Beyond Question Answering: Understanding the Information Need of the User

Shuguang Li

Submitted for the degree of Doctor of Philosophy

University of York

Department of Computer Science

September 2011

Abstract

Intelligent interaction between humans and computers has been a dream of artificial intelligence since the beginning of digital era and one of the original motivations behind the creation of artificial intelligence. A key step towards the achievement of such an ambitious goal is to enable the Question Answering systems understand the information need of the user.

In this thesis, we attempt to enable the QA system's ability to understand the user's information need by three approaches. First, an clarification question generation method is proposed to help the user clarify the information need and bridge information need gap between QA system and the user. Next, a translation based model is obtained from the large archives of Community Question Answering data, to model the information need behind a question and boost the performance of question recommendation. Finally, a fine-grained classification framework is proposed to enable the systems to recommend answered questions based on information need satisfaction.

Contents

List of Figures

List of Tables

Acknowledgements

Firstly, I would like to thank my parents and my brother for their endless love and unconditional support during my study in York.

I would like also to thank my supervisor Dr. Suresh Manandhar and my assessor Professor Helen Petrie for their helpful guidance, valuable advice and encouragement about my research.

Additionally, thanks to all the people in the AI group for all their help, support and friendship.

Special thanks go to the people who helps me make the decision to pursue PhD study.

At last, I would like to thank my wife Jing Yang for her support and encouragement during my PhD study. For this, I will never be grateful enough.

Declaration

I hereby declare that all the work in this thesis is solely my own, except where attributed and cited to another author. Some of the material in this thesis has been previously published by the author. A list of publications is as follows:

1. Shuguang Li and Suresh Manandhar. Improving Question Recommendation by Exploiting Information Need. *Proceedings of ACL, 2011.*

2. Shuguang Li and Suresh Manandhar. Automatic Generation of Information-seeking Questions Using Concept Clusters *Proceedings of ACL-IJCNLP: short paper, 2009.*

Chapter 1

Introduction

Science fiction movies often have scenes where people talk to a machine in natural language to get answers to their questions. An intelligent dialogue system powered by artificial intelligence advances has always been the dream of researchers during the past several decades.

The Jeopardy! quiz show which features an answer-and-question format is a well-known U.S. TV quiz show. Given some clues, the contestants will try to guess the answers. Another feature of this show is that the questions cover a broad range of topics such as sports and history. Table 1.1 shows some example quizzes for the contestants.

In 2011, Watson, a DeepQA system (Ferrucci et al., 2010a) from IBM beat Brad Rutter, biggest all-time money winner, and Ken Jennings, the record holder for the longest championship streak. The "magic" behind Watson is the recent advances in natural language processing (NLP), information retrieval (IR), machine learning, computational linguistics, knowledge representation and reasoning (Ferrucci et al., 2010b). However, the research work on answer-

Category: General Science

Clue: When hit by electrons, a phosphor gives off electromagnetic energy in this form.

Answer: Light (or Photons)

Category: Lincoln Blogs

Clue: Secretary Chase just submitted this to me for the third time; guess what, pal. This time I'm accepting it.

Answer: his resignation

Category: Head North

Clue: They're the two states you could be reentering if you're crossing Florida's northern border.

Answer: Georgia and Alabama

Table 1.1: Jeopardy! quiz show examples

ing complex questions and developing intelligent question answering dialogue systems is still in the early stages (Quartertoni, 2007) and (Chali et al., 2011).

In this thesis, inspired by several observed limits in Question Answering (QA) research (e.g. Burger et al. (2001a) and Unger and Cimiano (2011)) and the fast development of community QA services (e.g. Jeon et al. (2005a); Liu et al. (2008); Bunescu and Huang (2010a)), we show our work on analyzing and understanding the user's information need.

1.1 Motivation

Given a collection of documents (such as the Internet and corpus) a QA system should be able to retrieve answers to questions posed in natural language. QA is regarded as requiring more complex NLP techniques than other types of IR, such as document retrieval, and it is sometimes regarded as the next step in the evolution of search engines (Hirschman and Gaizauskas, 2001).

From the 1960s to the 1980s, QA research was restricted to specific domains (closed-domain) (Green et al., 1961; Simmons, 1965; Weizenbaum, 1966; Woods, 1973). In 1999, the annual Text Retrieval Conference (TREC[1]) included a question-answering track, which started to boost the research of open-domain QA. Since then, automatic QA[2] (QA) and human-computer dialogue aided QA

[1]http://www.trec.nist.gov

[2]The term "QA" mentioned next means automatic Question Answering. "CQA" means the online services for users to post questions and get answers from attributors. "CQA systems" are the systems which provide helpful interfaces for users to search for relevant contents from online CQA services.

have received a great deal of attention from the Computational Linguistics research community (TREC 1999-2007). QA research attempts to deal with a wide range of question types including: fact, list, definition, how, why, hypothetical, semantically-constrained and cross-lingual questions (Chali et al., 2011).

However, QA systems cannot always satisfy the users' information needs, as the question processing element may fail to classify the question properly or the information needed for extracting and generating the answer is not easily retrieved (Burger et al., 2001a). In the following subsections, after the discussion on the concept of information need in QA, the three observations which inspired the research in this thesis will be detailed.

1.1.1 Information Need

The concept of information need dated back in the 1960s and was first brought up by Taylor (1962). After the thorough study on the process of asking questions, the formation of a question by a user was deemed to have four levels: 1) the user's conscious and unconscious (visceral) need for information (actual but unexpressed); 2) the conscious mental description of the user (usually rambling and ambiguous); 3) a formal statement of the question (the user can formalize a question at this level); 4) the formalized but compromised question (this question is usually a compromised one in order to be answered by the system). Wilson (1994) did a survey on the research of information needs and uses, and information need was deemed to be within the user's information seeking process and drive the user's information seeking behaviour (identifiable, observable information-related activity). Figure 1.1 shows a diagram of

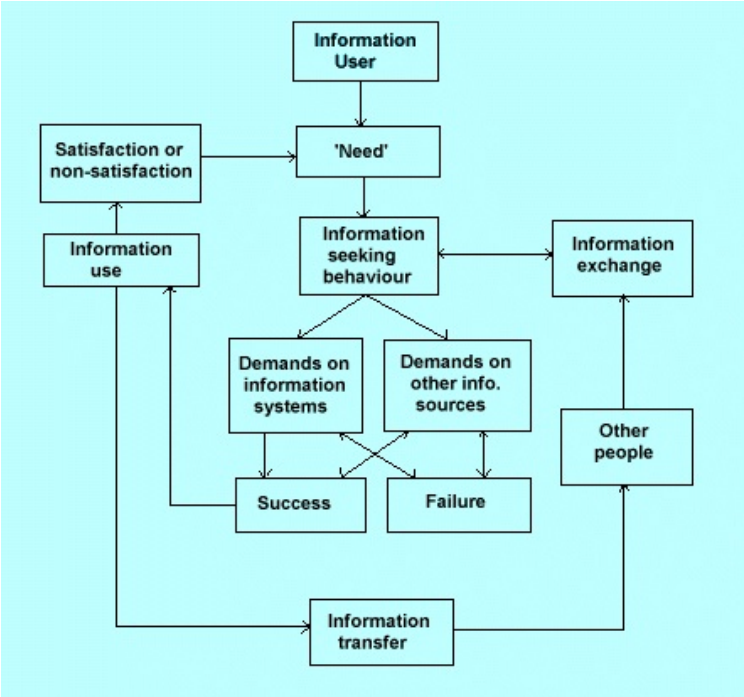

Figure 1.1: A model of information behaviour from Wilson (1994)

the user's information seeking processes.

Information need has been researched and defined from different aspects (Case, 2002). Atkin (1973) treated information seeking as the process of reducing uncertainty, and information need was defined as "*a function of extrinsic uncertainty produced by a perceived discrepancy between the individual's current level of certainty about important environmental objects and a criterion state that he seeks to achieve*". Dervin (1992) described information need as the compulsion of meaning production (sense making) of the user's current situation or the need to understand or make a choice: "*the individual, in her time and place, needs to make sense ... She needs to inform herself constantly. Her head is filled with questions. These questions can be seen as her information need*". As discussed in Wilson (1981) and Chiware (2008), the information

needs of the users were influenced not only by individual characteristics, but also "*linked to specific situations and that needs arise when the present level of knowledge is limited to deal with a new situation*".

Shneiderman et al. (1997) gave the definition of information need as "*perceived need for information that leads to someone using an information retrieval (IR) system in the first place*" in the research on improving user interfaces for textual database search. Figure 1.2 from Broder (2002) shows a classic model for IR, augmented for the web search context. We can see that the information need of a user is formulated into a query, when the user's intent is to locate and obtain information under a specific task. However, not all "user intent" behind the queries in the web search context are "informational". Between 39% and 60% of the queries in web search engines are informational, which focus on the user's goal of obtaining information about the query topic, and the rest queries are of another two categories: navigational and transactional (Broder, 2002) and (Rose and Levinson, 2004). According to Rose and Levinson (2004) and Agichtein (2006), the information needs under the classic IR context can be further classified into Directed (Closed or Open), Undirected, Advice, Locate and List.

As a typical IR application, we think the information needs in QA systems can also be classified into the above categories. For example, in the yearly TREC competition, the information need behind a factoid question can be seen as being Directed (a single, unambiguous answer) and the information need behind an definition question are usually fall into the category of Undirected (learning something about the topic). However QA systems need to understand and classify the user's information need into finer-grain categories in order to return an

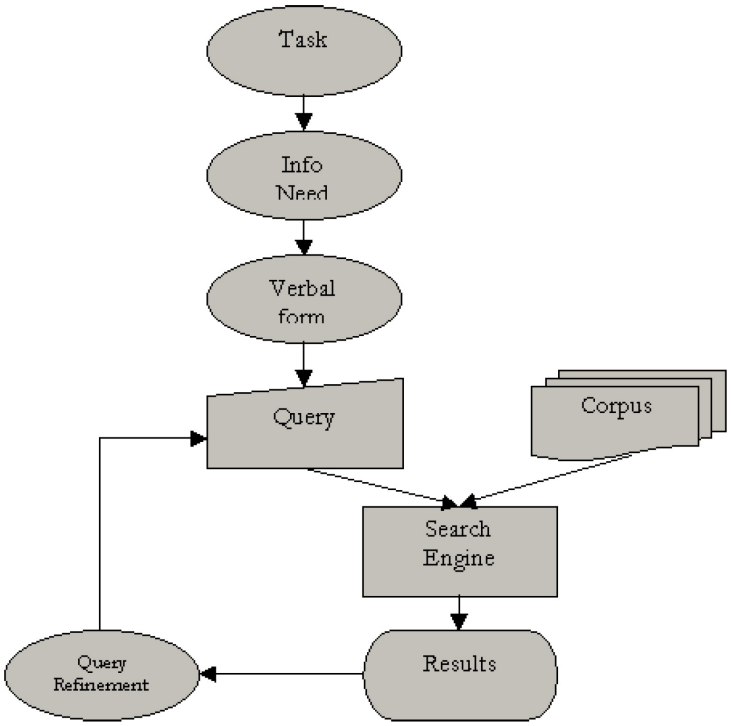

Figure 1.2: Classic model for IR, augmented for the web from Broder (2002)

exact answer instead of a ranked list of documents (Voorhees, 1999). The user's information need over supported answer types (i.e. PERSON, LOCATION) can be inferred in the TREC QA systems which provide varied strategies for different answer types (Agichtein, 2006). Hovy et al. (2002) proposed a new QA typology with fine-grain answer types and patterns associated. Agichtein (2006) did a manual analysis of the users' information needs over real-world questions. The information need behind a question was viewed as the relations between the name entities extracted from the question by Agichtein (2006). A set of fewer than 25 relations covered more than 50% of the real-world questions.

As detailed in the above paragraphs, the previous work on inferring the QA user's information need is restricted to the question itself. But the real-world questions can be quite complex and sometimes ambiguous, and the information need can not be easily inferred purely based on the question. The goal of our work is to understand the user's "*perceived need for information that leads to someone using an QA system in the first place*" which is similar to the definition by Shneiderman et al. (1997) for IR. In practice, we focus on the information need that can be inferred from the question and additional information including the answer contexts and collaborative resources regardless of answer types.

Recently, the user intent behind the questions in CQA services was further studied in Chen et al. (2012) and classified into three categories: "Subjective", "Objective" and "Social". The intent in the first two categories is to seek information, and the questions in these two categories can provide valuable resources for the study for user's information need.

1.1.2 Question Ambiguity

Ambiguity is inherent in natural language and the information need of a user can be expressed in various ways in QA (Burger et al., 2001a). Question ambiguity have been recognized as a critical issue for the automatic interpretation of natural language expressions and question processing in QA (Burger et al., 2001a) and (Unger and Cimiano, 2011).

The ambiguity problem in a natural language can be classified into lexical ambiguity, structural ambiguity, semantic ambiguity, and pragmatic ambiguity Kamsties (2001). Question ambiguities in QA varies: "*from various spellings to the presence of vocabulary terms that have different meanings in different domains (words sense ambiguity)*" (Burger et al., 2001a). Table 1.2 illustrates several forms of these ambiguity, associated with four levels of questioner sophistication. Unger and Cimiano (2011) claimed that question ambiguity can be classified into two categories: 1) the structural properties of an expression, e.g. different parse due to alternative preposition or modifier attachments and different quantifier scopings; 2) alternative meanings of lexical items (linguistic diversity).

Discovering and resolving question ambiguity has been shown to benefit QA systems (Burger et al., 2001a) and (Hori et al., 2003). However, although question context is a useful hint for disambiguation, not all ambiguities can be resolved by specifying the question context (Burger et al., 2001a). The solutions for resolving an ambiguity should vary according to different forms of the ambiguity. If the ambiguity results from "*linguistic diversity*", Burger

Level 1	**Q**: Where is Pfizer Pharmaceutical doing business?	**Ambiguity**: Location Granularity: Country vs. State vs. City
Level 2	**Q**: What recent drugs has Pfizer introduced on the market?	**Ambiguity**: Drugs developed by Pfizer or marketed by Pfizer.
Level 3	**Q**: When was Viagra introduced on the market? **Q**: Was Viagra a successful drug?	**Ambiguity**: Market location: Drug introduction in the U.S., in Europe, or in Mexico.
Level 4	**Q**: Why are pharmaceutical companies using Oracle Clinical Application?	**Ambiguity**: Impact on the customer satisfaction vs. impact on productivity.

Table 1.2: Ambiguity question examples from Burger et al. (2001a)

et al. (2001a) suggested that: "*linguistic ambiguity interacts with linguistic diversity. Various expressions of the same concept encountered in a question need to be conflated correctly, thus resolving most of the ambiguities*".

Through information need analysis, we will show that a clarification dialogue can be provided for the user to clarify the information need and reformulate the question in a way that helps the QA system to reolve ambiguity and find the correct answer. Burger et al. (2001a) claimed that the desirable answer for "*Where is the Taj Mahal?*" should depend on the information need of the user: "*If you are interested in the Indian landmark, it is in Agra, India; If instead you want to find the location of the Casino, it is in Atlantic City, NJ, USA; there are also several restaurants named Taj Mahal*".

1.1.3 Question Similarity Based on Information Need

In recent years, CQA such as Yahoo! Answers and WikiAnswers has established a new type of question QA service. A user posts a query (reference) question and then waits for the right contributors to answer it. This service shifts the inherent complexity of open domain QA from computer systems to volunteer contributors (Jeon et al., 2005a; Liu et al., 2008; Bunescu and Huang, 2010a). With more and more CQA services becoming available, QA research has shifted to question search and recommendation[3]. The users can search the large archives of data for similar questions, in order to minimize the time elapsed before getting the desired answer.

[3]We will provide the definitions for question search and question recommendation in Section 4.1

However, a question can be expressed in a number of ways, due to the variety and complexity in natural language. For example, "*Where can I get cheap aeroplane tickets?*" and "*Any travel website for low airfares?*" are semantically equivalent (reformulated) but have no common words. Therefore, without knowing that the users behind these two questions bear the same information need, it is quite hard for a CQA system to recommend one of the users the other question as the query. Moreover, in a real CQA scenario, users with quite similar information needs may express them using very different questions. For example in Table 1.3, question Q1 and question Q2 are both good recommendation questions for a user's question RefQ. The recommended questions are not necessarily identical or semantically equivalent to the query question, as long as the information needs behind the questions are similar.

Bag of words (BOW) overlapping based approaches (e.g. Monz and de Rijke (2001)) would fail to capture the similarity between the above reformulated questions and the the example questions in Table 1.3. Therefore, we think taking the information need similarity between questions into consideration can benefit question similarity measurement and question recommendation. The data from CQA services can provide valuable resources for boosting the research on analyzing, understanding and inferring the users' information needs.

1.1.4 Information Need Satisfaction

In QA research, the information need satisfaction research concerns whether extracted answers can satisfy the user's information need based on the question and the context (if provided). When more and more CQA services become available, the QA research has been shifted to question search and recommen-

RefQ	If I want a faster computer should I buy more memory or storage space? ...
RefInforNeed	I edit pictures and videos so I need them to work quickly. Any advice?
Answer	... If you are running out of space on your hard drive, then ... to boost your computer speed usually requires more RAM ...
Q1	What aspects of my computer do I need to upgrade ...
Infor Need1	... making a skate movie, my computer freezes, ...
Q2	What is the most cost effective way to expend memory space ...
Infor Need2	... in need of more space for music and pictures ...

Table 1.3: Yahoo! Answers question examples

dation and the CQA services can provide quite valuable resources for the information need satisfaction research (Liu and Agichtein, 2008). We think the information need satisfaction problem in CQA can then be defined as "*how to recommend questions found in CQA resources, so that the user's information need can be better satisfied, by giving suggested questions with their answers*".

The information need satisfaction problem in CQA question recommendation was first studied as a question usefulness ranking task by Bunescu and Huang (2010a). And the questions were thus recommended with respect to the degree of information need satisfaction. Table 1.4 shows an example from Bunescu and Huang (2010a) for the query (reference) question "*What bike should I buy for city riding?*" from their manually labeled data set. The "Reformulation" (paraphrasing) questions, which express the same information need as the user, should be recommended at the top. The "Useful" questions partially satisfy the user's information need, while the "Neutral" questions should not be recommended, as the user's information need will not be satisfied.

A pair of paraphrasing questions express the same or quit similar information need. We also find that a large number of questions in the "Useful" category in the dataset provided by Bunescu and Huang (2010a) have a textual entailment relation with their corresponding reference questions. Recommending these textual entailed questions can partially address the asker's information need (Bunescu and Huang, 2010a). Therefore, we think that question paraphrasing and textual entailment techniques can be considered to discover different relations of questions in CQA data and benefit the research on information need satisfaction. We will further study the question usefulness ranking problem and how fine-grain question usefulness classification can benefit question rec-

Reference Question: What bike should I buy for city riding?
Reformulation: What is the best bike to buy to get myself around campus in the city ? What is the best bike for traveling in a city ?
Useful: What bike should I buy for a starter bike ? What bike should I get as a beginner ?
Neutral: What is a good bike to start mountain biking ? What bike should I buy for hills ?

Table 1.4: Ordered question usefulness example from Bunescu and Huang (2010a)

ommendation.

WikiAnswers also provides CQA services, which relies on the Wiki technology used in Wikipedia. Users of WikiAnswers can tag question reformulations, in order to prevent the duplication of questions asking the same thing in a different way. It should be noted that contributors get no reward, in terms of trust points, for providing or editing alternate wordings for questions. We find that most of these tagged "reformulation" questions do not paraphrase the original questions. However, we think the wealth of such questions available on the WikiAnswers website can still be utilized for the research on question usefulness.

1.2 Contributions

This thesis contributes to the field of QA by analyzing the information needs of users. In order to understand the information needs of QA users, the approaches presented in this thesis make full use of the state-of-the-art in NLP and machine learning and the large amount of CQA data.

1.2.1 Question Disambiguation for Understanding the User's Information Need

We propose an approach to analyze the information need behind an ambiguous question from the context information around the retrieved answers. Using this approach, an interactive dialogue can be provided in order to help the user clarify the information need and reformulate the question to help the QA system find the correct answer. Furthermore, we also propose an efficient and effective feature selection algorithm, to improve the quality of the clustered context concepts.

1.2.2 Question Recommendation Based on Information Need

The CQA service data can provide valuable resources for the research of analyzing the user's information need behind a question. An IBM-4 machine translation based approach is proposed to model and infer the information need behind a question using CQA data as a parallel corpus. So that the similarity between questions can be measured based on the information need-

s inferred besides syntactic features. The Latent Dirichlet Allocation (LDA) based measures (Celikyilmaz et al., 2010) are compared with word based and knowledge based approaches, to calculate the similarity for question recommendation in CQA data.

1.2.3 Information Need Satisfaction by Usefulness Classification

We study the information need satisfaction problem in CQA question recommendation as a question "usefulness" classification problem. The ranking of the recommended questions are based on how far the information need of the query question can be satisfied, with respect to the recommended questions. A fine-grain question usefulness classification framework is proposed to help decompose this problem into a few subtasks, such as question paraphrasing and question textual entailment. For these subtasks, we compare different syntactic and semantic measures using a user-annotated gold standard. Moreover, a large amount of CQA data, without any further annotation is used to retrieve useful questions that can fulfill the user's need for information.

1.3 Thesis Organization

This thesis is organized into the following chapters:

Chapter 2 provides a review of related work, such as the development of QA research, interactive QA and concept clustering techniques. The related work arising with CQA services includes high quality content retrieval, question and

answer retrieval and user satisfaction prediction.

Chapter 3 presents an in depth study of the ambiguity problem in understanding the user's information need in QA. A clarification question generation method is proposed to help the user clarify the information need and reformulate the question in a way that helps the QA system find the correct answer.

Chapter 4 focuses on analyzing the users' information needs. By making use of the large archives of CQA data, a machine translation model can be obtained, to help boost the performance of question recommendation based on the information need similarity between questions.

Chapter 5 discusses the information need satisfaction problem in CQA question search. A fine-grain classification framework is proposed to formulate this problem as several binary classification subtasks using the WikiAnswers repository.

Chapter 6 summarizes the contributions of this thesis and presents the conclusions along with a discussion of open issues and future research directions.

Chapter 2

Background

2.1 Introduction

This chapter presents previous and recent methods relevant to this research project. The presentation focuses on examining the limitations of the approaches and highlighting the issues that strengthen the motivation for this research, addressed in the following chapters.

This chapter is divided into three parts:

1. Question Answering
2. Question Disambiguation
3. Community Question Answering Resources

2.2 Question Answering

The QA research spanned a history from closed-domain (Green et al., 1961; Simmons, 1965; Weizenbaum, 1966; Woods, 1973) to open-domain (e.g. Text REtrieval Conference TREC 1999-2007). QA research attempts to deal with a wide range of question types including: fact, list, definition, how, why, hypothetical, semantically-constrained and cross-lingual questions. And the challenges posed by answering different types of questions have been addressed by using a large variety of techniques, such as shallow and deep parsing, keyword extraction, name entity extraction, use of ontology, question typing and machine learning of answer patterns appropriate to question forms (Hirschman and Gaizauskas, 2001; Burger et al., 2001a; Harabagiu and Hickl, 2006; Quartertoni, 2007; Ferrucci et al., 2009; Chali et al., 2011). The history of the development of QA, the architecture of QA systems and the definition of "relevance" in QA will be detailed in this section.

2.2.1 Early Question Answering

From the 1960s to the 1980s, most QA systems were developed to answer questions in a restricted-domain. In the 1960s, BASEBALL answered questions about the US baseball league (Green et al., 1961). The information about every baseball event was all stored in a list structure database and the question was also transformed into a list structure. The question's list structure was used to search against the database for the desired answer. LUNAR was also such kind of QA system about lunar science projects (Woods, 1973). The development of comprehensive theories in computational linguistics led to the development of ambitious projects in text comprehension and QA, from the

1970s (Quartertoni, 2007). Simmons (1965) surveyed fifteen QA systems including conversational question answerers and front-ends to databases. Most of these systems depended on the semantic structure construction and canonical form matching in the databases.

Weizenbaum (1966)'s ELIZA was developed as a human computer conversation system which based on key word discovering and pattern matching. However ELIZA was not robust enough to answer open domain questions.

During this period, systems usually had a core database or human expert hand-craft knowledge system of the restricted-domain. The questions which can be answered are all restricted to some limited domain.

2.2.2 TREC Question Answering

In the late 1990s the annual Text Retrieval Conference (TREC) included a question-answering track which ran until 2007. This was the first large-scale open-domain QA system evaluation. The task of this track is to provide answers for the questions instead of only retrieving relevant documents. The answers for the questions are expected to be generated from a corpus of text. This competition had significant impact on open-domain QA research and QA system architectures (Burger et al., 2001a) and (Ferrucci et al., 2010b).

2.2.2.1 TREC-8 QA

In TREC-8, a QA track was introduced to provide a common evaluation framework for large-scale open-domain QA. The task for each participant was "given

200 fact-based, short-answer questions (factoid questions), each participants should return a ranked list of [document-id, answer-string] pairs per question such that each answer string was believed to contain an answer to the question. Answer strings were limited to either 50 or 250 bytes, and could either be extracted from the corresponding document or automatically generated from information contained in the document. Each question was guaranteed to have at least one document in the collection that explicitly answered the question. Human assessors read each string and made a binary decision as to whether the string actually did contain an answer to the question in the context provided by the document. Given a set of judgments for the strings, the systems were evaluated by the mean reciprocal rank (MRR) score" (Voorhees, 1999).

The factoid questions were fact-based, short-answer questions and limited to either 50 or 250 bytes, such as "How many calories are there in a Big Mac?" and "Who is the president of the State of America?". Most of the participants used a three-step architecture for their QA system. A system first classified the question according to the type of the answer, such as "person" and "location". Next, the question was transformed into queries to retrieve a batch of candidate documents from the TREC corpus. The last step is to detect entities from the candidate documents. The detected entities were returned as potential answers if the entities matched the type of the answer, otherwise retrieved documents associated with scores were returned in a ranking list. The most accurate systems in TREC-8 were able to find an answer for more than 70% of the questions (Voorhees, 1999).

2.2.2.2 TREC-9 QA

The basic QA task of TREC-9 was generally the same as TREC-8 except that the questions were drawn from different resources and the document collection was larger. The judgement criteria for the correct answers of TREC-9 was more strict and a small portion of the questions were more difficult. The systems were asked to have more robust question type classification module in order to handle reformation of the same question. For the questions whose answer strings were limited to 250 bytes, the best system missed 34% of them. And for the questions whose answer strings were limited to 50 bytes, the best system missed 14% of them (Voorhees, 2000).

Most of the participants' systems still followed the three-step architecture: question classification, candidate document retrieval, and answer generation. And these systems gained lots of improvement which came from different new approaches adopted in the three steps. WordNet Miller (1995) became widely used in both query generation in candidate document retrieval and answer generation. Kwok et al. (2000) used WordNet to add related concepts or synonyms for query expansion.

Falcon from Harabagiu et al. (2000) performed the best among all the submitted systems. The input question were transformed to a semantic form which was used to classify the answer types based on WordNet's taxonomy. The eventual answers must go through both semantic form unification and simple logic proof process. If any of the process failed, a new set of terms related to the concepts in the questions were generated again and again until the candidate document retrieval step was able to retrieve a mount of document in a pre-determined range and the final answers got approved.

2.2.2.3 TREC-10 QA

The main task followed previous year's basic QA task except that the answer string was limited to 50 bytes and the answer was not guaranteed to be found in the collection of documents. Most of the participants' systems still followed the three-step architecture including question classification, candidate document retrieval and answer generation. Similar to previous year, most of the systems achieved better performance by improving the methods used in each step (Voorhees, 2001).

Besides the main task, a context task was added. The systems were required to answer a set of related and sequential questions about the same context. However, as underlined in De Boni and Manandhar (2003), each of the questions in a serial was answered separately by completing questions containing anaphoric references with keywords from the previous question. A list task was also added. The systems were required to answer question about a list of entities such as "What are 9 novels written by John Updike?". However most of the systems simply apply their systems designed for the main task to the other tasks without any main modification.

Soubbotin (2001) performed best in the main task. The system was built on indicative patterns which was used to match text fragments. Each type of questions was linked to a set of predefined patterns. After the question type was determined, the key concepts in a question was filled in the corresponding patterns in order to match the potential answers in the retrieved paragraphs.

Last year's winner Harabagiu et al. (2001) won the second place in the main task. The correctness of an answer was still evaluated by a unification process between the question and the answer. A reference resolution algorithm was used to handle pronouns and different kinds of reference in the context questions.

2.2.2.4 TREC-11 QA

The main task followed previous year except that only one answer should be returned for each test question and the evaluation criteria was more restrict. This required the system to understand exact answers and recognize when it found the correct one. The answer string which "contains more than just the answer (or is missing bits of the answer)" was not deemed as exact answer (Voorhees, 2002).

Most of the systems still followed the three-step structure: question type classification, document retrieval and answer extraction. Some systems in the main task used sophisticated methods to understand the questions in order to locate the exact answers while others chose shallow, data-driven approaches. The LCC QA system from Moldovan et al. (2002) performed best in the main task. 83% of the factoid questions were correctly answered, well ahead of the second place system at 54.2% (Voorhees, 2002).

The general structure of the system remained the same as before Harabagiu et al. (2001). Both of the question and the retrieved answer passages were parsed by a customised parser and then transformed to their corresponding logic forms. These logic forms can incorporate the relations between concepts

appeared in the question and answers. These logic forms were further transformed to axioms. The lexical chains were also used to generate axioms to incorporate semantic information about related concepts. A customized logic prover guided by these axioms was able to rank candidate answers by their correctness and eliminate "unreasonable" answers (Moldovan et al., 2002).

2.2.2.5 TREC-12 QA

A passage task was introduced to test the systems' ability to retrieve short document extracts (250 characters) or indicate no answer existing in the corpus for factoid questions. The main task was divided into three subtasks: factoids, lists and definitions. The factoid task remained the same as previous year. The list question task became more difficult as it required the systems to decide the number of correct answers instead of explicitly providing. Definition questions asked about some aspect of a person or thing such as "Who is Vlad the Impaler?" and "What is a golden parachute?". As a definition question was more difficult to answer than a factoid one and the correctness was more difficult to judge, a more rigid evaluation procedure for definition questions was conducted (Voorhees, 2003).

Most of the QA systems still followed the three-step architecture. After each question was classified into one of the predefined types, a set of documents were retrieved based on the key words in the question or related concepts. At last the answers were extracted from the set of documents and validated afterwards. The methods used by the QA systems were similar to last year and the improvement from each participant resulted in better coverage and accuracy (Voorhees, 2003). These methods included tagging, parsing, name

entity finding, co-reference resolution, syntactic relation extraction and structured pattern extraction.

Harabagiu et al. (2003) performed the best in both factoid and list tasks, and ranked the third in the definition task. In the question processing step, the methods used for factoid and list questions were identical and got some improvement. A better name entity recognizer was added to assist question type determination. After a set of potential answers were discovered from candidate documents, a logic prover was used to remove incorrect ones through abdicative justification. For list questions, the similarities between answers were calculated in order to decide the number of returned answers. For definition questions, each parsed question was matched against a set of predefined question patterns which were linked with answer patterns. These answer patterns were used to match answers in the documents.

BNN's QA system (Ana et al., 2003) performed the best in the definition task. After question type classification, a set of sentences were retrieved by BNN's Information Retrival engine. Next, the sentences which did not mention the question target were filtered by some heuristics. The phrases which had some relations with the question target in the sentences were extracted as "kernel facts". These "kernel facts" were ranked based on their types and the distances to the question's profile which were a vector of related words appearing in the contexts of the question target from other resources. These ranked "kernel facts" were returned as the final answers.

A relevance-based language modeling approach from Zhang and Lee (2003) was used for passage task and gained quite good performance. Besides a quite fine-

grain named entity recogniser, the answer candidate sentences were selected and ranked based a QA event analysis approach from (Yang et al., 2003).

2.2.2.6 TREC-13 QA

The main task of TREC 2004 QA track put several different types of questions (factoid, list and definition questions) about the same nominal target (person or organization) into a series. The target can be viewed as a topic description which a TREC topic developer would write to add the context information for questions in a series, and also expresses the information need behind the questions. So the participant QA systems were not only required to handle these different types of questions but also deal with references and ellipses in the questions. The evaluation of each type of questions remained the same as previous year (Voorhees, 2004).

One of the changes of the QA system from National University of Singapore was in the answer ranking step. The similarity between a question and a potential answer was calculated based on the grammatical dependency relations extracted from both sides. Another change was in the retrieval step. For the definition questions, a set of definition sentences were extracted from both TREC corpus and external authentic resources. In addition to the hard pattern (manually crafted) matching approach used for extraction in previous year's system, a soft pattern matching method was added to boost the recall of definition sentences extraction. Moreover these sentences were added to the documents for answering factoid and list questions (Cui et al., 2003).

PIQUANT II from IBM's Watson Lab (Chu-Carroll et al., 2003) attended the QA TREC with interesting features. In the question precessing step, a new

question was generated from each question by anaphora resolution. Later a set of Answering Agents were applied to these generated questions to generate answers. For example, the Juru-Based Predictive Annotation Agent included named entity annotators, query generation and expansion, and document retrieval. The Profile Agent used highly correlated entities with the target to select documents for answering the definition questions. This approach did not perform well but showed quite promising in the future development. The QA-by-Dossier Agent generated different types of questions about the given targets before the questions were submitted (Chu-Carroll et al., 2003).

The QA system from Fudan University ranked second-place in the definition question task. A pattern-based method was used to extract more answers for the list questions. Knowledge base such as WordNet and a few online dictionaries were used to get related definitions. The Target was expanded so that the the recall of the answer and knowledge extraction can be largely improved (Wu et al., 2004).

2.2.2.7 TREC-14 QA

There were three tasks in TREC 2005 QA track: a main task, a document ranking task and a relation task. The details of the tasks were described in Voorhees and Dang (2005). The main task was the same as previous year's except that the target could be a nominal string (a person, a organization or a thing) or an event such as "return of Hong Kong to Chinese sovereignty" and the dependencies between questions in a series increased. Thus the QA systems were required not only to incorporate context information provided by the event target but also to handle anaphoric references or ellipses in a

question series. The participants were also required to incorporate temporal factor in their systems. The document ranking task was introduced to test the performance of document retrieval module in the QA systems. In the relation task, a topic statement was given as a context for the question. And then the question was "either a yes/no question, which was to be understood as a request for evidence supporting the answer, or a request for the evidence itself" (Voorhees and Dang, 2005).

The QA system from LCC (Harabagiu et al., 2005) performed the best in the main task. One of the features of LCC's QA system was the usage of a logic prover presented in Moldovan et al. (2003) and Moldovan et al. (2005) to filter answers based on their correctness. Their logic prover continue gaining some improvement. Both the questions and the answers were transformed to their corresponding logic forms after the process of syntactic parse, Named Entity Recognition, semantic parse and temporal context representation. The logic forms were further transformed to axioms which were used by the logic prover. The WordNet axioms, Ontology axioms and linguistic template axioms were also fed into the logic prover. In order to deal with the temporally constrained events introduced, a textual inference axioms generation module was added. Before answer selection, each of the answer was processed by the logic prover and thus obtained a confidence score. For the definition questions, LCC used pattern based approaches which proved to be a quite effective and efficient way to discovery answer nuggets. And LCC treated the relation task as answering complex questions. In the question processing step, the question context (text) were decomposed into several short questions. After a set of documents were retrieved three strategies were employed to help find the desired answers. The Keyword Density strategy ranked the documents based on the matching

score of keywords and dependency relations. The Topic Representation strategy selected answers based on the topic-relevant words and relations (topic signatures) obtained from the retrieved documents. The Lexicon Generation strategy generated lexicons which were critical for questions type determination. These expanded lexicons were used further to filter irrelevant answers.

The QA system from CL Research detailed in Litkowski (2005) performed the best in the relation question task. Each relation topic was transformed into a boolean query which contained a set of selected words joined by AND or OR operations. If no documents were found, the boolean query was modified until some documents were able to be found. Each relation topic was also transformed to a new question. Treating this new question as a definition one, the system compared two approaches. The first approach was a pattern matching method to find definition snippets in the documents. The second approach made use of the context features such as the surrounding adjectives, noun constituents, and prepositional relations.

The system from National University of Singapore performed quite well in the main task and the document ranking task. Their QA system described in Sun et al. (2005) continued to apply semantic relation analysis to answer extraction. The system used question expansion method to obtain more relations of the related terms in the questions in order to solve the matching problems caused by short questions. A bigram soft pattern matching model was improved from last year to further boost the recall of definition sentences extraction.

2.2.2.8 TREC-15 QA

TREC 2006 QA track consisted of two tasks piloted in Dang et al. (2006). The main task was the same as previous year except that the QA systems were required to consider temporal factors and return the most up-to-date answers for the questions in present tense. A complex relation question task was added to test the QA systems' performance of answering complex questions. In this task, a question was composed of a template part and a narrative part. The template part was a fixed question template with a few free name entity slots that varied for each question. These bracketed name entities were often called "facets". The narrative part was a few sentences elaborating on the information need. This complex relation task required the QA systems to capture the relations between these name entities in a question and understand the complex information need behind. This complex relation task allowed the system to interact with NIST assessors to get feedbacks about the initial submit results (initial run). The adjusted results from these feedbacks was also shown as the second run.

One QA system from LCC performed the best in both factoid and list question tasks detailed in Moldovan et al. (2006). In question analysis step, one of the new features of their system was a temporal recognition and resolution module to solve the temporal constraints in questions. Each question was reformulated after the time of the target was resolved. Another new feature was that the questions were reformulated based on previous answers and target information. The answer type detection module was further improved by combining heuristics and machine learning approaches. In the answer extraction step, temporal context analysis was improved to help unify the time between a question and an answer in different level of granularity including phrase lev-

el sentence level and word level. The logic prover mechanism also got some improvement including negation detection and representation, conditional detection and representation and etc. The negation detection and representation module was able to represent negative information in the documents. And the conditional detection and representation introduced conditional relations between clauses in the logic representation. The natural language axioms and lexical chain axioms were both improved to help the system capture syntactic equivalence and semantic related concepts.

Another QA system from LCC also attended the main task. As described in Hickl et al. (2006b), there are four main improvement in the system. First, topic-relevant words and relations (topic signatures) were used for query expansion in addition to traditional WordNet synonyms expansion approach. Second, a two-layer document retrieval strategy was used to enhance the recall of document retrieval followed by a ranking method to ensure precision. Third, content modeling was used to select relevant passages for answering definition questions. Finally, one state-of-the-art textual entailment method was used to re-rank the retrieved answers and select the answers which were more likely to be entailed by the question. While the abductive reasoning mechanism was not used. The AQUAINT corpus (Voorhees, 2002) was deep pre-processed in order to support the above improvement: syntactic and semantic parsing, name entity recognition and classification, and temporal expression recognition.

The QA system from Edinburgh University performed quite well in the list question task (Kaisser et al., 2006) and two new and complementary approaches were used. In the first approach, FrameNet (Baker et al., 1998), PropBank

(Palmer et al., 2005) and VerbNet (Schuler, 2005) were used to generate answer templates to retrieve potential answers in the documents. The second approach matched the semantic role labeling (PropBank) paths between a question and a potential answer.

The QA system from Fudan University (Zhou et al., 2006) used the same factoid question system as previous year. For list questions, the improvement was made in separate modules such as answer type classification, document retrieval, answer ranking and answer filtering. The strategy for answering definition questions still relied on external Web knowledge bases.

Vechtomova and Karamuftuoglu (2006) identified the lexical bond between sentences in order to find the answers talking about the relation between two entities. Two sentence ranking algorithms were compared in the initial run. The first algorithm considered the number of facets the sentences contained, the number of question words the sentences had and the number of lexical bonds in the sentences. While the second algorithm considered the average inverse document frequency score of the returned nuggets instead of the lexical bonds. After receiving feedback from the assessors, the system filtered the nuggets based on the nuggets the assessors selected or the keywords in the nuggets, and produced two submissions.

The automatic QA system from MIT performed quite well in the complex relation task during the last two years (Katz et al., 2006). After removing irrelevant words from the narrative part in the question analysis step, the documents were retrieved and ranked based on the matching of the selected keywords and synonym groups from lexical resources. Their system was tuned

by making use of pervious year's data.

2.2.2.9 TREC-16 QA

The TREC 2007 QA track still consisted of a main task and a complex question task piloted in Dang et al. (2007). The main task repeated the question series format but the document collection for answer retrieval changed to a mixture of news and blogs. This required the QA systems to deal with some poorly formed and noisy blog texts which were not as authentic as news collection. The complex relation question task remained the same as previous year.

Moldovan et al. (2007) performed the best in the main task with some improvement. A better name entity recognizer was employed to discover finer-grain types of name entities. Language models were built for each question class in order to rank the potential answers. For list questions, Wikipedia was used as an authentic resource to improve the coverage of answer types. Moreover the logic prover was added to select answers for list questions. For answering definition questions, the robustness of the nuggets extraction module was further improved by finer-grain answer patterns.

The QA system from Fudan University detailed in Qiu et al. (2007) performed quite well in the definition question task. The retrieved sentences were ranked based on three groups of features. The first group was based on language modeling approaches (Han et al., 2006), the second was based on dependency triples and the last one was based on the scores from the IR engine.

In the initial run of the complex relation question task, Zhang et al. (2007a)

used two approaches to rank candidate sentences. One heuristic approach was based on Vechtomova and Karamuftuoglu (2006) which was detailed in the previous section. And a machine learning approach treated answer quality prediction as a binary classification problem. In the second run, both of the submissions were automatic. One submission used the terms in the selected sentences to update query terms in order to re-rank the sentences. Another submission selected the sentences directly based on the selected terms.

The baseline system from MacKinnon and Vechtomova (2007) was a modified version of Vechtomova and Karamuftuoglu (2006) in the complex relation question task. Another run used synonyms from Wikipedia but the performance got no improvement. Interestingly, the second run submissions got no improvement from human assessors.

2.2.3 Architecture of TREC-style Question Answering Systems

As discussed in the previous section, most QA systems that can answer TREC-style questions (such as factoid, list and definition questions) are usually composed of three core components: the question processing (or query formation) component, the candidate document retrieval component and the answer selection component. The question processing component analyzes the input question and usually decomposes it into several queries. The candidate retrieval component queries multiple resources. The answer selection component analyzes the results from the candidate document retrieval component and usually extracts a ranked list of answers which sometimes go through a

series of validation. OpenEphyra, an open source release of the QA system Ephyra (Schlaefer et al., 2007), was used in the our question disambiguation work in an automatic QA environment, discussed in the next chapter.

2.2.3.1 Question Processing (or Query Formation)

The question processing component usually involves question type classification and query formation. As question type is crucial for query formation and answer extraction strategy selection, a question is first classified into one of predefined types such as person, location and time, quality and organization. Query formation is an important step for document retrieval involving keywords generation and expansion (Burger et al., 2001a). Most systems appended synonyms, alternate keywords and even related concepts to the original keywords in query expansion in order to improve recall in IR (Hickl et al., 2006b; Vechtomova and Karamuftuoglu, 2006; MacKinnon and Vechtomova, 2007).

2.2.3.2 Candidate Document Retrieval

The candidate retrieval component uses the question queries and retrieves a set of documents from unstructured knowledge sources (i.e. Search Engine, Corpus) and semi-structured knowledge sources (i.e. Database). High-quality document retrieval is helpful even using comparatively simple answer extraction techniques. TREC QA tracks retrieved documents from news collections and added blog data in TREC 2007. Documents are usually pre-processed using stop-word removal, term stemming and indexing. The open source LEMUR[1]

[1]http://www.lemurproject.org

toolkit, Indri[2] search engine and LUCENE[3] IR engine has been widely used. Web based QA systems discover answers from the results returned by commercial web search engines such as Google and Yahoo!.

2.2.3.3 Answer Selection

The answer selection component extracts snippets from relevant documents and generates a ranked list of answers. The candidate documents are processed by NLP techniques such as stop word removal, named entity recognition, Part-of-speech tagging, chunking and parsing to generate syntactic and semantic features (Voorhees, 2002; Chu-Carroll et al., 2003; Harabagiu et al., 2005; Kaisser et al., 2006). Based on the history of TREC QA tracks, we can see that questions of different types are usually treated separately. For factoid questions, name entities or short snippets are extracted and ranked based on confidence scores. While for questions of other types, the retrieved sentences or snippets are often ranked based on their syntactic or semantic similarity to the question. For complex questions, the answers are fused or summarized from multiple candidates.

2.2.4 Relevance in IR and Question Answering

A quite important concept of IR and QA is "relevance". Manning et al. (2008) described "relevance" as around the notion of relevant and nonrelecant documents with respect to the user's information need. They claimed that "relevance is assessed relative to an information need, not a query". A retrieved

[2]www.lemurproject.org/indri

[3]http://www.lucene.apache.org/

Category	Relation
Semantic Relevance	Considers how questions and answers are related through their meaning
Goal-directed Relevance	Considers the informational goals associated with a question and its associated answer, both in the mind of the questioner and the answerer
Logical Relevance	Considers the relationship that exists between the unknown information a question is asking about and an answer in virtue of the way we reason about answers in relation to questions
Morphic Relevance	Considers the way the answer is expressed, i.e. the outer form of the answer

Table 2.1: Relevance categories from De Boni (2004)

document is usually judged as either relevant or nonrelevant based on the gold standard or the ground truth predefined in an IR system.

De Boni (2004) did a survey on the different types of relevance in QA. In QA, "relevance" was defined as "*the concept which expresses the worthiness of an answer in relation to a previously asked question, a questioner and an answerer*" by De Boni (2004), and was further divided into several categories shown in Table 2.1 which was also viewed as the explanation for "*worthiness*".

2.3 Question Disambiguation

2.3.1 Interactive Question Answering

As the information needs of people are sometimes too complex to be formulated as one or more single questions, there is general agreement that QA systems could benefit greatly from providing a means of user interaction up to the actual dialogue capabilities (Burger et al., 2001b). QA systems may fail to provide the desired answer due to the failure of modeling the context of the question or the ambiguity of the question, interactive clarification can also help resolve question ambiguity (Hori et al., 2003) and (Ferrucci et al., 2009).

Unlike traditional QA applications, interactive QA systems must do more than cooperatively answer a single user question. Interactive QA systems need to understand a user's information need and model what a user wants to know over the course of a QA dialogue. Systems that fail to represent the user's knowledge base run the risk of returning redundant information, while systems that do not model a user's intent can end up returning irrelevant information (Hickl et al., 2006a). Some systems choose to provide the user with access to new types of information that is in some way relevant to the user's stated and unstated information needs.

FERRET (Hickl et al., 2006a) built an interactive QA system for real-world environments. FERRET provided answers to the user's questions from an QA system. In addition, FERRET also provided question-answer pairs (QUABs) that were either generated automatically or selected from a large database manually created offline. When presented with a set of QUABs, users were able to select a coherent set of follow-on questions to support or clarify their in-

formation needs. The dialogue fragment in Table 2.2 from Hickl et al. (2006a) provides an example of the kinds of follow-up questions.

Small et al. (2004) proposed to use event frames for their interactive QA system HITIQA in order to align the understanding of the question between the system and the user. The system mainly dealt with analytical questions such as "*ANALYST: What is the history of the nuclear arms program between Russia and Iraq? HITIQA: Do you want to see material on cooperation on development of weapons between Iraq and Russia?*". The dialogue was used to help solve the mismatch or conflict between the the user's question and the system's understanding. Details of this system are shown in Table 2.3 from Small et al. (2004). This was a way to give the user additional information about a domain that they were interested in, but only had a little knowledge about. As the automatic generation of event frames has been improved in recent research, this method will achieve enhanced results.

De Boni and Manandhar (2005) were inspired by the idea of follow-up questions. Instead of a single question, a user was able to pose a series of questions, and the system was able to handle anaphoric references or ellipses in the series of questions. As introduced in Section 2.2.2, follow-up questions were also included in the TREC 2001 competition as a context questions task.

As specified in Section 1.1.2, it is often the case that the user's question is too ambiguous for the systems to interpret and then provide the desired answer. In the past, there were a few methods proposed to generate disambiguation questions in an interactive QA system. In the HITIQA system (Small et al., 2004) which has been introduced above, the three kinds of dialogue derived

	UserQ1:	What has been the impact of job outsourcing programs on India's relationship with the U.S.?
	QUAB1:	How could India respond to U.S. efforts to limit job outsourcing?
	QUAB2:	Besides India, what other countries are popular destinations for outsourcing?
	UserQ2:	What industries are outsourcing jobs to India?
	QUAB3:	Which U.S. technology companies have opened customer service departments in India?
(h)	QUAB4:	Will Dell follow through on outsourcing technical support jobs to India?
	QUAB5:	Why do U.S. companies find India an attractive destination for outsourcing?
	UserQ3:	What anti-outsourcing legislation has been considered in the U.S.?
	QUAB6:	Which Indiana legislator introduced a bill that would make it illegal to outsource Indiana jobs?
	QUAB7:	What U.S. Senators have come out against anti-outsourcing legislation?

Table 2.2: Dialogue fragment from Hickl et al. (2006a)

1. The first question is submitted to the search engine, and 50 documents are returned. An answer pool is obtained after the documents are broken into passages. A combination of hierarchical clustering and classification is applied.

2. If the system discovers that there is only one topic in the answer pool, the system assumes that it has got the answer. But if several topics are found, clarification is used. The next step is to measure how well each of the aspects within the answer space matches up against the original question.

3. Three kinds of dialogue Narrowing, Expanding and Fact Seeking will be derived from the matching results (on-target, near-misses and outliers). The information asked by the system will be based on the missing part in the topic frames.

Table 2.3: Details of HITIQA algorithm from Small et al. (2004)

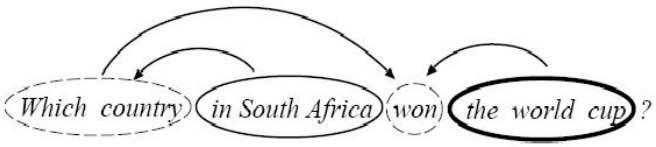

Figure 2.1: Dependency structure example from Hori et al. (2003)

from the matching results (on-target, near-misses and outliers) can be viewed as a type of disambiguation questions.

Another method was proposed by Hori et al. (2003), requiring additional information from the user. The lacking information was supplemented by the user's feedback and the correct answers were able to be distinguished. This method focused on the linguistic features of a question and made use of the dependency tree structure of the question sentence. The ambiguity of each phrase in the question was measured by structural ambiguity and a generality score for each phrase. A phrase that was not modified by other phrases was considered to be highly ambiguous, because less restriction on the information for the phrase was gained by the system. For the question "*Which country in South Africa won the world cup?*", Figure 2.1 shows the corresponding dependency structure, in which the question is separated into phrases. The system discovered that no phrase modified "*the world cup*" phrase, so the algorithm resulted in finding that this phrase had the highest ambiguity and the system generated questions about this phrase: "*What kind of world cup?*" or "*What year was the world cup held?*" For each generated question, a score combining both trigrams frequency features and unigrams features was calculated.

However, we can see that there are several problems or limitations with both

methods for generating disambiguation questions. The HITIQA system relied on manually constructed frames, so it was still only used in restricted domains at the moment. The questions asked by the system relied on the missing part of the frame. The problem with Hori et al. (2003) was that the ambiguity was only based on linguistic information about the user's question; no domain knowledge or analysis of the answer documents was used. In the above example "*Which country in South Africa won the world cup?*", the answers could belong to topic "*rugby*" or "*football*" due to the lack of information about "*world cup*". Although Hori's method could ask the question "*What kind of world cup?*", the template for this question "*What kind of __?*" was manually constructed.

2.3.2 Concept Clustering

Generating topic hierarchies from the retrieved documents has been applied to many IR applications. Lawrie and Croft (2003) proposed a method for automatically generating topic hierarchies from small collections of text. They then applied this technique to summarizing documents retrieved by a search engine. They used statistical methods to find the terms whose scores were quite high, to represent the topics. Chuang and Chien (2004) used a hierarchical clustering algorithm to generate well-formed topic hierarchy from retrieved text segments. As specified in Lawrie and Croft (2003) and Chuang and Chien (2004), the topic hierarchy is a description of the text, which summaries the text and provides a way to navigate through it and also improves the recall of the system.

As far as our knowledge, no research has been done on generating disambigua-

Rank 19: Books with Pictures From Space (Science U) ... of Our Cosmos, by Simon Goodwin A gallery of the most significant photographs taken by the Hubble telescope explains what Hubble's achievements can ...

Rank 34: Amazon. COM: buying info: Hubble's Universe: A Portrait of Our ... ... Ingram A gallery of the most significant photographs of space as taken by the Hubble telescope explains what Hubble's achievements can tell us about the ...

Rank 63: ESA Portal - Press Releases - HST's 10th anniversary, ESA and ... ... A public conference will take place in the afternoon to celebrate Hubble's achievements midway through its ... Notes for editors. The Hubble Space Telescope ...

Rank 100: FirstScience.com - The Hubble Decade ... astronauts' first view of the Earth from the Moon - and the Hubble Space Telescope's ... View from the top. On the scientific front, Hubble's achievements ...

Rank 111: The Hindu : Discoverer of expanding universe ... Hubble's achievements were recognized during his lifetime by the many honours conferred upon him In 1948 he was elected an ... 'The Hubble Space Telescope ...

Rank 140: HubbleSite- Science ... farther and sharper than any optical/ultraviolet/infrared telescope ... very specific goal (like the Cosmic Background Explorer), Hubble's achievement

Table 2.4: Hierarchy generated from snippets from Lawrie and Croft (2003)

tion questions based on hierarchical summaries or clustering information in the answer documents. The results from the CBC concept clustering (Pantel, 2003) and google's open-domain class acquisition (Pasca and Durme, 2008) will be used in our question disambiguation work in the next chapter. And we call these results "concept clusters" in our work, such as the cluster {blue, pink, red, yellow} with the label "color" (two-level topic hierarchy). The details of related research work is discussed next.

2.3.2.1 CBC Clustering Algorithm

The Clustering by Committee (CBC) algorithm (Pantel, 2003) was motivated by the goal of automatically extracting concepts and clusters of words from large databases. Broad-coverage lexical resources such as WordNet have been proved to be useful in many IR applications, such as word sense disambiguation (Leacock et al., 1998) and QA (Pasca and Harabagiu, 2001). But the coverage of such resources is quite small and sometimes these resources include the rare meaning of a word. One of the solutions for such problems is to use a clustering algorithm to automatically cluster words and from them discover semantic classes. In CBC, a number of subsets of elements were chosen to form committees which determined how the other elements were clustered later. The centroid of each cluster was the average features among the members of the committee.

The CBC algorithm consisted of three phases. In Phase 1, each element's top-k similar elements was computed using the top ranking features of all the elements. The dependency features of each word was calculated first for the word sense clustering task. Each word was then represented by a vector of

dependency frequency relationships. In Phase 2, a collection of tight clusters was collected using the top-k similar elements from Phase 1. The elements of each cluster formed a committee. The algorithm tried to form as many committees as possible, but different clusters were quite different from each other. In Phrase 3, each element was assigned to its most similar committee.

In Phrase 3, there were two versions, depending on the task. In the hard-clustering version, every element was assigned to the cluster containing the committee to which it was most similar. In this version every element was assigned to its closest centroid just like K-means. But unlike K-means, the number of clusters was not fixed and the centroids did not change (i.e. when an element was added to a cluster, it was not added to the committee of that cluster). This version could be applied to the document clustering application. Term frequency vectors were used to represent the documents. Then each of the centroids found in the CBC clustering algorithm was a set of terms which represent that committee (cluster). The second version was called soft-clustering. Each element was assigned to multiple senses (committees).

The second version of CBC was applied to word sense clustering. The dependency features of each word were calculated first. Then each word was represented by a vector of mutual information about the dependency frequency features. For example, "apple" used features like: {"V_release:subj:N_apple" 90 times, "computer:nn:apple" 20 times,...} (the other features of "apple" is presented in Appendix A). After assigning an element to a cluster, CBC algorithm removed any overlapping features from the element's feature vector before assigning it to another cluster. This allowed CBC to discover the less frequent senses of a word and to avoid duplicate senses.

2.3.2.2 Cluster Labeling

An quite important issue of clustering is to label or name the clusters, different algorithms based on different theories have been proposed.

Popescul and Ungar (2000) compared the χ^2 test with frequent and predictive words methods. If a word was equally likely to appear in the children nodes of the current node, this word was a more general word. Glover et al. (2002) used an information gain based approach to label clusters of web pages. The information gain of each feature (phrases or n-grams) was calculated in order to decide which feature could best separate a cluster from random documents. The "anchor text" information obtained quite good results using this method from the web achieves. Stein and Eissen (2004) proposed an ontology based approach which minimized the difference between the hierarchical results and a useful ontology. More specifically, if a unique matching was found between the categorization (constructed by a clustering algorithm for a set of documents) and the well-formed ontology, the categorization was labeled by the root of the matched ontology. Treeratpituk and Callan (2006) proposed an algorithm to distinguish general labels for parents from specific labels for children in hierarchical cluster labeling. This algorithm was inspired by the idea that "*a good descriptor for a cluster should not only indicate the main concept of the cluster, but also differentiate the cluster from its sibling and parent clusters*". And the hypothesis for this algorithm was similar to Popescul and Ungar (2000): the words for the parent and words in the child clusters had different distributions.

Following the previous work on CBC algorithm, Pantel and Ravichandran

(2004) proposed a semantic class labeling algorithm which tried to find context words to label the semantic classes. The key idea behind this method was to use features to get a list of context terms around the committees (classes), and these list of terms tended to represent the committees discovered. For example, "fruit" could be discovered to be the semantic class label for {apple, orange, banana} and "race" for {Preakness Stakes, Preakness, Belmont S-takes}. As described in (Pantel and Ravichandran, 2004), the top 5 features were:

1. Apposition (N:appo:N) e.g. ... Oracle, a company known for its progressive employment policies, ...
2. Nominal subject (-N:subj:N) e.g. ... Apple was a hot young company, with Steve Jobs in charge.
3. Such as (-N:such as:N) e.g. ... companies such as IBM must be weary ...
4. Like (-N:like:N) e.g. ... companies like Sun Microsystems do no shy away from such challenges, ...

Pasca and Durme (2008) implemented a new algorithm for generating class labels together with the set of class instances from the web. As described in Pasca and Durme (2008), the main features of the algorithm were:

1. It is an integrated method which enables the simultaneous extraction of class instances, associated labels and attributes; whereas, the methods introduced above are divided into separate clustering and labeling phases.
2. This method makes use of the Web Internet and queries from Google

users as a resource to acquire thousands of open-domain classes covering a wide range of topics and domains.

3. "*IS-A*" relationships patterns are used in order to discover the class label and class instance pairs.

2.4 Community Question Answering Resources

As introduced in Section 1.1.3, CQA websites, such as Yahoo! Answers, WikiAnswers, Naver and Baidu Zhidao, have grown quite fast during the past several years. CQA has become a new type of QA service, which is also called "social question answering" in some researcher's work such as Shah et al. (2009) and Gazan (2011). As specified in Liu and Agichtein (2011), a key to the success of CQA systems is to "*provide askers with efficient and helpful service in order to minimize the response latency in searching the CQA repository*".

The large amount of question and answer pairs from these CQA websites provide valuable resources for the research on QA and user behaviors in a new setting. How to efficiently make use of them has become an increasingly important research issue for the whole community. In the past, related research mainly involved discovering high quality content (questions, answers and users) and similar and relevant content retrieval.

2.4.1 Discovering High Quality Content

Although the users can earn credit if their answers were voted as the best answers, there are still some users making fun of the asker and irrelevant advertisements posted as the answers. The users' votes are sometimes unreliable,

which involve unconscious and malicious spam attacks the detailed of which can be found Bian et al. (2008a).

Jeon et al. (2006) used the maximum entropy approach and kernel density estimation to predict answer quality. Without using textual information from the question and answers, non-textual features such as answerer's acceptance ratio, answer length, questioner's self evaluation, answerer's activity level and answerer's category specialty were shown highly correlated with an answer's quality.

Zhang et al. (2007b) used link analysis algorithms to discover community experts. Jurczyk and Agichtein (2007) extended their work to discover authentic users under specific categories in a web-scale setting.

Agichtein et al. (2008) extended the work from Jeon et al. (2006) to find high quality question and answer content from Yahoo! Answers. More specifically, a set of classification algorithms were compared by combining contributor relationship feature with user's usage-based features and textual features. According to the χ^2 test for all the features, the most significant ones included answer length, the number of words in the answer with a corpus frequency larger than c, the number of "thumbs up" minus "thumbs down" received by the answerer (divided by the total number of "thumbs"s received) and the entropy of the trigram character-level model of the answer.

In order to achieve robust ranking which is resilient to various ways of vote spams, Bian et al. (2008a) studied the behaviors of vote spam attacks and proposed several approaches using three sets of features including content features

(i.e. the similarity between the query and the question), community features (i.e. history of the answerer and the number of resolved questions by the answerer) and user votes. They found the previous two were more important than the last one.

In order to improve the quality of answers in the QA community, Liu et al. (2010) explored role of browsing context in the answerers behavior in a CQA service. And the experiment showed that "*the relevant web browsing context can have significant positive effects on the answerers reported ability, effort, and willingness to answer questions*". And Liu and Agichtein (2011) explored different contextual features which influenced the answerer behaviour: the temporal activity patterns, textual information and the question's ranking position in the list. They proposed an approach to recommend the unanswered questions to right answerer in the community based on the effective features.

2.4.2 CQA Data Retrieval

There are lots of redundant information accumulated in all the CQA services. The duplicated questions are generated because some users post their questions without carefully searching existing collections or the "lexical chasm gap" between questions are quite hard to bridge as specified in Section 1.1.3. In the past, researchers proposed different approaches for retrieving questions, answers and other contents from the CQA repository.

2.4.2.1 Relevance

Some previous work focused on modeling the relevance between the query (sometimes questions) and contents in the CQA archives (Song et al., 2008). The related work can be viewed in the following paragraphs.

For automatically collecting similar questions, Jeon et al. (2005a) and Jeon et al. (2005b) compared: vector space model (TFIDF); the negative KL divergence between the language models; the output score of the query likelihood model; and, the score of the Okapi model based on the similarity between their answers. By applying this technique to many different question and answer collections, a large number of similar question pairs were gathered.

As described in the previous section, Jeon et al. (2006) proposed an answer quality measurement approach while other works only considered relevance factors. Their method was able to be further integrated into the retrieval model and shown to improve the performance of retrieving question and answer pairs.

A machine translation based model was trained using these collections to improve the performance of a question search system by Riezler et al. (2007). They developed two query expansion approaches in answer retrieval based on Statistical Machine Translation technologies.

Wu et al. (2008) presented an incremental automatic question recommendation[4] framework based on probabilistic latent semantic analysis. Question recommendation in their work considered both user interests and feedback.

[4]We will provide the definitions for question recommendation in Section 4.1, which is different from Wu et al. (2008) and Duan et al. (2008).

Duan et al. (2008) tackled the question recommendation problem in two steps, firstly, they represented questions as graphs of topic terms by making use of an MDL (Minimum Description Length) based Tree Cut Model; secondly, they ranked the recommended questions on the basis of the graphs.

Bian et al. (2008b) proposed an approach to improve the accuracy of retrieving well-formed, factual answers by studying the behaviors of vote spam attacks and incorporated both textual and non-textual features in the classification model.

Wang et al. (2009a) proposed a tree kernel framework to find similar questions in the CQA archive based on syntactic tree structures. Wang and Chua (2010) further mined lexical and syntactic features to detect question sentences in C-QA data.

Zhou et al. (2011) utilized a topic enhanced Translation-based Language Model (TopicTRLM) which combined lexical knowledge and latent semantic knowledge to measure relatedness between questions. The suggested questions can be semantically related to the queried question and can explore different aspects of a topic, tailored to users' information needs.

In order to make further use of the CQA data, we think a well study of the relations between all the questions is needed. Bernhard and Gurevych (2008) proposed an approach to retrieve paraphrased questions by treating the manually tagged "reformulated" (paraphrasing) questions in WikiAnswers as the gold standards. They claimed that "it is feasible to answer learners' questions

by retrieving question paraphrases from social QA sites". However, we find their approach to be problematic, as only a small number of the goad standard pairs are paraphrased questions detailed in Section 5. The other pairs are only "related" questions. As seen in Bian et al. (2008b), "related" questions are the ones which can be retrieved by the search engines provided by CQA services and usually share some common words with the reference (query) question. We will also show that some of these manually tagged "reformulated" questions in WikiAnswers are about different topics and should be eliminated in Chapter 5.

2.4.2.2 Usefulness

Song et al. (2008) proposed the notion of "question utility" for studying the usefulness of questions to improve question retrieval. Question utility was defined as "*the possibility that a question would be repeated by other people*". In order o measure question utility, a language model and a method based on the LexRank algorithm were examined. Finally, the relevance score and utility score regarding the question utility were combined into a log linear model for the task of question recommendation. Bunescu and Huang (2010b) also proposed to study the question usefulness problem, and "*question usefulness*" was defined as "*a question is deemed useful if its expected answer may overlap in information content with the expected answer of the reference question*".

2.4.2.3 Asker and searcher satisfaction

Some research work focused on "asker satisfaction" and "search satisfaction", such as in Liu and Agichtein (2011). CQA data can be reused to satisfy an askers information need, based on effective retrieval of relevant questions and

answers to the information need (Liu et al., 2008).

The question of whether we can predict if an asker in CQA will be satisfied with the answers proposed by the community has been frequently asked. Liu et al. (2008) first attempted to quantify and predict asker satisfaction in CQA services. The asker satisfaction problem was formally defined as "*a question submitted by an asker in CQA, predict whether the user will be satisfied with the answers contributed by the community*". Predicting the asker satisfaction was deemed highly personal, difficult and subjective. They studied this problem under a classification framework. Given a question thread posted by an asker, the features derived for the prediction included question related (title, description, posting time, and rating feedbacks), question and answer similarity, asker and answerer history and category information. The preliminary results showed that the asker's prior history had a big influence on predicting their satisfaction. The authors claimed that "their work can be viewed as opening a promising direction towards modeling personalized user's intent, expectation and satisfaction in Community QA".

Liu and Agichtein (2008) proposed a personalized models of asker satisfaction to predict whether "a particular question author will be satisfied with the answers contributed by the community participants". They also followed a classification framework and explored different sets of features such as "Question Features", "Question-Answer Relationship Features", "Asker User History", "Category Features" and "Text Features". Different from the work of Liu et al. (2008), two personalized approaches were compared. In the first personalized approach a separate classifier was trained for each user, while the second approach trained a separate classifier for each group of users.

The above work on CQA focused on the asker satisfaction while Liu et al. (2011) focused on search satisfaction which was defined as: "*given a search query S, a question Q, and an answer A originally posted in response to Q on a CQA site, predict whether A satisfies the query S*". They compared several search satisfaction prediction methods by making use of three sets of features: query clarity, query to question match, and answer quality. The authors claimed their work as "the first attempt to predict and validate the usefulness of CQA archives for external searchers, rather than for the original askers". However our work on information need satisfaction in Li and Manandhar (2011) is independent from the work of Liu et al. (2011). We treated the user's information need satisfaction as a usefulness ranking problem in Chapter 5 while not trying to classify the retrieved contents as relevant or nonrelevant.

Chapter 3

Question Disambiguation for Understanding the User's Information Need

3.1 Introduction

As demonstrated by Burger et al. (2001a) and Unger and Cimiano (2011), question ambiguity problem poses a big challenge for a QA system to bridge the information need gap between the system and the user. We discussed this problem in Section 1.1.2. In a thorough analysis of a few QA tasks by Ferrucci et al. (2009), the ambiguity challenge is associated with all the datasets assessed, especially with more complex questions.

In this chapter, inspired by the observation that QA systems would provide multiple answers for an ambiguous question, we analyze the information need behind ambiguous questions from the concept clusters found in the answer documents. And thus we propose to resolve question ambiguity by generating

clarification questions based on these concept clusters. For further improving the quality of the concept clusters, an efficient and effective feature selection algorithm is compared with state-of-the-art methods. This chapter is therefore divided into two parts:

1. Clarification Question Generation Based on Concept Clusters
2. Improving Concept Clusters by Feature Selection

3.2 Clarification Question Generation Based on Concept Clusters

3.2.1 Introduction

In a realistic Interactive QA (IQA) situation, one third of the users pose follow-up questions, that is, they go beyond a single question per dialogue (Kirschner and Bernardi, 2007). We show that interactive clarification dialogue can be successfully employed to clarify a user's information needs and help reformulate the question in order to efficiently find the correct answer. We will use the term "clarification question" to describe such disambiguation or clarification follow-up questions.

During the past few years, clarification IQA systems have received a lot of interests from NLP researchers. De Boni and Manandhar (2003) proposed a clarification dialogue recognition algorithm, through the analysis of collected data on clarification dialogues. They also examined the importance of clarification dialogue recognition for QA. Kirschner and Bernardi (2007) analyzed

informational transitions and the context dependency of interactive dialogue through the study of discourse phenomena. The goal is to enable us to predict the topics that users of an IQA system are likely to ask about next.

It has been discovered that due to the ambiguity and vagueness in user questions, as well as redundancy, variability, and possible contradictory information in the data, it is common for a question by a casual user to have multiple answers from a QA system (Dalmas and Webber, 2007). For example, Burger et al. (2001a) specified that the desirable answers for "*Where is the Taj Mahal?*" should look like the following:

> If you are interested in the Indian landmark, it is in Agra, India; if instead you want to find the location of the Casino, it is in Atlantic City, NJ, USA; there are also several restaurants named Taj Mahal. A full list is rendered by the following hypertable. If you click on the location, you may find the address.

If a QA system generates a list of all potential answers: "the city of Agra", "Atlantic City, NJ", "Mumbai" for the question, it is difficult for it to generate the desirable answer given above (Burger et al., 2001a). However, we can conclude that the answers (i.e. "the city of Agra", "Atlantic City, NJ" and"Mumbai") for a specific question tend to fall into distinctive but equivalent classes[1].

[1] "distinctive but equivalent classes" means all these answers belong to the same answer class (e.g. Location, Person). But answers are distinctive from each other, and there is not hyper-relation between any two of them.

We also found that the topics of clarification questions can be found in the context of the answers, based on their separability. For example, taking the above question, each answer of "the city of Agra", "Atlantic City, NJ" and "Mumbai" is associated with one of the subtopics "Palace", "Casino" and "Hotel" from the concept cluster "Building" in the context. Therefore, we can generate topics (i.e. type of building) for clarification questions based on this concept cluster. However, we show that only the topics that maximize the separation of distinctive equivalent classes of answers are good quality ones. In other words, we want to generate an information-seeking question whose topic makes each answer distinct from others. Thus, the new answers for this question can reduce the ambiguity and vagueness in the original question.

Besides ambiguous questions, we also use list questions (some researcher use "multiple answer questions") from TREC dataset to verify the effectiveness of our approach as QA systems generate a list of answers for both kinds. Our approach of generating concept clusters can also be viewed as the production of "class attributes" associated with different classes of objects, which is similar to work of Pasca and Durme (2008). The difference is that our approach gives higher ranking scores for the "class attributes" which better characterize (distinguish) different objects of a class.

The rest of the section is organised as follows. In Section 3.2.2, related research work is examined. In Section 3.2.3, we present our approach for discovering the context topics that cause the answers to fall into such distinctive classes and maximize the class separation between them. In Section 3.2.4, we empirically verify the effectiveness of our approach. Section 3.2.5 concludes by summarizing our work and discussing future directions.

3.2.2 Related Work

The several methods for generating the questions topic in an clarification dialogue have been detailed in Section 2.3.1. However, none of the methods considered the context of the list of answers in the documents returned by QA systems. Partially inspired by the observation that multiple answers are likely to fall into distinctive equivalent classes, we propose a method to discover the context topics that cause the answers to fall into distinctive equivalent classes and maximize the classification (separation) of the answers. Our system can then generate clarification questions based on these topics and manually constructed questions templates.

3.2.3 Topic Generation Based on Concept Clusters

3.2.3.1 Organizing answers

To make the answers fall into distinctive equivalent classes, we make use of the Open Directory Project[2] which is the largest, most comprehensive human-edited directory on the Web. There are several hierarchical structures for a specific noun phrase, as the phrase may have multiple meanings in different document contexts. After obtaining multiple answers from QA systems, we can organize them hierarchically. In this way we can not only obtain a group of distinctive equivalent classes of answers, but also gain useful information from the category information. Table 3.1 shows several answers for "*Which country won the world cup?*", and the answers are classified according to the

[2]www.dmoz.org

South Africa	Regional: Africa: South Africa
Australia	Regional: Oceania: Australia
Brazil	Regional: South America: Brazil
England	Regional: Europe: United Kingdom

Table 3.1: World cup winners

classes in Open Directory's category information.

3.2.3.2 Topic generation based on concept clustering

We make use of concept clustering techniques to find the topics in the answer documents. The topics are based on the labels of the concept clusters. Because we are dealing with open domain questions, the desirable concept clusters should have several characteristics: (a) one cluster represents one topic and the instances in this cluster represents different subtopics; (b) broad-coverage, as we apply the concept clusters to open-domain questions, which cover a wide range of topics; (c) coarse grained, as we found that casual users find coarse grained categories more useful to specify their information needs.

Recent research on automatically extracting concepts and clusters of words from large databases makes it feasible to be applied on a big set of concept clusters. Clustering by Committee (CBC) (Pantel, 2003) made use of the fact that words in the same cluster tend to appear in similar contexts. Pasca and Durme (2008) used Google logs and lexico-syntactic patterns to obtain clusters and labels simultaneously. Google also released Google Sets, which can be used to grow concept clusters of different sizes. We developed a system

to retrieve such concept clusters and labeled them using NLP patterns as described in Pasca and Durme (2008): <[..] "Class Label" [such as|including] "Class Instance" [and|,|.]>.

In our approach we combine the CBC algorithm, Google Sets and NLP patterns to generate our concept clusters. Some examples are shown in Table 3.2. We can see that similar clusters with different labels can be extracted by different algorithms.

3.2.3.3 Concept cluster document vector space modeling

In the statistically based vector-space model, a document is conceptually represented by a vector of keywords extracted from the document, with associated weights representing the importance of the keywords in the document and within the whole document collection. A document D_j in the collection is represented as $\{W_{0j}, W_{1j}, ..., W_{nj}\}$, and W_{ij} is the weight of word i in document j. Here we use our concept clusters to create concept cluster vectors. We define the concept clusters as $\{C_1, C_2, ..., C_n\}$. C_i=$\{e_{i1}, e_{i2}, ..., e_{im}\}$, e_{ij} is j^{th} subtopic of cluster C_i and m is the size of C_i. A document D_j is now represented as $<WC_{1j}, WC_{2j}, ..., WC_{nj}>$, and WC_{ij} is the score vector of document D_j for concept cluster C_i:

$$WC_{ij} = <Score_j(e_{i1}), Score_j(e_{i2}), ...Score_j(e_{im})>$$

$Score_j(e_{ip})$ is the weight of subtopic e_{ip} in document D_j.

An example of our concept cluster vector is shown in Table 3.3. Note that the

concept clusters we use here are only subsets of the ones in our local storage structure.

3.2.3.4 Concept cluster separability measure

We can view different concept clusters as different groups of features that can be used to classify the answer documents. We would like to choose the cluster (group of features) that maximize the classification of the answers. In other words, the goal of our approach is to compare different groups of features by their separability of the answers. Usually class-separability may be measured by the intraclass distance S_w and the interclass distance S_b. The greater S_b is and the smaller S_w is, the better the separability of the data set. Therefore, the ratio of S_w and S_b can be used to measure the distinctiveness of the classes: the smaller the ratio, the better the separability (Wang et al., 2007). In both of the experiments in Section 3.2.4, we retrieve the answers from Google search snippets for each quesiton, and each snippet is quite short. So we combine all the snippets for one answer into one document. One answer is associated with one document. This means there is only one document in one class and the intraclass distance S_w becomes zero. So we propose another average interclass measure to compare the separability of different groups:

$$Score(C_i) = D\frac{1}{N}\sum_{p<q}^{N} Dis(A_p, A_q),$$

D is the Dimension Penalty score, $D = \frac{1}{M}$,

M is the size of cluster C_i,

N is the combined total number of classes from all the answers

CBC algorithm	C_{1437}(UK County): {Oxfordshire, Buckinghamshire, Cambridgeshire, Bedfordshire, Hertfordshire, Lincolnshire, Suffolk Dorset, Staffordshire, Herefordshire, Devon Shropshire, ...}
	C_{18}(Computer Company): {IBM, Intel, Motorola, Hewlett-Packard, Compaq, Sun, Microsystems, Cisco, Microsoft, Cisco Systems, Oracle, Nortel, Dell, Lucent Technologies, ...}
	C_{46}(MANUFACTURER / CAR): {Mercedes-Benz, BMW, Nissan, Toyota, Peugeot, Audi, Volkswagen, Opel, Mazda, VW, Mitsubishi...}
Google Sets	C_{10007}(Movie genre): {romantic, comic, thriller, comedy, action, drama, romance, horror, adventure, western, crime, fantasy, love, ...}
	C_{10020}(Car Brand): {honda, bmw, ford, toyota, nissan, mitsubishi, mazda, chevrolet, volkswagen, audi, renault, chrysler, hyundai, subaru, volvo, opel, ...}
	C_{10010}(Characteristic): {characteristic, aspect, trait, feature, attribute, element, peculiarity, variation, facet, distinctive feature, Hallmark, nuance, difference, ...}
NLP Patterns	C_{10005}(Dam): {Kelly Barnes Dam, South Fork Dam, Redlands Dam, Deja Vu Dam, Teton Dam, Buffalo Creek Dam, Rainbow Lake Dam, Silver Lake Dam, Toccoa Falls Dam, St. Francis Dam, Walnut Lake Dam,...}
	C_{10021}(Rivers in China): {Jialing, Huangpu River, Wu jiang, Yanglong River, Huanghe, Chang Jiang, ...}
	C_{10101}:(Provinces in China): {Shaanxi, Tangshan, Indian Ocean, Aleppo, Damghan, Gansu, Ardabil, ...}

Table 3.2: Concept clusters

Google Snippets	The Taj Mahal is a prestigious luxury **hotel** located in the Colaba region of Mumbai, India, next to the Gateway of India.
	1 Feb 2009 ... The Trump Taj Mahal **Casino** Resort is located at 1000 Boardwalk , in Atlantic City, New Jersey, in the bustling casino area along the shore.
	The Taj Mahal was built on a parcel of land to the south of the walled city of Agra. Shah Jahan presented Maharajah Jai Singh with a large **palace** in the
Concept Cluster	{hotel, casino, palace}
D_j's Vector	$<Score_j$(hotel), $Score_j$(casino), $Score_j$(palace)$>$

Table 3.3: Concept cluster vector example

$$Dis(A_p, A_q) = \sqrt{\sum_{m=0}^{n} (Score_p(e_{im}) - Score_q(e_{im}))^2}$$

Because we will later treat every cluster as a group of features for classification, we introduce D, the "Dimension Penalty" score which will give a higher penalty to bigger clusters. We use the reciprocal of the size of the cluster. The second part is the average pairwise distance between answers. N is the total number of classes of answers. Below, we describe in detail how to use concept cluster vector modeling and the separability measure to rank clusters.

3.2.3.5 Cluster ranking algorithm

Input: A, D and CS; Θ_1, Θ_2; Q; QS

Output: T = $\{< C_i, Score >\}$, $(C_i \in CS)$; QS

Variables: X, Y;

Steps:

1. CS = CS - QS
2. For each cluster C_i in CS
3. X = No. of answers whose context contains the subtopics from C_i;
4. Y = No. of subtopics from C_i that occur in the answers' context;
5. If X $< \Theta_1$ or Y $< \Theta_2$
6. delete C_i from CS; continue
7. Represent the documents on C_i
8. Calculate the Score(C_i) using our separability measure
9. put $< C_i, Score >$ in T
10. return T the medoid.

Algorithm: Concept Cluster Ranking

In the above algorithm, A is an answers set. A = $\{A_1, A_2, ..., A_p\}$ which is associated with the answers documents set D = $\{D_1, D_2, ..., D_p\}$. CS is a concept cluster set, and CS = $\{C_i \mid$ some of the subtoptics from C_i occurs in D$\}$. Θ_1 and Θ_2 are threshold values. QS is the output concept clusters, and QS = $\{C_i \mid$ some of the subtoptics from C_i occurs in Q$\}$.

The Concept Cluster Ranking Algorithm ranks the concept clusters based on their separability score. Previous to that we applied our concept clusters vector model to documents D of the answers and then retrieved the concept clusters. For document modeling, we use the variant version of TFIDF scheme in Yang and Liu (1999) to calculate the weight of subtopic e from cluster C in document D:

$$Score(e) = \begin{cases} log(tf_{e,D} + 1) log\frac{n}{x_e} & \text{if } tf_{e,D} \geq 1 \\ 0 & \text{otherwise} \end{cases}$$

Where where $tf_{e,D}$ is the frequency of e in D, n is the number of documents and x_e is the number of documents where e occurs.

In the first step of this algorithm we delete all the clusters which are in QS from CS so that we only consider the context clusters. However, in some questions we find the concept clusters in QS can also be useful to distinguish different answers from each other. In addition, we can also present the topic of such clusters to the users. Next, we calculate X (the number of answers whose con-

text contains subtopics from C_i) and Y (the number of subtopics from C_i that occurs in the answers' context). More specifically, we calculate Y by finding the occurrence of the subtopics from C_i in the documents associated with all the answers. We expect the clusters to hold two characteristics: (a) at least occur in Θ_1 answers, as we want to have a cluster whose subtopics are widely distributed in the answers. We set Θ_1 as half the number of answers in both of the experiments in Section 3.2.4; (b) at least have Θ_2 subtopics occurring in the answer documents. If a cluster has the above two characteristics, we view the cluster as a group of features. Next, we use our separability measure to calculate a score for this cluster. The final ranking of all of the clusters is based on this score. The higher a cluster's score the better separability it has.

This means we have selected a cluster which has several subtopics occurring in the answers and the answers are distinguished from each other because they belong to the different subtopics.

3.2.4 Experiment

We conducted experiments to verify the effectiveness of our approach on questions modified from the TREC list questions and an ambiguous questions set.

3.2.4.1 Data set and baseline method

To the best of our knowledge, the only available test data of multiple answer questions are the list question collection from TREC Data. For our first list question collection we randomly select the questions which have at least 3 answers from TREC 2005 and TREC 2006 data. But some of the questions

have ellipsis and reference. So we modified these list questions to factoid ones with additional words from their context questions to eliminate ellipsis and reference. We manually chose the ambiguous questions from TREC data and added the questions discussed as examples in Hori et al. (2003) and Burger et al. (2001b).

Our system takes a question and its corresponding list of answers as input; then retrieves Google snippet documents for each of the answers with respect to the question. We compare two different ways of ranking the clusters: (1) a baseline system using traditional clustering methods such as a vector space model. Similar to these clustering methods our baseline system does not rank the clusters by the above separability score. The preferred cluster occurs in more answers and has more subtopics distributed in the answer documents. If we use X to represent the number of answers whose context contains the subtopics from one cluster and Y to represent the number of subtopics from this cluster that occur in the answer context, we will use X$\times$Y to rank all the concept clusters found in the context. (2) the approach proposed in Section 3.2.3, in which we rank the clusters according to each cluster's separability on the classes of answers.

We manually developed a set of question templates for generating clarification questions based on the concept clusters in the above two systems, such as "Who is [Person]?", "Which year did that happen?" and etc. We made use of four measures for evaluating the ranking methods. They are mean average precision, precision of the top one cluster, precision of the top three clusters and the number of failure cases.

3.2.4.2 Statistical significance test

In order to verify whether some approaches are significantly better than other methods and avoid the cases that some methods by chance perform better, we use a statistical significance test method in Smucker et al. (2007) in our experiment. More specifically, we use the Fisher's Randomization Test, the details of which can be found in Efron and Tibshirani (1994), Box et al. (1978), and Cohen (1995).

Taking the mean of average precisions (MAP) experiment for example, in Table 3.4, our approach outperforms the baseline by 0.19 ($0.603 - 0.413$). we first make a null hypothesis which is "*the baseline method and our propose approach are identical*". And then we randomly choose one approach for a question under this hypothesis. If there are 50 test questions, there are 2^{50} ways (permutations) to apply both approaches for the assessors to evaluate the Mean Average Precision of the results. After we randomly choose 10,000 permutations, we count the times T when the absolute difference between the MAP values are greater than 0.19 during the experiment. The value of $T/10,000$ is called *p-value*, and if *p-value* is less than 0.05 (Smucker et al., 2007) then the null hypothesis is rejected and we can conclude that our approach performs significantly better than the baseline approach (Actually we get $0.01652 < 0.05$ for this experiment). Next we report the statistical test decision directly next to each result, e.g. "significant at $p < 0.05$" or simply "$p < 0.05$".

Methods	MAP	P@1	P@3	#No
Baseline	41.3%	42.1%	27.7%	33.0%
Our Approach	60.3% ($p < 0.05$)	90.0% ($p < 0.05$)	81.3% ($p < 0.05$)	11.0%

Table 3.4: List question results

3.2.4.3 Results on list questions and error analysis

We applied our algorithm to the first collection of questions to verify the performance of our approach and used manual judgement. Two assessors were involved in the manual judgment. For each approach we obtain the top 20 clusters based on their scores. Given a cluster with its subtopics in the context of the answers, an assessor manually labeled each cluster 'good' or 'bad'. If it is labeled 'good', the cluster should be relevant to the question and the cluster label can be used as a dialogue seeking question topic, to distinguish one answer from the others (characterize one answer of the list). Otherwise, the assessor will label a cluster as 'bad'. We use the above two ranking approaches to rank the clusters for each question. Table 3.4 provides the results from the first list question collection. The second column is MAP over the set of clusters. The third column is the precision of the top cluster, while the fourth is the precision of the top three clusters. The last column is the percentage of questions whose top 3 clusters are all labeled 'bad'.

From Table 3.4, we can see that our approach significantly outperforms the baseline approach in MAP by 19%, in P@1 by 47.9% and in P@3 by 54.1%. Our approach can reduce the number of questions whose top 3 clusters generated are all labeled 'bad' by 22%. And we can see that about ten percent of the questions have no 'good' clusters generated. Two examples from this experiment are shown in Table 3.5.

LQ1:	Which professional teams has Warren Moon been a player?
1^{st}	C_{41}(American States):{houston, baltimore, los angeles, ...}
2^{nd}	C_{20}(Football Position):{defensive, receiver, quarterback, ...}
3^{rd}	C_{60}(Sports Career):{leading, pro, footballer, ...}
LQ2:	Which countries were visited by Hugo Chavez?
1^{st}	C_{210}(Leader):{leader, header, spokesman, president, ...}
2^{nd}	C_{210}(Organization):{government, support, union, organization, ...}
3^{rd}	C_{16}(Activity):{buildup, struggle, posturing, role, effort, ...}

Table 3.5: List question examples

After further analysis of the answers documents we find the reasons for generating 'bad' clusters is as following: (1) most of the clusters were automatically extracted by the CBC algorithm and some clusters are quite noisy. The members (subtopics) in a noisy cluster often span a quite wide range of topics. Some of the noisy clusters have quite high separation scores for some of the answers and cause random errors; (2) sometimes the answer documents retrieved from Google for a specific question are quite similar. So that the subtopics extracted for each answer are quite similar; (3) the unstructured Google snippets are quite noisy. It is often the case that some subtopic randomly co-occur with an answer in a retrieved snippet. In the experiment we only look for context words in the Google snippet, while not using any scheme to specify whether there is a relationship between the answer and the subtopics.

3.2.4.4 Results on ambiguous questions and error analysis

To the best of our knowledge, there are no standard ambiguous question collections available for us to evaluate ambiguous questions. Therefore, we manually selected questions from TREC 8, TREC 9, TREC 2003 and articles. However, even using all of these questions, it did not provide enough information to distinguish one answer from the others. Table 3.6 provides the statistics for the performance on this question collection. It shows that our approach significantly outperforms the baseline approach in MAP by 22.5%, in P@1 by 37.9% and in P@3 by 17.1%. Our approach can reduce the number of questions whose top 3 clusters generated are all labeled 'bad' by 17.4%. We provide some results for ambiguous questions in Table 3.7.

Methods	MAP	P@1	P@3	#No
Baseline	31.1%	33.2%	21.8%	47.1%
Our Approach	53.6% ($p < 0.05$)	71.1% ($p < 0.05$)	64.2% ($p < 0.05$)	29.7%

Table 3.6: Ambiguous question results

From Table 3.6, we can see that both of the baseline and our approach perform worse on the ambiguous questions than the list questions with respect to all the measures. After further analysis of the failure cases, we find out that most of the 'bad' clusters for ambiguous questions are generated for the same reasons as the list questions discussed in the previous section. We also find that for some questions the separability scores of all the generated concept clusters are low, and our algorithm fail to provide good clarification question topics. This is because the answers retrieved for each of these ambiguous questions belong to quite different topics (e.g. some are about fruit and others are about animal). In the cases like these, the best disambiguation questions should directly ask the user to choose the topic that may interest him/her.

3.2.5 Summary

Inspired by the fact that answers from QA systems for ambiguous questions usually fall into distinctive but equivalent classes, we propose an approach to generate clarification questions based on the context information in the answers' associated documents. Our approach can help the user clarify his or her information need and bridge the understanding of the user's information need. As explained in section 3.2.4.1, the clarification questions are generated based on the discovered concept clusters and manually constructed templates. Our empirical results show that our approach leads to significant improvement

AQ1:	Where is Taj Mahal?
1^{st}	C_{20}(BUILDING):{casino, resort, palace, hotel, ...}
AQ2:	Which country in Europe won the world cup?
1^{st}	C_{234}(SPORTS):{football, rugby, hockey, ...}
AQ3:	Which country in Europe won the football world cup?
1^{st}	C_{10023}(YEAR):{1994, 1998, 2002, 2006, ...}

Table 3.7: Ambiguous question examples

on the TREC collections and the ambiguous question collections. The contribution of this work can be summarized as: (1) a new concept cluster vector model, so that a document can be represented in different groups of subtopics; (2) a new ranking scheme was developed to rank the context concept clusters for ambiguous questions, according to each cluster's separability. The labels of such clusters will be used as topics for clarification questions later in the research. Finally the approach significantly outperformed the baseline method.

3.3 MDS based Feature Comparison and Comparison for Clustering Concepts

3.3.1 Introduction

NLP clustering and classification tasks often face the challenging research problem of a large number of features. The dimensionality of the feature space is usually very high and features are often extremely sparse. To estimate any parameter, a large number of samples are necessary to achieve a reasonable level of accuracy. Such high dimensional data often contains irrelevant and redundant information that degrades the accuracy of learning algorithms. State-of-the-art classifiers perform poorly on high dimensional datasets as discussed in Fuka and Hanka (2001) and Dhillon and Guan (2003). Feature analysis or feature reduction approaches such as Principle Component Analysis (PCA) and Linear Discriminant Analysis (LDA) suffer from the curse of dimensionality (Charles Bouveyron and Schmid, 2007), because they become computationally more expensive as the feature space grows.

Classification problems require choosing a set of features as a pre-processing step. Feature selection is a process of choosing a subset of the original features, so that the feature space is optimally reduced according to a given evaluation criterion (Yu and Liu, 2003). Feature comparison is the process of using such evaluation criterion to compare features. Most feature selection algorithms are either computationally expensive or require computing a large correlation score matrix.

Supervised feature selection approaches can be divided into two kinds. For

wrapper approaches, the quality of every candidate feature subset is assessed, by investigating the performance of a specific clustering algorithm on this subset, and each candidate subset is obtained by conducting a combinatorial search through the space of all feature subsets (Mladenić, 2006). These algorithms have shown success on low dimensional data. But wrapper approaches tend to be computationally expensive and the performance relies on the chosen learning algorithm (Langley, 1994).

In contrast, filter approaches such as Information Gain (IG), χ^2-test and Mutual Information are more efficient in dealing with the high dimensional data. An extensive analysis of feature selection algorithms is given in Mladenić (2006). The performance of a certain selection method also depends highly on the task. Several feature selection approaches are analyzed for the task of text classification (Li et al., 2009). A revised Mutual Information method is employed to remove noisy features in concept clustering (Pantel, 2003). In most NLP applications, especially when dealing with high dimensional data, the most popular feature selection algorithms are the traditional Information Gain (IG) and χ^2-test (CHI) that have been proven to be superior to other approaches to text classification (Yang and Pedersen, 1997). We propose an effective supervised feature comparison and feature selection algorithm to analyze high dimensional features. Moreover, our algorithm can outperform the above two feature selection approaches in the task of dependency feature selection for concept clustering.

In Section 3.3.2, we briefly describe the related work on dependency feature analysis. In Section 3.3.3 we describe our feature comparison algorithm to compare and analyze high dimensional feature sets. This algorithm is fur-

ther extended for supervised feature selection. In Section 3.3.4, we apply our algorithm to select features for the task of concept clustering and provide comparative evaluation. We also make a thorough analysis of dependency features of different types and path lengths for concept clustering.

3.3.2 Related Work on Analysis of Dependency Features

Dependency features have been shown to be very effective in concept clustering in Lin (1998) and Pantel (2003). But the construction of the dependency feature spaces has been either based on all relations or a fixed subset and there is no quantitative distinction between different relations. In Lin (1998) and Pantel (2003), only direct (path length 1) dependency features are taken into account. But the context information, such as the subject and object of a verb, are ignored. Such longer path dependency features are more accurate and usually incorporate more accurate context information.

Padó and Lapata (2007) conducted a small-scale experiment on synonymy detection and reported very small performance difference between dependency features of different path lengths. Rothenhäusler and Schütze (2009) showed that dependency relation based feature spaces also farely better if the dependency relations used are selected carefully. They chose dependency features beyond the direct dependency path, to conduct clustering experiment on two sets of nouns. The sets of features were compared using the wrapper approach. Only 12 types of features were chosen to be compared, and the performance was based on one purity score using a specific clustering approach. A thorough

analysis of different types of dependency features is missing.

In this section, we develop an effective feature comparison algorithm to make a thorough analysis of dependency features of different path lengths and types. We also note that the method to compare different features proposed in Rothenhäusler and Schütze (2009) is not reliable, because the classification or clustering results are not only determined by the similarity (distance) measures, but also influenced by the chosen classification or clustering algorithms as discussed in Fuka and Hanka (2001), Dhillon and Guan (2003) and Padó and Lapata (2007). Our algorithm can also be extended for high dimensional dependency feature selection.

3.3.3 MDS Based Feature Comparison and Selection Algorithm

Our feature comparison and selection algorithm is based on Multidimensional Scaling (MDS) (Borg and Groenen, 2005), which is used to map the data points to a lower dimensional space. Feature comparison and selection are based on the scatter measure defined.

3.3.3.1 Multidimensional Scaling

MDS takes a similarity matrix or distance matrix as input. If we represent the M objects in n dimensions as $(x_1, ..., x_M)$. D is a similarity or a distance matrix containing Minkowski distance $\|x_i - x_j\|$ for each x_i and x_j in M. The Minkowski distance metric provides a general way to define the distance:

$$d_{ij} = [\sum_{l=1}^{k} |x_{il} - x_{jl}|^r]^{1/r}$$

where k is the dimension of the data. For the special case of $r = 2$, the Minkowski distance metric gives the Euclidean distance. A MDS mapping from n dimensions to k ($k \ll n$ dimensions attempts to preserve distances. d_{ij} is the pairwise distance in the mapped k dimensional space between object x_i and x_j. Usually, MDS is formulated as an optimization problem. MDS is a minimizer of the squared loss function:

$$min_{x_1,\ldots,x_M} \Sigma(\|x_i - x_j\| - d_{ij})^2$$

Several iterative minimization algorithms exist to move the object points in a multidimensional space to minimize loss (Borg and Groenen, 1997). MDS has many purposes and the most widely used is data visualization, if the data is plotted in the first two or three dimensions. This application of MDS is much like PCA, and in fact, when the Euclidean distances between the points is used, the results are identical to PCA, up to a change in sign. But PCA needs to compute a feature correlation matrix that is computationally expensive in high dimensional data (Jung and Marron, 2009). On the other hand, MDS requires only a similarity or distance matrix of the instances.

3.3.3.2 Feature Scoring Measure

We define S_w as the within class scatter matrix and S_b as the between class scatter matrix:

$$S_b = \sum_{p=1}^{M} K_p(\tilde{b_p} - \tilde{b})(\tilde{b_p} - \tilde{b})^T$$

$$S_w = \sum_{p=1}^{M} \sum_{k=1}^{K_p} (b_k - \tilde{b_p})(b_k - \tilde{b_p})^T$$

Given N samples, M classes, class $p = \{b_k | k = 1, ...K_p\}$ (the size of class p is K_p). Given a feature set F, each sample is represented by a feature vector b_k. $\tilde{b_p}$ is the mean for class p and $\tilde{b}$ is the mean for all samples. S_w measures how the samples are separated from their cluster means or how close the samples surround their centers. S_b measures how the resulting clusters are separated from each other. Devijver and Kittler (1982) introduced four scatter measures given a feature set F:

$$J_1(F) = trace(S_w + S_b),$$

$$J_2(F) = \frac{trace(S_b)}{trace(S_w)},$$

$$J_3(F) = trace(S_w^{-1} S_b),$$

$$J_4(F) = \frac{|S_w + S_b|}{|S_w|}.$$

J_1 ignores the class separability. J_2 ignores the effect on the actual separability caused by the correlation of the feature components. J_4 is more reliable than J_3 as stated in Devijver and Kittler (1982). We compare the effectiveness of both J_3 and J_4 scatter measures later in this section.

3.3.3.3 Feature Set Comparison

Input:

feature set $\tilde{F}_i$;

N samples, K classes;

K classes;

chosen dimension L

Output:

$fcscore_L(\tilde{F}_i)$ on dimension L

Steps:

1. Construct data matrix M for all samples given feature set $\tilde{F}_i$;
2. Compute similarity matrix D from M;
3. Use MDS to generate Y_i and e_i;

4. Generate feature Set F from the top L columns of Y_i;

5. $fcscore_L(\tilde{F}_i) = log(J_4(F))$;

Concept Cluster Ranking Algorithm

The Concept Cluster Ranking Algorithm describes the first step of our feature comparison algorithm based on Multiple Dimensional Scaling. In order to compare and analyze different sets of features, for each feature set, i.e. $\tilde{F}_i$, we construct data matrix M by representing all samples from $\tilde{F}_i$. If the size of $\tilde{F}_i$ is $S_{\tilde{F}_i}$, M is an $N \times S_{\tilde{F}_i}$ matrix. $N \times N$ similarity or dissimilarity symmetric matrix D is computed from M. Next D will be fed into MDS algorithm to produce a new configuration (data matrix) Y_i associated with eigenvalue vector e_i. Y_i is an $N \times \tilde{N}$ ($\tilde{N} \ll S_{\tilde{F}_i}$) matrix. This means that the examples have been mapped to a $\tilde{N}$ dimension space. e_i is an descending eigenvalue vector, and higher eigenvalue means the corresponding dimension is more important. After choosing the first L dimensions (columns) in Y_i as the new feature Set F, we can compute the corresponding scatter score $fcscore_L(\tilde{F}_i) = log(J_4(F))$ for F.

In the second step, we compare the feature sets on the same dimension based on their $fcscore$ socore. For example, if we want to compare and analyze feature set F_i and F_j on dimension L, we get $fcscore_L(F_i)$ and $fcscore_L(F_j)$ from step 1. The general performance of F_i and F_j will be further compared and analyzed by choosing different dimensions.

3.3.3.4 Feature Selection

There are many feature selection algorithms for data pre-processing of clustering and classification problems. The two most widely used, Information Gain (IG) and χ^2-test (CHI) have been proved to be very efficient and effective. Thus in this section, we use the IG, CHI and a random feature selection approach as our baseline algorithms.

Information gain (IG) measures the number of bits of information obtained for class prediction by knowing the presence or absence of a feature in the dependency space of a word. There are K classes $\{C_1, C_2, ...C_K\}$. IG requires the calculation of the class probability $P(C_i)$ of class C_i and the conditional probability $p(C_i|f)$ of a class C_i given feature f. The information gain of a feature f is given by:

$$\begin{aligned} IG(f) = &- \sum_{i=1}^{K} P(C_i) + \\ & P(f) \sum_{i=1}^{K} P(\bar{C}_i|f) \log P(\bar{C}_i|f) \\ & + P(f) \sum_{i=1}^{K} P(C_i|f) \log P(C_i|f) \end{aligned}$$

For a given feature f and a category C_i, suppose A is the number of times f and C_i co-occur, B is the number of times the f occurs without C_i, C is the number of times C_i occurs without f, D is the number of times neither C_i nor f occurs and N is the number of samples. The χ^2 statistics is:

$$\chi^2(f, C_i) = \frac{N(AD - BC)^2}{(A + C)(B + D)(A + B)(C + D)}$$

$$\chi^2(f) = \sum_{i=1}^{K} P(C_i)\chi^2(f, C_i)$$

Our feature comparison algorithm can be further extended to be used for feature selection. There are several ways to do this. The most efficient way is to treat each of the features as a single feature set (size 1). The feature comparison algorithm is still used to calculate the scatter scores for the feature sets on a given dimension. We compare all the features on the same dimension L. Another way is to group the features to fixed size feature sets and compare these feature sets using their scores on the same dimension. The second way is not an optimal solution, however, it can deal with very high dimensional data in an efficient way.

3.3.4 Evaluation

In order to demonstrate the effectiveness of our algorithm, we conduct two experiments. In the first experiment we compare dependency features for the task of concept clustering. In the second experiment we apply our feature selection algorithm to the same task. Three baseline systems, including two widely used feature selection algorithms are used.

Concept clustering is the process of generating semantic classes (concepts) such as *countries*, *universities*, *fruits* and *sports* from unlabeled samples such as *apple, pear, CMU, MIT, basketball, baseball, ...* (Lin, 1998). It has been proven to be useful in applications such as word sense disambiguation (Leacock et al., 1998) and QA (Li and Manandhar, 2009). Recent research on automatical-

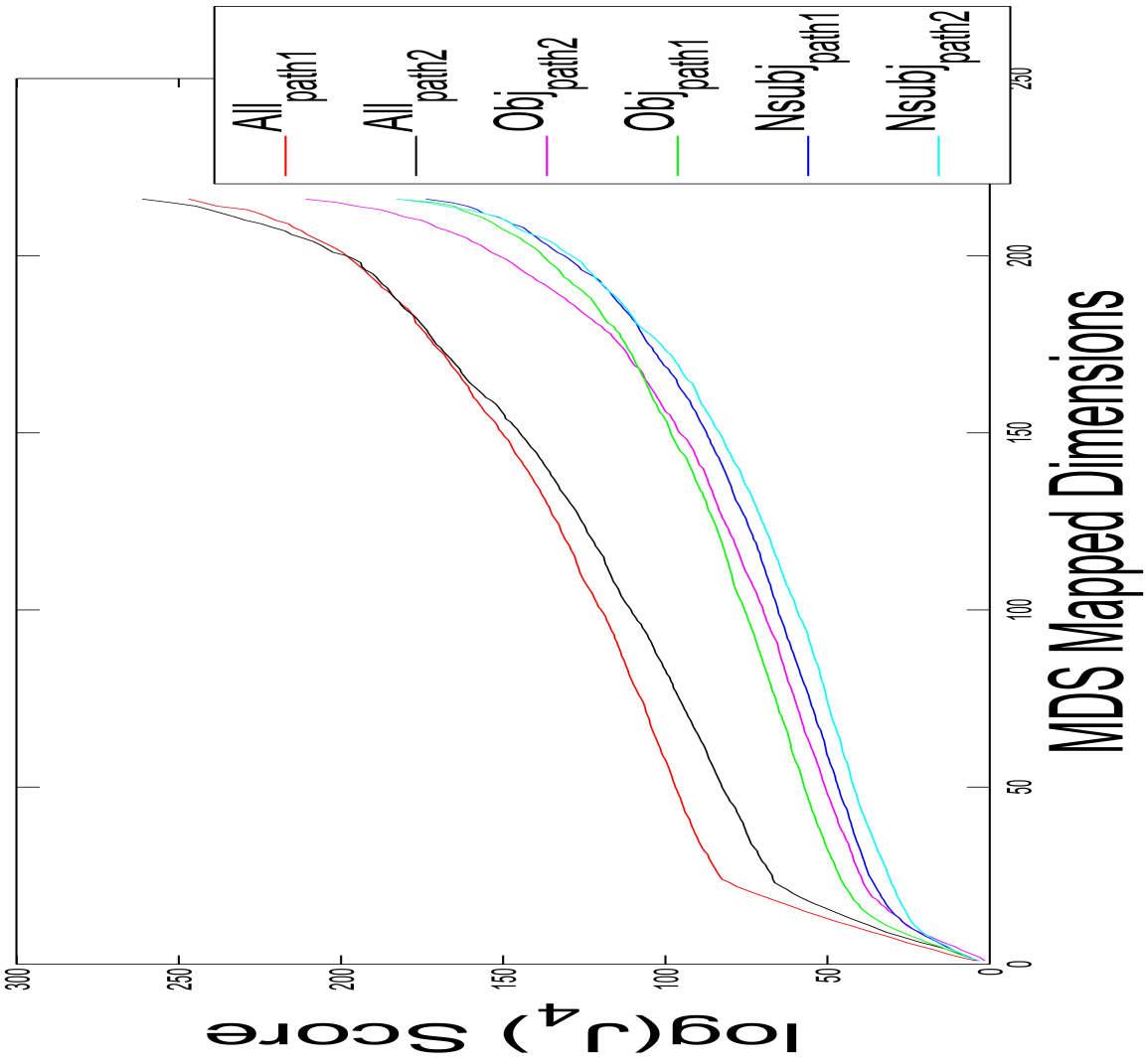

Figure 3.1: Feature sets comparison results on different MDS dimensions

ly extracting concepts based on syntactic context information in large corpora makes it feasible to grow a large set of concept clusters. Clustering by Committee (CBC) makes use of the fact that words in the same cluster tend to appear in similar contexts. Padó and Lapata (2007) presented a general framework for the construction of semantic space models. Dependency-based semantic space models can capture syntactic relationships between words thus are shown to outperform mere word co-occurrence models as specified in Padó and Lapata (2007) and Rothenhäusler and Schütze (2009).

Dependency relations can be categorized into a fixed number of types, such as *amod (adjective modifier)*, *nn (noun-noun modifier)*, *obj (object of a verb)* and *subj (subject of a verb)*, the resulting feature sets can be further divided into several subsets. For example, *apple* has a dependency feature *(obj:eat)* with the type *subj* in our corpus. But dependency features can have different path lengths (sequences of dependency edges). The above feature *(subj:eat)* is a path length 1 feature while *(obj:eat:subj:we)* is a path length 2 feature as it spans two dependency relations. A key problem with dependency context features is to choose the path length or the desirable dependency feature type. But this problem is not the same as traditional feature selection or feature reduction. It should be summarized as a feature set comparison or analysis task, which compares the performance of feature sets of different path lengths or of different dependency types.

3.3.4.1 Experiment Setup

To create the gold standard data set, we chose 481 nouns that needed to be clustered into 30 manually derived classes. We parsed the complete Aquaint

Concepts	**Members**
UK County	{Oxford shire, Buckingham shire, Cambridge shire, Bedford shire, Hertford shire, Lincoln shire, Suffolk Dorset, Stafford shire, ...}
Computer Company	{IBM, Intel, Motorola, Hewlett-Packard, Compaq, Sun, Microsystems, Cisco, Microsoft, Cisco Systems, Oracle, Nortel, Dell, Lucent Technologies, ...}
Movie genre	{romantic, comic, thriller, comedy, action, drama, romance, horror, adventure, western, ...}
Car Brand	{honda, bmw, ford, toyota, nissan, mitsubishi, mazda, chevrolet, volkswagen, audi, renault, chrysler, hyundai, subaru, volvo, opel, ...}
Universities	{UCL, Yale, UCLA, Oxford, Cambridge, New York University, ...}
Newspaper	{Wall Street Journal, USA Today, Daily News, Los Angles Times, New York Times, ...}
Clothes	{robe, wedding dress, coat, jacket, blouse, underwear, T-shirt, ...}
Feeling	{shame, pleasure, happiness, sadness, fear, anger, heaviness, joy...}
Body Parts	{eye, chest, ear, stomach, shoulder, ankle, elbow, ...}
Career	{waitress, waiter, clerk, planner, organizer, }
Music	{rock, rap, pop, folk, blues, country music, jazz, ...}
Sports	{volleyball, tennis, ski, golf, sailing, rugby, soccer, ...}

Table 3.8: Concept clusters examples in our experiment

	MDS Dimensions						
Features	1	2	5	10	50	80	120
All_{path1}	4.1	9.0	20.9	39.6	96.9	110.2	130.9
All_{path2}	3.2	6.7	16.2	34.4	82.5	98.2	122.6
Obj_{path1}	3.3	6.7	15.4	29.5	57.1	67.9	82.9
Obj_{path2}	1.4	3.1	10.6	23.8	50.7	62.2	79.0
$Nsubj_{path1}$	3.5	6.1	13.4	23.7	46.7	57.6	72.8
$Nsubj_{path2}$	3.0	6.6	12.7	21.9	41.9	51.8	67.7
Features	170	180	190	200	210	216	
All_{path1}	166.5	175.7	185.6	198.2	220.1	246.9	
All_{path2}	165.3	174.5	184.9	198.3	229.5	261.2	
Obj_{path1}	109.2	115.8	125.3	137.8	154.7	182.7	
Obj_{path2}	110.2	119.5	134.2	151.4	175.2	210.9	
$Nsubj_{path1}$	101.1	108.4	117.3	131.1	149.2	173.8	
$Nsubj_{path2}$	96.8	107.8	117.3	129.0	149.3	182.7	

Table 3.9: Scores on different MDS dimensions (481 nouns)

corpus (Voorhees, 2002) using the Stanford Parser (Klein and Manning, 2003) to generate dependency features associated with words. We chose the 2,000 most frequent nouns. Three assessors were asked to manually cluster the nouns. In order to obtain a reliable gold standard of high quality, a set of 481 nouns in 30 classes were collected by performing intersection operation from the results by the three assessors. In another word, we removed a noun from a cluster if the noun was not agreed by the three assessors. WordNet (Miller, 1995) is the most widely used gold standard for concept clustering. A set of nouns can be clustered into one concept cluster, if they share the same antecedent node in the WordNet data structure. Wikipedia[3] also provides some concept lists, such as list of fruits, list of inventors, list of newspapers, etc. The assessors were asked to refer to these two sources as gold standards for their manual concept clustering. The generated classes include *weather*, *animals*, *illness* and *feeling*. Some examples are shown in Table 3.8. For the feature comparison experiment, we use the whole set of nouns. For the feature selection experiment, we divided the whole set of nouns into two sets. The training set contains 241 nouns in 30 classes and the testing set includes the other 240 nouns in 24 classes. The testing data set is also used for the data visualization in Figure 3.1.

3.3.4.2 Semantic Space Model

In constructing the semantic feature spaces, we use dependency features and follow the formalization and terminology developed in Padó and Lapata (2007). The dependency parse p of a sentence s is a directed graph $ps = (V_s, E_s)$, where $E_s \subseteq V_s \times V_s$. The nodes $v \in V_s$ are labeled with individual words w_i. Each edge $e \in E_s$ bears a label l which is parser-specific. For example, [Det,det,N]

[3]www.wikipedia.org

and [N,subj,V] are examples for labels provided by MINIPAR (Lin, 1993) in Lin (1998) and Pantel (2003), while labels produced by Stanford Parser such as [nsubj] and [amod] are used in this section.

As defined in Padó and Lapata (2007), the dependency features of the targeted words varies due to different dependency types and path length functions. If we have selected the dependency feature set F for n nouns (W) in our experiment, we represent each (i.e. the kth noun in W) of the nouns in the following feature vector:

$$b_k = (v(f_1), ..., v(f_n)),$$

v is the value function for feature f_i. In this section, v is Lin and Pantel's unbiased mutual information score measure between a word and each of its features.

3.3.4.3 Feature Comparison and Analysis

In this section we compare and analyse six feature sets:

- All_{path1} contains all the path length 1 dependency features;
- All_{path2} contains all the dependency features of path length 2;
- Obj_{path1} contains all the path length 1 object dependency relation features (such as feature *obj:eat* for noun *apple*);
- Obj_{path2} contains all the object dependency relation features of path length 2;

- $Nsubj_{path1}$ contains all the nsubj dependency relation features of path length 1;

- $Nsubj_{path2}$ contains all the nsubj dependency relation features of path length 2.

We can view the first step of our algorithm as a feature mapping process. The size of All_{path1} is 907152 while All_{path2} is 4105980.

In the first step of our algorithm, each feature set is mapped to a new feature set F and the nouns are mapped to a new feature space (generated by MDS). Next we can analyze the performance on the new feature space. Our algorithm is used to compare the clustering performance of different feature sets. However, we need to compare the sets of features on the same dimension. We used the Matlab[4]'s implementation of MDS. We found out that the cosine similarity measure on dependency feature space works better on pilot experiments; therefore, we use this the similarity measure in the following experiment. The scores for the six feature sets are shown in Table 3.9. A higher score means nouns with the same class are better clustered together, while nouns not in the same class are better separated. Figure 3.2 and 3.3.4.3 provide a 2D visualization of our noun collection using different feature sets. The x-axis is the first feature (dimension) of the new feature space and the y-axis is the second feature. We can see that the data are more widely separated in the first two dimensions of the $Nsubj_{path2}$'s mapped feature space (dimension). This conclusion agrees with the scores in Table 3.9. The first two dimension scores of $Nsubj_{path2}$ is 6.6 higher than 3.1 of Obj_{path2}. We can obtain a similar conclusion for All_{path1} and All_{path2}.

[4]www.mathworks.com

In this section we only compare feature sets of the same dimension. As reported in Dy and Brodley (2004), J_3 is shown to be biased to higher dimensional feature sets. We also found from Figure 3.1 that J_4 also prefers higher dimensions. One reason for this is that data is more scattered in higher dimensions; another reason is related to the importance of each new feature. From Section 3.3.3.3 we know that each value in e_i shows the importance of each feature (dimension) in Y_i. If the first k values are much larger than the remaining elements, we can get rid of the remaining features. But in our mapped spaces, the values in e_i only decrease a little from the beginning element. This means the clustering performance will increase if we use more features. Therefore, J_4 will monotonically increase as we add more features.

Padó and Lapata (2007) reported very small differences for different path length functions. However, our experiment shows that there are big differences for different path length functions. From Figure 3.1 and Table 3.9 we can see that if we compare the performance on the first few dimensions, feature sets of path length 1 are better than path length 2. However, if we continue adding features to the new feature space, features of path length 2 outperform those of path length 1 on the clustering task. The reason is that feature sets of longer paths capture more accurate and less noisy syntactic information, so they generally perform better than the sets of shorted paths.

After calculating the difference value between the first value and the last non zero value in e_i for both mapped spaces, we found that the difference value of path length 1 is much higher than that of path length 2. This means that the feature space of path length 2 is much more sparse. When both of the

feature spaces are mapped onto the same dimensional space, the clustering power of path length 2 is scattered more evenly on the new space. Even the last dimensions in the mapped space from path length 2 feature sets can also perform well in clustering.

As stated in section 3.3.3, sometimes MDS generates the same results as PCA. This can be useful, especially when only similarity or dissimilarity information of the examples is known. If we choose the mapped features to conduct the clustering task, this algorithm can also be viewed as a feature reduction process (e.g. removing noisy features, only preserving the most a few prominent features).

3.3.4.4 Feature Selection

In order to compare the performance of different feature selection algorithms, we use cluster purity to measure the quality of a clustering solution. Given a set of selected features, we use a specific clustering algorithm to cluster the samplers and calculate the corresponding cluster purity score. As defined in Zhao and Karypis (2002). If there are Kclusters of the dataset D (N samples) and let the size of cluster C_j be $|C_j|$ and $|C_j|_{class=i}$ be the number of items of class i assigned to cluster j. The purity of a cluster solution S is:

$$purity(S) = \frac{1}{N}\sum_{j=1}^{k}\max_{i}(|C_j|_{class=i})$$

The purity evaluation approach is quite similar to the wrapper method, which requires one predetermined learning algorithm in feature selection and uses its

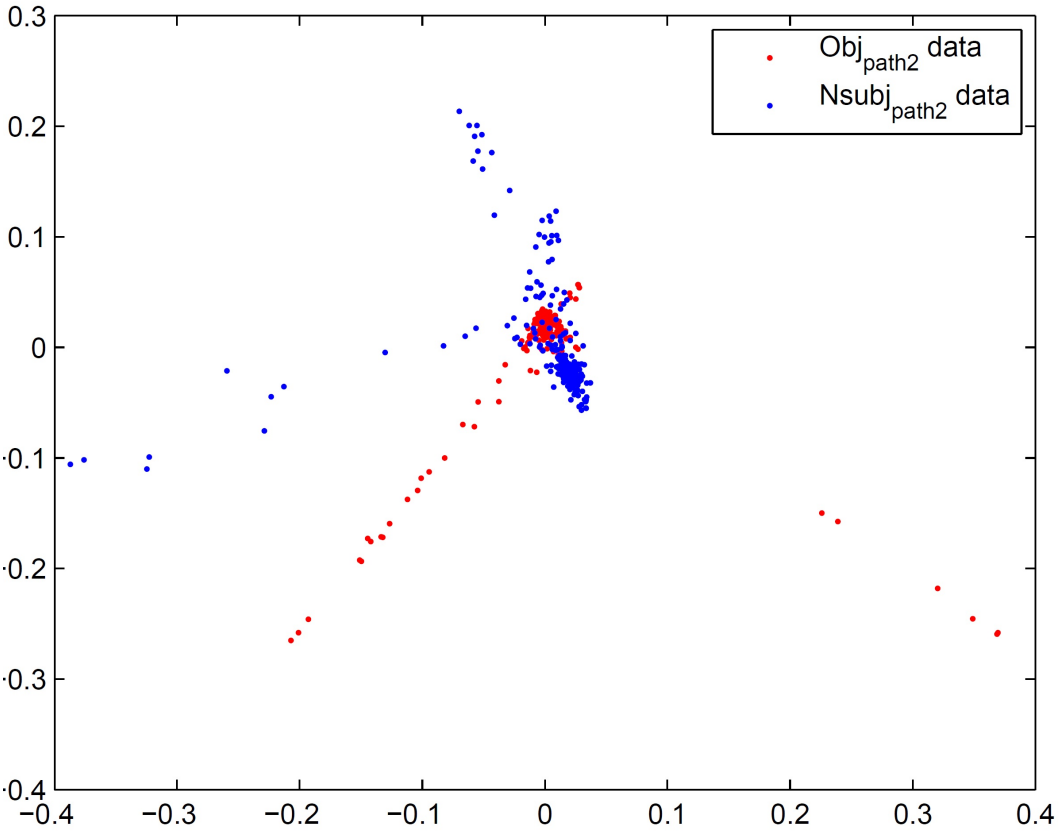

Figure 3.2: 2-Dimensional data visualization using feature set: Obj (path2) and Nsubj (path2)

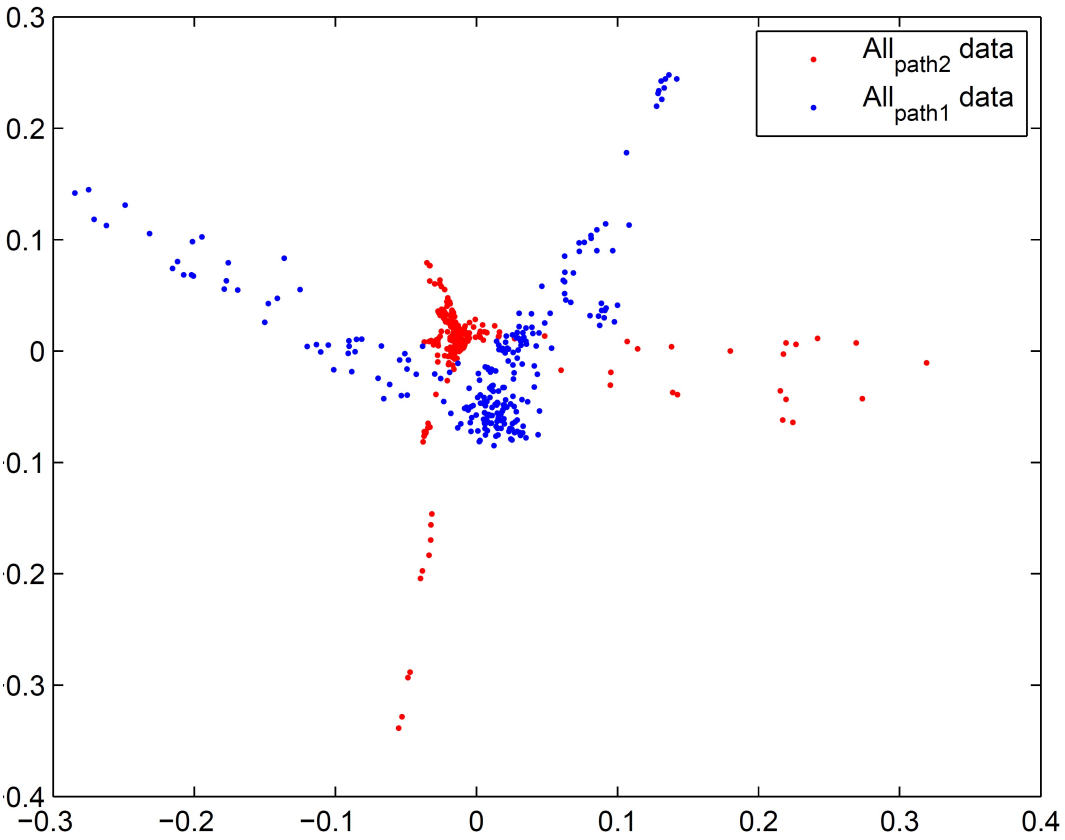

Figure 3.3: 2-Dimensional data visualization using feature set: All (path1) and All (path2)(down)

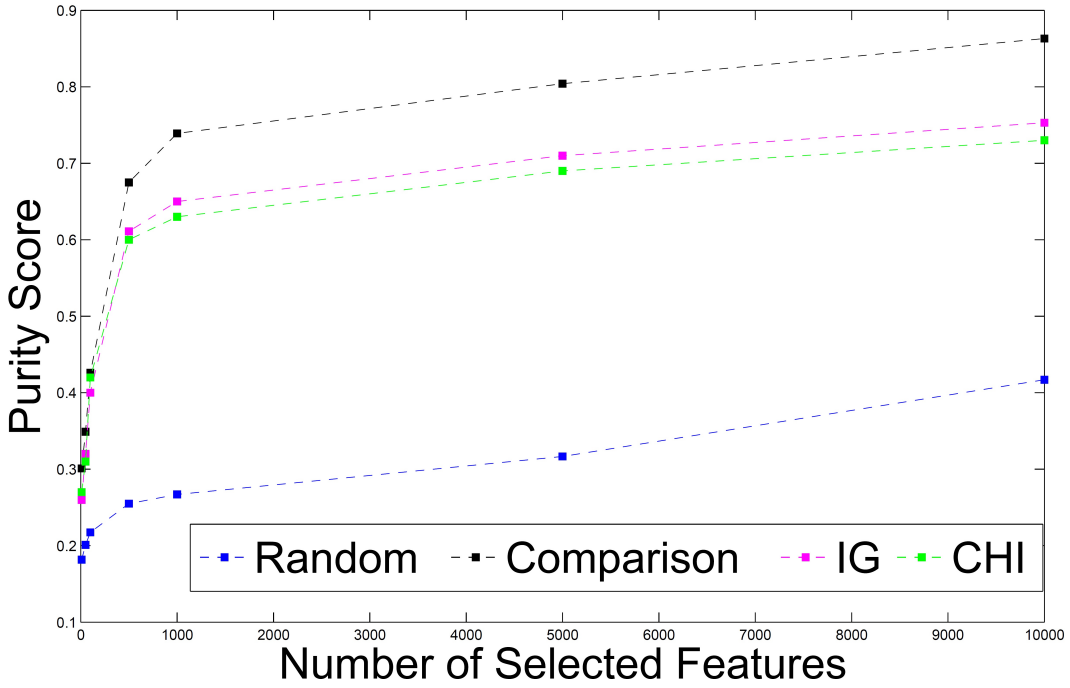

Figure 3.4: Feature selection on the testing data (240 nouns)

performance to evaluate and determine which features are selected. However, the wrapper approach suffers from expensive computation, but we only compare the performance on a few selected feature sets.

In this section, we randomly selected 20,000 dependency features of path length 2 from all training data feature sets. To use our feature comparison algorithm to do feature selection in this section, we rank the features using the comparison score of each single feature. If the dimension is too high, we change to ranking different sets of features, instead of comparing them one by one. After the feature selection step, we map the testing nouns on the new selected feature space, and then use K-means (MacQueen, 1967) to conduct clustering. The results using the purity measure are shown in Figure 3.4.

The comparison algorithm results are in black squares. It can be seen that IG and CHI have a similar performance. Furthermore, our selection result outperforms all the baseline methods.

3.3.5 Conclusion

This section proposed a novel feature comparison algorithm, based on MDS. Our method was shown to be quite effective in comparisons of high dimensional dependency features of different path lengths (sequences of dependency edges) or different types. Despite of the fact that dependency feature sets of longer paths are less noisy and incorporate more accurate syntactic information than the sets of shorter paths, we found that the features of longer paths do not always outperform the features of shorter paths. By analyzing their clustering performance and MDS scores on different dimensions, we found the features of

longer dependency paths perform better than those of shorter paths in general. Yet, features of shorter paths can be a better choice if we only select a small set of features, after applying the feature reduction process on the whole feature set. Our approach was also shown to be quite effective in feature selection and outperforms existing widely used algorithms on high dimensional dependency feature spaces.

Chapter 4

Question Recommendation Based on Information Need

4.1 Introduction

CQA services have built up very large archives of questions and their answers. Table 4.1 is an example from the Yahoo! Answers website. The question title part (Q Title) is the user's reference (query) question, and the user's information need is usually expressed using natural language statements mixed with questions expressing their interests in the question body part (Q Body).

In order to avoid the lag time involved in waiting for a personal response and to enable high quality answers from the archives to be retrieved, we need to search CQA archives of previous questions that are closely associated with answers (Jeon et al., 2005a,b; Duan et al., 2008). If a question is found to be interesting to the user, then a previous answer can be provided with very little delay. Question search and question recommendation are proposed to facilitate finding highly relevant or potentially interesting questions (Section

Q Title	If I want a faster computer should I buy more memory or storage space? ...
Q Body	I edit pictures and videos so I need them to work quickly. Any advice?
Answer	... If you are running out of space on your hard drive, then ... to boost your computer speed usually requires more RAM ...

Table 4.1: Yahoo! Answers question example

). Given a user's question as the query, we define question search as "*trying to retrieve the most syntactically or semantically similar questions from the question archives*". As the complement of question search, we also define question recommendation as *recommending questions whose information need is the same or similar to the user's original question.* For example, the question "*What aspects of my computer do I need to upgrade ...*" with the information need "*... making a skate movie, my computer freezes, ...*" and the question "*What is the most cost effective way to extend memory space ...*" with information need "*... in need of more space for music and pictures ...*" are both good recommendation questions for the user in Table 4.1. Therefore, the recommended questions are not necessarily identical or similar to the query question.

In this section, we analyze the user's information need behind a question by

making use of CQA data. We discuss methods for recommending questions based on the similarly between the information needs behind. More specifically, we treat the question title part as the query question while the question body part as the information need behind the query question. We propose a machine translation based approach to model and infer the user's the information need behind the query question. We show that with the proposed models it is possible to recommend questions that have the same or a similar information need.

The remainder of the section is structured as follows. In section 4.2, we briefly describe the related work and limits. Section 4.3 addresses in detail how we measure the similarity between short texts in order to recommend questions based on the information need similarity. Section 4.4 describes our machine learning based approach for information need modeling. Section 4.5 is the experiment part. Section 4.6 is the conclusion.

4.2 Related Work

The related work for this chapter can be found in Section 4.1. Different researchers provide different ways to retrieve questions from large archives of QA data. However, none of them considers the similarity between questions by modeling their information needs using CQA question title and body pairs. We will show that question recommendation can be improved by information need modeling.

4.3 Short Text Similarity Measures

In question retrieval systems accurate similarity measures between documents are crucial. Most traditional techniques for measuring the similarity between two documents focus on comparing word co-occurrences. Metzler et al. (2005) compared the performance of a proposed statistical machine translation based approach with several simplistic approaches such as word overlapping, relative-frequency measures, and TFIDF measures. However, the state-of-the-art techniques usually fail to achieve the desired results for short texts (Jeon et al., 2005a).

In order to measure the similarity between short texts, in this section we make use of three kinds of text similarity measures: TFIDF based, knowledge based and LDA based similarity measures. We will compare their performance in the question recommendation task in the experiment section.

4.3.1 TFIDF

In the popular TFIDF scheme (Salton and McGill, 1983), the frequency count of a word in a document was compared to an inverse document frequency count, which as able to measure the number of occurrences of the word in the corpus. Baeza-Yates and Ribeiro-Neto (1999) provided a method to calculate the similarity between two texts based on TFIDF. Each document is represented by a term vector using TFIDF score. The similarity between two text D_i and D_j is the cosine similarity in the vector space model:

$$cos(D_i, D_j) = \frac{D_i^T D_j}{\|D_i\|\|D_j\|}$$

TFIDF is one of the most widely used term weighting schemes in todays IR systems, as it is both efficient and effective. However, if the query text contains only one or two words, this method will be biased to shorter answer texts (Jeon et al., 2005a). We also find that in CQA data, short contents in the question body cannot provide any information about the users' information needs. For the above two reasons, in the test data sets, we do not include questions whose information needs contain only a few noninformative words (e.g. stop words).

4.3.2 Knowledge-based Measure

Mihalcea et al. (2006) proposed several knowledge-based methods for measuring the semantic level similarity of texts, in order to solve the lexical chasm problem between short texts. These knowledge-based similarity measures were derived from word semantic similarity by making use of WordNet (Miller, 1995). The evaluation on a paraphrase recognition task showed that knowledge-based measures outperform the simpler lexical level approach.

We follow the definition in Mihalcea et al. (2006) to derive a text-to-text similarity metric mcs for two given texts D_i and D_j:

$$mcs(D_i, D_j) = \frac{\sum_{w \in D_i} maxSim(w, D_j) * idf(w)}{\sum_{w \in D_i} idf(w)} + \frac{\sum_{w \in D_j} maxSim(w, D_i) * idf(w)}{\sum_{w \in D_j} idf(w)}$$

For each word w in D_i, $maxSim(w, D_j)$ computes the maximum semantic similarity between w and any word in D_j. In this section we choose lin (Lin, 1998) and jcn (Jiang and Conrath, 1997) to compute the word-to-word semantic similarity.

We only choose nouns and verbs for calculating mcs. Additionally, when w is a noun we restrict the words in document D_i (and D_j) to just nouns. Similarly, when w is a verb, we restrict the words in document D_i (and D_j) to just verbs.

4.3.3 Probabilistic Topic Model

Celikyilmaz et al. (2010) presented probabilistic topic model based methods to measure the similarity between question and candidate answers. The candidate answers were ranked based on the hidden topics discovered by LDA methods.

In contrast to the TFIDF method which measures "common words", short texts are not compared to each other directly in probabilistic topic models. Instead, the texts are compared using some "third-party" topics that relate to them. A passage D in the retrieved documents (document collection) is represented as a mixture of fixed topics, with topic z getting weight $\theta_z^{(D)}$ in passage D and each topic being a distribution over a finite vocabulary of words, with word w having a probability $\phi_w^{(z)}$ in topic z. Gibbs Sampling can be used to estimate the corresponding expected posterior probabilities $P(z|D) = \hat{\theta}_z^{(D)}$ and $P(w|z) = \hat{\phi}_w^{(z)}$ (Griffiths and Steyvers, 2004).

In this section we use two LDA based similarity measures (Celikyilmaz et al., 2010) to measure the similarity between short information need texts. The first LDA similarity method uses KL divergence to measure the similarity between two documents under each given topic:

$$sim_{LDA1}(D_i, D_j) = \frac{1}{K}\sum_{k=1}^{K} 10^{W(D_i^{(z=k)}, D_j^{(z=k)})}$$

$$W(D_i^{(z=k)}, D_j^{(z=k)}) = -KL(D_i^{(z=k)} \| \frac{D_i^{(z=k)} + D_j^{(z=k)}}{2}) - KL(D_j^{(z=k)} \| \frac{D_i^{(z=k)} + D_j^{(z=k)}}{2})$$

$W(D_i^{(z=k)}, D_j^{(z=k)})$ calculates the similarity between two documents under topic $z = k$ using KL divergence measure. $D_i^{(z=k)}$ is the probability distribution of words in document D_i given a fixed topic z.

The second LDA similarity measure (Griffiths and Steyvers, 2004) treats each document as a probability distribution of topics:

$$sim_{LDA2}(D_i, D_j) = 10^{W(\hat{\theta}^{(D_i)}, \hat{\theta}^{(D_j)})}$$

where $\hat{\theta}^{(D_i)}$ is probability distribution of topics of document D_i, as defined earlier.

4.4 Information Need Modeling

There are two reasons to model the information needs behind the questions. It is often the case that the query question does not have a question body, therefore, we need to model the information need behind a query question, in order to recommend questions based on the similarity of their information needs. Another reason is that information need analysis plays a crucial part, not only in QA, but also in IR (Liu et al., 2008). In this section we propose an information need modeling method based on a statistical machine translation model.

4.4.1 Statistical Machine Translation Model

($\mathbf{f}^{(s)}$, $\mathbf{e}^{(s)}$), s = 1,...,S is a parallel corpus. In a sentence pair ($\mathbf{f}$, $\mathbf{e}$), source language String, $\mathbf{f} = f_1 f_2 ... f_J$ has J words, and $\mathbf{e} = e_1 e_2 ... e_I$ has I words. Alignment $\mathbf{a} = a_1 a_2 ... a_J$ represents the mapping information from source language words to target words.

Statistical machine translation models estimate $Pr(\mathbf{f}|\mathbf{e})$, the translation probability from source language string $\mathbf{e}$ to target language string $\mathbf{f}$ (Och et al., 2003):

$$Pr(\mathbf{f}|\mathbf{e}) = \sum_{\mathbf{a}} Pr(\mathbf{f}, \mathbf{a}|\mathbf{e})$$

The EM-algorithm is usually used to train the alignment models to estimate lexicon parameters $p(f|e)$.

In E-step, the counts for one sentence pair ($\mathbf{f}$,$\mathbf{e}$) are:

$$c(f|e;\mathbf{f},\mathbf{e}) = \sum_{\mathbf{a}} Pr(\mathbf{a}|\mathbf{f},\mathbf{e}) \sum_{i,j} \delta(f,f_j)\delta(e,e_{a_j})$$

$$Pr(\mathbf{a}|\mathbf{f},\mathbf{e}) = Pr(\mathbf{f},\mathbf{a}|\mathbf{e})/Pr(\mathbf{a}|\mathbf{e})$$

In the M-step, lexicon parameters become:

$$p(f|e) \propto \sum_{s} c(f|e;\mathbf{f}^{(s)},\mathbf{e}^{(s)})$$

Different alignment models such as IBM-1 to IBM-5 (Brown et al., 1993) and H-MM model (Och and Ney, 2000) provide different decompositions of $Pr(\mathbf{f},\mathbf{a}|\mathbf{e})$. For different alignment models, different approaches were proposed to estimate the corresponding alignments and parameters. The details can be found in Och and Ney (2003) and Brown et al. (1993).

4.4.2 Community Question Answering Parallel Corpus

We model the information need behind a question by making use of the C-QA data. More specifically, the question title and information need (question body) pairs are considered as a parallel corpus, which is used for estimating the statistical translation probabilities. The probability of a word w which describes the information need of a give question Q is:

$$P(w|Q) = \lambda \sum_{t \in Q} P_{tr}(w|t)P(t|Q) + (1 - \lambda)P(w|C)$$

The word-to-word translation probability $P_{tr}(w|t)$ is the probability that word w is translated from word t in question Q using the translation model. The above formula uses linear interpolation smoothing of the document model with the background language model $P(t|C)$. λ is the smoothing parameter. $P(t|Q)$ and $P(t|C)$ are estimated using the maximum likelihood estimator.

One important consideration is that statistical machine translation models first estimate $Pr(\mathbf{f}|\mathbf{e})$ and then calculate $Pr(\mathbf{e}|\mathbf{f})$, using Bayes' theorem to minimize ordering errors (Brown et al., 1993):

$$Pr(\mathbf{e}|\mathbf{f}) = \frac{Pr(\mathbf{f}|\mathbf{e})Pr(\mathbf{e})}{Pr(\mathbf{f})}$$

However, in this section, we skip this step, as we found the order of words in the information need (question body) part is not an important factor. More specifically, we estimated the IBM-4 model by *GIZA++* (Och and Ney, 2003) with the question part as the source language and information need (question body) part as the target language.

	Test_c		
Methods	MRR	Precision@5	Precision@10
TFIDF	84.2%	67.1%	61.9%
Knowledge1	82.2%	65.0%	65.6%
Knowledge2	76.7%	54.9%	59.3%
LDA1	92.5%	68.8%	64.7%
LDA2	61.5%	55.3%	60.2%
	Test_t		
Methods	MRR	Precision@5	Precision@10
TFIDF	92.8%	74.8%	63.3%
Knowledge1	78.1%	67.0%	69.6%
Knowledge2	61.6%	53.3%	58.2%
LDA1	91.8%	75.4%	69.8%
LDA2	52.1%	57.4%	54.5%

Table 4.2: Question recommendation results without applying information need modeling

	Test_c		
Methods	MRR	Precision@5	Precision@10
TFIDF	86.2%	70.8%	64.3%
Knowledge1	82.2%	65.0%	66.6%
Knowledge2	76.7%	54.9%	60.2%
LDA1	95.8%	72.4%	68.2%
LDA2	61.5%	55.3%	58.9%
	Test_t		
Methods	MRR	Precision@5	Precision@10
TFIDF	95.1%	77.8%	69.3%
Knowledge1	76.7%	68.0%	68.7%
Knowledge2	61.6%	53.3%	58.2%
LDA1	96.2%	79.5%	69.2%
LDA2	68.1%	58.3%	53.9%

Table 4.3: Question recommendation results with applying information need modeling

4.5 Experiments and Results

4.5.1 Text Preprocessing

The questions posted on community QA sites often contain spelling or grammar errors. Zhao et al. (2007) and Bunescu and Huang (2010a) showed that these errors influence the calculation of similarity and the performance of IR. In this section, we use open source software, *afterthedeadline*[1], to automatically correct the spelling errors in the question and information need texts first. We also made use of Web 1T 5-gram[2] to implement an N-Gram based method (Cheng et al., 2008) to further filter out the false positive corrections and re-rank correction suggestions (Mudge, 2010). The texts are tagged using Brill's Part-of-Speech Tagger (Brill, 1995), as the rule-based tagger is more robust than the state-of-the-art statistical taggers for raw web contents. This tagging information is only used for WordNet similarity calculation. Stop word removal and lemmatization are applied to all the raw texts before feeding into the machine translation model training, the LDA model estimating and similarity calculation.

4.5.2 Construction of Training and Testing Sets

We made use of questions from Yahoo! Answers for the estimating models and evaluation. More specifically, we obtained 2 million questions under two categories at Yahoo! Answers: 'travel' (1 million), and 'computers&internet' (1 million). Depending on whether the best answers had been chosen by the

[1]http://afterthedeadline.com

[2]http://www.ldc.upenn.edu/Catalog/CatalogEntry.jsp?cata
logId=LDC2006T13

asker, questions from Yahoo! answers can be divided into 'resolved' and 'unresolved' categories. From the above two categories, we randomly selected 200 resolved questions to construct two testing data sets: 'Test_t' ('travel'), and 'Test_c' ('computers&internet'). In order to measure the information need similarity, in our experiment we selected only those questions whose information needs part contained at least 3 informative words after stop word removal. The rest of the questions, 'Train_t' and 'Train_c', in the two categories, are left for estimating the LDA topic models and the translation models. We will show how we obtain these models later.

4.5.3 Experimental Setup

For each question (query question) in 'Test_t' or 'Test_c', we used the words in the question title as the main search query and the other words in the information need as a search query expansion, to retrieve candidate recommended questions from the Yahoo! Answers website. We obtained an average of 154 resolved questions in 'travel' and 'computers&internet' categories and three assessors were involved in the manual judgments.

Given a question returned by a recommendation method, two assessors are asked to label it 'good' or 'bad'. The third assessor will judge any conflicts. The assessors are also asked to read the information need and answer parts. If a recommended question is considered to express the same or similar information need, the assessor will label it 'good'; otherwise, the assessor will label it as 'bad'.

Three measures for evaluating the recommendation performance are utilized.

They are Mean Reciprocal Rank (MRR), top five prediction accuracy (precision@5) and top ten prediction accuracies (precision@10) (Voorhees and Tice, 2000). In MRR the reciprocal rank of a query question is the multiplicative inverse of the rank of the first 'good' recommended question. The top five prediction accuracy for a query question is the number of 'good' recommended questions out of the top five ranked questions and the top ten accuracy is calculated out of the top ten ranked questions.

4.5.4 Similarity Measure

The first experiment conducted question recommendation, based on their information need parts. Different text similarity methods were used to measure the similarity between the information need texts.

In the TFIDF similarity measure (TFIDF), the IDF values for each word were computed from frequency counts over the entire *Aquaint* corpus. To calculate the word-to-word knowledge-based similarity, a WordNet::Similarity Java implementation[3] of the similarity measures lin (Knowledge2) and jcn (Knowledge1) is used in this section. To calculate topic model based similarity, we estimated two LDA models from 'Train_t' and 'Train_c', using *GibbsLDA++*[4]. We treated each question, including the question title and the information need part, as a single document of a sequence of words. The documents were preprocessed before being fed into LDA model; 1,800 iterations for Gibbs sampling 200 topics parameters were set for each LDA model estimation.

[3]http://cogs.susx.ac.uk/users/drh21/

[4]http://gibbslda.sourceforge.net

Q1:	If I want a faster computer should I buy more memory or storage space?
InfoN	If I want a faster computer should I buy more memory or storage space? Whats the difference? I edit pictures and videos so I need them to work quickly. ...
RQ1	Would buying 1gb memory upgrade make my computer faster?
InfoN	I have an inspiron B130. It has 512mb memory now. I would add another 1gb into 2nd slot ...
RQ2	whats the difference between memory and hard drive space on a computer and why is.....?
InfoN	see I am starting edit videos on my computer but i am running out of space. why is so expensive to buy memory but not external drives? ...
Q2:	Where should my family go for spring break?
InfoN	... family wants to go somewhere for a couple days during spring break ... prefers a warmer climate and we live in IL, so it shouldn't be SUPER far away. ... a family road trip. ...
RQ1	Whats a cheap travel destination for spring break?
InfoN	I live in houston texas and i'm trying to find i inexpensive place to go for spring break with my family.My parents don't want to spend a lot of money due to the economy crisis, ... a fun road trip...
RQ2	Alright you creative deal-seekers, I need some help in planning a spring break trip for my family
InfoN	Spring break starts March 13th and goes until the 21st ... Someplace WARM!!! Family-oriented hotel/resort ... North American Continent (Mexico, America, Jamaica, Bahamas, etc.) Cost= Around $5,000 ...

Table 4.4: Question recommendation results by LDA measuring the similarity between information needs

The results in Table 4.2 show that the TFIDF and LDA1 methods perform better for recommending questions than the other methods. After further analysis of the questions recommended by both methods, we discovered that the ordering of the recommended questions from TFIDF and LDA1 are quite different. The TFIDF similarity method prefers texts with more common words, while the LDA1 method can find the relation between non-common words between short texts, based on a series of third-party topics. The LDA1 method outperforms the TFIDF method in two ways: (1) the top recommended question information needs to share less common words with the query question; (2) the top recommended questions span wider topics. The questions highly recommended by LDA1 can suggest more useful topics to the user.

Knowledge-based methods are also shown to perform worse than TFIDF and LDA1. We found that some words were mis-tagged so that they were not included in the word-to-word similarity calculation. Another reason for the poorer performance is the words out of the WordNet dictionary were also not included in the similarity calculation.

The Mean Reciprocal Rank score for TFIDF and LDA1 are more than 80%. That is to say, we are able to recommend questions to users by measuring their information needs. The first two recommended questions for Q1 and Q2 using the LDA1 method are shown in Table 4.4. InfoN is the information need associated with each question.

In the preprocessing step, some words were successfully corrected such as "*What should I do this saturday? ... and staying in a* ***hotell*** *...*" and "*my* ***faimly*** *is traveling to* ***florda*** *...*". However, there are still a small number of

texts such as "*How come my **Gforce** visualization doesn't work?*" and "*Do i need an Id to travel from new york to **maimi**?*" failed to be corrected. Therefore, in the future, a better method is required to correct these failure cases.

4.5.5 Information Need Modeling

Some of the retrieved questions had information need parts which were empty or became empty or almost empty (one or two words left) after the pre-processing step. The average number of such retrieved questions for each query question is 10 in our experiment. The similarity ranking scores of these questions are quite low or zero in the previous experiment. In this experiment, we will apply information need modeling approach to the questions whose information needs are missing, in order to find out whether we can improve question recommendation

The question and information need pairs in both the 'Train_t' and 'Train_c' training sets were used to train two IBM-4 translation models by *GIZA++* toolkit. These pairs were also pre-processed before training, and pairs whose information need part became empty after pre-processing were disregarded.

During the experiment, it was found that some of the generated words describing the information need existed in the question's body. This is caused by a self translation problem in the translation model as specified in Xue et al. (2008): the highest translation score for a word is usually given to itself, if the target and source languages are the same. This has always been a tough question: not using self-translated words can reduce retrieval performance, as the information need parts need the terms to represent the semantic mean-

ings; using self-translated words does not take advantage of the translation approach. To tackle this problem, we control the number of the words predicted by the translation model to be exactly twice the number of words in the corresponding pre-processed question.

The words describing the information needs of the retrieved questions produced by our translation based approach are shown in Table 4.5. In Q1, the information need behind the question "*recommend website for custom built computer parts*" may imply that users need to know some information about building computer parts such as "*ram*" and "*motherboard*" for a different purpose such as "*gaming*". While in Q2, the user may want to compare computers from different brands such as "*dell*" and "*mac*" or consider the "*price*" factor for "*purchasing a laptop for a college student*".

We also did a small scale comparison between the generated information need describing words and real questions whose information need parts are not empty. Q3 and Q4 in Table 4.4 are two examples. The original information need for Q3 is "*looking for beautiful beaches and other things to do such as museums, zoos, shopping, and great seafood*" in CQA. The generated content for Q3 contains words in wider topics such as '*wedding*', '*surf*' and the price information ('*cheap*'). This reflects that there are some other users asking similar questions with the same or other interests.

From the results in Table 4.3, we can see that the performance of most similarity methods were improved by making use of information need prediction. Different similarity measures received different degrees of improvement. LDA1 obtained the highest improvement, followed by the TFIDF based method.

These two approaches are more sensitive to the contents generated by a translation model.

However, we found out that in some cases the LDA1 model failed to give higher scores to good recommendation questions. For example, Q5, Q6, and Q7 in Table 4.5 were retrieved as recommendation candidates for the query question in Table 4.1. All three questions were good recommendation candidates, but only Q6 ranked fifth while Q5 and Q7 were out of the top 30 using the LDA1 method. Moreover, in a small number of cases, bad recommendation questions received higher scores and jeopardized the performance. For example, for query question "*How can you add subtitles to videos?*" with information need "*... add subtitles to a music video ... got off youtube ...download for this ...*", a retrieved question "*How would i add a music file to a video clip. ...*" was highly recommended by the TFIDF approach, as the predicted information need contained '*youtube*', '*video*', '*music*', '*download*', ...

The MRR score got improved from 92.5% to 95.8% in the 'Test_c' and from 91.8% to 96.2% in 'Test_t'. This means that the top question recommended using our methods can very well cater to the users' information needs. The top five precision and the top ten precision scores using the TFIDF and LDA1 methods also showed different degrees of improvement. Thus, we can improve the performance of question recommendation by information need modeling.

Q1:	Please recommend A good website for Custom Built Computer parts?
InfoN	custom, site, ram, recommend, price, motherboard, gaming, ...
Q2:	What is the best laptop for a college student?
InfoN	know, brand, laptop, college, buy, price, dell, mac, ...
Q3:	What is the best Florida beach for a honeymoon?
InfoN	Florida, beach, honeymoon, wedding, surf, cheap, fun, ...
Q4:	Are there any good clubs in Manchester
InfoN	club, bar, Manchester, music, age, fun, drink, dance, ...
Q5:	If i buy a video card for my computer will that make it faster?
InfoN	nvidia, video, ati, youtube, card, buy, window, slow, computer, graphics, geforce, faster, ...
Q6:	If I buy a bigger hard drive for my laptop, will it make my computer run faster or just increase the memory?
InfoN	laptop, ram, run, buy, bigger, memory, computer, increase, gb, hard, drive, faster, ...
Q7:	Is there a way I can make my computer work faster rather than just increasing the ram or harware space?
InfoN	space, speed, ram, hardware, main, gig, slow, computer, increase, work, gb, faster, ...

Table 4.5: Information need prediction parts generated by the IBM-4 translation based approach

4.6 Conclusions

In this chapter we tried to deal with the problem of recommending questions from large archives of CQA data, based on users' information needs. In order to do that we used CQA data to analyze the user's information need and proposed a machine translation based approach to model the information need given only the user's query question. Different similarity measures were compared to prove that it is possible to satisfy the user's information need by recommending questions from large archives of community QA. The LDA based approaches were proved to perform better on measuring the similarity between short texts, on the semantic level, than BOW overlapping based and knowledge based methods. Experiments showed that the proposed machine translation based information need modeling approach further enhanced the performance of question recommendation methods.

Chapter 5

Information Need Satisfaction by Usefulness Classification

5.1 Introduction

Retrieving questions from the CQA repositories to satisfy the user's information need is by no means an easy task. One reason is that the user's information need is usually quite complex. The query questions are sometimes vague and ambiguous (Burger et al., 2001a). Another reason is the lexical gap or word mismatch problem between questions; the same information need can be expressed by two questions with few common words (Lee et al., 2008). Bag of words (BOW) overlapping based approaches (e.g. Monz and de Rijke (2001)) would fail to capture the similarity between reformulated questions. Consider the two following examples of questions from Lee et al. (2008) that are semantically similar to each other:

1. Where can I get cheap airplane tickets?

2. Any travel website for low airfares?

This information need satisfaction problem using CQA data was first studied as a question usefulness ranking task in Bunescu and Huang (2010a). The retrieved questions fell into three predefined classes representing different level of "usefulness" for fulfilling the user's information need. However, we find this categorization too coarse-grain. We think a fine-grain framework is needed in order to further investigate usefulness of different questions. We want to further study the question usefulness ranking problem and how the question usefulness can benefit the question recommendation. We also find that more than 1/4 of the "Useful" questions from Bunescu and Huang (2010a) have a textual entailment relation with the reference question in this dataset and the size of the manually labeled dataset is too small.

Users of WikiAnswers can mark question reformulations, however, we discover that most of the "Reformulation" question pairs are not paraphrasing. We also discover that the most "Useful" questions often have a textual entailment relation with the reference question. So we think question paraphrasing and textual entailment techniques can be considered to improve the performance of usefulness classification.

In this chapter, we focus on the problem of information need satisfaction in question recommendation: *given a user's question, we want to rank the retrieved questions based on the degree of information need satisfaction.* More specifically, we propose a fine-grain question usefulness classification framework based on the overlapping of the information needs behind the questions. In another word, we formulate the problem of information need satisfaction in question recommendation as a question usefulness classification task, and

different classes represent different degree of information need satisfaction. Moreover, we compare the effectiveness of different question similarity measures. We are also able to investigate how the large WikiAnswers archives can benefit information need satisfaction by retrieving "useful" questions.

The remainder of the chapter is organized as follows. Section 5.2 discusses the work on question usefulness ranking and question paraphrasing and textual entailment. Section 5.3 presents the CQA repositories we will use in the experiment. The classification framework to deal with the information need satisfaction problem is detailed in Section 5.4. The syntactic and semantic question similarity measures used in the framework are also presented in Section 5.5. Finally, we present and analyze the experimental results obtained in Section 5.6 and conclude with Section 5.7.

5.2 Related Work

5.2.1 Question Usefulness Ranking

Bunescu and Huang (2010a) present a machine learning approach for the task of ranking previously answered questions in a CQA resource, with respect to a new, unanswered reference question. They manually labeled 60 groups of questions. The relations between the reference question and the ranked questions are divided into three categories:

1. Reformulation: paraphrasing questions which use alternative words or expressions;

2. Useful: useful questions;

3. Neutral: neutral questions which are less useful than the ones in the "Useful" category.

Table 5.1 shows an example from this dataset.

5.2.2 Question Paraphrasing

Paraphrases are alternative ways of expressing the same information. In recent years, paraphrasing is quite important for many NLP applications, including document summarization, text generation machine translation and question reformulation in QA (Lin and Pantel, 2001), (Barzilay, 2003), (Callison-Burch, 2007) and (Zhao et al., 2007). As a subtask of paraphrasing, question paraphrasing is the research of finding reformulated questions expressing the same information. Zhao et al. (2007) presented a novel method for automatically extracting question paraphrases from a search engine's logs and generating templates for question reformulation. A SVM model was trained, by making use of different features extracted from the questions and the most effective combination of features was identified. However, this method could not be used, as it requires access to Microsoft's search engine logs. Bernhard and Gurevych (2008) evaluated various string similarity measures and vector space based similarity measures on the task of retrieving question paraphrases from the WikiAnswers repository. As discussed in the previous section, most of the gold standard question pairs were not paraphrasings. However, we will show that the wealth of this CQA data repository is still valuable for information need satisfaction research, after further annotation and processing.

Reference1:
What bike should I buy for city riding?
Reformulation1:
What is the best bike to buy to get myself around campus in the city ?
What is the best bike for traveling in a city ?
Useful1:
What bike should I buy for a starter bike ?
What bike should I get as a beginner ?
What bike should I get if I'm a bigger person ?
What bike should I buy for free riding ?
What bike can I ride as a 16 year old ?
What bike should I buy to participate in my first triathlon ?
Neutral1:
What is a good bike to start mountain biking ?
What bike should I buy for hills ?
What bike should I buy my toddler ?
What bike should I buy for working out ?
What exercise bike should I buy, upright or recumbent ?
Which stationary bike Should I buy ?
What bike should I buy for working out ?
What exercise bike should I buy, upright or recumbent ?
What racing bike should I buy ?
What motorcycle jacket is better for a sport bike ?
What is the best tire to buy for a city bike ?

Table 5.1: Question usefulness ranking example from Bunescu and Huang (2010a)

5.2.3 Textual Entailment

Textual entailment is another important research field in NLP. Recognizing Textual Entailment (RTE) (Dagan et al., 2006) detects whether a Hypothesis (H) can be inferred (or entailed) by a Text (T). It is shown to be helpful for QA by Harabagiu and Hickl (2006).

In the work of Monz and de Rijke (2001), Corley and Mihalcea (2005) and Glickman et al. (2005), the methods proposed for RTE tasks relied on the similarity measures between texts, making use of the bag-of-words approach. Kouylekov and Magnini (2006) exploited the use of syntactic features and proposed a syntactic tree editing distance measure to detect entailment relations. Wang and Neumann (2007) proposed a subsequence kernel method approach, to incorporate structural features extracted from syntactic dependency trees for this task. Wang et al. (2009b) combined syntactic and semantic features to capture the key information shared between texts.

As far as can be ascertained, no previous work has been done in RTE from CQA question pairs. We will make the first attempt to exploit this direction using our annotated WikiAnswers dataset and the first fine-grain framework of question usefulness.

5.3 Question and Answer Repositories

WikiAnswers is a social QA website similar to Yahoo! Answers. As of February 2008, it contained 1,807,600 questions, sorted into 2,404 categories (Answers Corporation, 2008). Compared with its competitors, the main originality of

WikiAnswers is that it relies on the wiki technology used in Wikipedia, which means that answers can be edited and improved over time by the contributors (Bernhard and Gurevych, 2008). WikiAnswers allows users to mark question pairs that they think are rephrasings ("alternate wordings", or paraphrases) of existing questions. For example, the following questions are marked as paraphrasings for the reference question "*How do vaccines work?*":

1. How does the flu shot work?
2. Is there a vaccine to protect against swine flu?
3. How does the body get rid of viruses like the cold or flu?
4. What steps involving the immune system and white blood cells help people with the swine flu recover?
5. What is an example of how a vaccine works?

They also evaluated various string similarity measures and vector space based similarity measures on the task of retrieving question paraphrases from the WikiAnswers repository.

After a thorough investigation of this repository, we found that only a small number of the marked pairs are paraphrasings. Therefore, the WikiAnswers repository cannot be directly used for paraphrasing training. However, most of the marked questions are useful for answering the reference question. The repository will be used in this chapter for several purposes.

We found that if a question Q is marked on more question pages as "rephrasing", Q is more likely to be a general question. In contrast, if a question is rarely marked for any other questions, it is usually more specific. This is

similar to PageRank (Page et al., 1999), in which a webpage with more incoming hyperlinks tends to be more important. Taking $Q1$ *How do you write a good concluding sentence or paragraph?* and $Q2$ *How do you write good beginnings and endings for paragraphs and essays?* in Table 5.2 for example, $Q1$ is marked by other 74 questions as "rephrasing" and $Q2$ is hase not been marked as "rephrasing" by any other question. $Q1$ is more general than $Q2$, and $Q2$ talks about a more specific aspect of $Q1$. Based on this property, we collected more than 1,500 groups of questions from WikiAnswers. Actually we did not crawl the whole WikiAnswers website in order to get the "rephrasing" questions. Instead we relied on the indexed 'rephrasing" question pages from Google as we found that the webpages containing "rephrasing" questions had special "anchor" words. We ran our collecting system for three days and collected a total of 1,582 groups of questions. In each group, one reference question has been marked as "rephrasing" for the other questions. Some examples are shown in 5.2. In the first group, "*How do you write a good concluding sentence or paragraph?*" is marked as a "rephrasing" for 74 other questions in the WikiAnswers website; five out of the 74 are shown following it, while the Question "*How do you write good beginnings and endings for paragraphs and essays?*" has not been marked as a "rephrasing" on any pages.

For our experiments, we also use another two datasets created by Bunescu and Huang (2010a), which contain 60 groups of questions spanning a wide range of topics. Each group consists of a reference question followed by a partially ordered set of questions. For each reference question, the corresponding partially ordered set was created from questions in Yahoo! Answers and other online repositories that have a high cosine similarity with the reference question. As illustrated in the previous section, the questions are mainly divided into three

How do you write a good concluding sentence or paragraph?	74
How do you write good beginnings and endings for paragraphs and essays?	0
What is a conclusion of managing conclusion?	0
What is a good way to close an essay about Robert Hooke?	0
What is a good closing paragraph on your educational goals?	0
How do you write conclusion for bottle bio mes?	0
Who should not get a swine flu vaccination?	54
Can swine flu vaccination be taken during pregnancy?	2
Can you get the H1N1 vaccine if you are currently sick with the swine flu?	2
Is it safe for a pregnant woman to get the H1N1 flu shot?	4
Is a flu shot safe when you are pregnant?	2
Is the swine flu shot active?	0
What should you feed a rabbit?	51
How often to feed a 45 days old rabbit?	0
Can bunnies eat cantaloupe?	0
What do Rex bunnies eat?	0
Can bunnies eat potatoes?	0
What do backyard bunnies eat?	0

Table 5.2: WikiAnswers dataset examples

categories. The "Reformulation" questions are thought to be more useful than the other questions, and the "Useful" questions are deemed to better satisfy the user's information need than "Neutral" questions. The relations between questions in the "Useful" category were more finely annotated. However, only one example was given for this annotation, and no guiding standard was provided. We think the finer annotation is fairly essential for fulfilling the user's information need, but needs further definition and annotation guidelines.

5.4 Question Usefulness Classification Framework

Given a user's question, we want to rank the retrieved questions based on the degree of information need satisfaction. In this section, we propose a classification framework to solve the information need satisfaction problem in the question recommendation. Within a fine-grain question usefulness classification framework, we are able to formulate this problem as a question usefulness classification task. Within this framework we compare the effectiveness of different question similarity measures for classification. We also investigate how the large WikiAnswers archives can benefit information need satisfaction by retrieving "useful" questions.

5.4.1 Fine-grain Question Usefulness Definition

Inspired by the work of Bunescu and Huang (2010a), we further categorize the relations between the reference question and the other questions. The catego-

rization is based on the overlap of the information need behind the questions. We want to investigate how complex NLP technologies (Textual Entailment, Paraphrasing, etc) benefit information need satisfaction in question recommendation.

The relations between questions are divided into several categories. In the whole spectrum, we may define the following semantic relations between a question pair $< Q_r, Q_i >$:

1. $Q_r, = Q_i$ (E): the two questions are (almost) the same, asking for the same thing;

2. $Q_r > Q_i$ (G): the input question is more general than the reference question. For example, the reference question is asking about the apple, while the input question is asking about fruit. In other words, the answer to the reference question is a subset of the answer to the input question;

3. $Q_r < Q_i$ (S): similar to the previous annotation, but the other way around. The reference question is asking for more specific things than the input question. In particular, yes-no questions can be viewed as verifications of concrete facts, which are very specific;

4. $Q_r <> Q_i$ (C): the two questions are about related things, probably the same level on the ontology, e.g. apple vs. pear. In other words, there is a more general question Q_j, which subsumes both Q_r and Q_i, $Q_j > Q_r$, and $Q_j > Q_i$;

5. $Q_r \leftarrow Q_i$ (P): the input question is asking about a presupposition of the reference question, for example, a definition of a concept mentioned in the reference question;

6. $Q_r - Q_i$ (O): the input question is useful for answering the reference question, but the relation is not one of the above mentioned ones;

7. $Q_r \neq Q_i$ (N): the input question is useless for answering the reference question, although the topics of both questions are the same. e.g. summer camp in Florida vs. summer camp in Canada;

Some examples from WikiAnswers are shown in Table 5.3. We also manually relabeled the datasets used in Bunescu and Huang (2010a) using this new categorization. We found that most of the "Useful" questions have a textual entailment relation with the reference question. An example is shown in Table 5.4.

This multi-class classification problem can be decomposed into several binary classification problems. We focus on the following binary classification subtasks, based on the classes defined in this chapter:

1. Question Paraphrasing Identification: the questions in E are treated as positive data, while the others are negative;

2. Question Textual Entailment: the questions in G or S are thought to have textual entailment relationship with the reference question;

3. Question Usefulness: this is a simple task to distinguish the related questions from the questions in N.

5.5 Question Similarity Measures

In our question usefulness classification task, we use both syntactic and semantic similarity measures to calculate the relevance between the reference

Did Prince become a Jehovah's Witness?	Reference
Is Prince a Jehovah's Witness?	E
Is Prince still a practicing Jehovah's Witness?	*E*
Are there celebrity Jehovah's Witnesses?	*G*
What kingdom hall does prince go to?	*N*
Are there some 'stars' who have become jehovah's witnesses?	*G*
What did dolphins evolve from?	**Reference**
Where did dolphins come from?	*E*
When did dolphins evolve?	*N*
Where are dolphins located?	*N*
Did dolphins evolve into anything?	*N*
How did dolphins evolve from a dinosaur?	*S*
How does the flu shot work?	**Reference**
What is a flu shot?	*P*
Is there a vaccine to protect against swine flu?	*N*
How does the body get rid of viruses like the cold or flu?	*U*
What steps involving the immune system and white blood cells help people with the swine flu recover?	*U*
What is an example of how a vaccine works?	*U*

Table 5.3: WikiAnswers question examples with annotation

Reference1: Whats a nice summer camp to go to in Florida?
Reformulation1: What camps are good for a vacation during the summer in FL? (E) What summer camps in FL do you recommend? (E)
Useful1: Does anyone know a good art summer camp to go to in FL? (S) Are there any good artsy camps for girls in FL? (S) What are some summer camps for like singing in Florida? (S) What is a good cooking summer camp in FL? (S) Do you know of any summer camps in Tampa, FL? (S) What is a good summer camp in Sarasota FL for a 12 year old? (S) Can you please help me find a surfing summer camp for beginners in Treasure Coast, FL? (S) Are there any acting summer camps and/or workshops in the Orlando, FL area? (S) Does anyone know any volleyball camps in Miramar, FL? (S) Does anyone know about any cool science camps in Miami? (S) Whats a good summer camp youve ever been to? (G)
Neutral1: Whats a good summer camp in Canada? (N) Whats the summer like in Florida? (N)

Table 5.4: Re-Categorization question examples

question and other questions in the CQA data. The scores of these similarity measures are used as features for training the classifiers.

5.5.1 TFIDF

The TFIDF method used in the chapter is the same as the one introduce in Section 4.3.1.

5.5.2 Knowledge-based Measures

Mihalcea et al. (2006) proposed several knowledge-based methods for measuring the semantic level similarity of texts, to solve the lexical chasm problem between short texts. These knowledge-based similarity measures were derived from word semantic similarity by making use of WordNet. The evaluation on a paraphrase recognition task showed that knowledge-based measures outperform the simpler lexical level approach.

We follow the definition in Mihalcea et al. (2006) to derive a text-to-text similarity metric mcs for two given texts D_i and D_j:

$$mcs(D_i, D_j) = \frac{\sum_{w \in D_i} maxSim(w, D_j) * idf(w)}{\sum_{w \in D_i} idf(w)} + \frac{\sum_{w \in D_j} maxSim(w, D_i) * idf(w)}{\sum_{w \in D_j} idf(w)}$$

For each word w in D_i, $maxSim(w, D_j)$ computes the maximum semantic similarity between w and any word in D_j. In this section we choose lin (Lin,

1998) and *jcn* (Jiang and Conrath, 1997) to compute the word-to-word semantic similarity.

We only choose nouns and verbs for calculating *mcs*. Additionally, when w is a noun we restrict the words in document D_i (and D_j) to just nouns. Similarly, when w is a verb, we restrict the words in document D_i (and D_j) to just verbs.

5.5.3 Bag-of-Words (BOW) Overlapping Measure

Similar to the popular TFIDF scheme of Salton and McGill (1983), a widely used similarity measure for textual entailment is bag-of-words (BOW), which calculates the similarity based on the ratio of overlapping words as seen in Monz and de Rijke (2001), Corley and Mihalcea (2005) and Glickman et al. (2005). $NumOverLap(D_i, D_j)$ is the number of overlapping words between D_i and D_j.

$$BOW_1(D_i, D_j) = \frac{NumOverLap(D_i, D_j)}{Lengthof D_i}$$

$$BOW_2(D_i, D_j) = \frac{NumOverLap(D_i, D_j)}{Lengthof D_j}$$

5.5.4 Dependency Matching Measure

Dependency features have been explained in detail in the previous feature selection section. Each sentence or question can be parsed into a series of dependency triples. A dependency matching similarity measure was defined in Wang and Neumann (2007). We use the Typed Dependency component in Stanford Parser to generate the dependency triples.

A dependency triple in document D_i is in the form of $< word_{m_i}, relation_{r_i}, word_{n_i} >$ and a triple in document Dj is $< word_{m_j}, relation_{r_j}, word_{n_j} >$. A function called triple set matcher can be defined as follows:

if ($word_{m_i} = word_{m_j}$ and $word_{n_i} = word_{n_j}$ and $word_{r_i} = word_{r_j}$)

return FullMatch

elseif ($word_{m_i} = word_{m_j}$ and $word_{r_i} = word_{r_j}$)

return LeftMatch

elseif ($word_{n_i} = word_{n_j}$) and $word_{r_i} = word_{r_j}$)

return RightMatch

elseif ($word_{m_i} = word_{m_j}$ and $word_{r_i} = word_{r_j}$)

return ArgsMatch

Consequently, the two similarity functions can be defined more precisely, based on the sum of the matched triple elements normalized by the total number of triples.

$$DepSimi_1(D_i, D_j) = \frac{OverLapScore(D_i, D_j)}{NumberOfTriple(D_i)}$$

$$DepSimi_2(D_i, D_j) = \frac{OverLapScore(D_i, D_j)}{NumberOfTriple(D_j)}$$

$$OverLapScore(D_i, D_j) = a_1 \times Num_F + a_2 \times Num_L + a_3 \times Num_R + a_4 \times Num_A$$

The two similarity functions use different normalizations. $OverLapScore(D_i, D_j)$ is the sum of the different matching cases discovered by the above triple set matcher; a_1 to a_4 are the different weights for the different matching cases; Num_F, Num_L, Num_R and Num_A are the total number of full matching, left matching, right matching and argument matching cases.

5.5.5 Predicate-argument Structure Measure

Wang and Neumann (2007) proposed a text relatedness scoring method for textual entailment. Predicate-argument structures (PAS) were used to calculate the semantic similarity between texts.

5.5.5.1 Semantic Role Labeling

In order to obtain the predicate-argument structures for the textual entailment corpus, we use the semantic role labeler described in Zhang et al. (2008). The SRL system was trained on the Wall Street Journal sections of the Penn Treebank using PropBank and NomBank annotation of verbal and nominal

predicates, and relations to their arguments, and produces the semantic dependencies as outputs. The head words of the arguments (including modifiers) were annotated as a direct dependent of the corresponding predicate words, labeled with the type of the semantic relation (Arg0, Arg1 . . . , and various ArgMs).

As input, the SRL system requires syntactic dependency analysis. We use the open source MSTParser (McDonald, 2006), also trained on the Wall Street Journal sections of the Penn Treebank, using a projective decoder with second order features. The SRL system then goes through a pipeline of 4-stage processing: predicate identification (PI) identifies words that evoke a semantic predicate; argument identification (AI) identifies the arguments of the predicates; argument classification (AC) labels the arguments with the semantic relations (roles); and predicate classification (PC) further differentiates different uses of the predicate word. All components are built as maximal entropy based classifiers, with their parameters estimated by the open source TADM system, with feature sets selected on the development set. Evaluation results from CoNLL[1] 2006 Shared task show that the MSTParser achieved state-of-the-art performance, especially for its out-domain applications.

5.5.5.2 Predicate-argument Structure

After the semantic parsing described in the previous section, we obtain a PAS for each sentence.

A predicate-argument graph (PAG) is defined, with nodes which are predi-

[1]the Conference on Computational Natural Language Learning

cates, arguments or sometimes both, and the edges of which are labeled semantic relations. Notice that each predicate can dominate zero, one, or more arguments, and each argument has one or more predicates which dominate it. Furthermore, the graph is not necessarily fully connected. Thus, the similarity measure is calculated based on the similarity between the PAG semantic relevance.

In order to compare the two graphs and reduce the alignment complexity by breaking the graphs into sets of trees, two types of decomposed trees were considered in Wang and Neumann (2007). One is to take each predicate as the root of a tree and the arguments as child nodes; the other is to take each argument as the root and their governing predicates as child nodes. The first is called Predicate Trees (P-Trees) and the latter, Argument Trees (A-Trees).

To obtain the P-Trees, each predicate is considered, in order to find all the arguments which it directly dominates, and then a P-Tree is constructed. The algorithm to obtain A-Trees works in a similar way. Subsequently, we have a set of P-Trees and a set of A-Trees for each PAG, both of which are simple trees with a depth of one.

5.5.5.3 PAS Similarity Measure

Based on the P-Trees and A-Trees, a semantic similarity measure was proposed in Wang and Neumann (2007). This similarity measure is the maximum value of the relatedness scores of all pairs of trees in two sentences (P-trees and A-trees).

$$PASSimi(D_i, D_j) = \max_{1 \leq m \leq r, 1 \leq n \leq s} \{(R(Tree_{im}, Tree_{jn}))\}$$

$Tree_{im}$ is one of the PAG trees from document D_i, and $Tree_{jn}$ is from document D_j. $R(Tree_{im}, Tree_{jn})$ calculates the relatedness between two PAG trees, based on their decompensated P-trees and A-trees. In order to compare two P-Trees or A-Trees, each predicate-argument pair contained in a tree is treated as a semantic triple in the form of $< predicate, relation, argument >$. The relatedness function $R(Tree_{im}, Tree_{jn})$ between two trees is defined as the minimum value of the relatedness scores of all the triple pairs from the two trees.

5.6 Experiments

5.6.1 Statistical significance test

For the statistical significance test, We use the Fisher's Randomization Test approach described in Section 3.2.4.2. Smucker et al. (2007) provided the details of using this test for a IR task. We will explain how we use this test for our experiment.

Taking the textual entailment task in Table 5.5 for example, our approach outperforms the baseline by 0.047 ($0.604 - 0.557$). We first make a null hypothesis which is "*the method using syntactic features (SynF) and the approach using both syntactic and semantic features (SynF+SemF) are identical*". Then we

randomly apply one approach to a test question pair under this null hypothesis. If there are 50 test question pairs, there are 2^{50} ways (permutations) to apply both approaches for the assessors to evaluate. After we randomly 10,000 permutations, we count the times T when the absolute difference between the average accuracy are greater than 0.047 during the experiment. The value of $T/10,000$ is called *p-value*, and if *p-value* is less than 0.05 then the null hypothesis is rejected and we can conclude that our approach performs significantly better than the baseline approach (Smucker et al., 2007). We report the statistical test decision directly next to each result, e.g. "significant at $p < 0.05$" or simply "$p < 0.05$".

5.6.2 Experiment on the WikiAnswers collection

We randomly select 50 groups of questions from our WikiAnswers collection described in Section 5.3. From each group, we also randomly select 4-5 questions, as well as the reference question. The resulting dataset is similar to the examples shown in Table 5.2. Within each selected group, two annotators are employed to mark the relations between the reference question and other questions. A total of 213 pairs of questions were annotated. However, because the two annotators did not always agree with each other on the annotations, the inter agreement ratio of our annotated collection is 76.5% (163 out of 213). The 163 question pairs whose annotations are agreed by both annotators are chosen as the dataset $Wiki_s$. We used the program by Rui Wang[2] for calculating the dependency matching and predicate-argument structure matching similarities for the following experiments.

[2]www.coli.uni-saarland.de/∼rwang/

	SynF	SynF+SemF
$Result_T$	55.7%	60.4% ($p < 0.05$)
$Result_P$	56.0%	67.0% ($p < 0.05$)

Table 5.5: Paraphrasing and textual entailment results using syntactic and semantic features

For our experiment on question textual entailment, we run 10 5-fold cross validations using LIBSVM (Chang and Lin, 2011) on the $Wiki_s$ dataset. There are 83 textual entailment pairs out of the 163 pairs in $Wiki_s$. The results for question textual entailment is shown in Table 5.5 on the $Result_T$ row. The average accuracy is 55.7% when making use of all the syntactic features (SynF), including bag-of-words overlapping and dependency matching. The average accuracy can be further improved by 4.7% when the predicate-argument structure semantic feature (SemF) is added.

We also test the performance of syntactic and semantic features on question paraphrasing identification. As there are less than 10 paraphrasing question pairs in $Wiki_s$, similarly to the previous annotation method, we collect a dataset of 200 question pairs, including 100 paraphrasings. The results are shown on the $Result_P$ row in Table 5.5. Using only syntactic features we obtain an average accuracy of 56.0%. And the performance is improved by 11% after adding the feature of SemF.

We did significance test on both task, and the results showed that the method using both set of features significantly outperformed the one using only syntactic features. We find that some of the false positive cases in the results are caused by the unsatisfying performance of a SRL system (Pradhan et al.,

2011) and (Srikumar and Roth, 2011). During our experiments, we find that due to the limited coverage of the verb frames or predicates, sometimes the SRL system fails to (or not correctly) identify some some important predicates or arguments in a question. More specifically, if question $Q1$ entails question $Q2$ and some important predicates or arguments in $Q2$ fail (or not correctly) to be identified by the SRL system, fewer P-Trees and A-Trees are matched against some part of $Q1$, thus, the relatedness of the whole pair could easily be satisfied.

One of the most important things in question recommendation is to retrieve questions. Similarly we run another two experiments on a binary classification task: predict whether a question is useful or not for the reference question (similar to the "Useful" category in Bunescu and Huang (2010b)).

As well as $Wiki_s$, we used the two datasets QSimiple and QComplex from Bunescu and Huang (2010b). As specified in Section 5.3, the relations between questions are mainly divided into three classes: "Reformulation", "Useful" and "Neutral". In QComplex, unanswered questions (reference questions) tend to be longer, whereas other questions in the group are shorter. However, the QSimple set is the opposite. There 1,329 question pairs in QSimple and 1,970 pairs in QComplex.

In order to run the binary classification experiment on QSimiple and QComplex, the questions in "Reformulation" and "Useful" are treated as useful questions, while the questions in "Neutral" are treated as negative ones. For the questions annotated in $Wiki_s$, the questions in E, G and S are treated as positive examples, while the other questions are negative. The results of using

different datasets for training and testing are shown from Row 2 to Row 7 in Table 5.6. The last two columns show the results when making use of different sets of features introduced in Section 5.5. The column of "SynF+SemF" shows the result of using syntactic and semantic features while the last column shows the result of using all the features.

Training ⇒ Testing	SynF+SemF	SynF+SemF+TFIDF+Knowledge
$Wiki_s$ ⇒ QSimple	48.2%	50.9% ($p < 0.05$)
$Wiki_s$ ⇒ QComplex	43.1%	38.7% ($p < 0.05$)
QSimple ⇒ QComplex	63.7%	63.2% ($p < 0.05$)
QComplex ⇒ QSimple	59.0%	59.0%
QSimple ⇒ $Wiki_s$	64.8%	63.8% ($p < 0.05$)
QComplex ⇒ $Wiki_s$	38.0%	38.0%
$Wiki_W$ ⇒ $Wiki_s$	71.0%	72.9% ($p < 0.05$)
$Wiki_W$ ⇒ QSimple	73.1%	75.0% ($p < 0.05$)
$Wiki_W$ ⇒ QComplex	68.4%	71.3% ($p < 0.05$)

Table 5.6: Question usefulness classification results

Row 2 ($Wiki_s$ ⇒ QSimple) shows the performance on QSimple using $Wiki_s$ as training data. We can see that the classification performance on QSimple gets improved by 2.7% after introducing TFIDF and Knowledge based features. Row 3 ($Wiki_s$ ⇒ QComplex) shows the performance on QComplex using the same training data. However using all of the features causes a drop in accuracy of 0.5%. We also did statistical significant test in Row 2, 3, 4 and 6 in which the performance is different using different sets of features. For example, in Row 2, we can conclude that the method using all the features significantly outperforms the one only using syntactic and semantic features. And in Row

3 the method using syntactic and semantic features significantly outperforms the one using all the features. From Row 2 to Row 7 in Table 5.6, we can conclude that the classification performance rarely gets improved and sometimes is harmed by introducing TFIDF and Knowledge based features.

In $Wiki_s$, about half of the question pairs are paraphrasings or have a textual entailment relation. Therefore, we think that WikiAnswers might be a good resource for learning to retrieve useful questions. Thus, we randomly select 25,000 question pairs from the WikiAnswers dataset, detailed in Section 5.3 as positive examples for training a binary SVM classifier. We also randomly select 25,000 question pairs ($Wiki_W$) which are not marked as "rephrasing" questions on the WikiAnswers website as negative examples. The results are also shown in the last three rows of Table 5.6. The second column shows the results using a combination of both syntactic and semantic similarity measures, while the third one shows the results of using all the similarity measures described in Section 5.5. The $Wiki_W \Rightarrow Wiki_s$ row shows the results of testing the SVM classifier on our annotated dataset, and the other two rows show the accuracy for testing on the pairs in QSimple and QComplex. We also did statistical significant test for the results in the last three rows. Using $Wiki_W$ as training data, we can conclude that the methods using and all the features significantly outperformed the ones using only syntactic and semantic features. We can also conclude that $Wiki_W$, which has not been annotated, is a valuable resource for learning to retrieve useful questions.

5.7 Conclusions

In this chapter, we focused on the problem of information need satisfaction in question recommendation. The contributions of this chapter are as follows. Firstly we formulated this problem as a fine-grain question usefulness classification task. The classification task was further decomposed into a few binary classification subtasks, such as question paraphrasing and question textual entailment. Secondly, we compared the effectiveness of different syntactic and semantic features on these subtasks using a user-annotated gold standard. The large archives of question pairs collected from WikiAnswers were shown to be a valuable resource for learning to retrieve useful questions and helping solve the information need satisfaction problem.

Chapter 6

Conclusions and Future Research

This chapter presents a set of conclusions based on the thesis work and proposes several directions for future work. It shows the contributions made in the field of understanding user information need in QA with respect to the state-of-the-art in NLP, IR and machine learning and future directions for research in this area.

6.1 Contributions

This thesis makes the following contributions to the area of understanding user information need in QA:

6.1.1 Question Disambiguation for Understanding the User's Information Need

The aim of this part of work is to understand the information need behind ambiguous questions and thus resolve question ambiguity.

We propose an approach for generating clarification questions to help verify the users' information needs. The concept clusters discovered from the context of the list of answers are analyzed for ambiguous questions. The labels of the concept clusters also contribute to topics of the clarification questions. Therefore, based on a set of manually crafted question templates, an interactive dialogue can be provided to help the user clarify the information need and reformulate the question in a way that makes it possible for the QA system to find the correct answer. The empirical results show that our approach leads to good performance on the test question collection.

As the success of question topic generation relies on the quality of the concept clusters, we also propose a novel feature comparison algorithm based on MDS. This approach is effective in feature selection for concept clusters and outperforms existing widely used algorithms in high dimensional spaces such as dependency features.

6.1.2 Question Recommendation Based on Information Need

The aim of this part of work is to analyze the information need of the user by making use of the large archives of CQA data and recommend questions based

on information needs.

We use the CQA data to analyze the user's information need behind a question. By treating the question title and body parts as a parallel corpus, we propose a machine translation based approach to model the information need given only the user's query question. Different similarity measures are compared to recommend questions from CQA repository based on the information need. The experiments show that (1) two LDA based approaches are proved to perform better on measuring the similarity between short texts, on the semantic level, than traditional methods; (2) it is possible to satisfy the user's information need by recommending questions from large archives of community QA.

6.1.3 Information Need Satisfaction by Usefulness Classification

The aim of this part of work is to deal with the information need satisfaction by making use of CQA data.

For studying the the information need satisfaction problem in CQA question recommendation, we propose a fine-grain question usefulness classification framework to decompose this problem into few subtasks. The retrieved questions from the repository are ranked based on how much of the information need can be satisfied. We compare different syntactic and semantic measures of question paraphrasing and textual entailment, using a user-annotated gold standard. Moreover, the large amount of WikiAnswers data without any additional human annotation is shown to enhance the systems performance on

retrieving useful questions that fulfill the users information need.

6.2 A Look into the Future

Our work suggests interesting directions for future research, outlined below. We will discuss some immediate extensions of our work and also suggest longer-term research directions that naturally build on techniques developed in this thesis.

6.2.1 Question Disambiguation

We applied the MDS based feature selection approach to concept clustering in order to reduce the noise in the clusters. During the experiment of resolving question ambiguity, we discovered that the noise in the concept clusters caused some bad cases in the results, and the number of concept clusters also influence the performance. So we plan to evaluate the quality (e.g. noise in the concept clusters, noisy cluster labels) and quantity (e.g. number of concept clusters) features on the performance of our proposed clarification generation approach.

After a thorough analysis of the experiment results, we found out some concepts appeared in the context of an answer by chance. So we plan to investigate the statistic dependency relationships between the answer and the concepts found in the context. By removing the concepts which have no relationship with the answers (e.g. we only consider the concepts if they have subject or object relations with the answer), we intend to further improve the performance. We also plan to develop an automatic approach to generate clarification ques-

tion templates in addition to the manually created ones.

As specified in Burger et al. (2001b), the ambiguities range from various spellings to the presence of vocabulary terms that have different meanings in different domains (word sense disambiguation). We plan to investigate how word sense disambiguation work can benefit our research. For example, we want to find out whether the concepts found in retrieved documents can help disambiguate the sense of the terms in the question.

6.2.2 Improving Question Recommendation

Jeon et al. (2005a) gathered a large number of similar question pairs from a Korean CQA service website using a translation based approach. We plan to apply their approach to the WikiAnswers question and answer pairs we collected, and perform a thorough analysis on the results. And we plan to extend our work in Chapter 4 by considering the similarity between the generated answers. More specifically, we plan to treat the question and answer pairs in the WikiAnswers repository as a parallel corpus. So that, based on the reference (query) question, the word-to-word translation probability can be used to produce sequence of words which can be viewed as the generated answer. We will add this "pseudo answer" as a new feature to rank questions with answers besides in addition to the previous features we used (e.g. calculating the similarity between the "pseudo answer" and other answers).

Pseudo-relevance feedback (or blind feedback) is an technique in IR system for improving the retrieval accuracy (Rocchio, 1971), (Buckley et al., 1994), (Robertson et al., 1994) and (Lavrenko and Croft, 2001). The basic idea is

that the initial set of retrieved results are considered as relevant, and the terms in this initial set are used to retrieve more results (Manning et al., 2008). In our work of question similarity based on information need, we plan to compare blind relevance feedback with other approaches for identifying information needs.

We also plan to improve our work in Chapter 5 by investigating more effective features and considering other textual entailment and paraphrasing algorithms.

6.2.3 WikiAnswers Repository

The previous work on CQA mainly focused on the user's history in the Yahoo! Answers website. We plan to verify the importance of the asker and answerer's histories from the WikiAnswers service in user behavior, user preference and question search. After a thorough analysis of the WikiAnswers website, we plan to consider the following community features from WikiAnswers:

1. Past professions (e.g. Electronic Repair)
2. Hobbies (e.g. Crossword Puzzles)
3. Contributions of the user
4. Badges and credits
5. Categories which the user supervises
6. Categories which the user is most active

The contributions of the user consist of the the user's past activities of "trashes", "unflagged", "Removed", "made the change to", "added", "added alter-

nate wording" and "split" some questions or the corresponding answers.

6.2.4 Domain Adaption

In Chapter 5, we used the two datasets in Bunescu and Huang (2010a) from Yahoo! Answers as training data and obtained average accuracies of 64.8% and 38% respectively when testing on our annotated dataset from WikiAnswers. We feel that there is still plenty of room for improvement. Although Yahoo! Answers and WikiAnswers have lots in common, we find the two services are different in several aspects. First, most questions in WikiAnswers are written in a line, while Yahoo answers has more complex and personalized questions. Second, WikiAnswers has a smaller user base than Yahoo! Answers and contains more personal questions. Third, the users' activities in the two services differ from each other as seen in Section 6.2.3 and Liu et al. (2008). We think domain adaption techniques (Pan and Yang, 2010) can be used to further improve our system's classification performance on WikiAnswers data by 1) transferring knowledge from Yahoo! Answers to WikiAnswers; 2) overcoming the lacking of labeled WikiAnswers data; 3) making use of the large archives of unlabeled data in WikiAnswers.

Many machine learning methods work well only under a common assumption: the training and test data are drawn from the same domain and the same distribution (Pan and Yang, 2010). If the domain changes where the distribution may be different, the statistical models usually need to be rebuilt from the newly collected and labeled data. However this process may be time consuming and expensive. And sometimes there is sufficient training data in one domain (source domain), but there is little or no labeled data in another

related domain (target domain) (Blitzer et al., 2006) and (Jiang and Zhai, 2007). Domain adaption techniques are proposed to transfer knowledge across domains and have been applied to many NLP applications (Ando and Zhang, 2005; Jiang and Zhai, 2007; Daumé III, 2009; Li et al., 2012).

Domain adaption techniques can be roughly divided into the fully supervised and the semi-supervised categories (Daumé III, 2009). In a fully supervised setting, domain adaption techniques have access to a large, annotated corpus of data from a source domain and a small annotated corpus in the target domain. While in a semi-supervised setting, the corpus in the target domain are not annotated. Jiang and Zhai (2007) proposed a general instance weighting framework for domain adaptation which can support adaptation with some target domain labeled instances as well as that without any labeled target instances. Their empirical results on three NLP tasks showed that incorporating unlabeled target instances using their approach were more effective than excluding misleading training examples from the source domain. Agirre and de Lacalle (2009) obtained up to 22% error reduction when using both source and target domain data compared to a classifier trained on the target data alone in a word sense disambiguation task. Furthermore, the authors showed that using the source and 40% of the target data was able to obtain the same result as using the target data alone. In their experiment, both of the source and target domain data were manually labeled. Imamura et al. (2012) used the approach from Daumé III (2009) to improve grammar error correction by treating pseudo-error sentences as the source and real-error sentences as the target.

First we plan to annotate a small dataset from WikiAnswers based on the clas-

sification framework proposed in Bunescu and Huang (2010a): reformulated, useful, not useful. Next we plan to adapt the models trained on the manually labeled Yahoo Answers! dataset from Bunescu and Huang (2010a) to the small annotated WikiAnswers dataset.

We also plan to annotate a textual entailment dataset within several categories (e.g. animals, cars) from WikiAnswers, which is treated as the corpus in the source domain, and treat other categories as the target domain. We try to investigate whether incorporating unlabeled instances in the target domain can improve the performance of textual entailment classification. Similarly we plan to apply domain adaption techniques to question paraphrasing classification task using WikiAnswers data.

Bibliography

Eugene Agichtein. Web information extraction and user modeling: Towards closing the gap. *IEEE Data Engineering Bulletin*, 29(4):37–44, 2006.

Eugene Agichtein, Carlos Castillo, Debora Donato, Aristides Gionis, and Gilad Mishne. Finding high-quality content in social media. In *Proceedings of WSDM*, pages 183–194, New York, NY, USA, 2008. ACM.

Eneko Agirre and Oier Lopez de Lacalle. Supervised domain adaption for wsd. In *Proceedings of EACL*, pages 42–50, Stroudsburg, PA, USA, 2009. Association for Computational Linguistics.

Jinxi Xu Ana, Ana Licuanan, and Ralph Weischedel. Trec2003 qa at bbn: Answering definitional questions. In *Proceedings of TREC*, 2003.

Rie Kubota Ando and Tong Zhang. A framework for learning predictive structures from multiple tasks and unlabeled data. *Journal of Machine Learning Research*, 6:1817–1853, December 2005.

Charles Atkin. Instrumental utilities and information seeking. *New models for communication research*, pages 205C–242, 1973.

Ricardo Baeza-Yates and Berthier Ribeiro-Neto. *Modern Information Retrieval*. Addison Wesley Longman Publishing Co. Inc., 1999.

Collin F. Baker, Charles J. Fillmore, and John B. Lowe. The berkeley framenet project. In *Proceedins of COLING-ACL*, pages 86–90, 1998.

Regina Barzilay. *Information fusion for multidocument summarization: paraphrasing and generation*. PhD thesis, New York, NY, USA, 2003.

Delphine Bernhard and Iryna Gurevych. Answering learners' questions by retrieving question paraphrases from social q&a sites. In *Proceedings of the Third Workshop on Innovative Use of NLP for Building Educational Applications*, pages 44–52, Stroudsburg, PA, USA, 2008. Association for Computational Linguistics.

Jiang Bian, Yandong Liu, Eugene Agichtein, and Hongyuan Zha. A few bad votes too many?: towards robust ranking in social media. In *Proceedings of AIRWeb*, pages 53–60, New York, NY, USA, 2008a. ACM.

Jiang Bian, Yandong Liu, Eugene Agichtein, and Hongyuan Zha. Finding the right facts in the crowd: factoid question answering over social media. In *Proceedings of WWW*, pages 467–476, New York, NY, USA, 2008b. ACM.

John Blitzer, Ryan McDonald, and Fernando Pereira. Domain adaptation with structural correspondence learning. In *Proceedings of EMNLP*, pages 120C–128, 2006.

Ingwer Borg and Patrick Groenen. *Modern Multidimensional Scaling: Theory and Application (Second Edition)*. Springer Series in Statistics, 2005.

George E. P. Box, William G. Hunter, and J. Stuart Hunter. *Statistics for Experimenters*. John Wiley & Sons, 1978.

Eric Brill. Transformation-based error-driven learning and natural language

processing: A case study in part-of-speech tagging. *Computational Linguistics*, 21(4):543–565, 1995.

Andrei Broder. A taxonomy of web search. *SIGIR Forum*, 36(2):3–10, September 2002.

Peter F. Brown, Vincent J. Della Pietra, Stephen A. Della Pietra, and Robert L. Mercer. The mathematics of statistical machine translation: parameter estimation. *Computational Linguistics*, 19(2):263–311, June 1993.

Chris Buckley, Gerard Salton, James Allan, and Amit Singhal. Automatic query expansion using smart: Trec 3. In *Proceedings of TREC*, pages 0+, 1994.

Razvan Bunescu and Yunfeng Huang. A utility-driven approach to question ranking in social qa. In *Proceedings of COLING*, pages 125–133, Stroudsburg, PA, USA, 2010a. Association for Computational Linguistics.

Razvan Bunescu and Yunfeng Huang. Learning the relative usefulness of questions in community qa. In *Proceedings of EMNLP*, 2010b.

John Burger, Claire Cardie, Vinay Chaudhri, Robert Gaizauskas, Sanda Harabagiu, David Israel, Christian Jacquemin, C. Y. Lin, Steve Maiorano, George Miller, and Al. Issues , tasks and program structures to roadmap research in question & answering (q & a). *New York*, pages 1–35, 2001a.

John Burger, Claire Cardie, Vinay Chaudhri, Robert Gaizauskas, Sanda Harabagiu, David Israel, Christian Jacquemin, Chin-Yew Lin, Steve Maiorano, George Miller, Dan Moldovan, Bill Ogden, John Prager, Ellen Riloff, Amit Singhal, Rohini Shrihari, Tomek Strzalkowski, Ellen Voorhees, and

Ralph Weischedel. Issues, tasks and program structures to roadmap research in question & answering (Q&A). Technical report, National Institute of Standards and Technology, 2001b.

Chris Callison-Burch. *Paraphrasing and Translation*. PhD thesis, UK, 2007.

Donald O. Case. Amsterdam: Academic Press, 2002.

Asli Celikyilmaz, Dilek Hakkani-Tur, and Gokhan Tur. Lda based similarity modeling for question answering. In *Proceedings of HLT-NAACL 2010 Workshop on Semantic Search*, pages 1–9, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics.

Yllias Chali, Sadid A. Hasan, and Shafiq R. Joty. Improving graph-based random walks for complex question answering using syntactic, shallow semantic and extended string subsequence kernels. *Information Processing and Management*, 47(6):843–855, 2011.

Chih-Chung Chang and Chih-Jen Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2: 27:1–27:27, 2011.

Stephane Girard Charles Bouveyron and Cordelia Schmid. High-dimensional discriminant analysis. *Communications in Statistics: Theory and Methods*, 36:2607–2623, 2007.

Long Chen, Dell Zhang, and Levene Mark. Understanding user intent in community question answering. In *Proceedings of Proceedings of WWW (Companion Volume)*, pages 823–828, New York, NY, USA, 2012. ACM.

Charibeth Cheng, Cedric Paul Alberto, Ian Anthony Chan, and Vazir Joshua

Querol. Spellchef: Spelling checker and corrector for filipino. *Journal of Research in Science, Computing and Engineering*, 4, 2008.

Elisha Rufaro Tembo Chiware. *Business information needs, seeking patterns and information services in the small medium and micro enterprises sector (SMME) in Namibia*. PhD thesis, 2008.

Jennifer Chu-Carroll, Krzysztof Czuba, John Prager, and Sasha Blair-Goldensohn. Ibm's piquant ii in trec 2004. In *Proceedings of TREC*, 2003.

Shui-Lung Chuang and Lee-Feng Chien. A practical web-based approach to generating topic hierarchy for text segments. In *Proceedings of CIKM*, pages 127–136, New York, NY, USA, 2004. ACM.

Paul R. Cohen. *Empirical methods for artificial intelligence*. MIT Press, Cambridge, MA, USA, 1995.

Courtney Corley and Rada Mihalcea. Measuring the semantic similarity of texts. In *Proceedings of the ACL 2005 Workshop on Empirical Modeling of Semantic Equivalence and Entailment*, pages 13–18, Stroudsburg, PA, USA, 2005. Association for Computational Linguistics.

Hang Cui, Keya Li, Renxu Sun, Tat-Seng Chua, and Min-Yen Kan. National university of singapore at the trec-13 question answering main task. In *Proceedings of TREC*, 2003.

Ido Dagan, Oren Glickman, and Ramat Gan. The pascal recognising textual entailment challenge. *Lecture Notes in Computer Science: Machine Learning Challenges*, 2006.

Tiphaine Dalmas and Bonnie L. Webber. Answer comparison in automated question answering. *Journal of Applied Logic*, 5(1):104–120, 2007.

Hoa Trang Dang, Jimmy Lin, and Diane Kelly. Overview of the trec 2006 question answering track. In *Proceedings of TREC*, 2006.

Hoa Trang Dang, Diane Kelly, and Jimmy Lin. Overview of the trec 2007 question answering track. In *Proceedings of TREC*, 2007.

Hal Daumé III. Frustratingly easy domain adaptation. *CoRR*, 2009.

Marco De Boni. *Relevance in Open Domain Question Answering: Theoretical Framework and Application*. PhD thesis, UK, 2004.

Marco De Boni and Suresh Manandhar. An analysis of clarification dialogue for question answering. In *Proceedings of HLT-NAACL*, pages 48–55, Stroudsburg, PA, USA, 2003. Association for Computational Linguistics.

Marco De Boni and Suresh Manandhar. Implementing clarification dialogues in open domain question answering. *Journal of Natural Language Engineering*, 11(4):343–361, December 2005.

Brenda Dervin. From the minds eye of the user: the sense making qualitative and quantitative methodology in glazier. *Qualitative research in information management*, 1992.

Pierre Devijver and Josef Kittler. *Pattern Recognition: A statistical Approach*. Presentice Hall, 1982.

Inderjit Dhillon and Yuqiang Guan. Information theoretic clustering of sparse co-occurrence data. In *Proceedings of ICDM*, page 517, 2003.

Huizhong Duan, Yunbo Cao, Chin-Yew Lin, and Yong Yu. Searching questions by identifying question topic and question focus. In *Proceedings of ACL*, pages 156–164, 2008.

Jennifer G. Dy and Carla E. Brodley. Feature selection for unsupervised learning. *Journal of Machine Learning Research*, 5:845–889, December 2004.

Bradley Efron and R. J. Tibshirani. *An Introduction to the Bootstrap (Chapman & Hall/CRC Monographs on Statistics & Applied Probability)*. Chapman and Hall/CRC, 1 edition, may 1994.

David Ferrucci, Eric Nyberga, James Allan, Ken Barker, Eric Brown, Jennifer Chu-Carroll, Arthur Ciccolo, Pablo Duboue, James Fan, David Gondek, Eduard Hovy, Boris Katz, Adam Lally, Michael McCord, Paul Morarescu, Bill Murdock, Bruce Porter, John Prager, Tomek Strzalkowski, Chris Welty, and Wlodek Zadrozny. Towards the open advancement of question answering systemstowards the open advancement of question answering systems. Technical report, New York, USA, 2009.

David A. Ferrucci, Eric W. Brown, Jennifer Chu-Carroll, James Fan, David Gondek, Aditya Kalyanpur, Adam Lally, J. William Murdock, Eric Nyberg, John M. Prager, Nico Schlaefer, and Christopher A. Welty. Building watson: An overview of the deepqa project. *AI Magazine*, 31(3):59–79, 2010a.

David A. Ferrucci, Eric W. Brown, Jennifer Chu-Carroll, James Fan, David Gondek, Aditya Kalyanpur, Adam Lally, J. William Murdock, Eric Nyberg, John M. Prager, Nico Schlaefer, and Christopher A. Welty. Building watson: An overview of the deepqa project. *AI Magazine*, pages 59–79, 2010b.

Karel Fuka and Rudolf Hanka. Feature set reduction for document classification problems. In *Proceedings of IJCAI Workshop: Text Learning: Beyond Supervision*, USA, 2001.

Rich Gazan. Social q&a. *American Society for Information Science & Technology*, 62:2301–2312, 2011.

Oren Glickman, Ido Dagan, and Moshe Koppel. Web based probabilistic textual entailment. In *In Proceedings of RTE 2005 Workshop*, 2005.

Eric J. Glover, Kostas Tsioutsiouliklis, Steve Lawrence, David M. Pennock, and Gary W. Flake. Using web structure for classifying and describing web pages. In *Proceedings of WWW*, pages 562–569, New York, NY, USA, 2002. ACM.

Bert F. Green, Jr., Alice K. Wolf, Carol Chomsky, and Kenneth Laughery. Baseball: an automatic question-answerer. In *Proceedings of IRE-AIEE-ACM (Western)*, pages 219–224, New York, NY, USA, 1961. ACM.

Thoma L. Griffiths and Mark Steyvers. Finding scientific topics. *National Academy of Sciences*, 101(Suppl. 1):5228–5235, April 2004.

Kyoung-Soo Han, Young-In Song, and Hae-Chang Rim. Probabilistic model for definitional question answering. In *Proceedings of SIGIR*, pages 212–219, New York, NY, USA, 2006. ACM.

Sanda Harabagiu and Andrew Hickl. Methods for using textual entailment in open-domain question answering. In *Proceedings of COLING/ACL*, pages 905–912, 2006.

Sanda Harabagiu, Dan Moldovan, Marius Paşca, Mihai Surdeanu, Rada Mihalcea, Roxana Gîrju, Vasile Rus, Finley Lăcătuşu, Paul Morărescu, and Răzvan Bunescu. Answering complex, list and context questions with LCC's question-answering server. In Ellen M. Voorhees and Donna Harman, editors, *Proceedings of TREC*, pages 355–361, 2001.

Sanda Harabagiu, Dan Moldovan, Christine Clark, Mitchell Bowden, Andrew Hickl, and Patrick Wang. Employing two question answering systems in trec-2005. In *Proceedings of TREC*, 2005.

Sanda M. Harabagiu, Dan I. Moldovan, Marius Pasca, Rada Mihalcea, Mihai Surdeanu, Razvan C. Bunescu, Roxana Girju, Vasile Rus, and Paul Morarescu. Falcon: Boosting knowledge for answer engines. In *Proceedings of TREC*, 2000.

Sanda M. Harabagiu, Dan I. Moldovan, Christine Clark, Mitchell Bowden, John Williams, and Jeremy Bensley. Answer mining by combining extraction techniques with abductive reasoning. In *Proceedings of TREC*, pages 375–382, 2003.

Andrew Hickl, Patrick Wang, John Lehmann, and Sanda M. Harabagiu. Ferret: Interactive question-answering for real-world environments. In *Proceedings of ACL*, 2006a.

Andrew Hickl, John Williams, Jeremy Bensley, Kirk Roberts, Ying Shi, and Bryan Rink. Question answering with lccs chaucer at trec 2006. In *Proceedings of TREC*, 2006b.

Lynette Hirschman and Robert Gaizauskas. Natural language question answering: the view from here. *Journal of Natural Language Engineering*, 7 (4):275–300, December 2001.

Chiori Hori, Takaaki Hori, Hajime Tsukada, Hideki Isozaki, Yutaka Sasaki, and Eisaku Maeda. Spoken interactive odqa system: Spiqa. In *Proceedings of ACL*, pages 153–156, Stroudsburg, PA, USA, 2003. Association for Computational Linguistics.

Eduard Hovy, Ulf Hermjakob, and Deepak Ravichandran. A question/answer typology with surface text patterns. In *Proceedings of HLT*, pages 247–251, San Francisco, CA, USA, 2002. Morgan Kaufmann Publishers Inc.

Kenji Imamura, Kuniko Saito, Kugatsu Sadamitsu, and Hitoshi Nishikawa. Grammar error correction using pseudo-error sentences and domain adaptation. In *Proceedings of ACL*, pages 388–392, Stroudsburg, PA, USA, 2012. Association for Computational Linguistics.

Jiwoon Jeon, W. Bruce Croft, and Joon Ho Lee. Finding semantically similar questions based on their answers. In *Proceedings of SIGIR*, pages 617–618, 2005a.

Jiwoon Jeon, W. Bruce Croft, and Joon Ho Lee. Finding similar questions in large question and answer archives. In *Proceedings of CIKM*, pages 84–90, New York, NY, USA, 2005b. ACM.

Jiwoon Jeon, W. Bruce Croft, Joon Ho Lee, and Soyeon Park. A framework to predict the quality of answers with non-textual features. In *Proceedings of SIGIR*, pages 228–235, New York, NY, USA, 2006. ACM.

Jay J. Jiang and David W. Conrath. Semantic similarity based on corpus statistics and lexical taxonomy. In *Proceedings of COLING*, 1997.

Jing Jiang and ChengXiang Zhai. Instance weighting for domain adaptation in nlp. In *Proceedings of ACL*, Prague, Czech Republic, June 2007.

Sungkyu Jung and Steve Marron. Pca consistency in high dimension, low sample size context. *The Annals of Statistics*, 37:4104–4130, 2009.

Pawel Jurczyk and Eugene Agichtein. Discovering authorities in question answer communities by using link analysis. In *Proceedings of CIKM*, pages 919–922, New York, NY, USA, 2007. ACM.

Michael Kaisser, Silke Scheible, and Bonnie Webber. Experiments at the u-

niversity of edinburgh for the trec 2006 qa track. In *Proceedings of TREC*, 2006.

Erik Kamsties. *Surfacing Ambiguity in Natural Language Requirements*. PhD thesis, 2001.

Boris Katz, Gregory Marton, Sue Felshin, Daniel Loreto, Ben Lu, Federico Mora, Ozlem Uzuner, Michael McGraw-Herdeg, Natalie Cheung, Yuan Luo, Alexey Radul, Yuan Shen, and Gabriel Zaccak. Question answering experiments and resources. In *Proceedings of TREC*, 2006.

Manuel Kirschner and Raffaella Bernardi. An empirical view on iqa follow-up questions. In *Proceedings of SIGdial Workshop on Discourse and Dialogue*, 2007.

Dan Klein and Christopher D. Manning. Accurate unlexicalized parsing. In *Proceedings of ACL*, pages 423–430, Stroudsburg, PA, USA, 2003. Association for Computational Linguistics.

Milen Kouylekov and Bernardo Magnini. Tree edit distance for recognizing textual entailment: Estimating the cost of insertion. In *Proc. of the PASCAL RTE-2 Challenge*, pages 68–73, 2006.

Kui-Lam Kwok, Laszlo Grunfeld, Norbert Dinstl, and M. Chan. Trec-9 cross language, web and question-answering track experiments using pircs. In *Proceedings of TREC*, pages 26–35, 2000.

Pat Langley. Selection of relevant features in machine learning. In *Proceedings of the AAAI Fall symposium on relevance*, pages 140–144. AAAI Press, 1994.

Victor Lavrenko and W. Bruce Croft. Relevance based language models. In *Proceedings of SIGIR*, pages 120–127, New York, NY, USA, 2001. ACM.

Dawn J. Lawrie and W. Bruce Croft. Generating hierarchical summaries for web searches. In *Proceedings of SIGIR*, pages 457–458, New York, NY, USA, 2003. ACM.

Claudia Leacock, George A. Miller, and Martin Chodorow. Using corpus statistics and wordnet relations for sense identification. *Computational Linguistics*, 24(1):147–165, March 1998.

Jung-Tae Lee, Sang-Bum Kim, Young-In Song, and Hae-Chang Rim. Bridging lexical gaps between queries and questions on large online q&a collections with compact translation models. In *Proceedings of EMNLP*, pages 410–418, Stroudsburg, PA, USA, 2008. Association for Computational Linguistics.

Fangtao Li, Sinno Jialin Pan, Ou Jin, Qiang Yang, and Xiaoyan Zhu. Cross-domain co-extraction of sentiment and topic lexicons. In *Proceedings of ACL*, pages 410–419, 2012.

Shuguang Li and Suresh Manandhar. Automatic generation of information-seeking questions using concept clusters. In *Proceedings of ACL-IJCNLP Short Papers*, pages 93–96, Stroudsburg, PA, USA, 2009. Association for Computational Linguistics.

Shuguang Li and Suresh Manandhar. Improving question recommendation by exploiting information need. In *Proceedings of ACL*, pages 1425–1434, Stroudsburg, PA, USA, 2011. Association for Computational Linguistics.

Dekang Lin. Principle-based parsing without overgeneration. In *Proceedings of ACL*, pages 112–120, Stroudsburg, PA, USA, 1993. Association for Computational Linguistics.

Dekang Lin. An information-theoretic definition of similarity. In *Proceedings of ICML*, pages 296–304, San Francisco, CA, USA, 1998. Morgan Kaufmann Publishers Inc.

Dekang Lin and Patrick Pantel. Dirt discovery of inference rules from text. In *Proceedings of KDD*, pages 323–328, New York, NY, USA, 2001. ACM.

Kenneth C. Litkowski. Exploring document content with xml to answer questions. In *Proceedings of TREC*, 2005.

Qiaoling Liu and Eugene Agichtein. Modeling answerer behavior in collaborative question answering systems. In *Proceedings of ECIR*, pages 67–79, Berlin, Heidelberg, 2011. Springer-Verlag.

Qiaoling Liu, Yandong Liu, and Eugene Agichtein. Exploring web browsing context for collaborative question answering. In *Proceedings of IIiX*, pages 305–310, New York, NY, USA, 2010. ACM.

Qiaoling Liu, Eugene Agichtein, Gideon Dror, Evgeniy Gabrilovich, Yoelle Maarek, Dan Pelleg, and Idan Szpektor. Predicting web searcher satisfaction with existing community-based answers. In *Proceedings of SIGIR*, pages 415–424, New York, NY, USA, 2011. ACM.

Yandong Liu and Eugene Agichtein. You've got answers: towards personalized models for predicting success in community question answering. In *Proceedings of ACL: Short Papers*, pages 97–100, Stroudsburg, PA, USA, 2008. Association for Computational Linguistics.

Yandong Liu, Jiang Bian, and Eugene Agichtein. Predicting information seeker satisfaction in community question answering. In *Proceedings of SIGIR*, pages 483–490, New York, NY, USA, 2008. ACM.

Ian MacKinnon and Olga Vechtomova. Complex interactive question answering enhanced with wikipedia. In *Proceedings of TREC*, 2007.

James MacQueen. Some methods for classification and analysis of multivariate observations. In *Proceedings of the 5th Berkeley Symposium on Mathematical Statistics and Probability*, volume 1, pages 281–297. Univ. of Calif. Press, 1967.

Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. *Introduction to Information Retrieval*. Cambridge University Press, New York, NY, USA, 2008.

Ryan McDonald. *Discriminative training and spanning tree algorithms for dependency parsing*. PhD thesis, 2006.

Donald Metzler, Yaniv Bernstein, W. Bruce Croft, Alistair Moffat, and Justin Zobel. Similarity measures for tracking information flow. In *Proceedings of CIKM*, pages 517–524, New York, NY, USA, 2005. ACM.

R. Mihalcea, C. Corley, and C. Strapparava. Corpus-based and knowledge-based measures of text semantic similarity. In *Proceedings of AAAI*, June 2006.

George A. Miller. Wordnet: a lexical database for english. *Communications of The ACM*, 38(11):39–41, November 1995.

Dunja Mladenić. Feature selection for dimensionality reduction. *Subspace, Latent Structure and Feature Selection*, pages 84–102, 2006.

Dan Moldovan, Mitchell Bowden, and Marta Tatu. A temporally-enhanced poweranswer in trec 2006. In *Proceedings of TREC*, 2006.

Dan Moldovan, Christine Clark, and Mitchell Bowden. Lymbas poweranswer 4 in trec 2007. In *Proceedings of TREC*, 2007.

Dan I. Moldovan, Sanda M. Harabagiu, Roxana Girju, Paul Morarescu, V. Finley Lacatusu, Adrian Novischi, Adriana Badulescu, and Orest Bolohan. Lcc tools for question answering. In *Proceedings of TREC*, 2002.

Dan I. Moldovan, Christine Clark, Sanda M. Harabagiu, and Steven J. Maiorano. Cogex: A logic prover for question answering. In *Proceedings of HLT-NAACL*, 2003.

Dan I. Moldovan, Christine Clark, and Sanda M. Harabagiu. Temporal context representation and reasoning. In *Proceedings of IJCAI*, pages 1099–1104, 2005.

Christof Monz and Maarten de Rijke. Lightweight entailment checking for computational semantics. In *Proceedings of the ICICS*, 2001.

Raphael Mudge. The design of a proofreading software service. In *Proceedings of HLT-NAACL Workshop on Computational Linguistics and Writing: Writing Processes and Authoring Aids*, pages 24–32, Morristown, NJ, USA, 2010. Association for Computational Linguistics.

Franz Josef Och and Hermann Ney. A comparison of alignment models for statistical machine translation. In *Proceedings of COLING*, pages 1086–1090, Stroudsburg, PA, USA, 2000. Association for Computational Linguistics.

Franz Josef Och and Hermann Ney. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51, mar 2003.

Sebastian Padó and Mirella Lapata. Dependency-based construction of semantic space models. *Computational Linguistics*, 33(2):161–199, June 2007.

Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The pagerank citation ranking: Bringing order to the web. Technical report, Stanford InfoLab, 1999.

Martha Palmer, Daniel Gildea, and Paul Kingsbury. The proposition bank: An annotated corpus of semantic roles. *Computational Linguistics*, 31(1): 71–106, March 2005.

Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10):1345–1359, October 2010.

Patrick Pantel and Deepak Ravichandran. Automatically labeling semantic classes. In *Proceedins of the 2004 HLT-NAACL*, pages 321–328, 2004.

Patrick Andre Pantel. *Clustering by committee*. PhD thesis, Edmonton, Alta., Canada, 2003. AAINQ82151.

Marius Pasca and Benjamin Van Durme. Weakly-supervised acquisition of open-domain classes and class attributes from web documents and query logs. In *Proceedings of ACL*, pages 19–27, 2008.

Marius Pasca and Sanda H. Harabagiu. The informative role of wordnet in open-domain question answering. In *Proceedings of HLT-NAACL Workshop on WordNet and Other Lexical Resources: Applications, Extensions and Customizations*, 2001.

Alexandrin Popescul and Lyle H. Ungar. Automatic labeling

of document clusters. Unpublished manuscript, available at: http://citeseer.nj.nec.com/popescul00automatic.html, 2000.

Sameer Pradhan, Lance Ramshaw, Mitchell Marcus, Martha Palmer, Ralph Weischedel, and Nianwen Xue. Conll-2011 shared task: modeling unrestricted coreference in ontonotes. In *Proceedings of CoNLL: Shared Task*, pages 1–27, Stroudsburg, PA, USA, 2011. Association for Computational Linguistics.

Xipeng Qiu, Bo Li, Chao Shen, Lide Wu, Xuanjing Huang, and Yaqian Zhou. Fduqa on trec2007 qa track. In *Proceedings of TREC*, 2007.

Silvia Quartertoni. *Advanced Techniques For Personalized, Interactive. Question Answering.* PhD thesis, UK, 2007.

Stefan Riezler, Alexander Vasserman, Ioannis Tsochantaridis, Vibhu O. Mittal, and Yi Liu. Statistical machine translation for query expansion in answer retrieval. In *Proceedings of ACL*, 2007.

Stephen E. Robertson, Steve Walker, Susan Jones, Micheline Hancock-Beaulieu, and Mike Gatford. Okapi at trec-3. In *Proceedings of TREC*, 1994.

J. Rocchio. *Relevance Feedback in Information Retrieval.* 1971.

Daniel E. Rose and Danny Levinson. Understanding user goals in web search. In *Proceedings of WWW*, pages 13–19, New York, NY, USA, 2004. ACM.

Klaus Rothenhäusler and Hinrich Schütze. Unsupervised classification with dependency based word spaces. In *Proceedings of the Workshop on Geometrical Models of Natural Language Semantics*, pages 17–24, Athens, Greece, March 2009. Association for Computational Linguistics.

Gerard Salton and Michael J. McGill. *Introduction to Modern Information Retrieval*. McGraw-Hill, 1983.

Nico Schlaefer, Jeongwoo Ko, Justin Betteridge, Guido Sautter, Manas Pathak, and Eric Nyberg. Semantic extensions of the ephyra qa system for trec 2007. In *Proceedings of TREC*, 2007.

Karin Kipper Schuler. *Verbnet: a broad-coverage, comprehensive verb lexicon*. PhD thesis, Philadelphia, PA, USA, 2005.

Chirag Shah, Sanghee Oh, and Jung S. Oh. Research agenda for social q&a. *Library & Information Science Research*, 31(4):205–209, December 2009.

Ben Shneiderman, Don Byrd, and W. Bruce Croft. Clarifying search: A user-interface framework for text searches. *D-Lib Magazine*, pages 3–10, January 1997.

Robert F. Simmons. Answering english questions by computer: a survey. *Communications of the ACM*, 8(1):53–70, January 1965.

Sharon Small, Tomek Strzalkowski, Ting Liu, Sean Ryan, Robert Salkin, Nobuyuki Shimizu, Paul B. Kantor, Diane Kelly, Robert Rittman, and Nina Wacholder. Hitiqa: Towards analytical question answering. In *Proceedings of COLING*, 2004.

Mark D. Smucker, James Allan, and Ben Carterette. A comparison of statistical significance tests for information retrieval evaluation. In *Proceedings of CIKM*, pages 623–632, New York, NY, USA, 2007. ACM.

Young-In Song, Chin-Yew Lin, Yunbo Cao, and Hae-Chang Rim. Question utility: a novel static ranking of question search. In *Proceedings of AAAI*, pages 1231–1236. AAAI Press, 2008.

Martion M. Soubbotin. Patterns of potential answer expressions as clues to the right answers. In *Proceedings of TREC*, pages 293–302, 2001.

Vivek Srikumar and Dan Roth. A joint model for extended semantic role labeling. In *Proceedings of EMNLP*, pages 129–139, Stroudsburg, PA, USA, 2011. Association for Computational Linguistics.

Benno Stein and Sven Meyer Zu Eissen. Topic identification: Framework and application. In *Proceedings of ICKM*, 2004.

Renxu Sun, Jing Jiang, Yee Fan Tan, Hang Cui, Tat-Seng Chua, and Min-Yen Kan. Using syntactic and semantic relation analysis in question answering. In *Proceedings of TREC*, 2005.

Robert S. Taylor. The process of asking questions. *American Documentation*, 13(4):391–396, 1962.

Pucktada Treeratpituk and James P. Callan. An experimental study on automatically labeling hierarchical clusters using statistical features. In *Proceedings of SIGIR*, pages 707–708, 2006.

Christina Unger and Philipp Cimiano. Representing and resolving ambiguities in ontology-based question answering. In *Proceedings of TIWTE*, pages 40–49, Stroudsburg, PA, USA, 2011. Association for Computational Linguistics.

Olga Vechtomova and Murat Karamuftuoglu. Identifying relationships between entities in text for complex interactive question answering task. In *Proceedings of TREC*, 2006.

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

Ellen M. Voorhees. Overview of the trec-9 question answering track. In *Proceedings of TREC*, 2000.

Ellen M. Voorhees. Overview of the trec 2001 question answering track. In *Proceedings of TREC*, pages 42–51, 2001.

Ellen M. Voorhees. Overview of the trec 2002 question answering track. In *Proceedings of TREC*, 2002.

Ellen M. Voorhees. Overview of the trec 2003 question answering track. In *Proceedings of TREC*, pages 54–68, 2003.

Ellen M. Voorhees. Overview of the trec 2004 question answering track. In *Proceedings of TREC*, 2004.

Ellen M. Voorhees and Hoa Trang Dang. Overview of the trec 2005 question answering track. In *Proceedings of TREC*, 2005.

Ellen M. Voorhees and Dawn M. Tice. The trec-8 question answering track. In *Proceedings of LREC*, 2000.

Kai Wang and Tat-Seng Chua. Exploiting salient patterns for question detection and question retrieval in community-based question answering. In *Proceedings of COLING*, pages 1155–1163, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics.

Kai Wang, Zhaoyan Ming, and Tat-Seng Chua. A syntactic tree matching approach to finding similar questions in community-based qa services. In *Proceedings of SIGIR*, pages 187–194, New York, NY, USA, 2009a. ACM.

Rui Wang and Günter Neumann. Recognizing textual entailment using a subsequence kernel method. In *Proceedings of AAAI*, pages 937–942. AAAI Press, 2007.

Rui Wang, Yi Zhang, and Guenter Neumann. A joint syntactic-semantic representation for recognizing textual relatedness. In *Proceedings of TAC/RTE-5*, 2009b.

Joseph Weizenbaum. Eliza a computer program for the study of natural language communication between man and machine. *Communications of the ACM*, 9(1):36–45, January 1966.

Tom D. Wilson. On user studies and information needs. *Journal of Documentation*, 37:3–15, 1981.

Tom D. Wilson. Information needs and uses: fifty years of progress. information seeking behaviour, behavior, information needs, information use. In B. C. Vickery, editor, *Fifty years of information progress: a Journal of Documentation review*, pages 15–51. Aslib, London, 1994.

William A. Woods. Progress in natural language understanding: an application to lunar geology. In *Proceedings of AFIPS*, pages 441–450, New York, NY, USA, 1973. ACM.

Hu Wu, Yongji Wang, and Xiang Cheng. Incremental probabilistic latent semantic analysis for automatic question recommendation. In *Proceedings of RecSys*, pages 99–106, New York, NY, USA, 2008. ACM.

Lide Wu, Xuanjing Huang, Lan You, Zhushuo Zhang, Xin Li, and Yaqian Zhou. Fduqa on trec2004 qa track. In *Proceedings of TREC*, 2004.

Xiaobing Xue, Jiwoon Jeon, and W. Bruce Croft. Retrieval models for question and answer archives. In *Proceedings of SIGIR*, pages 475–482, New York, NY, USA, 2008. ACM.

Hui Yang, Hang Cui, Mstislav Maslennikov, Long Qiu, Min-Yen Kan, and Tat-Seng Chua. Qualifier in trec-12 qa main task. In *Proceedings of TREC*, 2003.

Yiming Yang and Xin Liu. A re-examination of text categorization methods. In *Proceedings of SIGIR*, pages 42–49, New York, NY, USA, 1999. ACM.

Yiming Yang and Jan O. Pedersen. A comparative study on feature selection in text categorization. In *Proceedings of ICML*, pages 412–420, San Francisco, CA, USA, 1997. Morgan Kaufmann Publishers Inc.

Lei Yu and Huan Liu. Feature selection for high-dimensional data: A fast correlation-based filter solution. In *Proceedings of ICML*, pages 856–863, Washington, D.C., 2003.

Chen Zhang, Matthew Gerber, Tyler Baldwin, Steve Emelander, Joyce Y. Chai, and Rong Jin. Michigan state university at the 2007 trec ciqa evaluation. In *Proceedings of TREC*, 2007a.

Dell Zhang and Wee Sun Lee. A language modeling approach to passage question answering. In *Proceedings of TREC*, 2003.

Jun Zhang, Mark S. Ackerman, and Lada Adamic. Expertise networks in online communities: structure and algorithms. In *Proceedings of WWW*, pages 221–230, New York, NY, USA, 2007b. ACM.

Yi Zhang, Rui Wang, and Hans Uszkoreit. Hybrid learning of dependency structures from heterogeneous linguistic resources. In *Proceedings of CoNLL*, pages 198–202, Stroudsburg, PA, USA, 2008. Association for Computational Linguistics.

Shiqi Zhao, Ming Zhou, and Ting Liu. Learning question paraphrases for qa from encarta logs. In *Proceedings of IJCAI*, pages 1795–1800, San Francisco, CA, USA, 2007. Morgan Kaufmann Publishers Inc.

Ying Zhao and George Karypis. Criterion functions for document clustering: Experiments and analysis. Technical report, University of Minnesota, 2002.

Tom Chao Zhou, Chin-Yew Lin, Irwin King, Michael R. Lyu, Young-In Song, and Yunbo Cao. Learning to suggest questions in online forums. In *Proceedings of AAAI*, 2011.

Yaqian Zhou, Xiaofeng Yuan, Junkuo Cao, Xuanjing Huang, and Lide Wu. Fduqa on trec2006 qa track. In *Proceedings of TREC*, 2006.