lemma form, and noun of preposition phrase. The details of these features are described in Table 5.3.

Features	Description
(A)	
(1) Word	Selected headword
(2) POS	Part-of-speech of selected headword
(4) Phrase type	Syntactic category of selected headword
(5) Maxtree	Biggest tree in syntactic tree contains headword and non-overlap with maxtree of other headwords
(6) Position	Relative position of selected headword with predicate, either before or after
(B)	
(8) Relation	Grammatical relations between headword and predicate
(11) IsOptimize	Is relation optimized by applying the observations or not
(C)	
(10) SemanticPObj	Semantic meaning of noun of preposition phrase
(D)	
(3) Ner	Name entity of selected headword
(7) Lemma predicate	Predicate lemma based on wordnet
(9) PObj	Noun of preposition phrase

Table 5.3: Features for headword classification

To classify the labels of the selected headwords, we use the same toolkit as the predicate prediction stage, TinySVM along with YamCha, as the SVM training and testing software. Each selected headword is classified into one of the 24 semantic roles such as A0, A1, etc. We have used Yamcha in the "One vs. All" method with all default parameters. We use 20 sections, from Section 2 to Section 21 in the data sets released in CoNLL 2005, to build up the training model. The features, which use for training and testing, correspond to the template of YamCha toolkit. For instance, features of selected headword of the sentence: "The $1.4 billions robot spacecraft faces a six-year journey to explore Jupiter and its 16 known moons." that are associated with the predicate "faces" are presented in Figure 5.7.

(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)	(11)	*Label*
spacecraft	NN	O	NP	NP	-1	face	nsubj	-	-	0	***A0***
faces	VBZ	-	VP	-	0	face	verb	-	-	0	***V***
journey	NN	O	NP	NP	1	face	dobj	-	-	0	***A1***
spacecraft	NN	O	NP	NP	-1	explore	nsubj	-	-	1	***A0***
explore	VB	-	VP	-	0	explore	verb	-	-	0	***V***
Jupiter	NNP	O	NP	NP	1	explore	dobj	-	-	0	***A1***

Figure 5.7: Features extracted for headword classification

In our system, we assume that one selected headword is delegated for one argument in a sentence. Therefore, we use the results of selected headword classification step as the results of argument classification.

5.4.2 Argument identification

To employ a SRL system with complete segments, we include the subtask of argument identification in GReSeA. We implement two algorithms to recognize the argument boundary: greedy search algorithm and machine learning using SVM.

5.4.2.1 Greedy search algorithm

Greedy search algorithm is the simplest implementation for argument identification. Based on the selected headword and syntactic tree, we search the suitable phrase for each headword. Furthermore, we apply the non-overlapping constraint that all argument boundaries are not overlap. Pseudo code of this algorithm is described in Table 5.4.

Figure 5.8 illustrates the Greedy search algorithm used for identifying the argument boundaries of the three headwords spacecraft, faces, and journey. Starting from the leaf of the syntactic tree, the headword journey searches bottom up until it reaches the noun phrase (step 1 and 2). However, when the headword journey reaches the verb phrase (step 3), it overlaps with the argument of the headword

```
Step 1:
for ( headword hwi in headword list listHw) {
maxtree[i] = phrase type of hwi
}
Step 2: search bottom up
do {
for ( headword hwi in headword list listHw) {
flag = true
for (j:0 → listHw.size & flag) {
if (i = j & maxtree[i] → parent.contain(maxtree[j])) {
flag = false
}
}
if (flag) {
maxtree[i] = maxtree[i] → parent
}
}
} until (no chance in maxtree)
return maxtree
```

Table 5.4: Greedy search algorithm

faces. We then conclude that the argument boundary of headword journey is the noun phrase "a six-year ... known moons".

5.4.2.2 Machine learning using SVM

In the recent SRL systems, from syntactic parsing, all of phrases related to the selected predicate were extracted based on the assumption that each phrase in the syntactic tree may be an argument. In contrast, GReSeA uses all words in the sentence as in W-by-W classification approach. Hence for each word, we extract the set of features. Beside the basic features introduced in (Gildea and Jurafsky, 2002), including predicate (1), voice (2), verb sub-categorization (3), phrase type (4), headword (5), path (7), and position (8), we add some additional features that

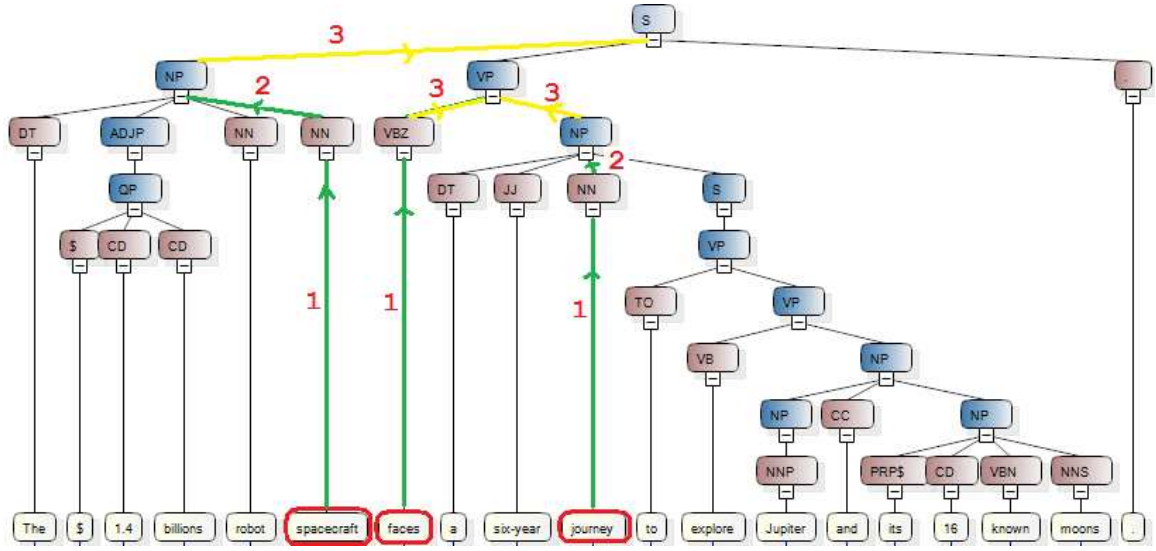

Figure 5.8: Example of Greedy search algorithm

are found to give significant improvement in argument detection in (Pradhan et al., 2004), including: headword POS (6), noun head of prepositional phrase (9), first word in constituent (10), first word POS in constituent (11), and parent phrase type (12) in GReSeA. Figure 5.9 gives the features extracted for argument prediction of sentence "The $1.4 billions robot spacecraft faces a six-year journey to explore Jupiter and its 16 known moons." that are associated with predicate "faces".

After the argument identification step, we have the boundary of all arguments in a sentence. Combine this with the results of the selected headword classification, we are able to recognize semantic role of the argument by using the constraints that: the argument has the same semantic role as the selected headword delegated for this argument.

In our experiments, we use TinySVM along with YamCha for SVM training and testing. The parameters of SVM kernel are the same as in the previous stages: predicate prediction, and headword classification. Similar to the predicate prediction stage, we use binary classification for each word in a sentence. Each word is classified into one of three categories: (1) B: the word is starting an argument; (2)

(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)	(11)	(12)	*Label*
face	1	VP:VBZ_NP	NP	spacecraft	NN	NP|S|VP|VBZ	-1	-	The	DT	S	*B*
face	1	VP:VBZ_NP	QP	billions	NNS	QP|ADJP|NP|S|VP|VBZ	-1	-	1.4	CD	ADJP	*I*
face	1	VP:VBZ_NP	QP	billions	NNS	QP|ADJP|NP|S|VP|VBZ	-1	-	1.4	CD	ADJP	*I*
face	1	VP:VBZ_NP	NP	spacecraft	NN	NP|S|VP|VBZ	-1	-	The	DT	S	*I*
face	1	VP:VBZ_NP	NP	spacecraft	NN	NP|S|VP|VBZ	-1	-	The	DT	S	*I*
face	1	VP:VBZ_NP	VP	faces	VBZ	-	0	-	faces	VBZ	S	*B*
face	1	VP:VBZ_NP	NP	journey	NN	NP|VP|VBZ	1	-	a	DT	VP	*B*
face	1	VP:VBZ_NP	NP	journey	NN	NP|VP|VBZ	1	-	a	DT	VP	*I*
face	1	VP:VBZ_NP	NP	journey	NN	NP|VP|VBZ	1	-	a	DT	VP	*I*
face	1	VP:VBZ_NP	VP	explore	VB	VP|S|NP|VP|VBZ	1	-	to	TO	S	*I*
face	1	VP:VBZ_NP	VP	explore	VB	VP|VP|S|NP|VP|VBZ	1	-	explore	VB	VP	*I*
face	1	VP:VBZ_NP	NP	Jupiter	NNP	NP|NP|VP|VP|S|NP|VP|VBZ	1	-	Jupiter	NNP	NP	*I*
face	1	VP:VBZ_NP	NP	Jupiter	NNP	NP|VP|VP|S|NP|VP|VBZ	1	-	Jupiter	NNP	VP	*I*
face	1	VP:VBZ_NP	NP	moons.	NNS	NP|NP|VP|VP|S|NP|VP|VBZ	1	-	its	PRP$	NP	*I*
face	1	VP:VBZ_NP	NP	moons.	NNS	NP|NP|VP|VP|S|NP|VP|VBZ	1	-	its	PRP$	NP	*I*
face	1	VP:VBZ_NP	NP	moons.	NNS	NP|NP|VP|VP|S|NP|VP|VBZ	1	-	its	PRP$	NP	*I*
face	1	VP:VBZ_NP	NP	moons.	NNS	NP|NP|VP|VP|S|NP|VP|VBZ	1	-	its	PRP$	NP	*I*

Figure 5.9: Features extracted for argument prediction

I: the word is belonging to an argument; and (3) O: the word is not belonging to an argument.

Similar to two previous stages, we use 20 sections, from Section 2 to Section 21 in the data sets released in CoNLL 2005, to build up the training model. All 35 semantic labels in the training corpus are replaced by the labels B, I and O for training classification model.

5.5 Experiment results

5.5.1 Experiment setup

In this section, first, we evaluate GReSeA in two stages: predicates prediction and arguments prediction. Second, we define the GReSeA baseline named GReSeA^b, which uses the same features introduced in Section 5.4.1 without applying the observations introduced in Section 5.2 to optimize the grammatical relations. We then compare GReSeA^b with GReSeA to evaluate the effects of the observations on SRL system. Lastly, we evaluate the robustness of GReSeA in handling ungrammatical

sentences.

Data sets: To evaluate the efficiency of the proposed approach, we use CoNLL 2005 data sets extracted from the PropBank corpus with 35 semantic labels classified into four clusters: core arguments, adjuncts, references, and verbs (Carreras and Màrquez, 2005). Since PropBank is one of the largest annotated corpus which serves many deep linguistic projects, the annotated tags selected in CoNLL 2005 data sets are diverse enough to serve a variety of needs. In our research, we develop an SRL system which is robust against ungrammatical sentences when apply to find the similar questions in cQA, and thus we focus on developing GReSeA to annotate only a smaller number of semantic labels that are useful to cQA task. Out of the 35 semantic labels, GReSeA uses a subset of 24 semantic labels, including: A0, A1, A2, A3, A4, AM-LOC, AM-TMP, AM-MNR, AM-CAU, AM-DIS, AM-NEG, AM-PNC, AM-ADV, R-A0, R-A1, R-A2, R-A3, R-A4, R-AM-TMP, R-AM-LOC, R-AM-MNR, R-AM-CAU, R-AM-PNC, R-AM-ADV due to the following two reasons. First, throughout the analysis, we found that some arguments are rarely used both in SRL and QA systems. For example, the number of arguments such as AA, AM, AM-REC, R-AA, R-AM-DIR, R-AM-EXT, etc. appearing in CoNLL 2005 data sets is very small[5]. Second, we recognize that some arguments are not meaningful in finding similar questions in cQA such as AM-MOD, AM-REC, etc. because two semantically similar questions may not share any common words. With this subset of semantic labels, GReSeA can achieve the following advantages:

- By reducing the semantic labels that rarely occur in the training corpus, we can reduce the noisy samples. Thus our model based on SVM will have less error during classification.

- By reducing the unused semantic labels, GReSeA is able to focus on the main arguments. Thus it helps to improve the quality of semantic information

[5]See more detail in 4.4

annotation and the accuracies of finding similar questions, and also reduce the processing time.

Similar to the recent SRL systems, we use 20 sections from Section 2 to Section 21 in CoNLL 2005 data sets for training and use Section 23 for testing. The semantic labels that did not use for training and testing are removed. The tasks to be evaluated are: predicate prediction, and argument prediction. We use the predicates provided in the CoNLL 2005 data sets for testing.

The different annotation: When we compare the result of our system GReSeA with the ground truth in CoNLL 2005, there are major differences in the annotation produced by GReSeA. In the data released in CoNLL 2005, chunks or phrases are considered as constituent arguments; they were annotated with all member words. In contrast, GReSeA focuses on annotating arguments based on headword selected from dependency relations. For instance, in the sentence "Born in 1937 in a Baltic Sea town now part of Poland, he was eight years old when World War II ended.", there are differences arising from recognizing the argument boundaries for the phrase "in 1937 in a Baltic Sea town now part of Poland". The difference is shown in Table 5.5.

Data released in CoNLL 2005	Results of GReSeA
[$_{TARGET}$ Born] [$_{AM-LOC}$ **in 1937 in a Baltic Sea town now part of Poland**], [$_{A1}$ he] was eight years old when World War II ended.	[$_{TARGET}$ Born] [$_{AM-TMP}$ **in 1937**] [$_{AM-LOC}$ **in a Baltic Sea town**] now part of Poland, [$_{A1}$ he] was eight years old when World War II ended.

Table 5.5: Comparison GReSeA results and data released in CoNLL 2005

Thus when we compare the results of GReSeA and data released in CoNLL 2005 in constituent-based system, there are two differences: (1) the argument temporal "in 1937", and (2) the argument location "in a Baltic sea town". Unfortunately, although GReSeA recognizes the argument location for phrase "in a Baltic

sea town", the boundary of this argument is different, hence, the result of argument segmentation is different. In contrast, when we compare GReSeA output and data released in CoNLL 2005 in selected headword, GReSeA has only one difference of excess recognized argument temporal "in 1937". These differences lead to superior performance for GReSeA which will be verified when we apply it to find similar questions in the cQA corpus in Chapter 6.

Second, in the data released in CoNLL 2005, the predicate verb "be" is omitted. For example, in the sentence illustrated above, the predicate-argument structure for verb "was" is missing. Therefore, we do not evaluate the accuracy of labeling for predicate verb "be". All the predicate-argument structures annotated in CoNLL 2005 for this sentence are:

(1) [$_{TARGET}$ Born] [$_{AM-LOC}$ in 1937 in a Baltic Sea town now part of Poland], [$_{A1}$ he] was eight years old when World War II ended.

(2) Born in 1937 in a Baltic Sea town now part of Poland, he was eight years old [$_{R-AM-TMP}$ when] [$_{A1}$ World War II] [$_{TARGET}$ ended].

5.5.2 Evaluation of predicate prediction

Information on predicate-argument structures in the data sets used in CoNLL 2005 was extracted from the PropBank corpus. It means that all the predicates annotated were the verbal predicates. Thus, there are no significant differences between the predicted accuracy of the system for predicates starting with POS "VB" and those starting with POS other than "VB". In this experiment, we evaluate the accuracy of two approaches, named heuristic and SVM, based on three metrics: precision, recall, and F1. The details of the experiment results in the Section 23 in CoNLL 2005 data sets with 2416 sentences and 5267 predicates are given in Table 5.6.

Using heuristic approach, GReSeA recognizes almost all the predicates that

	# of predicate	# of predicate predicted	# of predicate predicted correct	Precision	Recall	F1
Heuristic	5267	5183	5002	96.51	94.97	95.73
SVM	5267	5325	5098	95.74	96.79	96.26

Table 5.6: Accuracy of predicate prediction

have POS tag as "VB*" (95.73% vs. 98.43%[6]). The gap of 2.6% between the F1 accuracy and data statistics is caused by the use of automatic parser in GReSeA where some POS tags were not correctly recognized.

Recall that the heuristic approach cannot predict the predicates starting with POS tag other than "VB", so we introduce the method using SVM. Unfortunately, the data statistics show that the number of predicates started with POS tag "NN" is very small only 0.78%. Hence, the accuracy of GReSeA when using SVM to recognize the predicates including "NN*" and "VB*" is only slightly higher by 0.53%. However, we believe with other data sets such as the CoNLL 2008, where propositions were addressed around both verbal and nominal predicates, the accuracy of GReSeA using SVM should improve more significantly over the heuristic approach.

5.5.3 Evaluation of semantic argument prediction

To evaluate the accuracy of SRL systems, we use the accuracy of argument prediction which combines two steps: argument identification and argument classification. Specifically, we will compare our constituent-based system with similar SRL systems using the SVM approach: (1) Mitsumori: individual SRL system using W-by-W classification (Mitsumori et al., 2005); and (2) ASSERT: combined SRL system using C-by-C classification (Pradhan et al., 2004).

[6]See data statistics in Table 5.1

5.5.3.1 Evaluate the constituent-based SRL system

To evaluate the accuracy of GReSeA based on constituent, we evaluate two approaches for detecting the argument boundary where the first approach uses greedy search algorithm and the second one uses machine learning with SVM. However, there is a small difference in the ordering of processing steps. In the first approach, we select headword and classify the label before we search the boundary for the selected headword. In contrast, in the second approach, SVM was used to identify the boundary, and then, select headword for each argument to classify. We use the evaluating software released in CoNLL 2005 to calculate the accuracy of GReSeA in Section 23 of CoNLL 2005 data sets.

We compare GReSeA with two similar constituent-based SRL systems, one proposed by Mitsumori et al. (Mitsumori et al., 2005) named Mitsumori and the other by Pradhan et al. (Pradhan et al., 2004) named ASSERT. We choose these two systems because they both used the same SVM approach to address the semantic arguments. However, there is a slightly difference between them. While Mitsumori is an individual system, ASSERT is a combined system. To make the comparison fair, we report the results on 24 semantic labels. Table 5.7 shows the accuracy of the four systems, including GReSeA uses Greedy search algorithm (GReSeA Greedy), GReSeA uses machine learning (GReSeA SVM), Mitsumori, and ASSERT.

	precision	recall	F1
GReSeA Greedy	74.32	65.31	69.52
GReSeA SVM	75.00	69.86	72.34
Mitsumori	73.17	67.21	70.06
ASSERT	81.25	72.84	76.82

Table 5.7: Comparing similar constituent-based SRL systems

The first row in Table 5.7 shows the accuracy of GReSeA when using Greedy search algorithm to find the boundary of the arguments. As discussed in Section 5.5.1, there are differences in the annotation between GReSeA based on selected headword and the data released in CoNLL 2005. Moreover, the Greedy search algorithm is very simple in finding the argument boundary; thus GReSeA does not achieve good accuracy based on CoNLL 2005 test set. However, comparing with Mitsumori, the system used machine learning approach, the results of GReSeA using Greedy search algorithm are approximately the same (69.52% vs. 70.06%).

The accuracy of GReSeA when using machine learning approach for detecting argument boundary is given in the second row. As compared to the greedy search algorithm, the machine learning approach achieves an improvement of 2.82% (72.34% vs. 69.52%) in F1 measure.

Mitsumori et al. reported results for a system using the same features as in (Gildea and Jurafsky, 2002), which are also the initial set of features used in our system. Moreover, both Mitsumori and GReSeA are individual systems that use the same machine learning approach for estimating the local scores in both training and testing stages. However, GReSeA, based on grammatical relations, not only reduces the processing time, but also outperforms Mitsumori by 2.28% (72.34% vs. 70.06%) in F1 measure. Therefore, we can conclude that the features extracted from grammatical relations could achieve significant improvement among the individually SRL systems using the SVM approach.

Table 5.7, however, shows that GReSeA has lower performance than ASSERT by 4.5% in F1. In interpreting the results, we must consider two main differences in the architecture between GReSeA and ASSERT. First, while GReSeA uses W-by-W classification, ASSERT uses C-by-C classification. Second, GReSeA is an individual system, while ASSERT is a combined system. Recall that the accuracy of the system using W-by-W classification is lower than those using C-by-C

classification (Pradhan et al., 2005) and the systems using the combination are better than the individuals (Carreras and Màrquez, 2005), the lower performance of GReSeA as compared to ASSERT is to be expected.

5.5.3.2 Discussion

In this section, we evaluate the accuracy of GReSeA as compared to two SRL systems using SVM approach. Although GReSeA achieves a lower accuracy than the combined system such as ASSERT, we have analyzed the reasons that lead to the lower performance. In contrast, comparing with other individual systems such as Mitsumori, GReSeA improves the accuracy by 2.28% in F1 measure.

Basically, we develop GReSeA as a SRL system that annotating semantic arguments based on the selected headword. It is nontrivial when comparing the dependency-based SRL system and constituent-based SRL system (Johansson and Nugues, 2008). Therefore, we conducted another evaluation of semantic arguments annotated based on the selected headword. In this evaluation method, an argument is considered as: (1) correct if we pick out the correct headword of a correct argument; (2) extra if we pick out the wrong headword; and (3) missing if we do not pick out the headword of a correct argument. For instance, with the gold sentence S1 and the result S2, Table 5.8 illustrates the details of our evaluation. We do not count the selected headword with label *TARGET*.

S1: w0 $[_{A1}$ *w1 w2 s3]* $[_{TARGET}$ *add]* $[_{A2}$ *w5 w6 w7 w8]* $[_{A4}$ *w9 w10 w11]*

S2: $[_{AM-TMP}$ *w0]* $[_{A1}$ *w1] w2 w3* $[_{TARGET}$ *add] w5* $[_{A2}$ *w6] w7* $[_{AM-LOC}$ *w8] w9 w10 w11*

The results of testing in Section 23 are presented in Table 5.9. From the Table, we can see that, GReSeA could achieve a high accuracy of 78.27% in F1. However, we do not have good baseline system and gold corpus to evaluate the effectiveness of our dependency-based SRL system directly. Thus, in chapter 6 we

Word	Predicted label	Gold label	Result
w0	*	AM-TMP	extra
w1	A1	A1	correct
w6	A2	A2	correct
w8	A2	AM-LOC	extra
w9	A4	*	missing

Table 5.8: Example of evaluating dependency-based SRL system

apply our dependency-based SRL system in similar question finding task and test the effectiveness indirectly through the performance of cQA.

	precision	recall	F1
GReSeA	84.89	72.62	78.27

Table 5.9: Dependency-based SRL system performance on selected headword

5.5.4 Comparison between GReSeA and GReSeA^b

When applying three of our observations to GReSeA, each observation has a different effect. The first observation is used to improve the accuracy of core arguments. The second focuses on improving the accuracy of adjuncts arguments, including location and temporal; while the third observation is used to improve the SRL labeling for predicates from the verb "be". However, since CoNLL 2005 data sets omits the verb "be", we are not able to evaluate the effect of the third observation.

In this section, we use Section 23 in CoNLL 2005 data sets to evaluate the difference between GReSeA and GReSeA^b. We compare the results of core arguments, location arguments, and temporal arguments based on two evaluation systems: the dependency-based and the constituent-based. Table 5.10 and Table 5.11 show the results.

	precision	recall	F1
GReSeA^b	83.86	73.94	78.59
GReSeA	86.86	75.21	80.61

Table 5.10: Compare GReSeA and GReSeA^b on dependency-based SRL system in core arguments, location and temporal arguments

	precision	recall	F1
GReSeA^b	66.77	62.78	64.72
GReSeA	75.82	67.27	71.29

Table 5.11: Compare GReSeA and GReSeA^b on constituent-based SRL system in core arguments, location and temporal arguments

In these Tables, GReSeA, the system that uses the observations to optimize the grammatical relations, achieves significant improvements in performance over GReSeA^b. The GReSeA results are better than GReSeA^b in both dependency-based and constituent-based. Using the observations, GReSeA achieves the higher accuracies by a large margin over GReSeA^b by 2% and 6.57% for the dependency-based system and constituent-based system respectively. It reaffirms that the application of those observations to grammatical relations have strong positive effects in SRL systems.

5.5.5 Evaluate with ungrammatical sentences

One challenge of the current SRL systems is to handle the ungrammatical sentences. To demonstrate the robustness of GReSeA, we randomly select some sentences from Section 23 in CoNLL 2005 data sets; and then use open-source software (Foster and Andersen, 2009) to generate the ungrammatical sentences. We define 3 types of basic grammatical errors: (1) errors resulting from deleting one word such as delete article before noun, etc; (2) errors resulting from inserting one word such as

insert adjective before noun; and (3) errors resulting from substituting one word for another such as change the verb form, change the preposition, etc. The position of selected word in the sentence was picked randomly. We define a set of grammar rules to generate the ungrammatical data sets based on the description of the software. We assume that all ungrammatical sentences generated automatically have the same annotation results with the original sentences in the data sets; thus we use the gold annotated data sets to evaluate the performance. Table 5.12 illustrates an example of ungrammatical sentences generated automatically in our data sets.

Type	Content
Original	The finger-pointing has already begun.
Delete	finger-pointing has already begun.
Insert	The classified-ad finger-pointing has already begun.
Substitute	The finger-pointing has already beging.

Table 5.12: Examples of ungrammatical sentences generated in our testing data sets

The results of GReSeA and ASSERT on ungrammatical test set are reported in Table 5.13 and Figure 5.10. We evaluate the accuracy in F1 value for each data set. The delete data sets is the ungrammatical sentences generated by using the deleting rules; insert data sets and sub data sets are generated by using the inserting rules and substituting rules respectively. We randomly select 100, 500, 1000, 1500, 2000 sentences from the CoNLL 2005 test set to generate our testing data.

From the Figures, we can see that as compared to ASSERT, GReSeA achieves higher accuracy. Because our test set is generated based on CoNLL 2005 data sets that come from the Wall Street Journal, there is no significant difference between the two comparing systems. GReSeA outperforms the ASSERT by only a small margin in accuracy (0.94%). However, in the real data from forum, there are more types of grammatical errors besides the basic errors that were automatically gen-

	del		*insert*		*sub*	
# of sentences	ASSERT	GReSeA	ASSERT	GReSeA	ASSERT	GReSeA
100	67.72	70.75	68.72	70.87	64.64	67.32
500	69.87	69.71	69.94	70.39	65.39	66.20
1000	69.72	69.64	70.17	69.94	65.18	65.73
1500	69.64	70.97	70.12	71.23	65.76	67.05
2000	69.91	70.18	70.47	70.69	66.14	66.80
Avarage	69.37	70.25	69.88	70.62	65.42	66.62

Table 5.13: Evaluate F1 accuracy of GReSeA and ASSERT in ungrammatical data sets

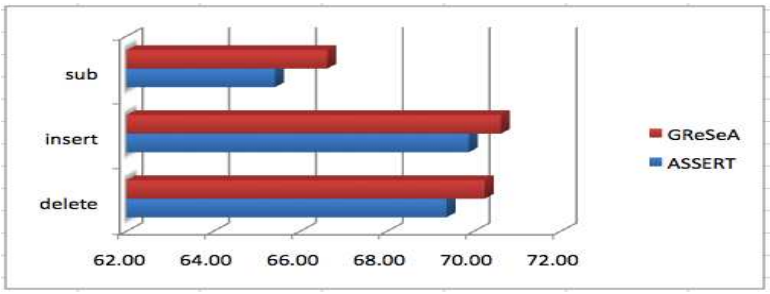

Figure 5.10: Compare the average F1 accuracy in ungrammatical data sets

erated in our test set. Table 5.14 gives the examples and the annotation results for the sentences selected from Yahoo! Answer website. In forum, the grammatical errors such as the preposition error presented in the first row ("*to healthy*"), subject-verb agreement presented in the second and third rows ("*Anyone have*", "All natural *way lose*") are very common. It is noted that GReSeA can handle these errors by using the relations between the selected headwords to annotate the semantic information. On the other hand, ASSERT fails to parse these sentences. Hence, it further demonstrates that GReSeA possesses good robustness in handling grammatical errors as compared to the current SRL systems.

GReSeA	ASSERT
[$_{A0}$ eating banana] [$_{TARGET}$ is] [$_{A1}$ good to healthy]?	Null
[$_{A0}$ Anyone] [$_{TARGET}$ have] [$_{A1}$ any idea]?	Null
[$_{A0}$ All natural way] [$_{TARGET}$ lose] [$_{A1}$ products]? PLEASE ANSWER?	Null

Table 5.14: Examples of semantic parses for ungrammatical sentences

5.6 Conclusion

In this chapter, we developed an efficient implementation of the observations and present a grammatical relations-based semantic role labeling system. By exploiting the grammatical relations, we achieved competitive results on the standardized CoNLL 2005 data sets. With less features extracted, GReSeA is better than the individual SRL systems using the same SVM approach. It achieved an improvement in F1 score of about 2.28% over (Mitsumori et al., 2005). Moreover, we reported an increase in accuracy between GReSeA and one of the best current SRL system (Pradhan et al., 2004) when testing on ungrammatical data set.

We observed that the accuracy when applying semantic analysis to finding similar questions in cQA is not determined only by the types of the annotation such as constituent-based or dependency-based systems. It means that detecting the argument boundaries cannot improve the performance. In contrast, handling the challenge of forum language such as the ungrammatical sentences is the primary problem that we need to tackle for achieving higher accuracies.

We reaffirm that using the general view about the relation between the head-word and its predicate, GReSeA is robust to not only simple grammatical errors such as the article, tense, plurality, but also to complex grammatical errors such as the preposition, subject-verb agreement.

Chapter 6

Applying semantic analysis to finding similar questions in community QA systems

Applying semantic information to traditional QA has been demonstrated to be effective in (Kaisser and Webber, 2007; Shen and Lapata, 2007; Li and Roth, 2006). Using semantic roles combined with dependency paths, questions and candidate answers are annotated with semantic arguments. Finding the correct answers thus becomes the problem of matching predicate-argument structures annotated in the question and the answer candidates. Although many works have demonstrated the increase of performance in traditional QA by applying semantic information, many problems need to be tackled when applying semantic analysis to cQA such as determining the role of verbs in sentence analysis (Klavans and Kan, 1998), ensuring the effectiveness of the semantic parser in case of grammatical errors, etc.

With the characteristics of forums language, applying semantic information is a great challenge. Although semantic parsers achieved impressive performances in recent years, these results are obtained on standard corpus collected from newswire

(Wall Street Journal, Brown). There is a big gap between data collected from newswire and data collected from forums (Dang et al., 2007); and thus, applying SRL systems to cQA applications will face many challenges such as the handling of grammatical errors. To integrate semantic information in finding similar questions in cQA, we develop a SRL system by leveraging grammatical relations that is robust to grammatical errors. Then, we utilize the similarity score between user's question, also called query, and the candidate questions to choose the relevant questions.

6.1 Overview of our approach

Using semantic parser, all arguments were addressed around the predicates. If two sentences were considered to be similar, the pair of semantic predicates in both sentences should be highly similar too. In addition, the modified information such as the arguments and their semantic roles around the two predicates should also be similar. Based on the results of semantic parser, we propose the method for measuring the similarity score between two sentences with three elements, including predicates, arguments, and semantic labels.

The architecture of the semantic relation matching is shown in Figure 6.1.

- Stage 1: we apply semantic parsing to represent all questions and query in a predicate-argument frame. In this frame, each argument includes two elements: semantic label and words.

- Stage 2: we estimate the semantic similarity score between the query and all questions. The semantic similarity score is measured in a combination of: (1) the predicate similarity score, (2) word-word similarity score, and (3) semantic labels translation probability score.

While the measurement of (1) and (2) can be derived based on the resources such as WordNet and lexical, the measurement of (3) requires the training data to

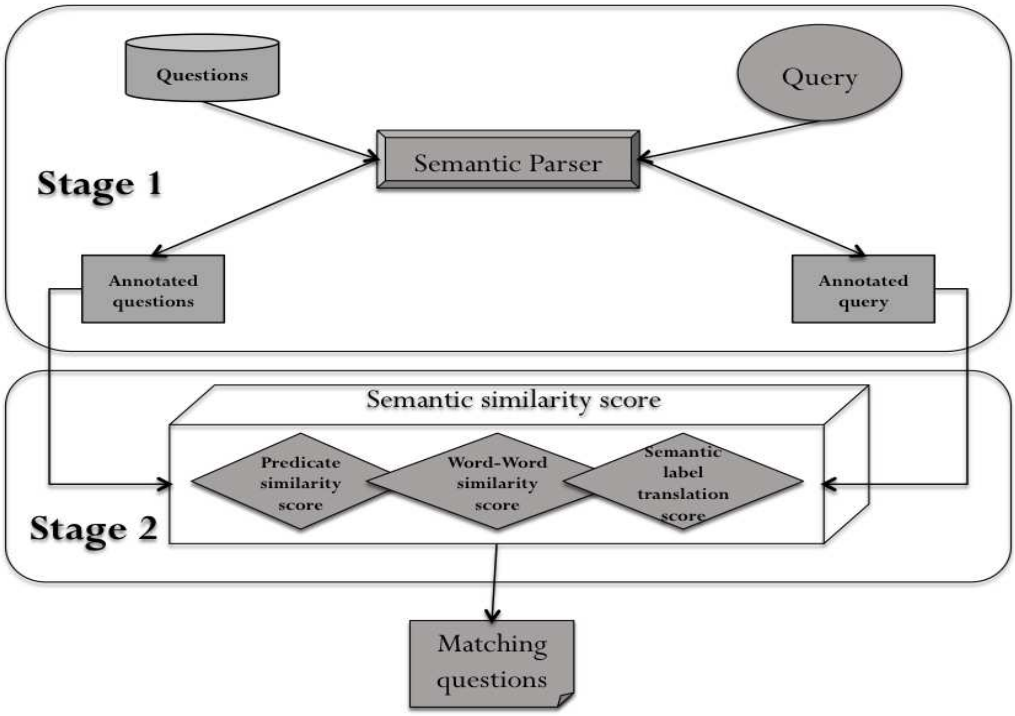

Figure 6.1: Semantic matching architecture

estimate the probabilities. For predicate similarity score, after we predict the predicate in two candidate questions, we use WordNet to expand the pair of predicate to measure their similarity. To estimate the similarity score between word-word, based on the lexical similarity, we calculate the score between pair of words in the arguments. For semantic label pair translation probability score, we iteratively train the expectation maximization (EM) method as presented in (Brown et al., 1993).

6.1.1 Apply semantic relation parsing

To capture the semantic structures contained in a sentence, we apply semantic parser to identify the predicates and their arguments. Since the semantic role of the argument has a special contribution in estimating the similarity, our system

needs to label the role for each argument in the predicate-argument structure. We define a semantic frame to represent a sentence in terms of semantic structures. Each frame includes predicate, and the set of its arguments. Each argument is associated with a semantic label such as A0, A1, etc. to indicate the semantic role. Therefore, a frame F can be represented as $F = (p, A)$, where p is the predicate and $A = (a_1, a_2, ..., a_n)$ is the set of arguments. Each argument $a_i = (l_i, w_i)$ has two elements: l_i is semantic label of argument and $w_i = (w_{i1}, w_{i2}, ..., w_{in})$ is the set of words in the argument.

In one sentence, we can have more than one predicate; hence, in the results of semantic parser, we can have more than one annotated sentence. Obviously, we can have more than one frame for each sentence. However, although one sentence can have more than one predicate, we consider that the most important predicate called "root" predicate to be more meaningful than the others. For instance, in the question "How can I resume playing video file in Youtube?", the main predicate is "resume" while the remaining predicate "playing" is just a modifier. Therefore, in our system, we have different weight for the similarity scores between the "root" predicate and the remaining predicates.

6.1.2 Measure semantic similarity score

6.1.2.1 Predicate similarity score

In natural language understanding, predicate identification is very important to understand the events described in the sentence. Normally, if two sentences are referred to the same event, they always have high semantic similarity score between the pair of predicates. In contrast, if two sentences possess the same semantic structure but the semantic relatedness of their predicates is small, then the two sentences might be different. Therefore, in our system, when matching semantic frames, we consider the semantic similarity between a pair of predicates as one of

the main element. We use WordNet to measure the similarity score based on their expansion relations. Our algorithm to measure the similarity score between the pair of predicate (p_1, p_2) is given in Table 6.1.

Calculate $R(p_1, p_2)$
Initial $R(p_1, p_2) = 0$
Step 1:
+ Select the synset S_1 that corresponds to p_1 in WordNet
+ All words in the same synset have the same score
+ If p_2 is found in S_1 then $R(p_1, p_2) = 1$
+ If p_2 cannot be found in S1 then $distance(S_1, S_2) = 1$, go to Step 2
Step 2:
+ Select the other synsets S_2 with relation such as hyponyms with S_1
+ $distance(S_1, S_2) = distance(S_1, S_2) + 1$
+ If p_2 is found in S_2 then $R(p_1, p_2) = 1/distance(S_1, S_2)$
+ If p_2 cannot be found in S_2 then update $S_1 = S_2$ and go to Step 2
We do the same steps to calculate $R(p_2, p_1)$
Finally, we have $Sim_p(p_1, p_2) = R(p_1, p_2) + R(p_2, p_1)/2$

Table 6.1: Algorithm to measure the similarity score between two predicates

6.1.2.2 Semantic labels translation probability

As we analysis above, another part of semantic frame matching is the translation probability between two semantic labels. Two arguments with two semantic labels in the same group such as core arguments have higher translation probability than two semantic labels in the different groups. For instance, $P(l_{A0}|l_{A0}) > P(l_{A0}|l_{A1}) > P(l_{A0}|l_{AM-TMP})$.

We use factoid question-answer pair from TREC-8 and TREC-9 QA task as the training data to measure the translation probabilities. In the training step, we use semantic parser to label all the training questions and the corresponding answers. After that, we employ GIZA, a statistical translation package, to train the paired labels with IBM translation model 1. We treat each label in a question

as a word in a source sentence and each corresponding label pair in an answer as a word in a target sentence. GIZA aligns the labels from the source sentences to the target sentences. The result of the alignment is a label translation probability table and we use this table to define the label pair mapping scores. GIZA performs an iterative training process using EM to learn the pairwise translation probabilities. In every iteration, the model automatically improves the probabilities by aligning the labels based on the current parameters. We initialize the training process by setting the translation probability between an identical labels to 1 and a small uniform value of 0.01 for all other cases, and then run EM to convergence.

6.1.2.3 Semantic similarity score

Let's define $F_{qi} = (p_{qi}, A_{qi})$ and $F_{qj} = (p_{qj}, A_{qj})$ to be the frame for two questions that we need to measure the similarity score. We divide the semantic similarity score into two components; one indicating the similarity score between the predicates, and the other is the similarity score between the arguments.

$$Sim(F_{qi}, F_{qj}) = \alpha * Sim_p(p_{qi}, p_{qj}) + (1 - \alpha) * Sim_A(A_{qi}, A_{qj}) \tag{6.1}$$

where $Sim_A(A_{qi}, A_{qj})$ denotes the similarity score between the two arguments sets, $A_{qi} = ((l_1^{qi}, w_1^{qi}), ((l_2^{qi}, w_2^{qi}), ..., ((l_n^{qi}, w_n^{qi}))$, and $A_{qj} = ((l_1^{qj}, w_1^{qj}), ((l_2^{qj}, w_2^{qj}), ..., ((l_m^{qj}, w_m^{qj}))$; and α is a weighting parameter that can be tuned.

In natural language, two similarity arguments can be expressed in different form with different ordering, thus, we choose the fuzzy matching by considering all arguments in a frame together. The equation for measuring the similarity score between the two sets of arguments $Sim_A(A_{qi}, A_{qj})$ is

$$Sim_A(A_{qi}, A_{qj}) = \sum_{u=1}^{n} \sum_{v=1}^{m} P(l_u^{qi} | l_v^{qj}) * e^{Sim_w(w_u^{qi}, w_v^{qj})} \tag{6.2}$$

where $P(l_u^{qi}|l_v^{qj})$ is the translation probability from a label l_u^{qi} to a label l_v^{qj}, and $Sim_w(w_u^{qi}, w_v^{qj})$ is the similarity between two sets of words w_u^{qi}, w_v^{qj}. We match each a_u in A_{qi} against each a_v in A_{qj}, considering that each arguments in question q_i has a probability to transform into an argument with a different label in question q_j. Finally, we sum up the score of translating each $a_u^{qi}(l_u^{qi}, w_u^{qi})$ to each $a_v^{qj}(l_v^{qj}, w_v^{qj})$ to get the overall score between the two arguments sets A_{qi} and A_{qj}.

To measure the similarity between two sets of words w_{qi}, w_{qj}, we use Jaccard coefficient. We remove all the stop words from w_{qi}, w_{qj} and get the stemmed forms of the remaining words. We define the equation for measuring the similarity between w_{qi}, w_{qj} as

$$Sim_w(w_u^{qi}, w_v^{qj}) = \frac{|w_*^{qi} \bigcap w_*^{qj}|}{|w_*^{qi} \bigcup w_*^{qj}|} \tag{6.3}$$

where w_*^{qi}, w_*^{qj} are the words in argument qi and qj after removing stop words.

Both questions can contain more than one semantic frame. Hence, we measure the pairwise frame semantic similarity scores, and pick up the highest similarity score as the score between the two questions. We then use this score to re-rank the retrieved questions and select the best similar questions.

6.2 Data configuration

In order to evaluate the performance of applying semantic parser in finding similar questions in cQA, we use the data published in (Wang et al., 2009). The data sets were collected by using Yahoo! Answer API to download QA threads from the Yahoo! Site that includes 0.5 million QA pairs from Healthcare domain over a 10-month period from 15/02/08 to 20/12/08. It covers six sub-categories including Dental, Diet&Fitness, Diseases, General Healthcare, Men's health, and Women's health.

To avoid the problems occurred when multiple questions were asked in a single question thread, we use a simple heuristic rules that segment each question thread into pieces of many single-sentence questions by using question mark and 5W1H words. Separating a multiple question into many single questions, we achieve the following two advantages: (1) different questions may ask about different aspects, thus, separating them may reduce the misunderstanding in annotation and is helpful to better match the question with user's query; and (2) the syntactic parsers is able to annotate short sentences better than the long sentences. In addition, the memory requirement and ambiguous syntactic structures are unlikely to occur.

These data sets are divided into two parts. The first part (0.3M), downloaded in the 3.5 months dated from 15/02/08 to 05/06/08, is used as the ground-truth setup; and the rest is used as test-bed for evaluation. For ground-truth, four annotators were asked to go through and check their similarities. To reduce the checking time, K-means text clustering method was used to first group similar answers with the assumption that two questions are considered similar if their answers in question threads are similar. The grouping answers, thus, help to find corresponding similar questions. For each sub-category, 20 representative groups were chosen in order to ensure well coverage on topics in each domain.

The statistics from the data sets are shown in Table 6.2[1]. There are a total of 301,923 question threads from 6 sub-categories presented in the first column. The second column presents the number of single questions. In these data sets, the average number of questions asked per question thread (referred to as "Q Ratio") is 1.96. The last column gives the number of questions annotated in the ground-truth.

In Table 6.3, we describe the instance question threads in the data sets. The first, second, and third rows are the examples in the same category and same group, but their number of words in the title is different. In these rows, the questions

[1]This table is adapted from (Wang et al., 2009)

Category	# of question thread	Est # of questions	Q Ration	# of Ground-truth
Dental	28879	59349	2.06	875
Diet&Fitness	105079	202331	1.93	4929
Diseases	31017	59259	1.91	407
General Healthcare	23004	45067	1.95	1008
Men's Health	42017	77342	1.84	819
Women's Health	71930	149880	2.08	1222
Total	301923	593228	(avg) 1.96	9260

Table 6.2: Statistics from the data sets using in our experiments

appear in both the title and subject fields. While the first and second rows are relevant, the third row is not relevant to the topic "bad breath". The fourth row is a sample without the subject field. The last row is a sample where question appears only in the title. The average length of question part is 2 to 3 sentences. Spelling errors are very common in these data sets.

To build up the testing questions, each annotator was also asked to indicate the topic of each group of similar questions. Then, these topics were used as a guidance to choose the testing questions. A total of 109 testing questions were selected, which ensure that all 109 groups are covered in the ground-truth. In these data sets, the questions are of various lengths and in various forms. Table 6.4[2] shows some example queries from the testing set.

6.3 Experiments

6.3.1 Experiment strategy

In our experiment, we use three different systems for comparison:

(1) BOW: a Bag-of-Word approach that simply matches stemmed words between query and questions.

[2]This table is adapted from (Wang et al., 2009)

Category	Group	Title	Subject
Dental	1	Bad breath?	How can I tell if I have bad breath? I am insecure about it and now that I'm dating, I feel like I constantly have to have gum in my mouth. Is there a way to tell by yourself if you have bad breath?
Dental	1	Why does my breath smell even right after brushing?	I have bad breath. I brush once every morning. After I brush it still smells, especially if I **dont** eat. I think my mouth is really dry or something, I **dont** understand. Could it be the food I eat, cheese? Is there a product that helps with this? I cant drink lots of water, I have a bladder problem, but mostly the only thing I drink is water. Please help. Thanks
Dental	1	Any products to help remove acid from saliva?	I believe I have pretty acidic saliva, I am curious if anyone happens to know a product out that can remove the acid content in ones spit. Thanks
Diet & Fitness	5	How many pounds is it likely for me to lose in 3 months on a strict diet with **excersize**? Any suggestions?	
General Healthcare	2	How do you go to bed early when you work till 1am?	I started a new job, I go in at 4pm and get home at about 1:30am. Then I am up till like 6am.. Please give me an idea besides sleeping pills....

Table 6.3: Example in the data sets using in our experiments

(2) ASSERT: use ASSERT to parse results follow by semantic matching.

(3) GReSeA: use GReSeA to parse results follow by semantic matching.

Recall that the Section 6.1, we design the strategy for applying semantic analysis in these steps: (1) choose the main test query from title and subject fields using question mark and 5W1H; (2) annotate the semantic information for all queries using semantic parser; and (3) estimate the similarity score between query and the archived questions by using semantic matching; and choose the top k archived questions that have the highest similarity scores. In case the semantic parser fails to analyze the query, we use the baseline as the backup to retrieve similar archived questions. In the baseline approach, step 2 is omitted.

Category	Topic	Query
Dental	Whitening strips	What is the best way to use crest while strips premium plus?
Diet&Fitness	Weight loss	Tips on losing weight?
Diseases	Pain in legs	Tingling in legs, sometimes pain, what is it?
General Healthcare	Felling tired	Why is it that at the same times afternoon or night I always go tired?
Men's Health	Advice on fitness	Any advice on a fitness schedule including weight lifting and diet plan?

Table 6.4: Example of testing queries using in our experiments

6.3.2 Performance evaluation

Queries tested. Table 6.5 presents statistics on the number of queries tested in the 6 sub-categories of the data sets. While all queries can be handled by the BOW naturally, many queries cannot be parsed by semantic parsers such as the ASSERT and GReSeA. However, note that the number of queries that can be parsed by GReSeA is much higher than that by ASSERT (82.57% vs. 68.81%), hence, demonstrating the robustness of GReSeA.

Category	# of query tested	# of query parsed by ASSERT	# of query parsed by GReSeA
Dental	20	13	16
Diet & Fitness	20	13	16
Diseases	20	10	15
General Healthcare	20	17	20
Men's Health	20	15	15
Women's Health	9	7	8
Total	109	75	90
Ratio	100%	68.81%	82.57%

Table 6.5: Statistic of the number of queries tested

System accuracy. We employ two performance metrics: Mean Average Precision (MAP[3]) and Precision at the top one retrieval results. The equation to compute the precision value is

[3]The MAP calculated on the returned top 10 questions

$$Precision = \frac{\#(relevant\ questions\ retrieved)}{\#(retrieved\ questions)} \tag{6.4}$$

	BOW	ASSERT	GReSeA
MAP(%)	54.09	52.42	56.72
Precision at top 1	73.43	81.85	86.11

Table 6.6: MAP on 3 systems and Precision at top 1 retrieval results

Table 6.6 shows the performance of the 3 different systems. From the Table, we draw the following observations:

(1) BOW model itself achieves mediocre precision at 54.09%. We conjecture that the mediocre precision obtained by BOW is because we use the heuristic rules such as question mark and 5W1H words to segment the question thread into the single-sentence questions. As the result, after removing the stop-words in the single-sentence questions, there are few meaningful words left for matching by BOW. Obviously, BOW does not capture the similarity among questions well.

(2) Since data collected from forum contain many grammatical errors, semantic parser cannot handle these sentences; and thus applying current semantic parse ASSERT obtains even lower accuracy as compared to the baseline system BOW (52.42% vs. 54.09%) in term MAP. GReSeA, with optimization based on grammatical relations, outperforms the ASSERT by 4.30% (56.72% vs. 52.42%), and BOW by 2.63% (56.72% vs. 54.09%) in terms of MAP. The accuracies demonstrate that using semantic parser cannot achieve better accuracy if the semantic parser cannot handle the grammatical errors in the forum language well.

(3) Applying semantic parser to finding similar questions achieves really good results with top 1 retrieved similar questions. Since two questions are actually similar when comparing in semantic similarity score, all semantic components have the high score; and thus semantic matching always returns a correct similar question. Comparing with the baseline approach in terms of top 1 precision, applying semantic parser improves by 8.42% when using ASSERT and 12.68% when using GReSeA. In addition, the improvement by 4.26% (86.11% vs. 81.85%) of GReSeA over ASSERT demonstrates the potential of GReSeA in capturing the similar questions in forum language.

6.3.3 System combinations

Choosing the threshold of similarity score. Many sentences in the data sets have small similarity scores when comparing with query. Because people believe that the QA systems, which have no answer, are better than those that provide the incorrect answers (Brill et al., 2002). Thus, the higher the precision of a QA system, the better it is. To reduce the non-relevant results, we use the threshold of similarity score to remove the retrieved questions that have very low similarity scores. The threshold is selected throughout our experiment empirically, which is given in Table 6.7 and Figure 6.2 with two metrics precision and F1.

With the high threshold score, the number of retrieved questions is smaller; hence, the precision is higher. In contrast, with the small retrieved questions, the recall and F1 accuracy are low. To select the threshold score, we increase the threshold score at intervals of 0.05 until the precision is stable.

From the Figure, we select threshold score at 0.3 because at this point, precision is high at 70.19%. In addition, with the threshold score higher than 0.3, the precisions do not show the significant improvement while the F1 values start to decrease by a large margin.

T	0	0.05	0.1	0.15	0.2	0.25	**0.3**	0.35	0.4	0.45
Precision	56.61	59.44	63.06	64.92	67.77	68.91	**70.19**	70.37	70.57	70.3
F1	55.06	49.4	43.92	37.66	34.84	31.83	**30.38**	28.36	25.65	23.56
Increase of Precision T_{t+1} -T_t	N.A	2.83	3.62	1.86	2.85	1.14	**1.28**	0.18	0.2	-0.27
Decrease of F1 T_{t+1} -T_t	N.A	-5.66	-5.48	-6.26	-2.82	-3.01	**-1.45**	-2.02	-2.71	-2.09

Table 6.7: Precision and F1 accuracy of baseline system with the different threshold of similarity scores

Combination system. We propose a combination system by first using the BOW as a filter in the retrieval questions, and then applying semantic parser to achieve the final results. The architecture of the combination system is shown in Figure 6.3.

To tackle the drawback of BOW when matching the similar questions in single-sentence questions, we change the combination system slightly. Instead of using single-sentence questions, in stage 1, we use BOW to index all the content including subject and title in question thread and get the initial results. Next, the heuristic rules (question mark and 5W1H) are applied to the initial results to recognize the single-sentence questions and then, semantic parser is used to annotate the semantic frame. In stage 2, we use semantic similarity score to estimate the similarity score and select the best results.

In our experiment, we compare three different systems:

(1) BOW+ASSERT: use BOW as the filter step, then apply ASSERT to parse the filtered results.

(2) BOW+GReSeA: use BOW as the filter step, then apply GReSeA to parse the filtered results.

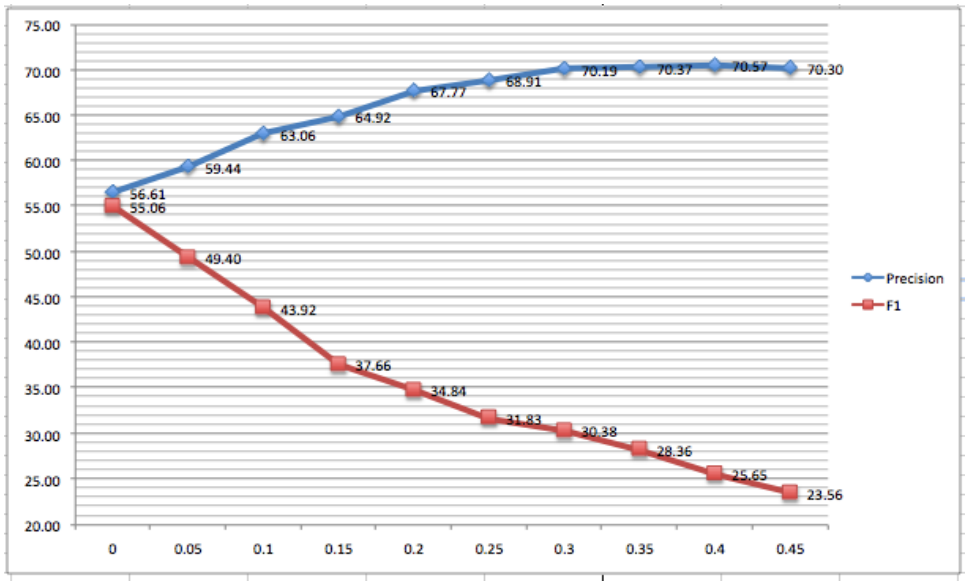

Figure 6.2: Illustration of Variations on Precision and F1 accuracy of baseline system with the different threshold of similarity scores

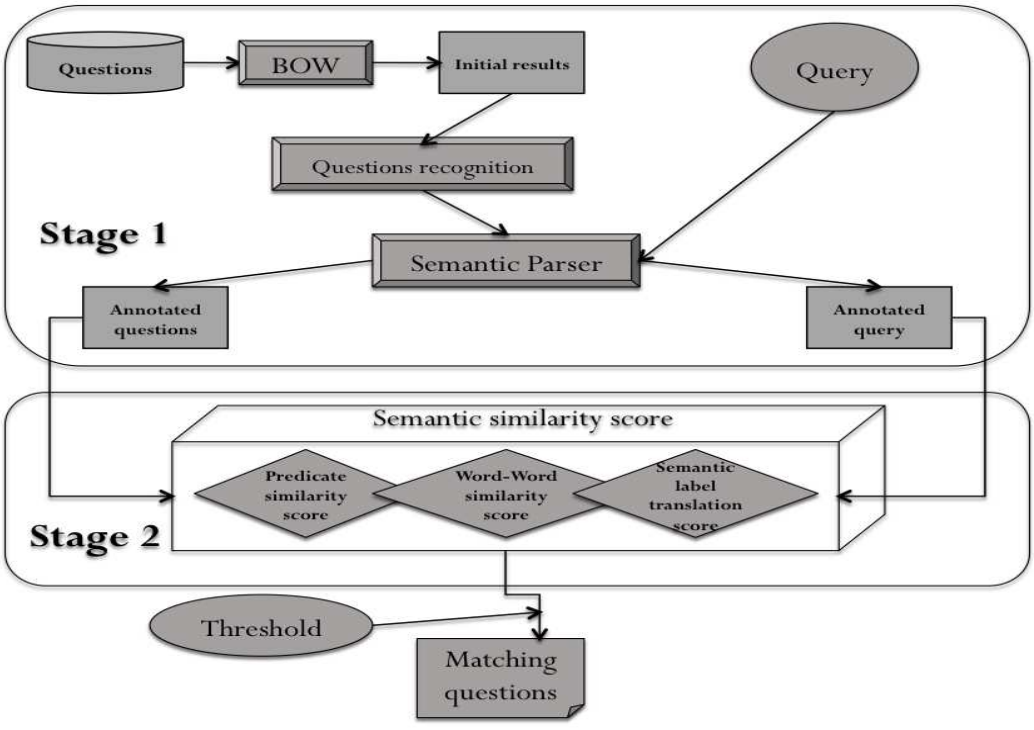

Figure 6.3: Combination semantic matching system

(3) Wang system: the best reported system presented in (Wang et al., 2009)

System accuracy. We use two performance metrics, including Mean Average Precision (MAP[4]) and Precision at the top one retrieval results to evaluate the performance of the three systems. The results are given in Table 6.8.

	BOW+ASSERT	BOW+GReSeA	Wang system
MAP(%)	80.85	82.53	88.56
Precision at top 1	89.72	91.30	89.17

Table 6.8: Compare 3 systems on MAP and Precision at top 1 retrieval results

From the experiment results, we have the following observations:

(1) BOW + ASSERT achieves the high precision of 80.85%, while BOW + GReSeA slightly improves this performance by 1.68%. We conjecture that the high precision obtained in the combination system is because of the higher quality initial results obtained by BOW. The combination of BOW as a filter gives an effective boosting, leading to a significant improvement in MAP by 25.81% (82.53% vs. 56.72%) as compared to the single system as discussed in Section 6.3.2. Specifically, GReSeA, which handles the ungrammatical errors in forum language well, always achieves the higher results as compared to ASSERT in terms of both the MAP (82.53% vs. 80.85%) and Precision at top 1 (91.30% vs. 89.72%).

(2) BOW + GReSeA achieves a lower MAP than Wang system by 6.03%. We conjecture that the lower accuracy is because Wang system integrated the features from answer matcher module. Using the features from answers gives the effective boosting and result in Wang's system achieving the significant

[4]The MAP calculated on the returned top 10 questions

improvement. The features extracted from the answers will be integrated into the proposed system in the future work.

(3) Finding similar questions by applying semantic relations matching always obtains high precision in top 1 retrieval results. From the Table, both combination systems achieve the higher performance as compared to Wang system. While BOW + ASSERT achieves a slight improvement of 0.55% (89.72% vx. 89.17%), BOW + GReSeA improves the performance by a large margin of 2.13% (91.3% vs. 89,17%). These results demonstrate that the effectiveness of the combination system of BOW and semantic parser in capturing the similar questions in forum language.

6.4 Discussion

Handling the forum language styles is not an easy problem. There are no standard templates for processing the forum languages. In our work, we presented a potential approach using semantic parser for finding similar questions. First, we observed that our results are very competitive. The results using GReSeA are better than both baseline BOW approach and using the best of current SRL system ASSERT. Since we handled the ungrammatical sentence well, we achieved an improvement in MAP of 2.63% over BOW and 4.3% over semantic matching system using ASSERT. Second, we further noted that the combination system outperforms the single system by a large margin. The combination system shows an improvement of 25.81% in MAP over the single system. In addition, we observed that the results of our combination system are very competitive, which improves by 2.13% (91.30% vs. 89.17%) on Precision at top 1 over the best system presented in (Wang et al., 2009).

From our experiments, we have two conclusions:

- Using semantic parser based on grammatical relations is a good direction to tackle the basic problems in forum languages such as the grammatical errors.
- A combination system of BOW and the semantic parser in finding similar questions is a potential approach because we can exploit both the statistical and semantic knowledge underlying the natural language.

Chapter 7

Conclusion

7.1 Contributions

In this thesis, we conjectured that grammatical relations could improve the performance of semantic role labeling system. In addition, we also proposed the potential approach for finding similar questions in cQA by applying semantic parser. The following are the contributions of this thesis to the field Semantic parsing and Question answering:

(1) Exploiting grammatical relations to developing SRL system that is robust to grammatical errors.

(2) Applying semantic parser to finding similar questions in cQA.

7.1.1 Developing SRL system robust to grammatical errors

In this work, we built a SRL system based on grammatical relations and some observations to optimize the grammatical relations between words. Grammatical relations are important to obtain the set of headwords that represent the semantic roles in the sentence. As compared to the performance of 19 participated SRL

systems in CoNLL 2005, our approach achieves competitive performance in CoNLL 2005 data sets in terms of F1-measures at 78.27% in dependency-based system. In addition, our system uses less number of features extracted and hence our system requires less computational time to process the corpus. For instance, our system requires 50% less processing time than ASSERT (Pradhan et al., 2004) in CoNLL 2005 testing set. This improvement is achieved because the grammatical relations we used are robust to possible classification errors in semantic labels.

There is a significant difference between our system and the current SRL systems. The current SRL systems tend to use the full syntactic parser tree that is sensitive to small change in sentence structure; hence these systems tend to get stuck when processing the ungrammatical sentences. In contrast, our system based on grammatical relations presents a general view from syntactic parser tree and hence our system is able to handle the ungrammatical sentences better. Overall, our results suggest that the use of grammatical relations can help to improve the performance of processing forum languages.

7.1.2 Applying semantic parser to finding similar questions in cQA

To the best of our knowledge, there is no cQA system that uses semantic analysis approach. In this thesis, we proposed a method for finding similar questions in cQA by applying semantic parser. Based on our SRL system named GReSeA, we proposed a potential approach for exploiting the semantic analysis by using semantic matching. To demonstrate the effectiveness of our approach, we employed the semantic matching algorithm and evaluated our system in Yahoo! Answer data sets. Our approach outperforms the baseline BOW system in terms of MAP by 2.63% and in Precision of top 1 retrieval results by 12.68%. Compared with the popular SRL system ASSERT (Pradhan et al., 2004) on the same task of finding

similar questions in Yahoo! Answer, our SRL system improves the performance in terms of MAP by 4.3% and in Precision at top 1 retrieval results by 4.26%. Additionally, our combination system achieves competitive results, which improves by 2.13% (91.30% vs. 89.17%) on Precision at top 1 retrieval results when compared with the state-of-the-art Syntactic Tree Matching (Wang et al., 2009) system in finding similar questions.

7.2 Directions for future research

The main purpose of our thesis is to demonstrate the role of grammatical relations in tackling the ungrammatical sentence for SRL system, and then apply the SRL system to improve the performance of cQA system in the task of finding similar questions. Based on our promising results, we suggest the following directions for future research:

(1) We currently detect the questions asked using 5W and question mark. However, replying on 5W and question mark is not satisfactory for this task. Our future work will investigate a new approach to detect the main question asked in the forums. Since context is an important part to improve the effectiveness of information retrieval, we will not only detect questions but also important sentences that contain the main information asked. These sentences will become the context to help in retrieving relevant questions in cQA. To achieve this, we will apply semantic parsing to get the semantic information and thus recognize the main information asked by using semantic information.

(2) We plan to better exploit the semantic information annotated in finding similar questions. This means that we will develop an algorithm to utilize the similarity score between two arguments. Instead of using only the word-to-word similarity, we will use phrase-to-phrase to estimate the similarity

score because we believe that phrase contains more information and linguistic knowledge than word. In this way, we can better exploit the effectiveness of semantic information annotated in cQA.

(3) As we analyze above, to understand natural language, an effective approach is to detect the event that is described in the sentence. The past research (Klavans and Kan, 1998) claimed that the role of verb is very important to represent the event in the sentence. In future research, we will develop the features to circumvent the problems in verb prediction. Furthermore, to exploit the semantic meaning in finding similar sentences, instead of using the verb-verb matching, we will implement the algorithm for phrasal verb matching. With phrasal verb matching, for instance, when comparing two verbs "give up" and "give", we will improve the accuracy in calculating similarity score. Thus, we will improve the overall performance in finding similar questions.

References

Ahn, Kisuh and Bonnie Webber. 2008. Topic indexing and retrieval for factoid qa. In *Coling 2008: Proceedings of the 2nd workshop on Information Retrieval for Question Answering*, pages 66–73, Manchester, UK. Coling 2008 Organizing Committee.

Attardi, G., A. Cisternino, F. Formica, M. Simi, and A. Tommasi. 2001. Piqasso: Pisa question answering system. In *Proceedings of TREC-2001*.

Bendersky, Michael and W. Bruce Croft. 2008. Discovering key concepts in verbose queries. In *SIGIR*, pages 491–498.

Berger, Adam, Rich Caruana, David Cohn, Dayne Freitag, and Vibhu Mittal. 2000. Bridging the lexical chasm: statistical approaches to answer-finding. In *SIGIR '00: Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 192–199, New York, NY, USA. ACM.

Brill, Eric, Susan Dumais, and Michele Banko. 2002. An analysis of the askmsr question-answering system. In *Proceedings of 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP*, pages 257–264.

Brown, P., V. Della Pietra, and R. Mercer. 1993. The mathematics of statistical machine translation: Parameter estimation. In *Proceeding of ACM SIGIR*.

Burke, Robin D., Kristian J. Hammond, Vladimir Kulyukin, Steven L. Lytinen, Noriko Tomuro, and Scott Schoenberg. 1997. Question answering from frequently-asked question files: Experiences with the faq finder system. Technical report, AI Magazine.

Carreras, Xavier and Lluís Màrquez. 2005. Introduction to the CoNLL-2005 shared task: Semantic role labeling. In *Proceedings of the Ninth Conference on*

Computational Natural Language Learning (CoNLL-2005), pages 152–164, Ann Arbor, Michigan. Association for Computational Linguistics.

Ciaramita, Massimiliano, Giuseppe Attardi, Felice DellOrletta, and Mihai Surdeanu. 2008. Desrl: A linear-time semantic role labeling system. In *Proceedings of the 12th Conference on Computational Natural Language Learning (CoNLL)*, Manchester, UK.

Collins, Michael and Nigel Duffy. 2001. Convolution kernels for natural language. In *Advances in Neural Information Processing Systems 14*, pages 625–632. MIT Press.

Cong, Gao, Long Wang, Chin-Yew Lin, Young-In Song, and Yueheng Sun. 2008. Finding question-answer pairs from online forums. In *SIGIR*, pages 467–474.

Cui, Hang, Renxu Sun, Keya Li, Min-Yen Kan, and Tat-Seng Chua. 2005. Question answering passage retrieval using dependency relations. In *SIGIR '05: Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 400–407, New York, NY, USA. ACM.

Dang, H. T., D. Kelley, and J. Lin. 2007. Overview of the trec 2007 question answering track. In *Proceedings of the Sixteen Text REtrieval Conference (TREC 2007)*.

de Marneffe, Marie-Catherine and Christopher D. Manning. 2008. The stanford typed dependencies representation. In *COLING 2008 Workshop on Cross-framework and Cross-domain Parser Evaluation*.

Foster, Jennifer and Oistein E. Andersen. 2009. Generrate: Generating errors for use in grammatical error detection. In *Proceedings of the NAACL Workshop on Innovative Use of NLP for Building Educational Applications*, Boulder, Colorado.

Gildea, Daniel and Daniel Jurafsky. 2002. Automatic labeling of semantic roles. *Computational Linguistics*, 28:245–288.

Haghighi, Aria, Kristina Toutanova, and Christopher Manning. 2005. A joint model for semantic role labeling. In *Proceedings of the Ninth Conference on Computational Natural Language Learning (CoNLL-2005)*, pages 173–176, Ann Arbor, Michigan. Association for Computational Linguistics.

Harabagiu, S., D. Moldovan, M. Pasca, R. Mihalcea, M. Surdeanu, R. Buneascu, R. Grju, V. Rus, and P. Morarescu. 2000. Falcon: Boosting knowledge for answer engines. In *Proceedings of the TREC-9 Conference*.

Huang, Jizhou, Ming Zhou, and Dan Yang. 2007. Extracting chatbot knowledge from online discussion forums. In *IJCAI*, pages 423–428.

Ittycheriah, Abraham and Salim Roukos. 2001. Ibms statistical question answering system. In *Proceedings of the TREC-10 Conference*.

Jeon, Jiwoon, W. Bruce Croft, and Joon Ho Lee. 2005. Finding similar questions in large question and answer archives. In *CIKM '05: Proceedings of the 14th ACM international conference on Information and knowledge management*, pages 84–90, New York, NY, USA. ACM.

Jiang, Zheng Ping, Jia Li, and Hwee Tou Ng. 2005. Semantic argument classification exploiting argument interdependence. In *Proceedings of the 19th International Joint Conference on Artificial Intelligence (IJCAI 2005)*, pages 1067–1072, Edinburgh, Scotland, UK.

Jiang, Zheng Ping and Hwee Tou Ng. 2006. Semantic role labeling of nombank: A maximum entropy approach. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing (EMNLP 2006)*, pages 138–145, Sydney, Australia.

Johansson, Richard and Pierre Nugues. 2007. Incremental dependency parsing using online learning. In *Proceedings of the CoNLL Shared Task Session of*

EMNLP-CoNLL 2007, pages 1134–1138, Prague, Czech Republic. Association for Computational Linguistics.

Johansson, Richard and Pierre Nugues. 2008. Dependency-based semantic role labeling of propbank. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 69–78, Honolulu.

Kaisser, Michael. 2008. The QuALiM question answering demo: Supplementing answers with paragraphs drawn from Wikipedia. In *Proceedings of the ACL-08: HLT Demo Session*, pages 32–35, Columbus, Ohio. Association for Computational Linguistics.

Kaisser, Michael and Bonnie Webber. 2007. Question answering based on semantic roles. In *Proceedings of the ACL 2007 Deep Linguistic Proceeding Workshop, ACL-DLP 2007*.

Kate, Rohit J. and Raymond J. Mooney. 2006. Using string-kernels for learning semantic parsers. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 913–920. Association for Computational Linguistics.

Klavans, Judith and Min-Yen Kan. 1998. Role of verbs in document analysis. In *Proceedings of the 17th international conference on Computational linguistics*, pages 680–686, Morristown, NJ, USA. Association for Computational Linguistics.

Ko, Jeongwoo, Eric Nyberg, and Luo Si. 2007. A probabilistic graphical model for joint answer ranking in question answering. In *SIGIR*, pages 343–350.

Li, Wei. 2002. Question classification using language model. Technical report, CiteSeerX - Scientific Literature Digital Library and Search Engine [http://citeseerx.ist.psu.edu/oai2] (United States).

Li, X. and D. Roth. 2002. Learning question classifiers. In *Proc. the International Conference on Computational Linguistics (COLING)*, pages 556–562.

Li, Xin and Dan Roth. 2006. Learning question classifiers: the role of semantic information. *Nat. Lang. Eng.*, 12(3):229–249.

Light, M., G. S. Mann, E. Riloff, and E. Breck. 2001. Analyses for elucidating current question answering technology. *Journal of Natural Language Engineering, Special Issue on Question Answering, FallWinter*.

Liu, Ting, Wanxiang Che, Sheng Li, Yuxuan Hu, and Huaijun Liu. 2005. Semantic role labeling system using maximum entropy classifier. In *Proceedings of the Ninth Conference on Computational Natural Language Learning (CoNLL-2005)*, pages 189–192, Ann Arbor, Michigan. Association for Computational Linguistics.

Liu, Yandong, Jiang Bian, and Eugene Agichtein. 2008. Predicting information seeker satisfaction in community question answering. In *SIGIR '08: Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 483–490, New York, NY, USA. ACM.

Lu, Wei, Hwee Tou Ng, Wee Sun Lee, and Luke. S. Zettlemoyer. 2008. A generative model for parsing natural language to meaning representations. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing (EMNLP 2008)*, Waikiki, Honolulu, Haiwai.

Manning, Christopher D. 2008. *Introduction to Information Retrieval*.

Màrquez, Lluís, Pere Comas, Jesús Giménez, and Neus Català. 2005. Semantic role labeling as sequential tagging. In *Proceedings of the Ninth Conference on Computational Natural Language Learning (CoNLL-2005)*, pages 193–196, Ann Arbor, Michigan. Association for Computational Linguistics.

Mitsumori, Tomohiro, Masaki Murata, Yasushi Fukuda, Kouichi Doi, and Hiro-

humi Doi. 2005. Semantic role labeling using support vector machines. In *Proceedings of the Ninth Conference on Computational Natural Language Learning (CoNLL-2005)*, pages 197–200, Ann Arbor, Michigan. Association for Computational Linguistics.

Miyao, Yusuke, Tomoko Ohta, Katsuya Masuda, Yoshimasa Tsuruoka, Kazuhiro Yoshida, Takashi Ninomiya, and Jun'ichi Tsujii. 2006. Semantic retrieval for the accurate identification of relational concepts in massive textbases. In *ACL-44: Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, pages 1017–1024, Morristown, NJ, USA. Association for Computational Linguistics.

Moschitti, Alessandro. 2004. A study on convolution kernels for shallow statistic parsing. In *Proceedings of the 42nd Meeting of the Association for Computational Linguistics (ACL'04), Main Volume*, pages 335–342, Barcelona, Spain.

Pizzato, Luiz Augusto, and Diego Mollá. 2008. Indexing on semantic roles for question answering. In *Coling 2008: Proceedings of the 2nd workshop on Information Retrieval for Question Answering*, pages 74–81, Manchester, UK. Coling 2008 Organizing Committee.

Pradhan, Sameer, Kadri Hacioglu, Valerie Krugler, Wayne Ward, James H. Martin, and Daniel Jurafsky. 2005. Support vector learning for semantic argument classification. *Machine Learning*, 60(1-3):11–39.

Pradhan, Sameer, Wayne Ward, Kadri Hacioglu, James H. Martin, and Dan Jurafsky. 2004. Shallow semantic parsing using support vector machines. In *Proceedings of the Human Language Technology Conference/North American, Chapter of the Association of Computational Linguistics (HLT/NAACL)*.

Qian, Longhua, Goudong Zhou, Fang Kong, Qiaoming Zhu, and Peide Qian. 2008.

Exploiting constituent dependencies for tree kernel-based semantic relation extraction. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 697–704, Honolulu.

Question-Answering-Wikipedia. 2009. Question answering from wikipedia. *http://en.wikipedia.org/wiki/Question_answering*.

Riedel, Sebastian and Ivan Meza-Ruiz. 2008. Collective semantic role labelling with markov logic. In *Proceedings of the 12th Conference on Computational Natural Language Learning (CoNLL)*, Manchester, UK.

Roth, D., G. Kao, X. Li, R. Nagarajan, V. Punyakanok, N. Rizzolo, W. Yih, C. O. Alm, and L. G. Moran. 2001. Learning components for a question answering system. In *TREC*, pages 539–548.

Shen, Dan and Mirella Lapata. 2007. Using semantic roles to improve question answering. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 12–21, Prague, Czech Republic. Association for Computational Linguistics.

Shrestha, Lokesh and Kathleen McKeown. 2004. Detection of question-answer pairs in email conversations. In *COLING '04: Proceedings of the 20th international conference on Computational Linguistics*, page 889, Morristown, NJ, USA. Association for Computational Linguistics.

Sneiders, Eriks. 2002. Automated question answering using question templates that cover the conceptual model of the database. In *NLDB '02: Proceedings of the 6th International Conference on Applications of Natural Language to Information Systems-Revised Papers*, pages 235–239, London, UK. Springer-Verlag.

Sun, Renxu, Jing Jiang, Yee Fan Tan, Hang Cui, Tat seng Chua, and Min yen Kan.

2005. Using syntactic and semantic relation analysis in question answering. In *Proceedings of the TREC*.

Sun, Renxu, Chai-Huat Ong, and Tat-Seng Chua. 2006. Mining dependency relations for query expansion in passage retrieval. In *SIGIR '06: Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 382–389, New York, NY, USA. ACM.

Sun, Weiwei, Hongzhan Li, and Zhifang Sui. 2008. The integration of dependency relation classification and semantic role labeling using bilayer maximum entropy markov models. In *Proceedings of the 12th Conference on Computational Natural Language Learning (CoNLL)*, Manchester, UK.

Surdeanu, Mihai, A Harabagiu, John Williams, and Paul Aarseth. 2003. Using predicate-argument structures for information extraction. In *Proceedings of ACL 2003*, pages 8–15.

Surdeanu, Mihai, Richard Johansson, Adam Meyers, Lluís Màrquez, and Joakim Nivre. 2008. The CoNLL-2008 shared task on joint parsing of syntactic and semantic dependencies. In *Proceedings of the 12th Conference on Computational Natural Language Learning (CoNLL)*, Manchester, UK.

Suzuki, Jun, Tsutomu Hirao, Yutaka Sasaki, and Eisaku Maeda. 2003. Hierarchical directed acyclic graph kernel: Methods for structured natural language data. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 32–39.

TREC-Overview. 2009. Text retrieval conference (trec) overview. *http://trec.nist.gov/overview.html*.

Wang, Kai, Zhaoyan Ming, and Tat-Seng Chua. 2009. A syntactic tree matching approach to finding similar questions in community-based qa services. In *ACM SIGIR 2009*.

Wong, Yuk Wah and Raymond J. Mooney. 2007. Learning synchronous grammars for semantic parsing with lambda calculus. In *ACL*, pages 960–967, Prague, Czech Republic.

Xue, Xiaobing, Jiwoon Jeon, and W. Bruce Croft. 2008. Retrieval models for question and answer archives. In *SIGIR*, pages 475–482.

Zhang, Dell and Wee Sun Lee. 2003. Question classification using support vector machines. In *SIGIR '03: Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, pages 26–32, New York, NY, USA. ACM.

Zhang, Min, Wanxiang Che, AiTi Aw, Chew Lim Tan, Guodong Zhou, Ting Liu, and Sheng Li. 2007. A grammar-driven convolution tree kernel for semantic role classification. In *ACL*, Prague, Czech Republic.