COMPLEX QUESTION ANSWERING: MINIMIZING THE GAPS AND BEYOND

SHEIKH SADID AL HASAN
Master of Science
University of Lethbridge, 2010

A Thesis
Submitted to the School of Graduate Studies
of the University of Lethbridge
in Partial Fulfillment of the
Requirements for the Degree

DOCTOR OF PHILOSOPHY

Department of Mathematics and Computer Science
University of Lethbridge
LETHBRIDGE, ALBERTA, CANADA

COMPLEX QUESTION ANSWERING: MINIMIZING THE GAPS AND BEYOND

SHEIKH SADID AL HASAN

Approved:

Name	*Signature*	*Rank*	*Date*
Dr. Yllias Chali Supervisor:			
Dr. Kevin Grant Committee Member:			
Dr. Dragomir Radev Committee Member:			
Dr. Sajjad Zahir Committee Member:			
Dr. Guy Lapalme External Examiner:			
Dr. Howard Cheng Chair, Thesis Examination Committee:			

This work is dedicated to my beloved parents and my dearest wife, Tanzila.

Abstract

Current Question Answering (QA) systems have been significantly advanced in demonstrating finer abilities to answer simple factoid and list questions. Such questions are easier to process as they require small snippets of texts as the answers. However, there is a category of questions that represents a more complex information need, which cannot be satisfied easily by simply extracting a single entity or a single sentence. For example, the question: "How was Japan affected by the earthquake?" suggests that the inquirer is looking for information in the context of a wider perspective. We call these *"complex questions"* and focus on the task of answering them with the intention to minimize the existing gaps in the literature.

The major limitation of the available search and QA systems is that they lack a way of measuring whether a user is satisfied with the information provided. This was our motivation to propose a reinforcement learning formulation to the complex question answering problem. Next, we presented an integer linear programming formulation where sentence compression models were applied for the query-focused multi-document summarization task in order to investigate if sentence compression improves the overall performance. Both compression and summarization were considered as global optimization problems. We also investigated the impact of syntactic and semantic information in a graph-based random walk method for answering complex questions. Decomposing a complex question into a series of simple questions and then reusing the techniques developed for answering simple questions is an effective means of answering complex questions. We proposed a supervised approach for automatically learning good decompositions of complex questions in this work. A complex question often asks about a topic of user's interest. Therefore, the problem of complex question decomposition closely relates to the problem of topic to question generation. We addressed this challenge and proposed a topic to question generation approach to enhance the scope of our problem domain.

Acknowledgments

All praise be to the Almighty who gave me the opportunity to survive in this world. Without his wish nothing is made possible. Other than that, I believe that I am able to finish this work just because of continuous encouragement from my beloved mother and father, although they were more than seven thousand miles away from me. I find no words to thank them for that.

I am very much grateful to my supervisor, Dr. Yllias Chali, who always inspired me to work hard and be in the right track. I cordially thank him for that.

I also thank my doctoral committee members Dr. Kevin Grant, Dr. Dragomir Radev, Dr. Sajjad Zahir, Dr. Guy Lapalme, and Dr. Howard Cheng for their valuable suggestions and guidance.

I received many helpful comments and suggestions from the anonymous reviewers during the submission process at different conferences and journals: COLING-2012, CIKM-2012, NLDB-2012, IUI-2011, CAI-2011, AIRS-2011, IJCNLP-2011, JNLE, and IPM. I owe my sincere gratitude to them.

I am also thankful to my dearest wife, Tanzila. I met her at the very start of this journey, and it was she who became my sole inspiration since then and was by my side through all the difficult times. I feel very lucky for that.

Lastly, I thank NSERC and the University of Lethbridge for their financial assistance, without which it would not have been possible for me to carry out the whole work.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Focus

This thesis is concerned with automatic answering of complex questions. Specifically, we focus on answering such complex questions that are essentially broader information requests about a certain topic. These are the questions whose answers need to be obtained from pieces of information scattered in multiple documents. For example, consider the following question:

Describe steps taken and worldwide reaction prior to the introduction of the Euro on January 1, 1999. Include predictions and expectations reported in the press.

This question is clearly requesting an elaboration about the topic "Introduction of the Euro", which can be answered by inferencing and synthesizing information from multiple documents. For example, an ideal expected answer (human-generated) to this question is as follows:

Most predictions prior to introduction of the euro on Jan. 1, 1999 were positive. By early 1996 a majority of Europeans accepted the idea and France's Prime Minister was strongly supportive. By 1997 most British commentary was favorable. In Germany the Bundesbank predicted that private investors would profit. Bank officials as far away as Zambia saw benefits. By 1998 the Chinese government had officially welcomed coming of the euro and the European Central Bank President predicted that the euro would eventually rival the dollar. Design of the euro note began in 1996. In 1998 the test printing of banknotes began and Germany minted some euro coins. By the end of the year Germany was still experimenting with euro coins but France was in full production followed by Finland, Belgium and Spain. In May 1998 two major banks began quoting prices in euros. In

October Bulgaria linked its currency, the lev, to the German mark, followed in December by fixing the value of the lev to the euro. In September key euro indicators were offered daily on the internet and in October Thailand announced that it was considering use of the euro in its foreign reserves. In November the Reserve Bank of India permitted bank transactions in euros followed in December by Romania's posting the euro in its rates of exchange. Small independent countries such as Monaco still faced some currency problems as Jan. 1, 1999 approached, but special arrangements were in process by European Union authorities.

From this example, it is clearly visible that the kind of complex questions we are dealing with in this thesis certainly cannot be answered by a single entity or a single sentence rather a query-focused summary of the source documents can essentially serve the purpose (Chali, Joty, and Hasan, 2009). This means that in contrast to complex questions there is obviously another category of questions, that can be termed as *simple* questions. For example, the question: *"Who is the president of Bangladesh?"* asks for a person's name. Again, the question: *"Which countries has Pope John Paul II visited?"* is a sample of a list question asking only for a list of small snippets of text. These questions are easier to answer as they require a single entity or a single sentence or small snippets of texts as the answers. On the other hand, as noted before, complex questions (such as for example, *"How was Japan affected by the earthquake?"*) suggest that the inquirer is looking for information in the context of a wider perspective and hence, answering such questions requires selecting important snippets of information from different parts of the document collection and synthesizing them. It is important to note that there are other possible ways of being a *complex* question apart from our definition of complex questions. For instance, *double-questions* (e.g. *"How old was Mozart when he died?"*, *"Who was the US president during the first World War?"*, *"How high is the highest US mountain?"*) are a completely different type of complex questions that also need complex analysis of the document to answer them.

These questions fall outside the scope of this thesis and should be an attractive subject for exploration in the future.

The experiments and evaluations conducted in this thesis are mainly influenced by the specific scenario proposed by the Document Understanding Conference (DUC[1]) (2005-2007) tasks. DUC has been conducted by the National Institute of Standards and Technology (NIST) since 2001. Although its vision was to progress in automatic text summarization, the query-focused summarization task proposed in DUC (2005-2007) was appropriate to simulate our complex question answering experiments. The query-focused summarization task was proposed to model a real-world complex question answering problem, in which a question cannot be answered by simply stating a name, date, quantity, etc. The task provided a simplification of real tasks with the assumption that a set of relevant documents containing the answer to the complex question are already given. It would be of our future interest to see how our proposed methods work on a document pool of lower quality. The DUC corpus is used for most of our experiments, which is comprised of newswire articles collected from several notable news agencies (e.g. the Associated Press, New York Times, Xinhua News Agency etc.). Specific descriptions of the corpora used in different experiments are presented in the later chapters of the thesis.

1.2 Background

We live in an information age now, where all kinds of information is easily accessible through the Internet. The increasing demand for access to different types of information available online have actually led researchers to an interest in a broad range of Information Retrieval (IR) related areas such as question answering, topic detection and tracking, summarization, multimedia retrieval, chemical and biological informatics, text structuring,

[1] http://duc.nist.gov/

and text mining. The traditional document retrieval systems cannot satisfy the end-users' information need to have more direct access into relevant documents. Question Answering (QA) systems can address this challenge effectively (Strzalkowski and Harabagiu, 2008) and for this reason, QA has received immense attention from the information retrieval, information extraction, machine learning, and natural language processing communities in the last 15 years (Hirschman and Gaizauskas, 2001).

The main goal of QA systems is to retrieve relevant answers to natural language questions from a collection of documents rather than employing keyword matching techniques to extract documents. A popular QA system in Korea, the Korean Naver's Knowledge iN search[2], allows users to ask almost any question and get answers from other users (Chali, Joty, and Hasan, 2009). Another widely known QA service is Yahoo! Answers[3] which is a community-driven knowledge market website launched by Yahoo!. As of December 2009, Yahoo! Answers had 200 million users worldwide and more than 1 billion answers[4]. Furthermore, Google launched a QA system[5] in April 2002 that was based on paid editors. However, the system was closed in December 2006. The main limitation of these QA systems is that they rely on human expertise to help provide the answers.

Automated QA research focuses on how to respond with exact answers to a wide variety of questions including: factoid, list, definition, how, why, hypothetical, semantically-constrained, and cross-lingual questions. Current QA systems have been significantly advanced in demonstrating finer abilities to answer simple factoid and list questions (Moldovan, Clark, and Bowden, 2007). For example, START[6], the world's first Web-based question answering system, can effectively answer millions of English questions about places, movies,

[2] http://kin.naver.com/

[3] http://answers.yahoo.com/

[4] http://yanswersblog.com/index.php/archives/2009/12/14/yahoo-answers-hits-200-million-visitors-worldwide/

[5] http://answers.google.com/

[6] http://start.csail.mit.edu/

people etc. Another well-known intelligent QA system is IBM's Watson[7], which is capable of answering natural language questions and is specifically developed to answer questions on the popular quiz show *Jeopardy!*.

Researchers' interest in developing natural language QA systems is not new. Simmons (1965) have surveyed a considerable number of English language QA systems developed between 1960 and 1965. A substantial amount of research has been conducted between the period of 1970 and 1990 in order to develop different types of QA systems that could be categorized as natural language database systems, dialogue systems, and reading comprehension systems (Greenwood, 2005). After that, researchers focused on answering simple factoid questions (e.g. *When was Barack Obama born?*, *In what year did Canada become an independent country?*) (Hirschman and Gaizauskas, 2001). One of the pioneer systems designed for open-domain factoid question answering was MURAX (Kupiec, 1993), which was built to draw answers to factoid questions from an on-line encyclopedia. There is a shift in research trend from answering simple factoid questions towards more complex type of questions such as definitional (e.g. *What is an adjective?*), list (e.g. *List the provinces of Canada*), scenario-based (i.e. extracting answers from a given scenario description to questions based on the scenario), and why-type questions (e.g. *Why is the sky blue?*) (Wang, 2006). In 1999, an annual QA evaluation track[8] was introduced at the Text Retrieval Conference (TREC), which caused a huge acceleration in QA research (Voorhees, 1999). The success of the TREC QA track led to the introduction of multilingual and cross-lingual QA tracks in the Conference and Labs of the Evaluation Forum (CLEF[9]) and NII Test Collection for IR Systems (NTCIR[10]) workshops.

The main goal of the TREC QA track was to promote research on systems that can retrieve exact answers rather than documents corresponding to a question. Initially, the

[7] http://www-03.ibm.com/innovation/us/watson/

[8] http://trec.nist.gov/data/qamain.html

[9] http://www.clef-initiative.eu

[10] http://research.nii.ac.jp/ntcir/index-en.html

TREC-8 (1999) and TREC-9 (2000) QA tracks solely focused on simple factoid question answering (e.g. *Who is the author of the book, "The Iron Lady: A Biography of Margaret Thatcher"?*, *What was the name of the first Russian astronaut to do a spacewalk?*). Along with the main task (i.e., factoid question answering), the TREC-2001 QA track introduced an additional task called the *list* task. In the list task[11], the questions asked for a certain number of instances to extract and the answers to the questions included an unordered list of those items (e.g. *Name 20 countries that produce coffee.*). The TREC-2002 QA track continued the same tasks as TREC-2001 wheres the main task of the TREC-2003 QA track included factoid, definitional and list type questions with the introduction of a new task called the *passages* task. The TREC-2004 QA track provided a target text and a series of factoid, list and other question types were asked about the target. The TREC-2005 QA track consisted of three tasks: 1) main task (same as 2004), 2) document ranking task (to return a ranked list of at most 1000 documents in response to a selected set of questions from the main task), and 3) relationship task[12] (same as the 2004 AQUAINT Relationship QA Pilot[13] where TREC-like topic statements (e.g. *The analyst is looking for links between Colombian businessmen and paramilitary forces. Specifically, the analyst would like to know of evidence that business interests in Colombia are still funding the AUC paramilitary organization.*) were given to establish a context, and then, the task was to respond with a set of information nuggets as evidence for the answer). The main task of the TREC-2006 QA track was the same as 2005 with an additionally proposed task called the *complex interactive QA task*[14] to address more complex information needs in an interactive manner. Here, the term *complex* in the *complex interactive QA task* denoted an information need called "topic" that consisted of a template (to provide the question in a canonical form) and a narrative (to provide additional context). An example topic is shown

[11] http://trec.nist.gov/data/qa/2001_qadata/list_task.html

[12] http://trec.nist.gov/data/qa/2005_qadata/qa.05.guidelines.html

[13] http://trec.nist.gov/data/qa/add_QAresources/README.relationship.txt

[14] *http://www.umiacs.umd.edu/˜jimmylin/ciqa/*

below:

```
<topic num="26">
<template id="1">
What evidence is there for transport of [smuggled VCDs] from
[Hong Kong] to [China]? </template>
<narrative>
The analyst is particularly interested in knowing the volume of
smuggled VCDs and also the ruses used by smugglers to hide their
efforts. </narrative>
</topic>
```

Given the topic, the QA system was supposed to respond with a set of information nuggets that provides evidence for the answer (similar to the TREC-2005 relationship QA task). The term *interactive* provided an opportunity that for each topic in consideration, a human assessor could interact with the system for five minutes. The TREC-2007 QA track continued the same tasks as TREC-2006 with a difference that in the complex interactive QA (ciQA) task, questions were asked over both blog documents and newswire articles[15]. For a deeper survey of other related question answering systems, interested readers should consider the articles by Simmons (1965), Greenwood (2005), Wang (2006), and Hirschman and Gaizauskas (2001).

From the above discussion, we can see how the research in question answering moved beyond simple factoid question answering towards addressing more complex information needs. As noted before, complex type of questions (specially those that are addressed in this thesis) typically focus on an issue that often relates to multiple entities, events and their complex relations. In fact, a complex question might ask about events, biographies, definitions, descriptions, or reasons (Chali, Joty, and Hasan, 2009). Researchers have shown that

[15] http://trec.nist.gov/data/qa/2007_qadata/qa.07.guidelines.html

multi-document summarization techniques can be applied to treat these questions successfully (Chali, Hasan, and Joty, 2011; Chali, Joty, and Hasan, 2009; Harabagiu, Lacatusu, and Hickl, 2006), where the goal is to form a summary of the given document collection in response to the complex information request.

Multi-document summarization is a technique to describe the information of a document collection in a concise manner (Wan, Yang, and Xiao, 2007a). Some web-based systems are already utilizing the potential of this technology. For example, Google News[16] and the Newsblaster[17] system automatically collect, cluster, categorize, and summarize news from several sites on the web, and help users find news of their interest. In the last decade, complex questions have received much attention from both the Question Answering (QA) and Multi-document Summarization (MDS) communities (Carbonell et al., 2000) as there is a significant synergy between text summarization and question answering systems. Summarization is a process of condensing multiple source texts into one shorter version in response to complex questions, while Question Answering provides a means for focus in query-oriented summarization. As complex questions cannot be answered using the same techniques that have successfully been applied to the answering of "factoid" questions, multi-document summarization techniques are applied to accomplish this task. Thus we focus more on the summarization aspects. The Information Retrieval phase for Question Answering falls outside the scope of this work. Therefore, in order to provide a simplification of real tasks we assume the given set of documents as relevant for the given complex questions.

Complex questions are in reality broader information requests about a topic of user's interest. Such requests could be addressed by traditional, TREC-style information retrieval techniques (e.g. by trying to retrieve only the sentence or a block of the target document that seems important). Typically, complex QA evaluation systems including the TREC-

[16] http://news.google.com

[17] http://newsblaster.cs.columbia.edu/

2004 AQUAINT Relationship QA Pilot, the TREC-2005 Relationship QA Task, and the TREC definition return unstructured lists of candidate answers in response to a complex question. However, such an approach would not necessarily provide an exact answer to the complex question as the user would have desired. To address this challenge, the DUC (2005-2007) MDS evaluations task systems with returning paragraph-length answers to complex questions that are responsive, relevant, and coherent. This way of complex question answering in the form of a query-focused multi-document summarization task is useful in the domain of document management and search systems. For example, it can provide personalized news services for different users according to the users' unique information need (Wan and Xiao, 2009). Moreover, users can obtain the news about a single event from different sources in the form of a summary containing multiple perspectives at the same time.

Research in the domain of automatic summarization has been carried out over the years. According to Mani (2001), automatic text summarization takes a partially-structured source text from multiple texts written about the same topic, extracts information content from it, and presents the most important content to the user in a manner sensitive to the user's needs. Although search engines do a remarkable job in searching through a heap of information, they have certain limitations like the TREC-style document retrieval systems. For example, if we ask for the impact of the current global financial crisis in different parts of the world, we can expect to sift through thousands of results for the answer. The process of getting a desired answer to a complex question would speed up considerably when the summary of the given documents is also available. Hence, the technology of automatic summarization is critical in dealing with this kind of problems.

The automatic text summarization task can be categorized into three types: Generic vs. Query-Oriented, Abstractive vs. Extractive, and Single vs. Multi-Document. A generic summary must contain the core information present in the document. The most notable approaches to generic summarization have been introduced by Hovy and Lin (1998), Zha (2002), Gong and Liu (2001), and Lee et al. (2009). While a generic summary includes information which is central to the source documents, a query-oriented summary should formulate an answer to the query. In recent years, the attention of the researchers has shifted from generic summarization toward query-based summarization (Nastase, 2008; Daumé III and Marcu, 2006; Wan, Yang, and Xiao, 2007b; Wan and Xiao, 2009; Conroy, Schlesinger, and O'Leary, 2006; Chali, Joty, and Hasan, 2009; Chali, Hasan, and Imam, 2012a).

An extract summary consists of sentences extracted from the document while an abstract summary may employ words and phrases that do not appear in the original document (Mani and Maybury, 1999). Abstract summarization involves understanding an article and then selecting the key points. Existing research has tried to emulate the human approach to the task with little success as several complicated factors such as word sense and grammatical structure have to be taken into consideration. The abstract summary that has all the characteristics of a good summary is the ultimate goal of automatic text summarization (Genest and Lapalme, 2012). On the other hand, extract summarization involves assigning scores to the original source sentences using some method and then picking the top-ranked sentences for the summary. Although this kind of summary may not be necessarily smooth or fluent, extractive summarization is currently a general practice among the automatic text summarization researchers for its simplicity (Edmundson, 1969; Kupiec, Pedersen, and Chen, 1995; Carbonell and Goldstein, 1998; Lin, 2003; Martins and Smith, 2009; Berg-Kirkpatrick, Gillick, and Klein, 2011).

The process of summarizing one document is termed as single document summarization whereas in multi-document summarization, multiple documents related to one main topic are used as sources. Single document summarization is useful in many situations such as summarizing e-mails, news articles or creating abstract of scientific research papers (Das and Martins, 2007; Kupiec, Pedersen, and Chen, 1995; Lin and Hovy, 1997; Lin, 1999; Conroy and O'Leary, 2001; Osborne, 2002; Hirao et al., 2002a; Hirao et al., 2002c). Currently, multi-document summarization is of greater interest since the amount of information present in the web is becoming huge (Hirao et al., 2003; Wan and Xiao, 2009; Chali, Joty, and Hasan, 2009; Chali, Hasan, and Imam, 2012a).

The automatic text summarization area has gone through several changes with the development of techniques and requirements since the year 1950. Typically, summarization approaches can be divided into three categories: Knowledge-based Methods, Classical Methods, and Modern Methods.

It is always desirable to emulate the process of summarization as humans do it. To accomplish the summarization task automatically the machine needs to understand the source texts, pick out the important points and generate sentences from these points. The whole approach relies on both natural language understanding and generation. These methods are termed as knowledge-based methods (Ferrier, 2001). McKeown and Radev (1995) have proposed such a system for summarizing multiple news articles on the same event. Their system works on sets of templates, which is built on the inherent important facts of the input text. The system is comprised of two major units: a content planner and a linguistic component. The content planner's function is to generate a conceptual representation of the input whereas the linguistic component organizes a selection of words to refer to the concepts using the lexical knowledge of English. This type of summarization is pretty hard for the machine to perform because they have to characterize a source text as a whole, and capture its important content. Hence, other methods have also been investigated in order to produce automatic summaries.

The methods that started the automatic text summarization research can be termed as the classical methods. Being motivated by the need to deal with the information overload problem, one of the first to perform such research was Luhn (1958). He used simple statistical techniques to determine the most significant sentences of a document. These sentences were then extracted from the text and printed out together to form the summary. Methods to find features of the input text have been developed since Luhn's work. Edmundson (1969) weighted sentences based on four different methods: cue phrase, keyword (i.e., term frequency based), location, and title. These early ground-breaking systems acted as the pioneers to the modern summarization systems.

The field of automatic text summarization has received immense attention from the researchers in the recent years. The vast increase of information in the web has acted as a fuel for this. The most notable modern methods of automatic summarization are as follows.

The graph-based methods, such as LexRank (Erkan and Radev, 2004) and TextRank (Mihalcea and Tarau, 2004), are applied successfully to generic, multi-document summarization. Erkan and Radev (2004) used the concept of graph-based centrality to rank a set of sentences for producing generic multi-document summaries. A similarity graph is produced for the sentences in the document collection. In the graph each node represents a sentence. An edge between two nodes measures the cosine similarity between the respective pair of sentences. The degree of a given node is an indication of how important the sentence is. A topic-sensitive LexRank is proposed in Otterbacher, Erkan, and Radev (2005). In this method, a sentence is mapped to a vector in which each element represents the occurrence frequency (TF*IDF) of a word. However, the major limitation of the TF*IDF approach is that it only retains the frequency of the words and does not take into account the sequence, syntactic and semantic structure, thus cannot distinguish between "The hero killed the villain" and "The villain killed the hero".

Latent Semantic Analysis (LSA) (Deerwester et al., 1990) based summarization methods were introduced in the 90s. LSA is a fully automatic statistical technique to extract and infer relations of expected contextual usage of words in passages of discourse. The first step towards the application of LSA is to represent a document as a document-term matrix

A, such that each row of matrix stands for a unique word present in the document and each column stands for a sentence. Each entry A_{ij} represents the frequency of term i in document j. Gong and Liu (2001) have proposed a scheme for automatic text summarization using LSA. Their approach classifies the document into different topics and picks the dominant sentence from each dominant topic sequentially until the summary length is reached.

Summarization models based on lexical chains were proposed in the late 90s. A lexical chain is a sequence of related words in the text, spanning short (adjacent words or sentences) or long distances (entire text). A chain is independent of the grammatical structure of the text and in effect it is a list of words that captures a portion of the cohesive structure of the text. Computing the lexical chains in a document is one technique that can be used to identify the central theme of a document. This in turn leads to the identification of the key section(s) of the document which can then be used for summarization purposes. The summarization systems based on lexical chains first extract the nouns, compound nouns and named entities as candidate words (Barzilay and Elhadad, 1997; Kolla, 2004; Li et al., 2007). The systems rank sentences using a formula that involves a) the lexical chains, b) keywords from query and c) named entities. For example, Li et al. (2007) used the following formula:

$$Score = \alpha P(chain) + \beta P(query) + \gamma P(namedEntity)$$

where $P(chain)$ is the sum of the scores of the chains whose words come from the candidate sentence, $P(query)$ is the sum of the co-occurrences of key words in a topic and the sentence, and $P(namedEntity)$ is the number of named entities existing in both the topic and the sentence. The three coefficients α, β and γ are set empirically. Then the top ranked sentences are selected to form the summary.

QA based summarization received huge attention from the researchers in the last decade. Typically, in a summarization system that is based on a question answering system (Molla and Wan, 2006), the topic sentences are converted to a sequence of questions as the underlined QA system is designed to answer only simple (i.e. factoid, list) questions. The QA system normalizes and classifies the questions, and finds the candidate answers along with the sentences in which the answers appeared. Instead of extracting the exact terms as answers, the systems extract the sentences for each of the questions in the topic to form the summary.

The idea of discourse-based summarization was first presented by Mann and Thompson (1988). They have argued that discourse structures of texts can be used effectively in summarization. Marcu (1998b) has proved this hypothesis by showing the role of text structures in selecting the most important sentences for the summary. He has proposed a discourse-based summarization approach that uses rhetorical parsing to derive the inherent text structure. Based on this structure, an importance score is assigned to each sentence in the text and the top *p%* sentences are selected to form a summary of the given text.

Machine Learning (ML) can be applied successfully for the task of summarization. The main goal of ML research is to design and develop algorithms and techniques such that computers can learn from data automatically. For the task of summarization, it is necessary to learn which part of the given source documents are the most relevant to be considered as the summary. In the 1990s, researchers have focused on employing statistical techniques for extractive single document summarization (Das and Martins, 2007). Most of these systems assumed feature independence while relying on the naive-Bayes methods (Kupiec, Pedersen, and Chen, 1995). Some other approaches were based on the choice of appropriate features while learning algorithms that make no independence assumptions (Lin and Hovy, 1997; Lin, 1999). Other significant approaches involved hidden Markov models and log-linear models to improve extractive summarization (Conroy and O'Leary, 2001; Osborne, 2002). Single document summarization systems using Support Vector Machines (SVMs) demonstrated good performance for both Japanese (Hirao et al., 2002a) and English documents (Hirao et al., 2002c). Hirao et al. (2003) showed the effectiveness of their multiple document summarization system employing SVMs for sentence extraction. Conroy and O'Leary (2001) used two kinds of states, where one kind corresponds to the summary states and the other corresponds to non-summary states. The motivation of applying CRF in text summarization came from observations on how humans summarize a document by posing the problem as a sequence labeling problem (Shen et al., 2007). The statistical technique such as Maximum Entropy (MaxEnt) works in a way that assumes nothing about the information of which it has no prior knowledge (Ferrier, 2001). Joty (2008) experimented with both empirical and unsupervised machine learning approaches (K-means and Expectation Maximization (EM) algorithms) to summarize texts. Hasan (2010) focused on query-oriented, extractive, multi-document summarization in order to combat the complex question answering problem by applying supervised machine learning techniques: Support Vector Machines (SVM), Hidden Markov Models (HMM), Conditional Random Fields (CRF), and Maximum Entropy (MaxEnt). As supervised systems rely on learning from a vast amount of labeled data, five automatic annotation techniques have been proposed using different textual similarity measurement techniques: ROUGE similarity measure (Lin, 2004), Basic Element (BE) overlap (Hovy et al., 2006), syntactic similarity measure (Moschitti and Basili, 2006), semantic similarity measure (Moschitti et al., 2007), and Extended String Subsequence Kernel (ESSK) (Hirao et al., 2003).

The other notable summarization systems are based on optimization techniques that perform integer linear programming formulations to the summarization task (McDonald, 2007; Gillick and Favre, 2009; Martins and Smith, 2009; Galanis, Lampouras, and Androutsopoulos, 2012). There are also some approaches that use the potential of sentence compression technology for the task of summarization (Madnani et al., 2007; Martins and Smith, 2009; Berg-Kirkpatrick, Gillick, and Klein, 2011).

All these researches described above have motivated us to focus on the task of answering complex questions with the intention to minimize several existing gaps in the literature.

1.3 Contribution

This thesis contributes to the domain of complex question answering in the following ways:

Reinforcement Learning Formulation: As the major limitation of the available search systems is that they lack a way of measuring whether a user is satisfied with the information provided, this has motivated us to propose a reinforcement learning formulation to the complex question answering problem. Given a set of complex questions, a list of relevant documents per question, and the corresponding human generated summaries (i.e. answers to the questions) as training data, our reinforcement learning module iteratively learns a number of feature weights in order to facilitate the automatic generation of summaries i.e. answers to previously unseen complex questions. A reward function is used to measure the similarities between the candidate (machine generated) summary sentences and the abstract summaries. In the training stage, the learner iteratively selects the important document sentences to be included in the candidate summary, analyzes the reward function and updates the related feature weights accordingly. The final weights are used to generate summaries as answers to unseen complex questions in the testing stage (Chali, Hasan, and Imam, 2011a). We have also effectively incorporated user interaction into the reinforcement learner to guide the candidate summary sentence selection process (Chali, Hasan, and Imam, 2012a).

Integer Linear Programming Formulation: The answer to the complex question must satisfy the requesting user the most. There might be several answers to the same question. It is desirable to identify an effective way in which one can tell which combination of the document sentences can fully satisfy the quest of the user. The use of optimization techniques can essentially serve this purpose. Therefore, we have formulated the complex question answering task using Integer Linear Programming (ILP) in a query-focused multi-document summarization setting. Sentence compression is a useful technique to discard redundant information in a sentence while keeping the most important information. Hence, we have applied sentence compression models for the task of query-focused multi-document summarization in order to investigate if sentence compression improves the overall summarization performance. Both compression and summarization are considered as global optimization problems and solved using ILP. Sentence compression models include lexical, syntactic and semantic constraints while summarization models include relevance, redundancy and length constraints. A comprehensive set of query-related and importance-oriented measures are used to define the relevance constraint whereas four alternative redundancy constraints are employed based on different sentence similarity measures using a) cosine similarity, b) syntactic similarity, c) semantic similarity, and d) extended string subsequence kernel (ESSK) (Chali and Hasan, 2012a).

Graph-based Random Walk Model with Syntactic and Semantic Information: In our previous two contributions, we have considered a document sentence as important by computing a measure of relevance with respect to the complex question and in terms of its characteristics in the document where it is present. However, there might be a case where a document sentence is not related to the question rather it has a strong inherent similarity with other important sentences in the document. This assumption can be effectively incorporated into a graph-based random walk model for summarization (Erkan, 2007). The traditional graph-based random walk models use basic similarity functions such as cosine measure in order to assess similarity between the sentences, which cannot capture the syntactic and semantic aspects of the sentence pair in consideration. This has motivated us to analyze the impact of syntactic and semantic information in a graph-based random walk method for answering complex questions. Initially, we apply tree kernel functions to perform the similarity measures between sentences in the random walk framework. Then, we extend our work further to incorporate the Extended String Subsequence Kernel (ESSK) to perform the task in a similar manner. Experimental results show the effectiveness of the use of kernels to include the syntactic and semantic information for this task (Chali, Hasan, and Joty, 2011; Chali, Hasan, and Imam, 2011c). In another experiment, we have successfully exploited a deeper semantic analysis of the source documents to select important concepts by using a predefined list of important aspects that act as a guide for selecting the most relevant sentences into the summaries (Chali, Hasan, and Imam, 2011b).

Learning Good Decompositions of Complex Questions: Previous works have demonstrated that decomposing a complex question into a series of simple questions and then reusing the techniques developed for answering simple questions is an effective means of answering complex questions. However, no study has developed any method to judge the significance of the decomposed questions by disregarding the fact that good decompositions are the prerequisites for more accurate answers. It is also not feasible to generate a list of good decompositions manually for all possible complex questions in order to guide the sentence selection step. In this thesis, we have addressed this challenge and proposed a supervised approach for automatically learning good decompositions of complex questions (Chali, Hasan, and Imam, 2012b). The training data generation phase mainly builds on three steps to produce a list of simple questions corresponding to a complex question: i) the extraction of the most important sentences from a given set of relevant documents (which contains the answer to a complex question), ii) the simplification of the extracted sentences, and iii) their transformation into questions containing candidate answer terms. Such questions, considered as candidate decompositions, are manually annotated (as good or bad candidates) and used to train a Support Vector Machine (SVM) classifier.

Topic to Question Generation: A complex question often asks about a topic of user's interest. Therefore, the problem of complex question decomposition closely relates to the problem of topic to question generation. We have addressed the challenge of automatically generating questions from topics (Chali and Hasan, 2012c) in order to enhance the scope of our problem domain. For example, given the topic *"Apple Inc. Logos"*, we generated questions such as *"What is Apple Inc.?"*, *"Where is Apple Inc. located?"*, *"Who designed Apple's Logo?"* etc. To simplify our task, we consider that each topic is associated with a body of texts containing useful information about the topic. Questions are generated by exploiting the named entity information and the predicate argument structures of the sentences present in the body of texts. To measure the importance of the generated questions, we use Latent Dirichlet Allocation (LDA) to identify the sub-topics (which are closely related to the original topic) in the given body of texts and apply the Extended String Subsequence Kernel (ESSK) to calculate their similarity with the questions. We use syntactic tree kernels for computing the syntactic correctness of the questions. The questions are ranked by considering their importance (in the context of the given body of texts) and syntactic correctness.

1.4 Thesis Outline

We give a chapter-by-chapter outline of the remainder of this thesis in this section.

Chapter 2: We give a detailed description of our reinforcement learning formulation for the complex question answering task.

Chapter 3: We provide a general review of the works performed previously with sentence compression for summarization. Then we present our ILP-based sentence compression models for the query-focused multi-document summarization task, and discuss our experiments and results.

Chapter 4: We take a closer look at the graph-based random walk model that was successful for the complex question answering task and then describe our improvements to the model by incorporating syntactic and semantic information.

Chapter 5: We discuss our approach for learning good decompositions of complex questions and present the evaluation results.

Chapter 6: We present our topic to question generation approach, and show the experimental results.

Chapter 7: We conclude the thesis by identifying some future directions of our research.

1.5 Published Work

Most of the materials presented in this thesis has been previously published. Chapter 2 to Chapter 6 expands on the materials published in Chali and Hasan (2012a, 2012b, 2012c), Chali, Hasan, and Imam (2012a, 2012b, 2011a, 2011b, 2011c), Chali, Hasan, and Joty (2011), and Ali, Chali and Hasan (2010) .

Chapter 2

Complex Question Answering using Reinforcement Learning

2.1 Introduction

The major limitation of the available search systems is that they lack a way of measuring whether a user is satisfied with the information provided. Measurement of user satisfaction in real time is a crucial component of the search systems since it can provide a direction towards improvement of the search policy dynamically. User satisfaction can be observed by monitoring user actions (e.g., clicking, copy-pasting, printing, saving, emailing) after the search results are presented. A user study can reveal the relationship between user satisfaction and retrieval effectiveness (Al-Maskari, Sanderson, and Clough, 2007). For example, Zaragoza, Cambazoglu, and Baeza-Yates (2010) performed a quantitative analysis about what fraction of the web search queries (posed to the current search engines) can lead to satisfactory results.

Computation of user satisfaction, as well as improvement to the search policy, is a difficult task to perform in real time. This motivates us to propose a reinforcement learning formulation to the complex question answering task so that the system can learn from user interaction to improve its accuracy according to user's information need. Formally, the complex question answering problem can be mapped to a reinforcement learning framework as follows: given a set of complex questions, a collection of relevant documents per question, and the corresponding human-generated summaries (i.e. answers to the questions), a reinforcement learning model can be trained to extract the most important sentences to form summaries (Chali, Hasan, and Imam, 2011a). During the learning stage, for simplicity, we assume that initially there is no actual user interaction provided to the system rather the importance of a candidate document sentence can be verified by measuring its similarity with the given human-made abstract summary sentences using a reward function. This assumption relies on the intuition that the users are fully satisfied with the abstract summaries. The original document sentences that are mostly similar to the abstract summary sentences are assigned good reward values. Using an appropriate reward function is essential in reinforcement learning since the learner is not actually aware of which actions (*sentence selection* in our case) to take, rather it must discover which actions deliver the most reward value by trying them (Sutton and Barto, 1998).

As noted before, real-time user interaction can help QA systems evolve by improving their policy automatically as time passes. Toward this end, we treat the complex question answering task as an interactive problem. Supervised learning techniques are alone not adequate for learning from interaction (Sutton and Barto, 1998). These techniques require a huge amount of human-annotated training data and it is often impossible to collect training examples of all desired kinds in which the agent has to act. Instead, a reinforcement learning approach can be used to sense the state of the environment and take suitable actions that affect the state. This is why, we believe that a reinforcement learning methodology is appropriate for the complex question answering task. We assume that a small amount of supervision is provided in the form of a reward function that defines the quality of the executed actions. In the training stage, the reinforcement learner repeatedly defines action sequences, performs the actions, and observes the resulting reward. In this phase, the learner's main goal is to estimate a policy that maximizes the expected future reward (Branavan et al., 2009).

In this thesis, we present a reinforcement learning framework for answering complex questions. In our initial experiments, we simplify the formulation by considering no real time user interaction as we assume that the human-generated abstract summaries are the gold-standard and the users (if they were involved) are satisfied with them. The proposed system tries to produce automatic summaries that are close to the abstract summaries. The relationship between these two types of summaries is learned and the final weights are used to output the machine generated summaries for the unseen data. We employ a modified linear, gradient-descent version of Watkins's $Q(\lambda)$ algorithm (Sutton and Barto, 1998) to estimate the parameters of our model. Experiments on the DUC benchmark datasets demonstrate the appropriateness and the effectiveness of the reinforcement learning approach. We also extend this work by proposing a model that incorporates user interaction into the reinforcement learner to guide the candidate summary sentence selection process. Evaluation results indicate that the user interaction component further improves the performance of the reinforcement learning framework. In the following sections, we discuss related work, our reinforcement learning formulation, feature space, user interaction modeling, and the experiments with results.

2.2 Related Work

We perform the complex question answering task using an extractive multi-document summarization approach within a reinforcement learning setting. Extractive summarization is simpler than abstract summarization as the process involves assigning scores to the given document sentences using some method and then picking the top-ranked sentences for the summary. Although this kind of summary may not be necessarily smooth or fluent, extractive summarization is currently a general practice due to its simplicity (Jezek and Steinberger, 2008). Over the years, various extraction-based techniques have been proposed for generic multi-document summarization. In recent years, researchers have become more interested in query-focused (i.e. topic-biased) summarization. The leading systems in the DUC[1] and TAC[2] tracks focus on the complex question answering task through

[1] http://duc.nist.gov/pubs.html

[2] http://www.nist.gov/tac/publications/index.html

multi-document summarization.

Other notable extraction-based summarization systems are as follows. Nastase (2008) expands a query by using the encyclopedic knowledge in Wikipedia and introduce a graph to generate the summary. Daumé III and Marcu (2006) present BAYESUM ("Bayesian summarization"), a sentence extraction model for query-focused summarization. On the other hand, Wan, Yang, and Xiao (2007b) propose a manifold-ranking method to make uniform use of sentence-to-sentence and sentence-to-topic relationships whereas the use of multi-modality manifold-ranking algorithm is shown in Wan and Xiao (2009). Other than these, topic-focused multi-document summarization using an approximate oracle score has been proposed in Conroy, Schlesinger, and O'Leary (2006) based on the probability distribution of unigrams in human summaries. In our proposed approach, we represent each sentence of a document as a vector of feature-values. We incorporate query-related information into our model by measuring the similarity between each sentence and the user query (i.e. the given complex question). We exploit some features such as title match, length, cue word match, named entity match, and sentence position to measure the importance of a sentence. We use a number of features to measure the query-relatedness of a sentence considering n–gram overlap, LCS, WLCS, skip-bigram, exact-word, synonym, hypernym/hyponym, gloss and Basic Element (BE) overlap, and syntactic information (See details in Section 2.6). These features have been adopted from several related works in the problem domain (Chali, Joty, and Hasan, 2009; Edmundson, 1969; Sekine and Nobata, 2001; Litvak, Last, and Friedman, 2010; Schilder and Kondadadi, 2008). We also considered a dynamic feature to lower the redundancy in the extract summary using the Maximal Marginal Relevance (MMR) model (Carbonell and Goldstein, 1998). This feature helps the learner to understand which document sentence is less similar to the sentences that are already present in the candidate answer space (i.e. the current summary pool). We use the relevant novelty metric that was previously shown in Goldstein et al. (2000). This metric measures relevance and novelty independently and provides a linear combination of them. A document sentence has a higher chance to be selected if it is both relevant to the given query and useful for a summary, while having minimal similarity to the previously selected sentences.

In the field of natural language processing, reinforcement learning has been extensively applied to the problem of dialogue management where the systems converse with a human user by taking actions that emit natural language utterances (Scheffler and Young, 2002; Roy, Pineau, and Thrun, 2000; Litman et al., 2000; Singh et al., 1999; Dethlefs et al., 2012). The state space defined in these systems encodes information about the goals of the user and what they say at each time step. The learning problem is to find an optimal policy that maps states to actions through a trial-and-error process of repeated interaction with the user. Branavan et al. (2009) presented a reinforcement learning approach for mapping natural language instructions to sequences of executable actions.

Our work was the first to propose a reinforcement learning formulation to the problem of complex question answering (Chali, Hasan, and Imam, 2011a). Recently, a related problem of automatic text summarization has been modeled using a reinforcement learning framework by Ryang and Abekawa (2012). Our approach is significantly different from their approach in a number of ways. Their formulation is applicable to generic summarization while our system considers a problem that is a kind of query-focused multi-document summarization. Moreover, the reward function of their model is not designed to take user feedback into account whereas the reward function of our reinforcement learning framework is specially designed for considering user feedback as the principle way of improving the search policy in real time.

As noted before, in our first experiment, we do not directly interact with a user making the cost of interaction lower. However, experiments in the complex interactive Question

Answering (ciQA) task[3] at TREC-2007 demonstrate the significance of user interaction in this domain. The technique of user modeling in an interactive QA system is not new (Hickl and Harabagiu, 2006; Webb and Strzalkowski, 2006; Dethlefs et al., 2012). An adaptive, open-domain, personalized, interactive QA system called YourQA[4] is an example of a deployed system where a QA module interacts with a user model and a dialogue interface (Quarteroni and Manandhar, 2009). Motivated by the effect of user interaction shown in the previous studies (Wang et al., 2003; Lin, Madnani, and Dorr, 2010; Sakai and Masuyama, 2004; Yan, Nie, and Li, 2011; Wu, Scholer, and Turpin, 2008; Harabagiu et al., 2005), we propose an extension to our reinforcement learning model by incorporating user interaction into the learner and argue that the user interaction component can provide a positive impact in the candidate summary sentence selection process (Chali, Hasan, and Imam, 2012a).

We compare our system with a SVM-based model. In the field of natural language processing, SVMs are applied to text categorization and syntactic dependency structure analysis. These approaches are reported to have achieved higher accuracy than previous approaches (Joachims, 1998). SVMs were also successfully applied to part–of–speech tagging (Giménez and Màrquez, 2003). Single document summarization systems using SVMs demonstrated good performance for both Japanese (Hirao et al., 2002b) and English documents (Hirao et al., 2002d). In (Hirao et al., 2003), they showed effectiveness of their multiple document summarization system employing SVMs for sentence extraction. A fast query-based multi-document summarizer called *FastSum* used SVM regression[5] to rank the summary sentences where the goal was to estimate the score of a sentence based on a given feature set (Schilder and Kondadadi, 2008). However, our SVM model treats the task as a classification problem where the classifier is trained on data pairs, defined by feature vectors and corresponding class labels. The need for a large amount of data to

[3] http://www.umiacs.umd.edu/~jimmylin/ciqa/

[4] http://www.cs.york.ac.uk/aig/projects/yourqa/

[5] Regression tasks tend to estimate the functional dependence of a dependent variable on a set of independent variables.

train a SVM-based system often makes it harder to use in practice. For this reason, we use an unsupervised summarization model to evaluate our proposed reinforcement system. A k-means clustering algorithm is used to build the unsupervised system (Chali, Joty, and Hasan, 2009).

2.3 Problem Formulation

We formulate the complex question answering problem by estimating an action-value function (Sutton and Barto, 1998). We define the value of taking action a in state s under a policy π (denoted $Q^{\pi}(s,a)$) as the expected return starting from s, taking the action a, and thereafter following policy π:

$$\begin{aligned} Q^{\pi}(s,a) &= E_{\pi}\{R_t|s_t = s, a_t = a\} \\ &= E_{\pi}\left\{\sum_{k=0}^{\infty} \gamma_k r_{t+k+1} | s_t = s, a_t = a\right\} \end{aligned} \qquad (2.1)$$

Here, E_{π} denotes the expected value given that the agent follows policy π, R_t is the *expected return* that is defined as a function of the reward sequence, $r_{t+1}, r_{t+2}, \cdots$, where r_t is the numerical reward that the agent receives at time step, t. We call Q^{π} the action-value function for policy π. γ stands for the discount factor that determines the importance of future rewards. We try to find out the optimal policy through policy iteration. Once we get the optimal policy (π^*) the agent chooses the actions using the Maximum Expected Utility Principle (Russell and Norvig, 2003). We show our reinforcement learning framework in Figure 2.1.

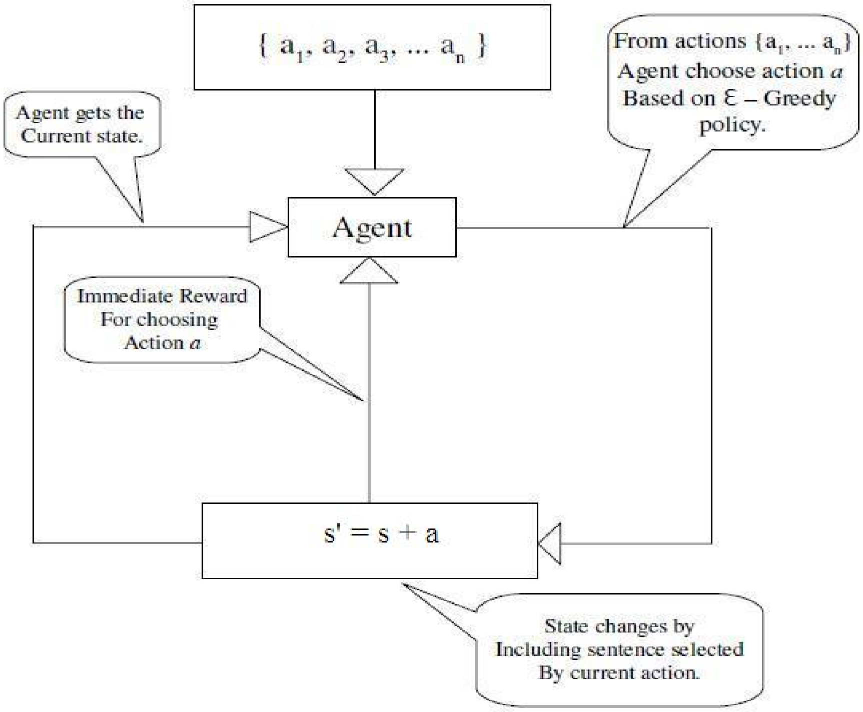

{ a_1, a_2, a_3, ... a_n }
From actions {a_1, ... a_n} Agent choose action a Based on ε – Greedy policy.
Agent gets the Current state.
Agent
Immediate Reward For choosing Action a
s' = s + a
State changes by Including sentence selected By current action.

Figure 2.1: Reinforcement learning framework

2.3.1 Environment, State & Actions

Given a complex question q and a collection of relevant documents $D = \{d_1, d_2, d_3, \ldots, d_n\}$, the task is to find an answer (extract summary). The *state* is defined by the current status of the answer pool. Initially, there is no sentence in the answer pool, i.e., the initial state s_0 is empty. In each iteration, a sentence from the given document collection is selected and added to the answer pool that in turn changes the *current state*. The environment is described by the state space. In each state, there is a possible set of actions that could be operated on the environment where a certain *action* denotes *selecting a particular sentence* (using the policy function of equation 2.1) from the remaining document sentences that are not yet included in the answer pool (i.e. candidate extract summary).

2.3.2 Reward Function

In the training stage of the reinforcement learning framework, for each complex question we are given a relevant document collection along with a set of human generated abstract summaries (see details in Section 2.7.2) as answers to the question. We consider these summaries (i.e. answers) as the gold-standard and assume that the users are satisfied with them. We utilize these summaries to calculate the immediate rewards. In a certain *state*, after taking an action *a* (i.e. selecting a sentence), we compute the immediate reward, *r* using the following formula:

$$r = w \times relevance(a) - (1 - w) \times redundancy(a) \tag{2.2}$$

where $relevance(a)$ is the textual similarity measure between the selected sentence and the abstract summaries, *redundancy*(a) is the similarity measure between the selected sentence and the *current state* (that includes the already chosen set of sentences) of the answer pool, and *w* is the weight parameter that denotes the importance of relevance and redundancy. By including redundancy in the immediate reward calculation we discourage redundancy in the final extract summary. In our experiments, the value of *w* is kept to 0.5 to provide *equal importance* to both relevance and redundancy. We measure the textual similarity using ROUGE (Recall-Oriented Understudy for Gisting Evaluation) (Lin, 2004).

2.3.3 Function Approximation

In many tasks such as the one to which we apply reinforcement learning, most of the states encountered will never have been experienced before. This occurs when the state or action spaces include continuous variables. As in our case the number of states and actions are infinite, the approximate action-value function is represented as a parameterized functional form with parameter vector, $\vec{\theta}_t$. Our approximate action-value function is a linear function of the parameter vector, $\vec{\theta}_t$. Corresponding to every state-action pair (s,a) , there is a column vector of features, $\vec{\varphi}_s = (\varphi_s(1), \varphi_s(2), \ldots, \varphi_s(n))^T$ with the same number of components as $\vec{\theta}_t$. The approximate action-value function is given by:

$$Q_t(s,a) = \vec{\theta}_t^T \vec{\varphi}_s = \sum_{i=1}^{n} \theta_t(i) \varphi_s(i) \tag{2.3}$$

2.3.4 Markov Decision Process (MDP)

Our environment has the Markov property, that is, given the current state and action we can predict the next state and expected reward. For our problem formulation, given the current state s if we take an action a, the next state will be $s^{'} = s + a$, since our action is to choose a sentence from the given document collection and adding it into the extract summary pool. Given any state and action, s and a, the transition model is defined by:

$$\rho^a_{ss'} = P_r \left\{ s_{t+1} = s^{'} | s_t = s, a_t = a \right\} \tag{2.4}$$

$\rho^a_{ss'}$ will be 1 when $s^{'} = s + a$. For all other states, the transition probability will be 0. Similarly, given any current state and action, s and a, together with any next state, $s^{'}$, the expected value of the next reward is:

$$R^a_{ss'} = E \left\{ r_{t+1} | s_t = s, a_t = a, s_{t+1} = s^{'} \right\} \tag{2.5}$$

2.4 Reinforcement Learning

We consider our task as an infinite horizon sequential decision making problem that finds a parameter vector $\vec{\theta}$ to maximize $Q(s,a)$ from equation 2.3. Policy gradient algorithms tend to estimate the parameters θ by performing a stochastic gradient ascent. The gradient is approximated by interacting with the environment, and the resulting reward is used to update the estimate of θ. Policy gradient algorithms optimize a non-convex objective and are only guaranteed to find a local optimum (Branavan et al., 2009). We use a modified linear, gradient-descent version of Watkins's $Q(\lambda)$ algorithm with ε-greedy policy to determine the best possible action i.e. to select the most important sentences. We use the ε-greedy policy (meaning that most of the time this policy chooses an action that has maximal estimated action value, but with probability ε an action is selected at random) to balance between exploration and exploitation during the training phase. We empirically set the value of $\varepsilon = 0.1$ during our experiments. We note that, 90% of the time our algorithm chooses an action with the best action-value and 10% of the time it chooses an action randomly.

The steps of our reinforcement learning algorithm are shown in Algorithm 1. Here, φ is a vector of feature-values (See details in Section 2.6) that is used to represent each document sentence and $\vec{\theta}$ is the vector of weights for the feature vector that the system will learn. γ is the discount factor that is used to calculate the reward of a state-action pair. The discount factor determines the importance of future rewards. We kept the initial value of γ as 0.1. The value of γ decreases by a factor of iteration counts. As long as the initial policy selects greedy actions, the algorithm keeps learning the action-value function for the greedy policy. However, when an exploratory action is selected by the behaviour policy, the eligibility traces[6] for all state-action pairs are set to zero. The eligibility traces are updated in two steps. In the first step, if an exploratory action is taken, they are set to 0 for all state-

[6] An eligibility trace is a temporary record of the occurrence of an event, such as the visiting of a state or the taking of an action (Sutton and Barto, 1998).

action pairs. Otherwise, the eligibility traces for all state-action pairs are decayed by γ^{λ}. In the second step, the eligibility trace value for the current state-action pair is incremented by 1 while accumulating traces. The original version of the Watkins's $Q(\lambda)$ algorithm uses a linear, gradient-descent function approximation with binary features. However, since we deal with a mixture of real-valued and boolean features (See Section 2.6), we modified the algorithm to induce a different update for the eligibility traces. In the second step of eligibility trace update, we increment the value by the corresponding feature score. The addition of a random jump step avoids the local maximums in our algorithm. The parameter λ defines how much credit we give to the earlier states. α is the step size parameter for the gradient descent method that is reduced by a factor of 0.99 as learning converges towards the goal.

To the best of our knowledge, the proposed formulation with the modified version of the Watkins's $Q(\lambda)$ algorithm is unique in how it represents the complex question answering task in the reinforcement learning framework.

2.5 Modeling User Interaction

In the basic reinforcement learning model for answering complex questions (discussed in the previous section), we have a set of possible actions in each state. Note that *state* refers to the current status (content) of the answer space while *action* refers to *choosing a candidate sentence*. Initially, there is no sentence in the answer pool. So, the initial state is empty. In each iteration, a new sentence is selected[7] from the document and added to the answer pool that in turn changes the state. We propose an extension to this model and add user interaction in the reinforcement learning loop in order to facilitate the candidate selection process. For a certain number of iterations during the training stage, the user is presented

[7] The sentence having the highest feature value is selected as the potential candidate.

ALGORITHM 1: Modified Watkins's $Q(\lambda)$ algorithm

```
**Input**: $\alpha$, $\vec{\theta}$, $\vec{e}$, $\lambda$, $\gamma$, $\delta$, $\varphi$, $\varepsilon$, num of iteration $T$
**Output**: A vector $\vec{\theta}$ of learned weights
Initialize: $\vec{\theta}$ to $\vec{0}$, $\alpha$ to 0.01, $\gamma$ to 0.1, $\lambda$ to 0.9
$\vec{e} = \vec{0}$
$s, a \leftarrow$ initial state and action of episode
$\varphi \leftarrow$ set of features present in $s, a$
**for** *each* $i = 1 \ldots T$ **do**
    **if** *s is not terminal* **then**
        **for** $i \in \varphi$ **do**
            $e(i) \leftarrow e(i) + \varphi(i)$
        **end**
        Take action $a$, observe reward $r$, and next state, $s$
        $\delta \leftarrow r - \sum_i \varphi(i)\theta(i)$
        **for** $a \in A(s)$ **do**
            $\varphi \leftarrow$ set of features present in $s, a$
            $Q_a \leftarrow \sum_i \varphi(i)\theta(i)$
        **end**
        $\delta \leftarrow \delta + \gamma max_a Q_a$
        $\theta \leftarrow \theta + \alpha\delta\vec{e}$
        $\alpha \leftarrow 0.99 * \alpha$
        **if** $probability \leq 1 - \varepsilon$ **then**
            **for** $a \in A(s)$ **do**
                $Q_a \leftarrow \sum_i \varphi(i)\theta(i)$
            **end**
            $a \leftarrow argmax_a Q_a$
            $\vec{e} \leftarrow \gamma\lambda\vec{e}$
        **else**
            $a \leftarrow$ a random action $\in A(s)$
            $\vec{e} \leftarrow 0$
        **end**
    **end**
**end**
return $\vec{\theta}$
```

with the top five candidate sentences (based on the learned Q function). The user can also see the complex question being considered and the current status (content) of the answer space (i.e. state). The task of the user at this point is to select the best candidate among the five to be added to the answer space. In the basic reinforcement learning model, the first candidate was selected to be added automatically as it was having the highest similarity score. In this way, there was a chance that a potentially unimportant sentence could be chosen that is not of user's interest. However, in the extended reinforcement learning model, the user interaction component enables us to incorporate the human viewpoint and thus, the judgment for the best candidate sentence is supposed to be perfect. Extensive experiments on DUC-2006 benchmark datasets support this claim. In Appendix B, we show an example of how exactly the user interaction component works. The outcome of the reinforcement learner is a set of weights that are updated through several iterations until the algorithm converges. During the user interaction experiment, the currently considered topic is shown to the user, followed by the complex question, current summary (i.e. answer) and the top five candidate sentences. At this point, the user selects a sentence to add to the answer space and the feature weights are updated based on this response. This process runs for three iterations for each topic during training. In the remaining iterations, the algorithm selects the sentences automatically and continues updating the weights accordingly.

2.6 Feature Space

We represent each sentence of a document as a vector of feature-values. We divide the features into two major categories: static and dynamic. Static features include two types of features, where one declares the importance of a sentence in a document and the other measures the similarity between each sentence and the user query. The usefulness of these features have been analyzed through a series of experiments in Chali, Joty, and Hasan

(2009). We use one dynamic feature that measures the similarity of already selected candidate with each remaining sentences. The dynamic feature is used to ensure that there is no redundant information present in the final summary.

2.6.1 Static Features: Importance

Position of Sentences: Sentences that reside at the start and at the end of a document often tend to include the most valuable information. We manually inspected the given document collection and found that the first and the last 3 sentences of a document often qualify to be considered for this feature. We assign the score 1 to them and 0 to the rest.

Length of Sentences: Longer sentences contain more words and have a greater probability of containing valuable information. Therefore, a longer sentence has a better chance of inclusion in a summary. We give the score 1 to a longer sentence and assign the score 0 otherwise. We manually investigated the document collection and set a threshold that a longer sentence should contain at least 11 words. The empirical evidence to support the choice of this threshold is based on the direct observation of our datasets[8].

Title Match: If we find a match such as exact word overlap, synonym overlap or hyponym overlap between the title and a sentence, we give it the score 1, otherwise 0. The overlaps are measured by following the same procedure as described in Section 2.6.2. We use the WordNet[9] (Fellbaum, 1998) database for the purpose of accessing synonyms and

[8] We understand that the cutoff of 11 words, and the choice of the first and last three sentences of a document are very specific for the newswire domain. Further experiments need to be conducted using datasets from different domains in order to come up with the choices for length and position of the sentences that are consistent across domains, genres of text and types of complex questions. In that case, additional features such as section title, and first sentences of individual sections might be considered when documents are categorized in different sections.

[9] WordNet (http://wordnet.princeton.edu/) is a widely used semantic lexicon for the English language. It groups English words (i.e. nouns, verbs, adjectives and adverbs) into sets of synonyms called synsets, provides short, general definitions (i.e. gloss definition), and records the various semantic relations between

hyponyms.

Named Entity (NE): The score 1 is given to a sentence that contains a Named Entity class among: PERSON, LOCATION, ORGANIZATION, GPE (Geo-Political Entity), FACILITY, DATE, MONEY, PERCENT, TIME. We believe that the presence of a Named Entity increases the importance of a sentence. We use the *OAK* System (Sekine, 2002), from New York University for Named Entity recognition[10]. The accuracy of the NE tagger used in the OAK system was reported to be of 72% recall and 80% precision (Sekine and Nobata, 2004).

Cue Word Match: The probable relevance of a sentence is affected by the presence of pragmatic words such as “significant”, “impossible”, “in conclusion”, “finally” etc. We use a cue word list[11] of 228 words. We give the score 1 to a sentence having any of the cue words and 0 otherwise.

2.6.2 Static Features: Query-related

n–gram Overlap: n–gram overlap measures the overlapping word sequences between the candidate document sentence and the query sentence where n stands for the length of the n-gram $(n = 1,2,3,4)$. We measured the recall based n–gram scores for a sentence S and a query Q using the following formula (Chali, Joty, and Hasan, 2009):

$$NgramScore(S,Q) = \frac{\sum_{gram_n \in S} Count_{match}\left(gram_n\right)}{\sum_{gram_n \in S} Count\left(gram_n\right)}$$

these synonym sets. We utilize the first-sense synsets. We use the WordNet version 3.0 in this research.

[10] We do not consider coreference resolution in this work.

[11] We constructed the cue word list from a list of transition words available at http://www.smart-words.org/transition-words.html

Where, n stands for the length of the n-gram $(n = 1,2,3,4)$ and $Count_{match}(gram_n)$ is the number of n-grams co-occurring in the query and the candidate sentence.

LCS: Given two sequences S_1 *and* S_2, the longest common subsequence (LCS) of S_1 *and* S_2 is a common subsequence with maximum length. We use this feature to calculate the longest common subsequence between a candidate sentence and the query. We used the LCS-based F-measure to estimate the similarity between a document sentence S of length m and a query sentence Q of length n as follows (Lin, 2004):

$$R_{lcs}(S,Q) = \frac{LCS(S,Q)}{m} \tag{2.6}$$

$$P_{lcs}(S,Q) = \frac{LCS(S,Q)}{n} \tag{2.7}$$

$$F_{lcs}(S,Q) = (1-\alpha) \times P_{lcs}(S,Q) + \alpha \times R_{lcs}(S,Q) \tag{2.8}$$

Where $LCS(S,Q)$ is the length of a longest common subsequence of S and Q and α is a constant that determines the importance of precision and recall. We set the value of α to 0.5 to give equal importance to precision and recall.

WLCS: Weighted Longest Common Subsequence (WLCS) improves the basic LCS method by remembering the length of consecutive matches encountered so far. Given two sentences X and Y, the WLCS score of X and Y can be computed using the similar dynamic programming procedure as stated in (Lin, 2004). The WLCS-based F-measure between a query and a sentence can be calculated similarly as LCS.

Skip-Bigram: Skip-bigram measures the overlap of skip-bigrams between a candidate sentence and a query sentence. Skip-bigram counts all in-order matching word pairs while LCS only counts one longest common subsequence. The skip bi-gram score between the document sentence S of length m and the query sentence Q of length n can be computed as follows (Chali, Joty, and Hasan, 2009):

$$R_{skip_2}(S,Q) = \frac{SKIP_2(S,Q)}{C(m,2)} \tag{2.9}$$

$$P_{skip_2}(S,Q) = \frac{SKIP_2(S,Q)}{C(n,2)} \tag{2.10}$$

$$F_{skip_2}(S,Q) = (1-\alpha) \times P_{skip_2}(S,Q) + \alpha \times R_{skip_2}(S,Q) \tag{2.11}$$

Where, $SKIP_2(S,Q)$ is the number of skip bi-gram matches between S and Q and α is a constant that determines the importance of precision and recall. We set the value of α as 0.5 to state equal importance to precision and recall. C is the combination function. We call the equation 2.11 skip bigram-based F-measure.

Exact-word Overlap: This is a measure that counts the number of words matching exactly between the candidate sentence and the query sentence. Exact-word overlap can be computed as follows:

$$Exact\ word\ overlap\ score = \frac{\sum_{w_1 \in WordSet} Count_{match}(w_1)}{\sum_{w_1 \in WordSet} Count(w_1)} \tag{2.12}$$

Where WordSet is the set of important[12] words in the sentence and $Count_{match}$ is the number of matches between the WordSet and the important query words.

Synonym Overlap: This is the overlap between the list of synonyms of the important words extracted from the candidate sentence and the query related[13] words. We use the

[12] Important words are essentially the content words (i.e. non stop words that are the nouns, verbs, adverbs and adjectives.)

[13] To establish the query related words, we took a query and created a set of related queries by replacing its important words by their first-sense synonyms using WordNet.

WordNet (Fellbaum, 1998) database for this purpose (Chali, Joty, and Hasan, 2009). Synonym overlap can be computed as follows:

$$Synonym\ overlap\ score = \frac{\sum_{w_1 \in SynSet} Count_{match}\left(w_1\right)}{\sum_{w_1 \in SynSet} Count\left(w_1\right)} \tag{2.13}$$

Where SynSet is the synonym set of the important words in the sentence and $Count_{match}$ is the number of matches between the SynSet and *query related words*.

Hypernym/Hyponym Overlap: This is the overlap between the list of hypernyms and hyponyms (up to level 2 in WordNet) of the nouns extracted from the sentence and the query related words. This can be computed as follows:

$$Hypernym/hyponym\ overlap\ score = \frac{\sum_{h_1 \in HypSet} Count_{match}\left(h_1\right)}{\sum_{h_1 \in HypSet} Count\left(h_1\right)} \tag{2.14}$$

Where HypSet is the hypernym/hyponym set of the nouns in the sentence and $Count_{match}$ is the number of matches between the HypSet and *query related words*.

Gloss Overlap: Our systems extract the glosses for the proper nouns from WordNet. Gloss overlap is the overlap between the list of important words that are extracted from the glossary definition of the nouns in the candidate sentence and the query related words. This can be computed as follows:

$$Gloss\ overlap\ score = \frac{\sum_{g_1 \in GlossSet} Count_{match}\left(g_1\right)}{\sum_{g_1 \in GlossSet} Count\left(g_1\right)} \tag{2.15}$$

Where GlossSet is the set of important words (i.e. nouns, verbs and adjectives) taken from the gloss definition of the nouns in the sentence and $Count_{match}$ is the number of matches between the GlossSet and query related words.

Syntactic Feature: The first step to calculate the syntactic similarity between the *query* and the *sentence* is to parse them into syntactic trees using a syntactic parser (Chali, Joty, and Hasan, 2009). We use the Charniak parser[14] (Charniak, 1999) for this purpose. Once we build the syntactic trees, our next task is to measure the similarity between the trees. For this, every tree T is represented by an m dimensional vector $v(T) = (v_1(T), v_2(T), \cdots v_m(T))$, where the i-th element $v_i(T)$ is the number of occurrences of the i-th tree fragment in tree T. The tree fragments of a tree are all of its sub-trees which include at least one production with the restriction that no production rules can be broken into incomplete parts. The similarity between two syntactic trees can be computed using the *tree kernel* function (Collins and Duffy, 2001). The *TK* (tree kernel) function gives the similarity score between the *query* and the *document sentence* based on their syntactic structures. The tree kernel of the two syntactic trees, T_1 and T_2 is actually the inner product of the two m-dimensional vectors, $v(T_1)$ and $v(T_2)$ (see details in Section 4.4.2) (Zhang and Lee, 2003; Moschitti et al., 2007; Moschitti and Basili, 2006):

$$TK(T_1, T_2) = v(T_1).v(T_2)$$

Basic Element (BE) Overlap: Basic Elements are defined as follows (Hovy et al., 2006):

- the head of a major syntactic constituent (noun, verb, adjective or adverbial phrases), expressed as a single item, or

[14] Available at ftp://ftp.cs.brown.edu/pub/nlparser/

- a relation between a head-BE and a single dependent, expressed as a triple: (head|modifier| relation).

We extract BEs for the sentences (or query) by using the BE package distributed by ISI[15]. We compute the Likelihood Ratio (LR) for each BE following (Zhou, Ticrea, and Hovy, 2005). We sort the BEs based on LR scores to produce a BE-ranked list. The ranked list contains important BEs at the top which may or may not be relevant to the complex question. We filter out the BEs that are not related to the query and get the BE overlap score (Zhou, Ticrea, and Hovy, 2005).

2.6.3 Dynamic Feature

For each sentence that is selected for the summary pool, we measure its similarity with the remaining non-selected sentences using ROUGE. The similarity value is encoded into the feature space of the non-selected sentences as the dynamic feature. The purpose of this feature is to ensure that the next sentence to be chosen into the summary is significantly different from the sentence that is already there. In other words, from the dynamic feature of a sentence we can understand whether the sentence can add any new information into the summary or not. The dynamic feature is updated each time a new sentence is added to the summary. We use the Maximal Marginal Relevance (MMR)[16] method (Carbonell and Goldstein, 1998) to balance this feature with query relevance. We give equal importance to query relevance and redundancy reduction such that the selected sentence can add some valuable as well as new information to the summary.

[15] BE website: http://www.isi.edu/˜cyl/BE

[16] A sentence has high marginal relevance if it is both relevant to the query and contains minimal similarity to previously selected sentences.

2.7 Evaluation Framework and Results

2.7.1 *Task Overview*

We consider the DUC-2007 task for our experiments. The task is as follows:

"Given a complex question (topic description) and a collection of relevant documents, the task is to synthesize a fluent, well-organized 250-word summary of the documents that answers the question(s) in the topic".

For example, given the topic description (from DUC 2007):

```
<topic> <num> D0703A </num>
        <title> steps toward introduction of the Euro </title>
        <narr>
        Describe steps taken and worldwide reaction prior to the
        introduction of the Euro on January 1, 1999. Include
        predictions and expectations reported in the press.
        </narr>
</topic>
```

and a collection of relevant documents, the task of the summarizer is to build a summary that answers the question(s) in the topic description. We consider this task[17] and apply a reinforcement approach to generate topic-oriented 250-word extract summaries (See sample summaries in Appendix C).

[17] For simplicity, our system does not attempt to address fluency in this research. However, fluency could be addressed using sentence ordering and co-reference resolution according to the procedure followed by Pingali, K., and Varma (2007).

2.7.2 Corpus for Training and Testing

The DUC-2006 and DUC-2007 document sets came from the AQUAINT corpus, which is comprised of newswire[18] articles from the Associated Press and New York Times (1998-2000) and Xinhua News Agency (1996-2000). In table 2.1, we present the description of the datasets used in our experiments. We use the DUC-2006 data to learn a weight for each of the features (described in Section 2.6) and then use these weights to produce extract summaries for the document clusters of DUC-2007. We also use the given abstract summaries for each topic as the training data. In DUC-2006, each topic (including a complex question) and its document cluster were given to 4 different NIST assessors, including the developer of the topic. Each assessor created a 250-word summary of the document cluster that satisfies the information need expressed in the topic. These multiple reference summaries were used in the training stage to calculate the numeric rewards.

Characteristics	Training Data	Testing Data
Conference	DUC-2006	DUC-2007
Number of clusters	50	25
Number of topics with associated complex questions	50	25
Number of documents per clusters	25	25

Table 2.1: Description of the datasets

2.7.3 Systems for Comparisons

Baseline System: We report the evaluation scores of one baseline system (used in DUC-2007) in each of the tables in order to show the level of improvement our system achieved. The baseline system generates summaries by returning all the leading sentences (up to 250

[18] Although we perform experiments on the newswire articles, we speculate that our feature space is also capable to take other types of datasets like *Yahoo! Answers* into consideration.

words) in the $\langle TEXT \rangle$ field of the most recent document(s).

SVM settings: We compare the performance of our reinforcement learning approach with a SVM-based technique to answer complex questions. A support vector based approach requires a huge amount of training data during the learning stage. Here, typically, the training data includes a collection of sentences where each sentence is represented as a combination of a feature vector and corresponding class label ($+1$ or -1). We use the same corpus (Section 2.7.2) for training and testing during the SVM experiments. We generate a training data set by automatically annotating (using ROUGE similarity measures) 50% of the sentences of each document cluster as positive and the rest as negative. The choice of 50% is based on the fact that SVM can learn well from a balanced (equal proportion of positives and negatives) set of examples (Chali and Hasan, 2012b). The annotation follows a similar procedure to (Chali, Hasan, and Joty, 2009). The same feature set (Section 2.6) is used to represent the document sentences as feature vectors except the dynamic feature. The dynamic feature seemed to be inappropriate for the SVM setting. However, to reduce the redundancy in the system-generated summaries, we use the MMR-approach during the summary generation process.

During the training step, we used the third-order polynomial kernel with the default value of C[19]. We used the SVM^{light}[20] (Joachims, 1999) package. We performed the SVM training experiments using WestGrid[21] to mitigate computation time. We used the *Cortex* cluster which is comprised of shared-memory computers for large serial jobs or demanding parallel jobs.

In the multi-document summarization task at DUC-2007, the required summary length was 250 words. In our SVM setup, we used $g(x)$, the normalized distance from the hyper-

[19] C is the trade-off parameter of SVM. We experimented with different kernels and found that the third-order polynomial kernel with the default value of C performs best.

[20] http://svmlight.joachims.org/

[21] http://westgrid.ca/

plane to x to rank the sentences (Chali, Hasan, and Joty, 2009). Then, we chose the top N sentences as the candidate summary sentences. Initially, the top-ranked sentence is added to the summary and then we perform an MMR-based computation to select the next sentence that is equally valuable as well as new (balancing the query relevance and redundancy factor). We continue this task until the summary length of 250-words is reached.

K-means Clustering: The k-means algorithm follows a simple way to cluster a given data set through a pre-specified number of clusters k. There are several approaches (such as "iK-Means" by (Mirkin, 2005), Hartigan's method (Hartigan and Wong, 1979) etc.) to estimate the number of clusters.

The k-means algorithm defines clusters by the center of mass of their members (Manning and Schutze, 2000). We start with a set of initial cluster centers that are chosen randomly and go through several iterations of assigning each object to the cluster whose center is the closest. After all objects have been assigned, we recompute the center of each cluster as the centroid or mean ($\boldsymbol{\mu}$) of its members. We use the squared Euclidean distance as the distance function. Once we have learned the means of the clusters using the K-means algorithm, our next task is to rank the sentences according to a probability model. A Bayesian model is used for this purpose (Chali, Joty, and Hasan, 2009).

2.7.4 Evaluation and Analysis

Automatic Evaluation (ROUGE): Similar to DUC-2006, in DUC-2007, each of the four assessors created a 250-word summary of the document cluster that satisfies the information need expressed in the topic statement. These multiple "reference summaries" were used in the evaluation of our system-generated summary[22] content. We considered

[22] The summaries are truncated to 250 words by ROUGE if the summary length reaches over the 250 word limit due to the inclusion of a complete sentence.

the widely used evaluation measures Precision (P), Recall (R) and F-measure for our evaluation task. *Recall* is defined as the ratio of the number of units (sentences/words) of the system-generated summaries in common with the reference summaries to the total number of units in the *reference* summary while *precision* is the ratio of the number of units of system-generated summaries in common with the reference summaries to the total number of units in the *system-generated* summaries. F-measure combines precision and recall into a single measure to compute the overall performance. We evaluate the system generated summaries using the automatic evaluation toolkit ROUGE (Lin, 2004) which has been widely adopted by DUC. ROUGE parameters were set as that of DUC-2007 evaluation setup. We report the scores of the two official ROUGE metrics of DUC, ROUGE-2 (bigram) and ROUGE-SU (skip bigram). All the ROUGE measures are calculated by running ROUGE-1.5.5 with stemming but no removal of stopwords.

ROUGE run-time parameters:

```
ROUGE-1.5.5.pl -2 -1 -u -r 1000 -t 0 -n 4 -w 1.2 -m -l 250 -a
```

Table 2.2 and table 2.3 show the ROUGE precision and recall scores of the reinforcement system, the supervised SVM system, and the unsupervised k-means system. In Table 2.4, we compare the ROUGE-F scores of the baseline system, SVM system, k-means system, and reinforcement system. From here, we find that the **reinforcement** system improves the ROUGE-2 and ROUGE-SU scores over the **baseline** system by 32.9% and 21.1%, respectively. On the other hand, the **reinforcement** system outperforms the supervised **SVM** system demonstrating improvements to the ROUGE-2 and ROUGE-SU scores by 28.4% and 2.7%, respectively besides performing very closely to the unsupervised **k-means** system.

Before starting these experiments, our intuition was that the reinforcement learning method should perform the best in comparison to both the supervised and unsupervised methods. However, this was not really the case all the time as we find that the ROUGE scores of the unsupervised k-means system is better than the reinforcement learning method. The reason for this might be that ROUGE might not be the right metric for this evaluation. This is why we conduct an extensive manual evaluation to analyze the performance of the systems. We also claim that the performance of our reinforcement learning method could further improve if we can improve upon the characteristics of the reward function since it plays a major role in the overall learning framework. As we used ROUGE for the reward value computation, it tried to find word-based similarity matching with the human-generated abstract summaries. Human-made summaries do not necessarily contain exact sentences from the document collection, rather a variation of them that involves paraphrasing. This might be the reason why ROUGE is not able to capture the true similarity between a candidate sentence and the abstract summary sentences. More effective textual similarity measures such as syntactic or semantic similarity matching could lead to accurate reward value calculation. We would like to explore this research in the future.

Systems	ROUGE-2	ROUGE-SU
SVM	0.0707	0.1477
K-means	0.1072	0.1742
Reinforcement	0.0878	0.1417

Table 2.2: ROUGE measures: Precision

Systems	ROUGE-2	ROUGE-SU
SVM	0.0641	0.1209
K-means	0.0779	0.1348
Reinforcement	0.0849	0.1319

Table 2.3: ROUGE measures: Recall

Statistical Significance: An approximate result to identify which differences in the competing systems' scores are significant can be achieved by comparing the 95% confidence intervals for each mean. In Table 2.4, we show the 95% confidence intervals of all the systems to report significance for doing meaningful comparison. ROUGE uses a randomized method named bootstrap resampling to compute the confidence intervals. Bootstrap resampling has a long tradition in the field of statistics (Efron and Tibshirani, 1994). We use 1000 sampling points in the bootstrap resampling. Two systems can be judged as significantly different if one of the two criteria becomes true: 1) their confidence intervals for the estimates of the means do not overlap at all, or 2) the two intervals overlap but neither contains the best estimate for the mean of the other (Schenker and Gentleman., 2001). Analyzing the reported confidence intervals of different systems, we see that the **reinforcement** system significantly outperforms the **baseline** system according to the first criterion. We also see that the confidence intervals of the **SVM**, **reinforcement** and **k-means** systems do overlap. However, according to the second criterion, we find that the **reinforcement** system is significantly better than the **SVM** system in terms of ROUGE-2 scores. On the other hand, there is no significant difference between the **reinforcement** system and the **k-means** system if we interpret the confidence intervals by considering the both criteria.

Systems	ROUGE-2	Confidence Intervals	ROUGE-SU	Confidence Intervals
Baseline	0.0649	0.0608 - 0.0688	0.1127	0.1084 - 0.1167
SVM	0.0672	0.0570 - 0.0787	0.1329	0.1218 - 0.1444
K-means	0.0902	0.0662 - 0.0953	0.1520	0.1241 - 0.1594
Reinforcement	0.0863	0.0740 - 0.0968	0.1365	0.1236 - 0.1478

Table 2.4: Performance comparison: F-Score with confidence intervals

Manual Evaluation: It might be possible to get state-of-the-art ROUGE scores although the generated summaries are bad (Sjöbergh, 2007). Therefore, we conduct an extensive manual evaluation in order to analyze the effectiveness of our approach. Two university graduate students judged the summaries for linguistic quality and overall responsiveness according to the DUC-2007 evaluation guidelines. “The given score is an integer between

1 (very poor) and 5 (very good) and is guided by consideration of the following factors: 1. Grammaticality, 2. Non-redundancy, 3. Referential clarity, 4. Focus, and 5. Structure and Coherence. They also assigned a content responsiveness score to each of the automatic summaries. The content score is an integer between 1 (very poor) and 5 (very good) and is based on the amount of information in the summary that helps to satisfy the information need expressed in the topic narrative[23]."

Table 2.5 presents the average linguistic quality and overall responsive scores of all the systems. From these results, we can see that the reinforcement system does not perform well compared to the baseline system in terms of linguistic quality. This fact is understandable since our approach did not consider any post-processing and sentence-ordering algorithms to fine-tune the system-generated summaries by ignoring the fluency component of the system task. However, in terms of overall content responsiveness, the reinforcement system outperformed all other systems indicating a better accuracy in meeting the user-requested information need. The differences between the systems were computed to be statistically significant[24] at $p < 0.05$ except for the difference between the **k-means** and the **reinforcement** system in terms of linguistic quality.

Systems	Linguistic Quality	Overall Responsiveness
Baseline	4.24	1.86
SVM	3.48	3.20
K-means	3.30	3.45
Reinforcement	3.32	3.80

Table 2.5: Average linguistic quality and overall responsiveness scores for all systems

Most Effective Features for Reinforcement Learning: After the training phase, we get the final updated weights corresponding to each feature. The smallest weight value indi-

[23] http://www-nlpir.nist.gov/projects/duc/duc2007/quality-questions.txt

[24] We tested statistical significance using Student's t-test and a p value less than 0.05 was considered significant.

cates that the associated feature can be eliminated because it does not contribute any relevant information for *action* selection. From this viewpoint we can infer that — weights reflect the effectiveness of a certain feature. Table 2.6 shows the top ten final feature weights (ranked by higher effectiveness) for this problem domain that we find after the training experiment. The table shows that the Basic Element overlap feature is the most effective feature followed by the syntactic feature and the sentence length feature. On the other hand, 1–gram overlap has the lowest weight value denoting the fact that this feature has little impact on the reinforcement learner.

Final Weight	Associated Feature
0.012837	Basic Element Overlap
0.007994	Syntactic Feature
0.007572	Length of Sentences
0.006483	Cue Word Match
0.005235	Named Entity Match
0.002201	2–gram Overlap
0.002182	Title Match
0.001867	Skip–Bigram
0.001354	WLCS
0.001282	1–gram Overlap

Table 2.6: Effective features

Experiments with User Interaction

System Description: The major objective of this experiment is to study the impact of the user interaction component in the reinforcement learning framework. To accomplish this purpose, we use the first 30 topics at most[25] from the DUC-2006 data to learn the weights respective to each feature and then use these weights to produce extract summaries for the next 15 topics (test data).

[25] We build several reinforcement systems by varying the training data size.

We follow six different ways of learning the feature weights by varying the amount of user interaction incorporated and the size of the training data: **1) SYS_0_20**, **2) SYS_10_20**, **3) SYS_20_0**, **4) SYS_20_10**, **5) SYS_30_0**, and **6) SYS_30_30**. The numbers in the system titles indicate how many user-interaction and non-user-interaction topics each system included during training, respectively. For example, the first system is trained with 20 topics of the DUC-2006 data without user interaction. Among these systems, the sixth system is different as it is trained with the first 30 topics of the DUC-2006 data without user interaction. The learned weights that are found from the **SYS_30_0** experiment are used as the initial weights of this system. This means that the **SYS_30_30** system is trained with 60 topics in a pseudo-manner (30 topics with interaction from **SYS_30_0** and 30 topics without interaction).

The outcomes of these systems are sets of learned feature weights that are used to generate extract summaries (i.e. answers) for the last 15 topics (test data) of the DUC-2006 data set. After the six learning experiments, we get six sets of learned feature weights which are used to generate six different sets of summaries for the test data (15 topics). We evaluate these six versions of summaries for the same topics and analyze the effect of user interaction in the reinforcement learning framework.

Evaluation: We report the two official ROUGE metrics of DUC-2006 in the results: ROUGE-2 (bigram) and ROUGE-SU (skip bigram). In Table 2.7, we compare the ROUGE-F scores of all the systems. In our experiments, the only two systems that were trained with 20 topics are **SYS_0_20** and **SYS_20_0** (the one has 20 unsupervised, the other has 20 supervised). From the results, we see that the **SYS_20_0** system improves the ROUGE-2 and ROUGE-SU scores over the **SYS_0_20** system by 0.57%, and 0.47%, respectively. Again, we see that the **SYS_20_10** system improves the ROUGE-2 and ROUGE-SU scores over the **SYS_10_20** system (both systems had 30 topics but where **SYS_20_10** had more human-supervised topics) by 0.96%, and 8.56%, respectively. We also find that the **SYS_30_0** system improves the ROUGE-2 and ROUGE-SU scores over the **SYS_20_10** system (both systems had 30 topics with **SYS_30_0** having more human supervision) by 0.25%, and 0.80%, respectively. The results show a clear trend of improvement when human interaction is incorporated. We can also see that the **SYS_30_30** system is performing the best since it starts learning from the learned weights that are generated from the outcome of the **SYS_30_0** setting. This denotes that the user interaction component has a positive impact on the reinforcement learning framework that further controls the automatic learning process efficiently (after a certain amount of interaction has been incorporated). In table 2.8, we report the 95% confidence intervals for ROUGE-2 and ROUGE-SU to show the significance of our results.

Systems	ROUGE-2	ROUGE-SU
SYS_0_20	0.0522	0.1186
SYS_10_20	0.0598	0.1146
SYS_20_0	0.0525	0.1192
SYS_20_10	0.0604	0.1244
SYS_30_0	0.0605	0.1254
SYS_30_30	0.0605	0.1257

Table 2.7: Performance comparison: F-Scores

Systems	ROUGE-2	ROUGE-SU
SYS_0_20	0.0407 - 0.0632	0.1106 - 0.1268
SYS_10_20	0.0462 - 0.0734	0.1144 - 0.1344
SYS_20_0	0.0413 - 0.0633	0.1113 - 0.1274
SYS_20_10	0.0467 - 0.0737	0.1144 - 0.1344
SYS_30_0	0.0463 - 0.0747	0.1148 - 0.1344
SYS_30_30	0.0500 - 0.0754	0.1177 - 0.1343

Table 2.8: 95% confidence intervals for different systems

We also conduct an extensive manual evaluation of the systems. Two university graduate students judged the summaries for linguistic quality and overall responsiveness according to the DUC-2007 evaluation guidelines. Table 2.9 presents the average linguistic quality and overall responsive scores of all the systems. Analyzing these results, we can clearly see the positive impact of the user interaction component in the reinforcement learning framework. We find that the performance of different systems gradually improves when more user interaction is provided as the guide for candidate sentence selection. From these comparisons we can also conclude that the reinforcement learning system can automatically learn well after a sufficient amount of training with user interaction. The improvements in the results are statistically significant ($p < 0.05$).

Systems	Linguistic Quality	Overall Responsiveness
SYS_0_20	2.92	3.20
SYS_10_20	3.45	3.40
SYS_20_0	3.12	3.39
SYS_20_10	3.50	3.72
SYS_30_0	3.68	3.84
SYS_30_30	3.96	4.10

Table 2.9: Linguistic quality and responsiveness scores

Discussion: The main goal of the reinforcement learning phase is to learn the appropriate feature weights that can be used in the testing phase. When the agent is in learning mode, in each iteration the weights get updated depending on the immediate reward it receives after selecting an action. To illustrate, when a user is interacting with the system (according to Section 2.5), in each iteration one sentence is chosen to be included into the summary space. The top five candidates vary based on the previously selected *action*. For each considered topic, the user provides feedback for three consecutive iterations while automatic learning continues in the remaining iterations. When the summary length reaches to 250 words, we obtain the weights learned from one topic. These weights become the initial weights for the next topic. The user again interacts with the system for three iterations and the process continues for a certain number of topics. Then, the agent starts automatic learning for the remaining topics.

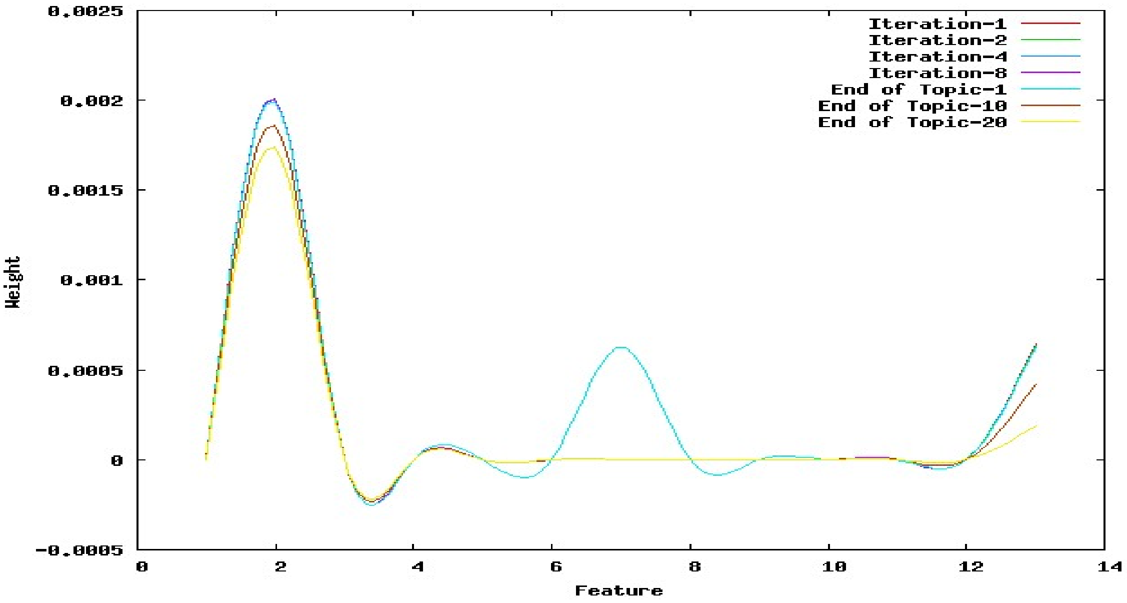


Figure 2.2: Effect of user feedback on feature weights

The effect of user feedback on the feature weights can be shown using a graph. We present the weights from different stages of the **SYS_20_0** experiment in figure 2.2. Note that the **SYS_20_0** system is trained with 20 topics of the DUC-2006 data with user interaction. We draw the graph using the *gnuplot*[26] graphing utility. To connect all data points smoothly, we used the *"smooth csplines"* option. The labels in the Y-axis refers to the features[27] in the following order: 1) 1-gram overlap, 2) 2-gram overlap, 3) LCS, 4) WLCS, 5) exact word overlap, 6) synonym overlap, 7) hypernym/hyponym overlap, 8) sentence length, 9) title match, 10) named entity match, 11) cue word match, 12) syntactic feature, and 13) BE overlap. We also show the weights from different stages of the **SYS_0_20** experiment in figure 2.3. Note that the **SYS_0_20** system is trained with 20

[26] http://www.gnuplot.info/

[27] We include those features that have at least one non-zero weight value in any of the considered stages.

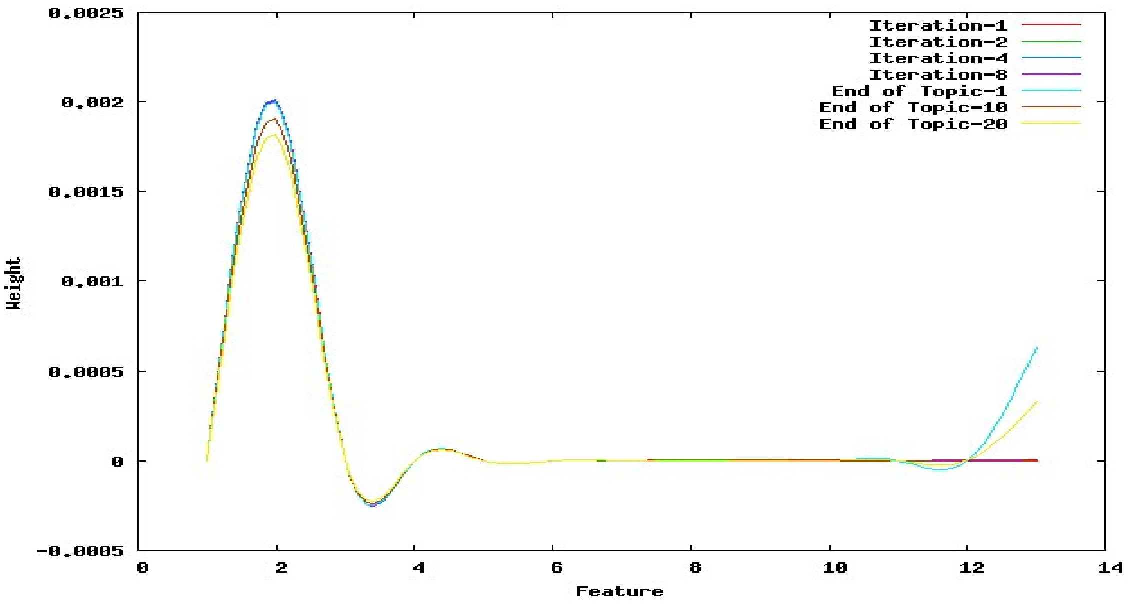


Figure 2.3: Effect of fully automatic learning on feature weights

topics of the DUC-2006 data without user interaction. This graph shows how automatic learning affects the weights during the same stages as shown in the previous graph. If we compare the two graphs, we find that both the graphs show a similar kind of trend, i.e., at the end of the learning phase (end of topic-20), all the feature weights converge to zero except for 2-gram overlap and BE overlap. However, the main point to notice here is how quickly they converged. From the figures, we can see that the **SYS_20_0** system converged quickly for the important two features (2-gram overlap and BE overlap) by immediately lowering the values in iteration-2. We can also see that the hypernym/hyponym overlap feature gets an abrupt increase in its value during iteration-8 while the value is lowered later. This phenomenon indicates that the reinforcement system is responsive to the user interests and actions. On the other hand, from figure 2.3 we understand that the **SYS_0_20** system converges slowly by following a fixed pattern. These experiments suggest that the incorporation of user feedback can guide the candidate sentence selection process in the right direction from the beginning of the learning phase. We can conclude that our reinforcement system can learn quickly and effectively from the provided user feedback[28]. In this experiment, we interacted with the system for three iterations for each topic. We claim that the learning performance will improve significantly if more user interaction is provided during the learning phase. The evaluations shown in Section 2.7.4 also supports this claim. In Appendix B, we show an example of how the ranking of the sentences in a final summary is affected as the user provides feedback. We also show a final summary that was generated using no interaction. Comparison of the summaries yields how the ranking of the sentences differed due to different training schemes.

2.8 Conclusion

We have presented a reinforcement learning formulation of the complex question answering problem. Our main motivation behind the proposal of the reinforcement learning formulation was in fact to enable learning from human interaction in real time as we believe that the incorporation of user feedback into the learning process can lead to a robust system that produces more accurate summaries to increase the level of user satisfaction. Initially, we have simplified our formulation by not taking the real user feedback into account with the assumption that users are satisfied with the given human-generated abstract summaries. Later, we have incorporated the actual user feedback during the candidate sentence selection step to provide effective guidance for generating summaries according to the user satisfaction. A series of experiments have been conducted on the DUC benchmark datasets which demonstrate the appropriateness of the reinforcement learning framework for the complex question answering task. A detailed analysis of the results has revealed that the performance of our proposed system would further improve by using a more subtle reward function that can consider syntactic and semantic properties of the sentences.

[28] Since the main intuition behind using a reinforcement learning methodology for the complex question answering task was in fact to enable learning from user interaction in real time (for providing an enhanced user satisfaction), quick convergence is a crucial characteristic of the proposed system.

Chapter 3

Complex Question Answering using Integer Linear Programming

3.1 Introduction

In this chapter, we describe how we formulate the complex question answering task using Integer Linear Programming (ILP). As discussed before, we treat the complex question answering task as a query-focused multi-document summarization task. Query-focused multi-document summarization aims to create a summary from the available source documents that can answer the requested information need (Chali, Hasan, and Imam, 2012a). Extraction-based automatic summarization has been a common practice over the years for its simplicity. Extraction of the most important sentences to form a summary can degrade the summary quality if there exists a longer sentence with partly relevant information to prevent inclusion of other important sentences (due to summary length constraint) (Martins and Smith, 2009). Sentence compression can be a good remedy for this problem where the task can be viewed as a single-sentence summarization (Jing, 2000; Clarke and Lapata, 2008). This motivates us to incorporate a sentence compression component into our ILP formulation to complex question answering.

Sentence compression[1] aims to retain the most important information of a sentence in the shortest form whilst being grammatical at the same time (Knight and Marcu, 2000; Knight and Marcu, 2002; Lin, 2003). Previous researches have shown that sentence compression can be used effectively in automatic summarization systems to produce more in-

[1] Although most of the works on sentence compression are mainly related to the English language, researchers have also worked on sentence compression related to languages other than English (Molina et al., 2011; Filippova, 2010; Bouayad-Agha et al., 2006). Our work is applied to the English language. However, we believe that the proposed techniques can be applicable to other languages provided that the lexical, syntactical and semantic properties of the corresponding language are considered.

formative summaries by reducing the redundancy in the summary sentences (Jing, 2000; Knight and Marcu, 2002; Lin, 2003; Daumé III and Marcu, 2005; Zajic et al., 2007; Madnani et al., 2007; Martins and Smith, 2009; Berg-Kirkpatrick, Gillick, and Klein, 2011). However, most of these researches either focused on the task of single document summarization and generic summarization or did not consider global properties of the sentence compression problem (Clarke and Lapata, 2008).

Due to the vast increase in both the amount of online data and the demand for access to different types of information in recent years, attention has shifted from single document and generic summarization toward query-based multi-document summarization. On the other hand, sentence compression can achieve superior performance if it can be treated as an optimization problem and solved using ILP to infer globally optimal compressions (Gillick and Favre, 2009; Clarke and Lapata, 2008).

ILP has recently attracted much attention in the natural language processing (NLP) community (Roth and Yih, 2004; Clarke and Lapata, 2008; Punyakanok et al., 2004; Riedel and Clarke, 2006; Denis and Baldridge, 2007). Gillick and Favre (2009) proposed to extend their ILP formulation for a concept-based model of summarization by incorporating additional constraints for sentence compression. However, to the best of our knowledge, there has not been a single research that deeply investigates the potential of using ILP-based sentence compression models for the task of query-focused multi-document summarization. In this thesis, we attempt to fill this gap in the literature by considering both compression and summarization as global optimization problems.

The sentence compression models used in the existing automatic summarization systems mostly exploit various lexical and syntactic properties of the sentences (Knight and Marcu, 2002; Mcdonald, 2006; Clarke and Lapata, 2008; Cohn and Lapata, 2008; Galanis and Androutsopoulos, 2010). A recent work has shown that discourse segmentation could be incorporated in a sentence compression system which can aid automatic summarization (Molina et al., 2011). Lin (2003) showed that pure syntactic-based compression does not improve a generic summarization system. A most recent work has shown that sentence compression can achieve better performance if semantic role information can be incorporated into the model (Yoshikawa et al., 2012). Inspired by their work, we recast their formulation as an ILP for sentence compression with semantic role constraints. We build three different ILP-based sentence compression models: 1) a bigram language model with lexical and syntactic constraints (derived from Clarke and Lapata (2008)), 2) the bigram language model with a topic signature modeling function (Lin and Hovy, 2000), and 3) the bigram language model with semantic role constraints (Yoshikawa et al., 2012). We choose to build them since the variation of these models were shown to achieve better results comparable to the state-of-the-art techniques (Clarke and Lapata, 2008; Yoshikawa et al., 2012). We perform a rigorous study to analyze the effectiveness of using these sentence compression models to generate query-focused summaries. For this study, we compose three different models depending on the order to perform sentence compression and extraction: 1) *ComFirst*, 2) *SumFirst*, and 3) *Combined*. The main motivation behind building these models is that we intend to study if the order of performing compression and extraction can affect the overall performance of the query-focused multi-document summarization. Martins and Smith (2009) argued that the two-step "pipeline" approaches such as *ComFirst* and *SumFirst* might often fail to select global optimal summaries.

Query-focused extractive multi-document summarization generally needs three essential criteria to be satisfied (McDonald, 2007): 1) Relevance: to contain informative sentences relevant to the given query, 2) Redundancy: to not contain multiple similar sentences, and 3) Length: should follow a fixed length constraint. We define a global optimization model that uses ILP to infer optimal summaries. The existing ILP formulations to the summarization task mostly rely on relevance and redundancy functions (such as word-level cosine similarity measure, word bigrams) that are primitive in nature (McDonald, 2007; Gillick and Favre, 2009; Martins and Smith, 2009). The major limitation of these approaches is that they do not consider the sequence of words (i.e. word ordering). They ignore the syntactic and semantic structure of the sentences and thus, cannot distinguish between “The police shot the gunman” and “The gunman shot the police”. The researchers speculate that the better the relevance and redundancy functions could be, the more the solutions would be efficient (Gillick and Favre, 2009). In the proposed optimization framework, we incorporate a comprehensive set of query-related and importance-oriented measures to define the relevance function. We employ four alternative redundancy constraints based on different sentence similarity measures using a) cosine similarity, b) syntactic similarity, c) semantic similarity, and d) extended string subsequence kernel (ESSK). We propose the use of syntactic tree kernel (Moschitti and Basili, 2006), shallow semantic tree kernel (Moschitti et al., 2007), and a variation of the extended string subsequence kernel (ESSK) (Hirao et al., 2003) to accomplish the task. Our empirical evaluation on the DUC benchmark datasets demonstrate the effectiveness of applying sentence compression for the task of query-focused multi-document summarization. The results also show that the quality of the generated summaries vary based on the use of alternative redundancy constraints in the optimization framework. The rest of the chapter is organized as follows: Section 3.2 focuses on our ILP-based sentence compression models, Section 3.3 discusses the proposed ILP formulation to query-focused multi-document summarization, and Section 3.4 presents the experiments and evaluation results.

3.2 ILP-based Sentence Compression Models

An ILP is a constrained optimization problem, where both the cost function and constraints are linear in a set of integer variables (McDonald, 2007; Clarke and Lapata, 2008). In this section we describe three ILP-based sentence compression models which we apply for the task of query-focused multi-document summarization. Our first model is a bigram language model derived from the work of Knight and Marcu (2002; Clarke and Lapata (2008). Our second model is close in spirit rather different in content to Clarke and Lapata (2008). In this model, we combine the bigram language model with a corpus-based topic signature modeling approach of Lin and Hovy (2000). Our first two models include various lexical and syntactical constraints based on the work of Clarke and Lapata (2008). In the third model, we add a set of semantically motivated constraints into the bigram language model based on the work of Yoshikawa et al. (2012).

3.2.1 Bigram Language Model

According to Clarke and Lapata (2008), the sentence compression problem can be formally defined as follows. Let $S = w_1, w_2, \cdots, w_n$ is an original sentence in a document. To represent the words to be included in the compressed version of this sentence, we define a set of indicator variables δ_i that are set to 1 if i-th word is selected into the compression, and 0 otherwise. To make decisions based on word sequences (rather than individual words), we define additional indicator variables a_i (that are set to 1 if i-th word starts the compression, and 0 otherwise), b_i (that are set to 1 if i-th word ends the compression, and 0 otherwise), and c_{ij} (that are set to 1 if sequence w_i, w_j is present in the compression, and 0 otherwise). Now the inference task is solved by maximizing the following objective function (that includes the overall sum of the decision variables multiplied by their log-transformed corpus bigram probabilities) (Clarke and Lapata, 2008):

$$Maximize \sum_i a_i \cdot P(w_i|start) + \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} c_{ij} \cdot P(w_j|w_i) + \sum_i b_i \cdot P(end|w_i) \quad (3.1)$$

such that $\forall i, j \in \{1 \cdots n\}$:

$$\delta_i, a_i, b_i, c_{ij} \in \{0, 1\} \quad (3.2)$$

$$\sum_i a_i = 1 \tag{3.3}$$

$$\delta_j - a_j - \sum_{i=1}^{j} c_{ij} = 0 \tag{3.4}$$

$$\delta_i - \sum_{j=i+1}^{n} c_{ij} - b_i = 0 \tag{3.5}$$

$$\sum_i b_i = 1 \tag{3.6}$$

$$\sum_i \delta_i \geq l \tag{3.7}$$

$$\sum_{i:w_i \in verbs} \delta_i \geq 1 \tag{3.8}$$

$$\begin{gathered} \delta_i = 1 \\ \forall i : w_i \in personal\, pronouns \end{gathered} \tag{3.9}$$

$$\begin{gathered} \delta_i = 0 \\ \forall i : w_i \in words\, in\, parentheses \end{gathered} \tag{3.10}$$

$$\begin{gathered} \delta_i - \delta_j = 0 \\ \forall i, j : w_j \in possessive\, mods\, of\, w_i \end{gathered} \tag{3.11}$$

The objective function in Equation 3.1 is maximized to find the optimal target compression where “start” and “end” denote w_0 and w_n, respectively. The above ILP formulation incorporates various constraints. The first constraint states that the variables are binary. The later constraints are defined to disallow invalid bigram sequences in the compression. Constraint 3.3 states that exactly one word can start a compression. Constraint 3.4 and Constraint 3.5 are responsible to ensure correct bigram sequences, whereas Constraint 3.6 denotes that exactly one word can end the compression. On the other hand, Constraint 3.7 forces the compression to have at least l words. We add some additional constraints (Constraint 3.8 to Constraint 3.11) from Clarke and Lapata (2008) to ensure that the target compressions are lexically and syntactically acceptable. To accomplish this purpose, we use the Oak system[2] (Sekine, 2002) and the Charniak parser[3] (Charniak, 1999) to obtain information regarding parts-of-speech and grammatical relations in a sentence.

3.2.2 Topic Signature Model

We use a topic signature modeling approach (Lin and Hovy, 2000) to identify the important content words from the original source sentence. The important words are considered to have significantly greater probability of occurring in a given text compared to that in a large background corpus. We incorporate this importance score into the objective function of the bigram language model (Section 3.2.1) to ensure that the target compression prefers to keep important content words. We use a topic signature computation tool[4] for this purpose. The background corpus that is used in this tool contains 5000 documents from the English GigaWord Corpus. Our modified objective function becomes:

$$Maximize \sum_i \delta_i \cdot I(w_i) + \sum_i a_i \cdot P(w_i|start) + \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} c_{ij} \cdot P(w_j|w_i) + \sum_i b_i \cdot P(end|w_i) \tag{3.12}$$

where $I(w_i)$ denotes the importance score of the i-th word.

[2] http://nlp.cs.nyu.edu/oak/

[3] Available at ftp://ftp.cs.brown.edu/pub/nlparser/

[4] Available at http://www.cis.upenn.edu/~lannie/topicS.html

3.2.3 Bigram Language Model with Semantic Constraints

Yoshikawa et al. (2012) have proposed a set of formulas called Markov Logic Network (MLN) to build a semantically motivated sentence compression model and showed that their model achieves improved performance. We recast their formulas as constraints of our ILP model and incorporate them into the bigram language model. The main idea is to utilize the predicate-argument relations of a sentence and define constraints based on semantic roles to improve the weaknesses of the lexical and syntactical constraints. In this manner, we can ensure that the target compression contains meaningful information. For this purpose, we parse the source sentence semantically using a Semantic Role Labeling (SRL) system (Kingsbury and Palmer, 2002; Hacioglu et al., 2003), ASSERT[5]. When presented with a sentence, ASSERT performs a full syntactic analysis of the sentence, automatically identifies all the verb predicates in that sentence, extracts features for all constituents in the parse tree relative to the predicate, and identifies and tags the constituents with the appropriate semantic arguments. We add the following additional constraints as the semantic constraints to our bigram language model (Section 3.2.1):

$$\begin{aligned} \delta_i = 1 \\ \forall i : w_i \, is \, a \, predicate \end{aligned} \tag{3.13}$$

$$\begin{aligned} \delta_i - \delta_j = 0 \\ \forall i, j : w_j \, is \, an \, argument \, of \, predicate \, w_i \end{aligned} \tag{3.14}$$

$$\begin{aligned} \delta_i = 1 \\ \forall i : w_i \in [ARG0 ... ARG5] \end{aligned} \tag{3.15}$$

[5] Available at http://cemantix.org/assert.html

$$\delta_i = 0 \tag{3.16}$$
$$\forall i : w_i \in optional\, arguments$$

Here, Constraint 3.13 guarantees that if a word is a predicate, it is included in the compression. Constraint 3.14 states that if a predicate is in compression, then its argument is also kept in the compression. In Constraint 3.15, we define that if a word denotes any of the possible semantic roles (i.e. *[ARG0...ARG5]* which are called *mandatory arguments*), it is included in the compression. On the other hand, we use Constraint 3.16 to restrict the inclusion of optional arguments[6] in the compression.

3.3 ILP for Query-focused Multi-document Summarization

The query-focused multi-document summarization inference problem can be formulated in terms of ILP. To represent the sentences included in the summary we define a set of indicator variables α_i that are set to 1 if i-th sentence is selected into the summary, and 0 otherwise. Let $Rel(i)$ be the relevance function that returns the relevance score of the i-th sentence. The score of a summary is the sum of the relevance scores of the sentences present in the summary. The inference task is solved by maximizing the overall score of a summary:

$$Maximize \sum_i Rel(i) * \alpha_i$$

such that $\forall i, j$:

[6] There are some additional arguments or semantic roles that can be tagged by ASSERT. They are called *optional arguments* and they start with the prefix *ARGM*. These are defined by the annotation guidelines set in (Palmer, Gildea, and Kingsbury, 2005).

$$\alpha_i \in \{0,1\} \tag{3.17}$$

$$Sim(i,j) * (\alpha_i + \alpha_j) \leq K \tag{3.18}$$

$$\sum_i Len(i) * \alpha_i \leq L \tag{3.19}$$

We incorporate three constraints into our formulation. The first constraint states that the variables are binary. The second constraint is the redundancy constraint that ensures that only one of the two similar sentences is chosen into the summary. $Sim(i,j)$ function returns a similarity score between the i-th and j-th sentences. Higher scores correspond to higher similarity between a pair of sentences. We assume a threshold K, that sets a tolerance limit to the acceptable similarity score between any two sentences. This value is empirically determined during experiments. The third constraint controls the length of the summary up to a maximum limit, L. $Len(i)$ denotes the length of the i-th sentence in words.

3.3.1 $Rel(i)$ Function

For each sentence, the $Rel(i)$ function returns a relevance score by combining a set of query-related and importance-oriented measures (as discussed in Section 2.6). The query-related measures calculate the similarity between each sentence and the given query while the importance-oriented measures denote the importance of a sentence in a given document (Chali, Hasan, and Imam, 2012a; Edmundson, 1969; Sekine and Nobata, 2001). For query-related measures, we consider n–gram overlap, longest common subsequence (LCS), weighted LCS, skip-bigram, exact word, synonym, hypernym/hyponym, gloss and basic elements (BE) overlap, and syntactic similarity. To measure the importance of a sentence, we consider its position, length, similarity with topic title, and presence of certain named entities and cue words. The mean of these scores denote the relevance of a sentence.

3.3.2 $Sim(i, j)$ Function

We employ four alternative redundancy constraints based on different sentence similarity functions (i.e. $Sim(i, j)$) using a) cosine similarity, b) syntactic similarity, c) semantic similarity, and d) extended string subsequence kernel (ESSK).

Cosine Similarity Measure (COS): The cosine similarity between the respective pair of sentences can be calculated by representing each sentence as a vector of term specific weights (Erkan and Radev, 2004). The term specific weights in the sentence vectors are products of local and global parameters. This is known as term frequency-inverse document frequency (tf-idf) model. The weight vector for a sentence s is $\vec{v_s} = [w_{1,s}, w_{2,s}, \ldots, w_{N,s}]^T$, where,

$$w_{t,s} = tf_t \times \log \frac{|S|}{|\{t \in s\}|}$$

Here, tf_t is the term frequency (*tf*) of the term t in a sentence s (a local parameter). $\log \frac{|S|}{|\{t \in s\}|}$ is the inverse document frequency (idf) (a global parameter). $|S|$ is the total number of sentences in the corpus, and $|\{t \in s\}|$ is the number of sentences containing the term t.

Syntactic Similarity Measure (SYN): Pasca and Harabagiu (2001) demonstrated that with the syntactic form one can see which words depend on other words. Syntactic features have been used successfully so far in *question answering* (Zhang and Lee, 2003; Moschitti et al., 2007; Moschitti and Basili, 2006). Inspired by the potential significance of using syntactic measures for finding similar texts, we get a strong motivation to use it as a redundancy measure in our optimization framework. The first step to calculate the syntactic similarity between two sentences is to parse the corresponding sentences into syntactic trees using the Charniak parser (Charniak, 1999). Once we build the syntactic trees, our next task is to measure the similarity between the trees using the tree kernel function (see details in Section 4.4.2). The tree kernel function gives the similarity score between a pair of sentences based on their syntactic structures.

Semantic Similarity Measure (SEM): Shallow semantic representations can prevent the sparseness of deep structural approaches and the weakness of cosine similarity based models (Moschitti et al., 2007). As an example, PropBank (PB) (Kingsbury and Palmer, 2002) made it possible to design accurate automatic Semantic Role Labeling (SRL) systems (Hacioglu et al., 2003). Therefore, we get the feeling that an application of SRL as a redundancy measure might suit well, since the textual similarity between a pair of sentences relies on a deep understanding of the semantics of both. So, applying semantic similarity measurement as a $Sim(i, j)$ function is another noticeable contribution of this research. To calculate the semantic similarity between two sentences, we first parse the corresponding sentences semantically using the Semantic Role Labeling (SRL) system, ASSERT. ASSERT is an automatic statistical semantic role tagger, that can annotate naturally occurring text with semantic arguments. We represent the annotated sentences using tree structures that are called semantic trees (ST). In the semantic tree, arguments are replaced with the most important word, often referred to as the semantic head. We look for noun, then verb, then adjective, then adverb to find the semantic head in the argument. If none of these is present, we take the first word of the argument as the semantic head. In the tree kernel method, common substructures cannot be composed by a node with only some of its children. Moschitti et al. (2007) solved this problem by designing the Shallow Semantic Tree Kernel (SSTK) which allows to match portions of a ST. The SSTK function yields the similarity score between a pair of sentences based on their semantic structures (see details in Section 4.4.2).

Extended String Subsequence Kernel (ESSK): ESSK is the simple extension of the Word Sequence Kernel (WSK) (Cancedda et al., 2003) and String Subsequence Kernel (SSK) (Lodhi et al., 2002). WSK receives two sequences of words as input and maps each of them into a high-dimensional vector space. WSK's value is just the inner product of the two vectors. But, WSK disregards synonyms, hyponyms, and hypernyms. On the other hand, SSK measures the similarity between two sequences of "alphabets". In ESSK, each "alphabet" in SSK is replaced by a disjunction of an "alphabet" and its alternative (Hirao et al., 2003). In ESSK, each word in a sentence is considered an "alphabet", and the alternative is its all possible senses. However, our ESSK implementation considers the alternative of each word as its disambiguated sense. We use a dictionary based Word Sense Disambiguation (WSD) System assuming one sense per discourse. We use WordNet (Fellbaum, 1998) to find the semantic relations (such as repetition, synonym, hypernym and hyponym, holonym and meronym, and gloss) for all the words in a text. We assign a weight to each semantic relation and used all of them. Our WSD technique is decomposed into two steps: (1) building a representation of all possible senses of the words and (2) disambiguating the words based on the highest score. To be specific, each candidate word from the context is expanded to all of its senses. A disambiguation graph is constructed as the intermediate representation where the nodes denote word instances with their WordNet senses and the weighted edges (connecting the senses of two different words) represent semantic relations. This graph is exploited to perform the WSD. We sum the weights of all edges leaving the nodes under their different senses. The sense with the highest score is considered to be the most probable sense. In case of a tie between two or more senses, we select the sense that comes first in WordNet, since WordNet orders the senses of a word by decreasing order of their frequency. We calculate the similarity score Sim(T_i, U_j) using ESSK where T_i and U_j are the two sentences. Formally, ESSK is defined as follows[7]:

$$K_{essk}(T,U) = \sum_{m=1}^{d} \sum_{t_i \in T} \sum_{u_j \in U} K_m(t_i, u_j)$$

$$K_m(t_i, u_j) = \begin{cases} val(t_i, u_j) & \text{if } m = 1 \\ K'_{m-1}(t_i, u_j) \cdot val(t_i, u_j) & \end{cases}$$

Here, $K'_m(t_i, u_j)$ is defined below. t_i and u_j are the nodes of T and U, respectively. The function $val(t,u)$ returns the number of attributes (i.e. words) common to the given nodes t and u.

$$K'_m(t_i, u_j) = \begin{cases} 0 & \text{if } j = 1 \\ \lambda K'_m(t_i, u_{j-1}) + K''_m(t_i, u_{j-1}) & \end{cases}$$

Here λ is the decay parameter for the number of skipped words. We choose $\lambda = 0.5$ for this research. $K''_m(t_i, u_j)$ is defined as:

$$K''_m(t_i, u_j) = \begin{cases} 0 & \text{if } i = 1 \\ \lambda K''_m(t_{i-1}, u_j) + K_m(t_{i-1}, u_j) & \end{cases}$$

Finally, the similarity measure is defined after normalization as below:

$$sim_{essk}(T,U) = \frac{K_{essk}(T,U)}{\sqrt{K_{essk}(T,T) K_{essk}(U,U)}}$$

[7] The formulae denotes a dynamic programming technique to compute the ESSK similarity score (Hirao et al., 2004) where d is the vector space dimension i.e. the number of all possible subsequences of up to length d.

3.4 Experiments

3.4.1 Task Description

We consider the query-focused multi-document summarization task defined in DUC-2007 (see Section 2.7.1 and Section 2.7.2 for details). We generate 250-word summaries for the topics of DUC-2007 (See sample summaries in Appendix C) using different combinations of sentence compression models (defined in Section 3.2) and alternative redundancy constraints (Section 3.3.2). As we intend to study if the order of performing compression and extraction can affect the overall performance of the query-focused multi-document summarization, we compose three different models depending on the order to perform sentence compression and extraction: **(1) ComFirst:** In this approach, document sentences are compressed first (using different models as described in Section 3.2) and then the most relevant compressions are selected to form the summaries (according to Section 3.3), **(2) SumFirst:** In this approach, we extract the most important sentences first from the source documents (according to Section 3.3) and then compress them (using different models as described in Section 3.2) to form the summaries, and **(3) Combined:** Here, we perform compression and extraction jointly by combining the objective functions of Section 3.2 and Section 3.3 according to Martins and Smith (2009). Then we optimize the combined objective function to select a small number of most important sentences (from the source documents) whose compressions should be used to form a summary.

3.4.2 Solving the ILPs

To solve the proposed ILP formulations, we use *lp_solve*[8], a widely used Integer Linear Programming solver that implements Branch-and-Bound algorithm. For summarization,

[8] *http://lpsolve.sourceforge.net/5.5/*

we solve an ILP for each topic in consideration and generate the corresponding query-focused summary. For a document cluster of average size (approximately 510 sentences), the solving process takes under 20 seconds on an Intel Pentium 4, 3.20 GHz desktop machine. For a larger document cluster (of size around 1000 sentences), it takes 90 – 120 seconds to solve the ILP. For a smaller document set, the ILP is solved in a few seconds. For compression, we solve an ILP for each sentence in consideration. The solving process takes less than a second per sentence on average for all the compression models. For the joint extraction and compression model, we solve an ILP for each topic in consideration. The solving process is generally slower than solving the ILPs for only sentence extraction or compression as it takes 300 – 1200 seconds depending on the document cluster size.

3.4.3 Evaluation Results and Discussion

Automatic Evaluation: The multiple "reference summaries" given by DUC-2007 are used in the evaluation of our summary content. We carried out the automatic evaluation of our summaries using the ROUGE (Lin, 2004) toolkit. Among different scores reported by ROUGE, unigram-based ROUGE score (ROUGE-1) could agree with human judgment the most (Lin, 2003). We report the widely adopted important ROUGE metrics in the results: ROUGE-1 (unigram), and ROUGE-2 (bigram). The comparison between the systems in terms of their F-scores is given in Table 3.1. We also include the results of the official baseline systems, the best system (Pingali, K., and Varma, 2007), and the average ROUGE scores of all the participating systems of DUC-2007. Baseline-1 returns all the leading sentences (up to 250 words) of the most recent document whereas baseline-2's main idea is to ignore the topic narrative while generating summaries using an HMM model[9].

The columns in Table 3.1 denote the use of alternative redundancy constraints in the

[9] http://duc.nist.gov/pubs/2004papers/ida.conroy.ps

	COS		SYN		SEM		ESSK		No Red.		Comp.	
Model	R1	R2	R1	R2	R1	R2	R1	R2	R1	R2	R1	R2
ComFirst												
bi	0.359	0.074	0.369	0.078	0.371	0.077	0.368	0.072	0.355	0.060		
topicS	0.372	0.080	0.366	0.081	0.378	0.079	0.373	0.076	0.360	0.071		
bi+sem	0.385	0.093	0.376	0.085	0.389	0.092	0.384	0.088	0.367	0.075		
SumFirst												
bi	0.368	0.076	0.365	0.079	0.388	0.096	0.370	0.088	0.362	0.071		
topicS	0.374	0.083	0.371	0.084	0.392	0.101	0.378	0.091	0.365	0.074		
bi+sem	0.388	0.096	0.382	0.091	0.405	0.113	0.391	0.101	0.374	0.083		
Combined												
bi	0.384	0.102	0.371	0.087	0.385	0.091	0.371	0.081	0.356	0.082		
topicS	0.389	0.105	0.374	0.089	0.398	0.103	0.368	0.084	0.364	0.078		
bi+sem	0.412	0.115	0.390	0.092	0.424	0.119	0.395	0.094	0.372	0.086		
No compr.	0.400	0.108	0.399	0.109	0.412	0.111	0.396	0.105	0.381	0.091		
Baseline1											0.334	0.060
Baseline2											0.400	0.093
AverageDUC											0.400	0.095
Best System											0.438	0.122

Table 3.1: Automatic Evaluation Results: Average ROUGE F-scores

optimization framework whereas the rows stand for the use of different compression models[10]. From these results, we can clearly see the impact of using different sentence compression models on the overall summarization performance. In the **ComFirst** approach, we can see that the bigram model with semantic constraints outperforms all the other alternative models by a clear margin. We can also see the impact of different redundancy constraints on the overall performance. We observe that the use of semantic measure as the redundancy constraint yields the best performance. On the other hand, we see a clear improvement in almost all the scores when we follow the **SumFirst** approach. This phenomenon suggests that compressing the document sentences at the beginning often tend to reduce relevant information in the sentences for which we get lesser similarity matching when we calculate the relevance scores according to Section 3.3.1. In the **Combined** approach, we achieve better summarization performance than the other two approaches which denotes that the overall summary quality can be improved if a global optimization frame-

[10] The last few rows and columns are used to accommodate the scores of the baselines and the state-of-the-art systems.

work is utilized having a joint compression and extraction model. Again, we see that the bigram language model with semantic constraints along with the semantic redundancy constraint (used in the summarization model) yields the best performance. We also report the results of a *"No compression"* and a *"No redundancy"* baseline. Comparisons with these baselines also suggest that our bigram compression model with semantic constraints can improve the overall summarization performance if a **Combined** optimization framework is used in presence of **COS** or **SEM** redundancy constraints. These results also demonstrate that the absence of a redundancy constraint in the ILP framework for summarization really hurts the overall quality of the summaries. We also compare the scores of our model with the state-of-the-art systems of DUC-2007. From the results, we see that our semantically motivated models can mostly outperform the **DUC baselines** and the **AverageDUC** scores to show a clear improvement in the overall summarization performance while achieving a comparable performance with respect to the DUC-2007 best system. The differences between the models are computed to be statistically significant at $p < 0.05$ (using Student's t-test) except for the differences between **topicSig+SYN** and **bigram+SYN**, and **topicSig+ESSK** and **bigram+ESSK** in all the three approaches, between **topicSig+COS** and **bigram+COS** in the **Combined** approach, and between **"bigram+sem"+SEM** and **DUC Best System** in the **Combined** approach.

Manual Evaluation: One of the important demerits of using sentence compression models is that they can degrade the linguistic quality of a summary by showing poor compression performance. ROUGE is not reliable to some researchers as there might be some linguistically bad summaries that get state-of-the-art ROUGE scores (Sjöbergh, 2007). So, we conduct an extensive manual evaluation in order to analyze the effectiveness of our approaches. Two self reported native English-speaking university graduate students judge the summaries for linguistic quality and overall responsiveness according to the DUC-2007 evaluation guidelines[11].

	COS		SYN		SEM		ESSK		No Redundancy		Comparison	
Models	LQ	Res.	LQ	Res.	LQ	Res.	LQ	Res.	LQ	Res.	LQ	Res.
ComFirst												
bigram	2.10	2.12	2.28	2.20	2.44	2.21	2.32	2.25	1.94	2.10		
topicSig	2.14	2.30	2.45	2.27	2.48	2.78	2.39	2.46	2.08	2.26		
bigram+sem	2.42	2.56	2.55	2.61	2.74	3.05	2.54	2.80	2.25	2.58		
SumFirst												
bigram	2.43	2.44	2.54	2.50	2.60	2.45	2.25	2.34	2.16	2.56		
topicSig	2.48	2.56	2.65	2.69	2.72	2.66	2.48	2.55	2.27	2.68		
bigram+sem	2.61	2.76	2.88	2.78	3.20	3.56	2.75	2.93	2.42	2.62		
Combined												
bigram	2.54	2.62	2.52	2.31	2.76	2.55	2.36	2.50	1.98	2.20		
topicSig	2.62	2.75	2.68	2.38	2.80	2.62	2.45	2.64	2.14	2.31		
bigram+sem	2.85	3.08	2.91	2.93	3.18	3.61	2.77	2.88	2.32	2.42		
No compression	3.30	3.38	3.42	3.15	3.64	3.50	3.38	3.21	2.28	2.15		
Baseline1											4.24	1.86
Baseline2											4.48	2.71
Best System											4.11	3.40

Table 3.2: Average linguistic quality (LQ) and responsiveness scores (Res.)

The given linguistic quality score is an integer between 1 (very poor) and 5 (very good) and is guided by consideration of the following factors: 1. Grammaticality, 2. Non-redundancy, 3. Referential clarity, 4. Focus, and 5. Structure and Coherence. The responsiveness score is also an integer between 1 (very poor) and 5 (very good) and is based on the amount of information in the summary that helps to satisfy the information need. The carried out user evaluation was subjective in nature specially while judging referential clarity, focus, coherence and overall responsiveness of the summaries. The inter-annotator agreement of Cohen's $\kappa = 0.43$ (Cohen, 1960) was computed that denotes a moderate degree of agreement (Landis and Koch, 1977) between the raters. Table 3.2 presents the average linguistic quality and overall responsive scores of all the systems. From these results, we can see that the use of different sentence compression models has a negative impact on the overall linguistic quality of the summaries. The reason behind this is that our bigram com-

[11] http://www-nlpir.nist.gov/projects/duc/duc2007/quality-questions.txt

pression models were less aware of the underlying context in a sentence and hence, some word deletions resulted a loss in focus and coherence of the overall summaries. However, we observe that the semantically motivated models are showing an improved summarization performance; also, their overall responsiveness scores are comparable to the state-of-the-art systems. This suggests that the manual evaluation results are corresponding well to the automatic evaluation results. Considering the work of Gillick and Favre (2009) for a relative comparison, we find that both our automatic and manual evaluation results are corresponding fairly well to their results obtained on the TAC[12]-2008 data. Their ILP model with additional constraints to include sentence compression achieved an improvement in ROUGE-2 score over the "no compression" alternative while having reductions in manual evaluation scores. We perform a statistical significance test on our manual evaluation results at $p < 0.05$ using Student's t-test. The differences between the models are statistically significant except for the differences between **topicSig+COS** and **bigram+COS**, and **topicSig+SYN** and **bigram+SYN** in all the three approaches. The manual evaluation results also demonstrate that the use of different redundancy constraints certainly affects the overall performance of the proposed optimization framework for summarization[13]. From these experiments we can conclude that the semantic similarity measure can be used effectively as the $Sim(i,j)$ function to improve the performance of the traditional cosine similarity based approaches.

3.5 Conclusion

In this chapter, we have discussed our ILP formulation to the complex question answering task. We have analyzed the effectiveness of using different ILP-based sentence compres-

[12] Text Analysis Conference, http://www.nist.gov/tac/

[13] The selection of sentences in the optimal summaries varied due to different redundancy measures, hence, the linguistic quality scores also varied to reflect the differences in coherence, redundancy etc.

sion models for the query-focused multi-document summarization task. Extensive experiments on the DUC benchmark datasets have demonstrated that the combined optimization framework with semantically motivated constraints can show state-of-the-art performance in comparison to the pipeline-based approaches having lexically and syntactically motivated constraints. We have also discovered that optimizing the summarization task before compression often yields better performance than that of optimizing the compression task before summarization.

Chapter 4

Complex Question Answering using Graph-based Random Walk Model

4.1 Introduction

The task of answering complex questions requires inferencing and synthesizing information from multiple documents. What this means is that it is necessary to search for the most important pieces of information from different parts of the given document collection and put them together so that the gathered information as a whole can satisfy the information need asked in the complex question. This process can be regarded as a kind of topic-oriented, informative multi-document summarization.

In case of generic summarization, the main goal is to produce a summary that can contain the most important topics of the document collection. As the most important topics are likely to be discussed in a document collection in a frequent manner, the general intuition behind having a good summary lies in that the summary should be formed using the most frequently discussed information in the document collection. The key aspect of this process involves computing the similarities among the sentences present in the document collection (Erkan, 2007). Graph-based methods such as LexRank (Erkan and Radev, 2004) and TextRank (Mihalcea and Tarau, 2004) are the most popular methods that can formalize the concept of sentence-to-sentence relationship effectively and hence, are applied successfully to generic, multi-document summarization. Note that in the graph-based representation of a document, each node in the graph stands for a sentence in the document whereas an edge between two nodes denotes the pairwise similarity relation between two sentences. In the LexRank method, a similarity graph is constructed according to the lexical similarities between the sentences. In this graph, a sentence is considered as important if it is mostly similar to other sentences of the document. This can be determined by examining the degree of a node in the graph, termed as the *lexical centrality value* or *LexRank*. The concept of direct relationship between two sentences in the graph can be further enhanced with the intuition of performing a random walk in the graph, which can lead to identifying the similarities among the sentences that are two or more edges apart from each other (Erkan, 2007). This idea is motivated from the centrality method used by the Google web search engine on the World Wide Web graph, which is known as PageRank (Page et al., 1999). A modified version of the LexRank to handle query-focused summarization is the topic-sensitive LexRank, which is proposed in Otterbacher, Erkan, and Radev (2005). This version is deemed necessary for addressing the complex question answering task where the centrality on a similarity graph is computed with regard to the given complex question.

In the traditional graph-based random walk methods discussed above, cosine-based similarity functions are used in order to calculate similarities among the sentences, where a sentence is mapped to a vector in which each element represents the occurrence frequency (TF*IDF) of a word. However, the major limitation of the TF*IDF approach is that it only retains the frequency of the words and does not take into account the sequence, syntactic and semantic structure. Thus, it cannot distinguish between “The hero killed the villain” and “The villain killed the hero”. For the task like *answering complex questions* that requires the use of more complex syntactic and semantics, the approaches with only TF*IDF are often inadequate to perform fine-level textual analysis (Chali and Joty, 2008).

We extensively study the impact of syntactic and semantic information in measuring similarity between the sentences in the random walk framework for answering complex questions. We apply the tree kernel functions and Extended String Subsequence Kernel (ESSK) to include syntactic and semantic information. We run our experiments on the DUC-2007 data and based on this we argue that for the complex question answering task, similarity measures based on syntactic and semantic information perform better and can be used to characterize the relation between a question and a sentence (answer) in a more effective way than the traditional TF*IDF based similarity measures. In another experiment, we successfully exploit a deeper semantic analysis of the source documents to select important concepts by using a predefined list of important aspects that act as a guide for selecting the most relevant sentences into the summaries (Chali, Hasan, and Imam, 2011b). Next sections give a detailed description of the related works in the domain, the graph-based random walk model, and our approaches.

4.2 Related Work

There are approaches in "Recognizing Textual Entailment", "Sentence Alignment", and "Question Answering" that use syntactic and/or semantic information in order to measure the similarity between two textual units. Indeed, this motivated us to include syntactic and semantic features to get the structural similarity between sentences. In MacCartney et al. (2006), they use typed dependency graphs (same as dependency trees) to represent the text and the hypothesis. They try to find a good partial alignment between the typed dependency graphs representing the hypothesis (contains n nodes) and the text (graph contains m nodes) in a search space of $O((m+1)n)$. They use an incremental beam search combined with a node ordering heuristic to do approximate global search in the space of possible alignments. A locally decomposable scoring function was chosen such that the score of an alignment is the sum of the local node and edge alignment scores. The scoring measure is designed to favor alignments which align semantically similar subgraphs, irrespective of polarity. For this reason, nodes receive high alignment scores when the words they represent are semantically similar. Synonyms and antonyms receive the highest score and unrelated words receive the lowest. Alignment scores also incorporate local edge scores which are based on the shape of the paths between nodes in the text graph which correspond to adjacent nodes in the hypothesis graph. In the final step they make a decision about whether or not the hypothesis is entailed by the text conditioned on the typed dependency graphs as well as the best alignment between them. To make this decision they use a supervised statistical logistic regression classifier (with a feature space of 28 features) with a Gaussian prior parameter for regularization.

The importance of syntactic and semantic features in finding textual similarity is described by Zhang and Lee (2003), Moschitti et al. (2007) and Moschitti and Basili (2006). An effective way to integrate syntactic and semantic structures in machine learning algorithms is the use of *tree kernel* functions (Collins and Duffy, 2001) which has been successfully applied to question classification (Zhang and Lee, 2003), (Moschitti and Basili, 2006). To the best of our knowledge, no study has used tree kernel functions to encode syntactic/semantic information for more complex tasks such as computing the relatedness between the query sentences and the document sentences. Another good way to encode some shallow syntactic information is the use of Basic Elements (BE) (Hovy et al., 2006) which uses dependency relations. Moreover, the study of shallow semantic information such as predicate argument structures annotated in the PropBank (PB) project (Kingsbury and Palmer, 2002) is a promising research direction.

In Hirao et al. (2004), they represent the sentences using Dependency Tree Path (DTP) to incorporate syntactic information. They apply String Subsequence Kernel (SSK) to measure the similarity between the DTPs of two sentences. They also introduce Extended String Subsequence Kernel (ESSK) to incorporate semantics in DTPs. In Kouylekov and Magnini (2005), they use the tree edit distance algorithms on the dependency trees of the text and the hypothesis to recognize the textual entailment. According to this approach, a text T entails a hypothesis H if there exists a sequence of transformations (i.e., deletion, insertion and substitution) applied to T such that we can obtain H with an overall cost below a certain threshold. In Punyakanok, Roth, and Yih (2004), they represent the question and the sentence containing answer with their dependency trees. They add semantic information (i.e., named entity, synonyms and other related words) in the dependency trees. They apply the approximate tree matching in order to decide how similar any given pair of trees are. They also use the edit distance as the matching criteria in the approximate tree matching. All these methods show the improvement over the Bag-Of-Words (BOW) scoring methods.

A substantial body of work on summarization using Information Extraction (IE) templates have been accomplished over the years in the Message Understanding Conferences (MUC[1]), DUC-2004 biography-related summarization task[2], as well as TREC[3]. In White et al. (2001), they discuss the use of MUC templates for summarization. In Zhou, Ticrea, and Hovy (2005), the authors define several biographical facts that should be included into a good biography. Filatova, Hatzivassiloglou, and McKeown (2006) automatically create templates for several domains and use summarization-like task to evaluate the quality of the created templates. All the templates and facts are used in these researches to generate more focused summaries. Nastase (2008) expands the query by using encyclopedic knowledge in Wikipedia and use the topic expanded words with activated nodes in the graph to produce an extractive summary. New features such as topic signature are used in the NeATS system by Lin et al. (2003) to select important content from a set of documents about some topic to present them in coherent order. An enhanced discourse-based summarization framework by rhetorical parsing tuning is proposed by Marcu (1998a). In our approach, we exploit the topic signature model and the rhetorical structure theory (Mann and Thompson, 1988) to assign weights to the document sentences. In Harabagiu, Lacatusu, and Hickl (2006), they introduced a paradigm for producing summary-length answers to complex questions. Their method operates on a Markov chain, by following a random walk with mixture model on a

[1] http://www-nlpir.nist.gov/related_projects/muc/proceedings/ie_task.html

[2] http://duc.nist.gov/duc2004/

[3] http://trec.nist.gov/

bipartite graph of relations established between concepts related to the topic of a complex question and subquestions derived from topic-relevant passages. Motivated by all these related researches, we propose to augment a predefined list of important aspects (that provides a better coverage of the topic on the entire document collection) into a random walk framework that no other study has used before to the best of our knowledge.

4.3 Graph-based Methods for Summarization

In Erkan and Radev (2004), the concept of graph-based centrality is used to rank a set of sentences, in producing generic multi-document summaries. A similarity graph is produced for the sentences in the document collection. In the graph, each node represents a sentence. The edges between nodes measure the cosine similarity between the respective pair of sentences where each sentence is represented as a vector of term specific weights. The term specific weights in the sentence vectors are products of local and global parameters. The model is known as term frequency-inverse document frequency (tf-idf) model. The weight vector for a sentence s is $\vec{v_s} = [w_{1,s}, w_{2,s}, \ldots, w_{N,s}]^T$, where,

$$w_{t,s} = tf_t \times \log \frac{|S|}{|\{t \in s\}|}$$

and

- tf_t is term frequency (*tf*) of term *t* in sentence *s* (a local parameter)
- $\log \frac{|S|}{|\{t \in s\}|}$ is inverse document frequency (idf) (a global parameter). $|S|$ is the total number of sentences in the corpus; $|\{t \in s\}|$ is the number of sentences containing the term *t*.

The degree of a given node is an indication of how much important the sentence is. Figure 4.1 shows an example of a similarity graph for 4 sentences.

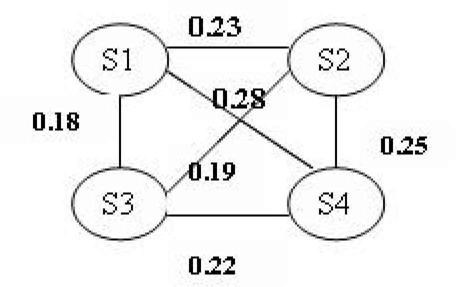


Figure 4.1: LexRank similarity

Once the similarity graph is constructed, the sentences are then ranked according to their eigenvector centrality. The LexRank performed well in the context of generic summarization.

To apply LexRank to query-focused context, a topic-sensitive version of LexRank is proposed in Otterbacher, Erkan, and Radev (2005). The score of a sentence is determined by a mixture model of the relevance of the sentence to the query and the similarity of the sentence to other high-scoring sentences.

4.3.1 Relevance to the question

Following Otterbacher, Erkan, and Radev (2005), we first stem out all the sentences in the collection and compute the word IDFs using the following formula:

$$idf_w = log\left(\frac{N+1}{0.5+sf_w}\right)$$

where, N is the total number of sentences in the document cluster, and sf_w is the number of sentences that the word w appears in.

We also stem out the questions and remove the stop words. The relevance of a sentence

s to the question q is computed by:

$$rel(s|q) = \sum_{w \in q} log\left(tf_{w,s}+1\right) \times log\left(tf_{w,q}+1\right) \times idf_w$$

where, $tf_{w,s}$ *and* $tf_{w,q}$ are the number of times w appears in s and q, respectively.

4.3.2 Mixture Model

In the previous section, we measured the relevance of a sentence to the question but a sentence that is similar to the high scoring sentences in the cluster should also have a high score. For instance, if a sentence that gets high score based on the question relevance model is likely to contain an answer to the question, then a related sentence, which may not be similar to the question itself, is also likely to contain an answer. This idea is captured by the following mixture model (Otterbacher, Erkan, and Radev, 2005):

$$p(s|q) \quad = \quad d\frac{rel(s|q)}{\sum_{z \in C} rel(z|q)} + (1-d)\sum_{v \in C}\frac{sim(s,v)}{\sum_{z \in C} sim(z,v)}p(v|q) \tag{4.1}$$

where, $p(s|q)$ is the score of a sentence s given a question q, is determined as the sum of its relevance to the question and the similarity to the other sentences in the collection. C is the set of all sentences in the collection. The value of the parameter d which we call “bias”, is a trade-off between two terms in the equation and is set empirically. For higher values of d, we prefer the relevance to the question to similarity to other sentences. The denominators in both terms are for normalization. We measure the cosine similarity weighted by word IDFs as the similarity between two sentences in a cluster:

$$sim(x,y) = \frac{\sum_{w \in x,y} tf_{w,x} tf_{w,y} (idf_w)^2}{\sqrt{\sum_{x_i \in x} (tf_{x_i,x} idf_{x_i})^2} \sqrt{\sum_{y_i \in y} (tf_{y_i,y} idf_{y_i})^2}}$$

Following Otterbacher, Erkan, and Radev (2005), Equation 4.1 can be written in matrix notation as follows:

$$\mathbf{p} = [d\mathbf{A} + (1-d)\mathbf{B}]^T \mathbf{p} \tag{4.2}$$

A is the square matrix such that for a given index *i*, all the elements in the i^{th} column are proportional to $rel(i|q)$. **B** is also a square matrix such that each entry **B***(i,j)* is proportional to *sim(i,j)*. Both matrices are normalized so that row sums add up to 1. Note that as a result of this normalization, all rows of the resulting square matrix $Q = [dA + (1-d)B]$ also add up to 1. Such a matrix is called *stochastic* and defines a Markov chain. If we view each sentence as a state in a Markov chain, then Q(i,j) specifies the transition probability from state *i* to state *j* in the corresponding Markov chain. The vector **p** we are looking for in equation 4.2 is the stationary distribution of the Markov chain. An intuitive interpretation of the stationary distribution can be understood by the concept of a random walk on the graph representation of the Markov chain.

With probability *d*, a transition is made from the current node to the nodes that are similar to the query. With probability *(1-d)*, a transition is made to the nodes that are lexically similar to the current node. Every transition is weighted according to the similarity distributions. Each element of the vector **p** gives the asymptotic probability of ending up at the corresponding state in the long run regardless of the starting state. The stationary distribution of a markov chain can be computed by a simple iterative algorithm, called power method (Erkan and Radev, 2004). The power method starts with a uniform distribution. At each iteration, the eigenvector is updated by multiplying with the transpose of the stochastic matrix. Since the Markov chain is irreducible and aperiodic, the algorithm is guaranteed to terminate.

4.4 Improvement over Cosine-based Methods

We claim that for a complex task like answering complex questions where the relatedness between the query sentences and the document sentences is an important factor, the graph-based method of ranking sentences would perform better if we could encode the syntactic and semantic information instead of just the BOW (i.e., TF*IDF) information in calculating the similarity between sentences. Thus, our mixture model for answering complex questions is:

$$p(s|q) \quad = \quad d \times KERSIM(s,q) + (1-d) \times \sum_{v \in C} KERSIM(s,v) \times p(v|q) \tag{4.3}$$

where, *KERSIM(s,q)* is the normalized syntactic (and/or semantic) similarity between the *query (q)* and *the document sentence (s)* and C is the set of all sentences in the collection. In cases where the query is composed of two or more sentences, we compute the similarity between the document sentence (s) and each of the query-sentences (q_i) then we take the average of the scores. In the next subsections, we describe how we can encode syntactic and semantic structures according to Moschitti et al. (2007) in calculating the similarity between sentences, and present our experiments and results.

4.4.1 Encoding Syntactic and Shallow Semantic Structures

Encoding syntactic structure is easier and straight forward. Given a sentence (or query), we first parse it into a syntactic tree using a parser like (Charniak, 1999) and then we calculate the similarity between the two trees using the *tree kernel* (discussed in Section 4.4.2). For this purpose, we re-implement the tree kernel model by following Moschitti et al. (2007).

Though introducing syntactic information gives an improvement on BOW by the use of syntactic parses, but these, too are not adequate when dealing with complex questions whose answers are expressed by long and articulated sentences or even paragraphs. Shallow semantic representations, bearing a more compact information, could prevent the sparseness of deep structural approaches and the weakness of BOW models (Moschitti et al., 2007).

Initiatives such as PropBank (PB) (Kingsbury and Palmer, 2002) have made possible the design of accurate automatic Semantic Role Labeling (SRL) systems like ASSERT (Hacioglu et al., 2003). Attempting an application of SRL to QA hence seems natural, as pinpointing the answer to a question relies on a deep understanding of the semantics of both. For example, consider the PB annotation:

```
[ARG0 all][TARGET use][ARG1 the french franc][ARG2 as their currency]
```

Such annotation can be used to design a shallow semantic representation that can be matched against other semantically similar sentences, e.g.

```
[ARG0 the Vatican][TARGET use][ARG1 the Italian lira][ARG2 as their currency]
```

In order to calculate the semantic similarity between the sentences, we first represent the annotated sentence (or query) using the tree structures like Figure 4.2 which we call Semantic Tree (ST). In the semantic tree, arguments are replaced with the most important word-often referred to as the semantic head. We look for noun first, then verb, then adjective, then adverb to find the semantic head in the argument. If none of these is present, we take the first word of the argument as the semantic head. This reduces the data sparseness with respect to a typical BOW representation.

However, sentences rarely contain a single predicate: it happens more generally that propositions contain one or more subordinate clauses. For instance, let us consider a slight modification of the second sentence: “the Vatican, located wholly within Italy uses the Italian lira as their currency.” Here, the main predicate is “uses” and the subordinate predicate is “located”. The SRL system outputs the following two annotations:

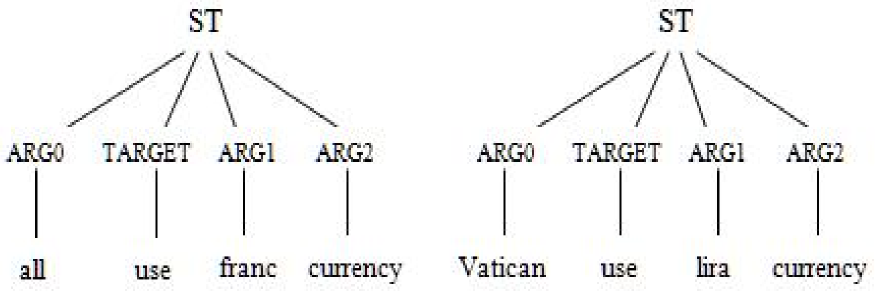


Figure 4.2: Example of semantic trees

```
(1) [ARG0 the Vatican located wholly within Italy][TARGET uses][ARG1 the Italian lira]
[ARG2 as their currency]
(2) [ARG0 the Vatican][TARGET located] [ARGM-LOC wholly][ARGM-LOC within Italy] uses
the Italian lira as their currency
```

giving the STs in Figure 4.3. As we can see in Figure 4.3(A), when an argument node corresponds to an entire subordinate clause, we label its leaf with ST, e.g. the leaf of ARG0. Such ST node is actually the root of the subordinate clause in Figure 4.3(B). If taken separately, such STs do not express the whole meaning of the sentence, hence it is more accurate to define a single structure encoding the dependency between the two predicates as in Figure 4.3(C). We refer to this kind of nested STs as STNs (Semantic Tree Networks).

4.4.2 Syntactic and Semantic Kernels for Text

Tree Kernels

Once we build the trees (syntactic or semantic), our next task is to measure the similarity between the trees. For this, every tree T is represented by an m dimensional vector $v(T) = (v_1(T), v_2(T), \cdots v_m(T))$, where the i-th element $v_i(T)$ is the number of occurrences of the

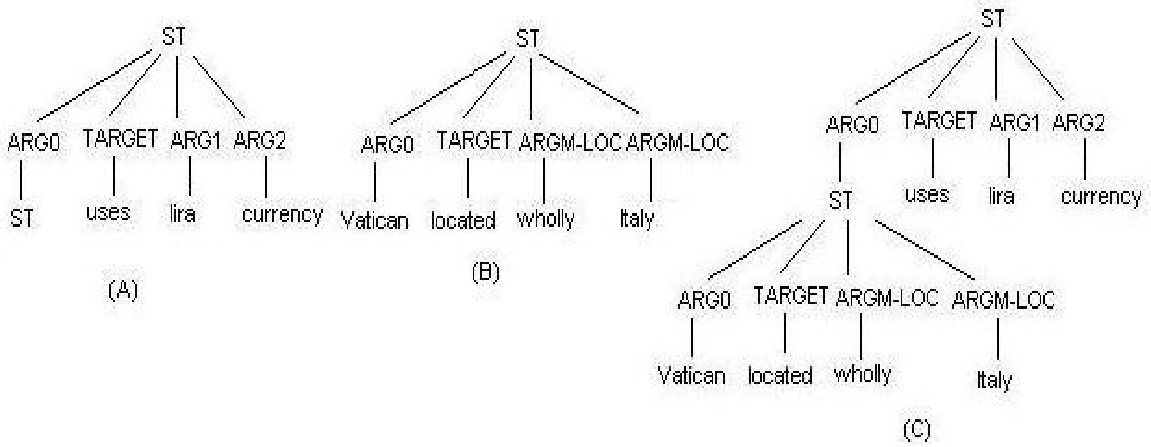


Figure 4.3: Two STs composing a STN

i-th tree fragment in tree T. The tree fragments of a tree are all of its sub-trees which include at least one production with the restriction that no production rules can be broken into incomplete parts.

Figure 4.4 shows an example tree and a portion of its subtrees.

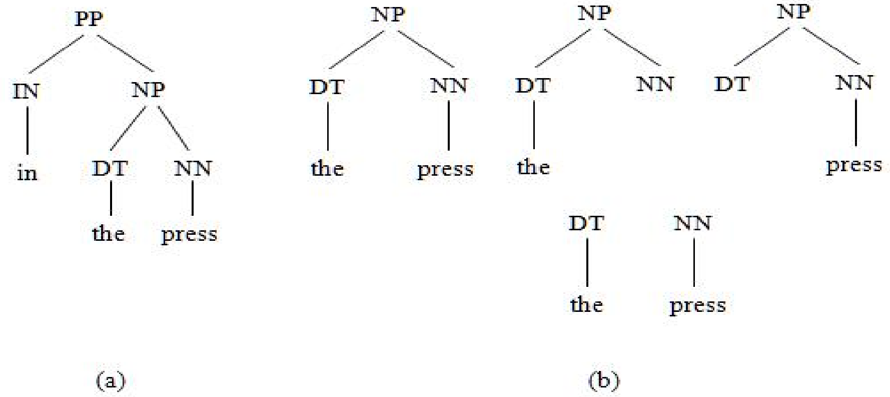


Figure 4.4: (a) An example tree (b) The sub-trees of the NP covering "the press".

Implicitly we enumerate all the possible tree fragments $1, 2, \cdots, m$. These fragments are the axis of this m-dimensional space. Note that this could be done only implicitly, since the number m is extremely large. Because of this, (Collins and Duffy, 2001) defines the tree kernel algorithm whose computational complexity does not depend on m. The tree kernel of two trees T_1 and T_2 is actually the inner product of $v(T_1)$ and $v(T_2)$:

$$TK(T_1,T_2) = v(T_1).v(T_2) \tag{4.4}$$

We define the indicator function $I_i(n)$ to be 1 if the sub-tree i is seen rooted at node n and 0 otherwise. It follows:

$$\begin{aligned} v_i(T_1) &= \sum_{n_1 \in N_1} I_i(n_1) \\ v_i(T_2) &= \sum_{n_2 \in N_2} I_i(n_2) \end{aligned}$$

where, N_1 and N_2 are the set of nodes in T_1 and T_2 respectively. So, we can derive:

$$\begin{aligned} TK(T_1,T_2) &= v(T_1).v(T_2) \\ &= \sum_i v_i(T_1)v_i(T_2) \\ &= \sum_{n_1 \in N_1} \sum_{n_2 \in N_2} \sum_i I_i(n_1)I_i(n_2) \\ &= \sum_{n_1 \in N_1} \sum_{n_2 \in N_2} C(n_1,n_2) \qquad (4.5) \end{aligned}$$

where, we define $C(n_1,n_2) = \sum_i I_i(n_1)I_i(n_2)$. Next, we note that $C(n_1,n_2)$ can be computed in polynomial time, due to the following recursive definition:

1. If the productions at n_1 and n_2 are different then $C(n_1,n_2) = 0$

2. If the productions at n_1 and n_2 are the same, and n_1 and n_2 are pre-terminals, then $C(n_1,n_2) = 1$

3. Else if the productions at n_1 and n_2 are not pre-terminals,

$$C(n_1,n_2) = \prod_{j=1}^{nc(n_1)} (1 + C(ch(n_1,j),ch(n_2,j))) \tag{4.6}$$

where, $nc(n_1)$ is the number of children of n_1 in the tree; because the productions at n_1 and n_2 are the same, we have $nc(n_1) = nc(n_2)$. The i-th child-node of n_1 is $ch(n_1,i)$.

As noted before, when the given complex question is composed of two or more sentences, we compute the similarity between the document sentence (s) and each of the query-sentences (q_i), and then we take the average of the scores as the syntactic feature value:

$$Syntactic\ similarity\ value = \frac{\sum_{i=1}^{n} TK(q_i, s)}{n}$$

where n is the number of sentences in the query q and s is the sentence under consideration. *TK* is the similarity value (tree kernel) between the sentence s and the query sentence q based on their syntactic structures. For example, for the following sentence s and query q, we get the score:

Query (q): Describe steps taken and worldwide reaction prior to introduction of the Euro on January 1, 1999. Include predictions and expectations reported in the press.

Sentence (s): Europe's new currency, the euro, will rival the U.S. dollar as an international currency over the long term, Der Spiegel magazine reported Sunday.

Scores: 90, 41

Average Score: 65.5

Note that, the tree kernel (TK) function computes the number of common subtrees between two trees. Such subtrees are subject to the constraint that their nodes are taken with all or none of the children they have in the original tree. Though, this definition of subtrees makes the TK function appropriate for syntactic trees but at the same time makes it not well suited for the semantic trees (ST) defined in Section 4.4.1. For instance, although the two STs of Figure 4.2 share most of the subtrees rooted in the *ST* node, the kernel defined above computes only one match (ST ARG0 TARGET ARG1 ARG2) which is not useful. The critical aspect of steps (1), (2) and (3) of the TK function is that the productions of two evaluated nodes have to be identical to allow the match of further descendants. This means that common substructures cannot be composed by a node with only some of its children as an effective ST representation would require. Moschitti et al. (2007) solve this problem by designing the Shallow Semantic Tree Kernel (SSTK) which allows to match portions of a ST.

Shallow Semantic Tree Kernel (SSTK)

The SSTK is based on two ideas: first, it changes the ST, as shown in Figure 4.5 by adding *SLOT* nodes. These accommodate argument labels in a specific order i.e., it provides a fixed number of slots, possibly filled with *null* arguments, that encode all possible predicate arguments. Leaf nodes are filled with the wildcard character * but they may alternatively accommodate additional information. The slot nodes are used in such a way that the adopted TK function can generate fragments containing one or more children like for example those shown in frames (b) and (c) of Figure 4.5. As previously pointed out, if the arguments were directly attached to the root node, the kernel function would only generate the structure with all children (or the structure with no children, i.e., empty).

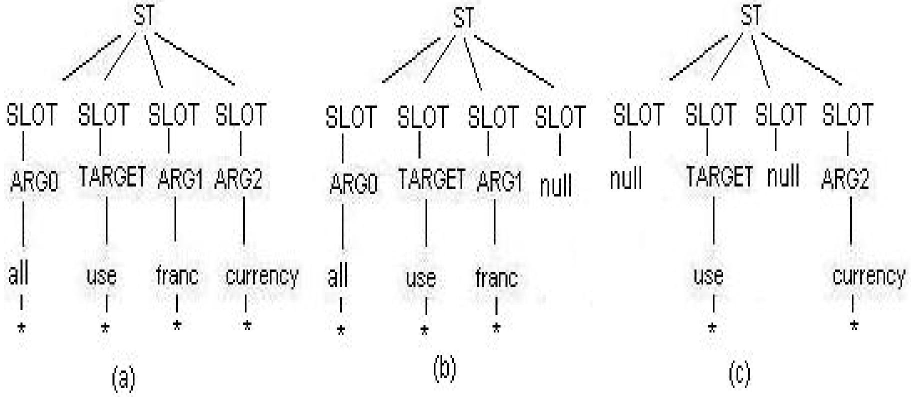


Figure 4.5: Semantic tree with some of its fragments

Second, as the original tree kernel would generate many matches with slots filled with the null label, we have set a new step 0 in the TK calculation:

(0) if n_1 (or n_2) is a pre-terminal node and its child label is *null*, $C(n_1,n_2) = 0$;

and subtract one unit to $C(n_1,n_2)$, in step 3:

$$(3)\ C(n_1,n_2) = \prod_{j=1}^{nc(n_1)} (1 + C(ch(n_1,j), ch(n_2,j))) - 1$$

The above changes generate a new C which, when substituted (in place of original C) in Eq. 4.5, gives the new SSTK. We re-implement the shallow semantic tree kernel model according to Moschitti et al. (2007). For the following example sentence s and query q we get the semantic score:

Query (q): Describe steps taken and worldwide reaction prior to introduction of the Euro on January 1, 1999. Include predictions and expectations reported in the press.

Sentence (s): The Frankfurt-based body said in its annual report released today that it has decided on two themes for the new currency history of European civilization and abstract or concrete paintings.

Scores: 6, 12

Average Score: 9

We can see that the above discussed kernels are designed by either choosing an explicit mapping function and incorporating it into an inner product or by directly defining the kernel function while making sure that it complies with the requirement of being a positive semi-definite function. A kernel $k : \chi \times \chi \to R$ is a symmetric and a positive semi-definite function, which simply computes an inner product in a reproducing kernel Hilbert space (Scholkopf and Smola, 2002). The constructed kernel matrices for the proposed kernel functions can be proved positive and semi-definite along with the kernel function's symmetricity using the theorem in (Shin and Kuboyama, 2008).

4.4.3 Extended String Subsequence Kernel (ESSK)

The similarity score between two sentences T_i and U_j, denoted by Sim(T_i, U_j), is calculated using ESSK according to the procedure discussed in Section 3.3.2.

4.4.4 Redundancy Checking and Summary Generation

Once the sentences are scored by the mixture model, the easiest way to create summaries is just to output the top most N sentences until the required summary length is reached. In that case, other factors such as redundancy and coherence would have been ignored.

The main goal of text summarization is to select the most salient information and form a coherent summary. The answer or summary can consist of multiple separately extracted sentences from different documents, where each of the selected text snippets might be individually important. However, when many of the competing sentences are included in the summary, the issue of information overlap between parts of the output comes up, and a mechanism for addressing redundancy is needed. Therefore, our summarization systems employ two levels of analysis: first, a content level, where every sentence is scored according to the features or concepts it covers and second, a textual level, when, before being added to the final output, the sentences deemed to be important are compared to each other and only those that are not too similar to other candidates are included in the final answer or summary (according to the "Maximum-Marginal-Relevancy (MMR)" concept of Goldstein et al. (1999)). For this purpose, we measure the BE overlap between an intermediate summary and a to-be-added candidate summary sentence according to Hovy et al. (2006). We call this overlap ratio R, where R is between 0 and 1 inclusively. Setting $R = 0.7$ means that a candidate summary sentence, s, can be added to an intermediate summary, S, if the sentence has a BE overlap ratio less than or equal to 0.7.

4.4.5 Experiments

We used the main task of DUC-2007 for evaluation. The purpose of our experiments is to study the impact of the syntactic and semantic representation introduced earlier for complex question answering task. To accomplish this, we generate summaries (See sample summaries in Appendix C) for 20 topics of DUC 2007 by each of our five systems defined below:

(1) TF*IDF: This system is the original topic-sensitive LexRank described in Section 4.3 that uses the similarity measures based on tf*idf (BOW) and does not consider the syntactic/semantic information. The mixture model for this system is given in Eq 4.1.

(2) SYN: This system measures the similarity between the sentences using the *syntactic tree* and the *general tree kernel* function defined in Section 4.4.2. The mixture model for this system is:

$$p(s|q) \quad = \quad d \times SYNSIM(s,q) + (1-d) \times \sum_{v \in C} SYNSIM(s,v) \times p(v|q) \tag{4.7}$$

where, *SYNSIM(s,q)* is the normalized syntactic similarity between the *query (q)* and *the document sentence (s)* and *C* is the set of all sentences in the collection.

(3) SEM: This system measures the similarity between the sentences using the *shallow semantic tree* and the *shallow semantic tree kernel* function defined in Section 4.4.2. Therefore, the mixture model for this system is:

$$p(s|q) \quad = \quad d \times SEMSIM(s,q) + (1-d) \times \sum_{v \in C} SEMSIM(s,v) \times p(v|q) \tag{4.8}$$

where, *SEMSIM(s,q)* is the normalized shallow semantic similarity between the *query (q)* and *the document sentence (s)* and *C* is the set of all sentences in the collection.

(4) SYNSEM: This system measures the similarity between the sentences using both the *syntactic* and *shallow semantic* trees and their associated *kernels*. Hence, the mixture model for this system is:

$$p(s|q) \quad = \quad d \times SYNSEM(s,q) + (1-d) \times \sum_{v \in C} SYNSEM(s,v) \times p(v|q) \tag{4.9}$$

where,
$$SYNSEM(s,q) = \frac{SYNSIM(s,q) + SEMSIM(s,q)}{2}$$
$$SYNSEM(s,v) = \frac{SYNSIM(s,v) + SEMSIM(s,v)}{2}$$

(5) ESSK: This system measures the similarity between the sentences using the *Extended String Subsequence Kernel (ESSK)* defined in Section 4.4.3. Therefore, the mixture model for this system is:

$$p(s|q) \quad = \quad d \times ESSK(s,q) + (1-d) \times \sum_{v \in C} ESSK(s,v) \times p(v|q) \tag{4.10}$$

where, *ESSK(s,q)* is the normalized similarity score between the *query (q)* and *the document sentence (s)* and C is the set of all sentences in the collection.

In our experiments, we set 0.7 as the value of d (bias) and R (overlap ratio).

4.4.6 Evaluation Results

The multiple "reference summaries" given by DUC-2007 are used in the evaluation of our summary content. We carried out automatic evaluation of our summaries using ROUGE (Lin, 2004) toolkit. We showed four of the ROUGE metrics in the experimental results: ROUGE-1 (unigram), ROUGE-L (LCS), ROUGE-W (weighted LCS with $weight = 1.2$) and ROUGE-SU (skip bi-gram). Table 4.1 and Table 4.2 show the ROUGE scores of the TF*IDF and SYN systems respectively. It can be noticed that almost in every cases (except ROUGE-SU) the SYN system outperforms the TF*IDF system which proves the effectiveness of syntactic similarity over the TF*IDF based similarity. Table 4.3 and Table 4.4 show the ROUGE scores of the SEM system and the SYNSEM system respectively. SEM system outperforms the TF*IDF and SYN systems in every measure. The SYNSEM system performs better than SYN system but not as good as the SEM system. It indicates that the shallow semantic similarity is more effective than the syntactic and TF*IDF based similarity. Table 4.5 shows the ROUGE scores of the ESSK system. Here, we find that the ESSK system performs better than the TF*IDF and SYN systems whereas it works worse than that of the SEM and SYNSEM systems.

Measures	ROUGE-1	ROUGE-L	ROUGE-W	ROUGE-SU
Precision	0.3762	0.3505	0.1857	0.1437
Recall	0.3442	0.3206	0.0933	0.1197
F-score	0.3594	0.3348	0.1242	0.1306

Table 4.1: ROUGE measures for TF*IDF system

Measures	ROUGE-1	ROUGE-L	ROUGE-W	ROUGE-SU
Precision	0.3849	0.3506	0.1885	0.1406
Recall	0.3557	0.3239	0.0956	0.1195
F-score	0.3696	0.3366	0.1268	0.1291

Table 4.2: ROUGE measures for SYN system

Measures	ROUGE-1	ROUGE-L	ROUGE-W	ROUGE-SU
Precision	0.4095	0.3748	0.1988	0.1614
Recall	0.3721	0.3406	0.0992	0.1332
F-score	0.3898	0.3567	0.1323	0.1458

Table 4.3: ROUGE measures for SEM system

The comparison between the systems in terms of their F-scores is given in Table 4.6. The SYN system improves the ROUGE-1, ROUGE-L and ROUGE-W scores over the TF*IDF system by 2.84%, 0.53% and 2.14% respectively. The SEM system improves

Measures	ROUGE-1	ROUGE-L	ROUGE-W	ROUGE-SU
Precision	0.3922	0.3549	0.1889	0.1484
Recall	0.3614	0.3270	0.0955	0.1254
F-score	0.3761	0.3403	0.1268	0.1359

Table 4.4: ROUGE measures for SYNSEM

Measures	ROUGE-1	ROUGE-L	ROUGE-W	ROUGE-SU
Precision	0.3920	0.3602	0.1928	0.1460
Recall	0.3577	0.3287	0.0945	0.1215
F-score	0.3740	0.3437	0.1268	0.1325

Table 4.5: ROUGE measures for ESSK system

the ROUGE-1, ROUGE-L, ROUGE-W, and ROUGE-SU scores over the TF*IDF system by 8.46%, 6.54%, 6.56%, and 11.68%, and over the SYN system by 5.46%, 5.98%, 4.33%, and 12.97% respectively. The SYNSEM system improves the ROUGE-1, ROUGE-L, ROUGE-W, and ROUGE-SU scores over the TF*IDF system by 4.64%, 1.63%, 2.15%, and 4.06%, and over the SYN system by 1.74%, 1.09%, 0%, and 5.26% respectively. The SEM system improves the ROUGE-1, ROUGE-L, ROUGE-W, and ROUGE-SU scores over the SYNSEM system by 3.65%, 4.84%, 4.32%, and 7.33% respectively which indicates that including syntactic feature with the semantic feature degrades the performance. On the other hand, the ESSK system improves the ROUGE-1, ROUGE-L, ROUGE-W, and ROUGE-SU scores over the TF*IDF system by 4.07%, 2.64%, 2.12%, and 1.52%, and over the SYN system by 1.19%, 2.10%, 0%, and 2.70% respectively.

Our experimental results show that, the graph-based random walk method for generating query-relevant summaries performs best when we measure the similarity between the sentences (and query) using their semantic structures. Similarity measure based on the syntactic structure also outperforms the traditional TF*IDF based measure for this task.

Systems	ROUGE-1	ROUGE-L	ROUGE-W	ROUGE-SU
TF*IDF	0.3594	0.3348	0.1242	0.1306
SYN	0.3696	0.3366	0.1268	0.1291
SEM	0.3898	0.3567	0.1323	0.1458
SYNSEM	0.3761	0.3403	0.1268	0.1359
ESSK	0.3740	0.3437	0.1268	0.1325

Table 4.6: ROUGE F-scores for systems

Confidence Interval: We show the 95% confidence interval of the important evaluation metrics for our systems to report significance for doing meaningful comparison. We use the ROUGE tool for this purpose. We also include the 95% confidence interval scores of one baseline system and the best system in DUC-2007 in order to show the level of performance our systems achieve. The baseline system generates summaries by returning all the leading sentences (up to 250 words) in the $\langle TEXT \rangle$ field of the most recent document(s). Table 4.7 reports the 95% confidence intervals of ROUGE-1 F-scores for all the systems.

Systems	ROUGE-1
Baseline	0.3266 - 0.3423
Best System	0.4316 - 0.4459
TF*IDF	0.3553 - 0.4008
SYN	0.3593 - 0.4077
SEM	0.3634 - 0.4169
SYNSEM	0.3562 - 0.4207
ESSK	0.3588 - 0.3900

Table 4.7: 95% confidence intervals for different systems

For a sample of 25 summaries[4] drawn from our different systems' generated summaries we conduct an extensive manual evaluation in order to analyze the effectiveness of our approaches. The manual evaluation comprised a Pyramid-based evaluation of contents and a user evaluation to get the assessment of linguistic quality and overall responsiveness.

[4] Randomly, we chose 5 summaries for each of these systems.

Pyramid Evaluation: In the DUC 2007 main task, 23 topics were selected for the optional community-based pyramid evaluation. Volunteers from 16 different sites created pyramids and annotated the peer summaries for the DUC main task using the given guidelines[5]. 8 sites among them created the pyramids. We used these pyramids to annotate our peer summaries to compute the modified pyramid scores[6]. We used the *DUCView.jar*[7] annotation tool for this purpose. Table 4.8 shows the modified pyramid scores of all our systems. A baseline system's score is also reported. The peer summaries of the baseline system are generated by returning all the leading sentences (up to 250 words) in the $\langle TEXT \rangle$ field of the most recent document(s). From these results we see that all our systems perform better than the baseline system and inclusion of semantic information always yields better scores whereas SYN and SYNSEM systems perform decently.

Systems	Modified Pyramid Scores
Baseline	0.1387
TF*IDF	0.5120
SYN	0.4848
SEM	0.5727
SYNSEM	0.4615
ESSK	0.5469

Table 4.8: Modified pyramid scores for all systems

User Evaluation: Two university graduate students judged the summaries for linguistic quality and overall responsiveness. The given score is an integer between 1 (very poor) and 5 (very good) and is guided by consideration of the following factors: 1. Grammaticality, 2. Non-redundancy, 3. Referential clarity, 4. Focus and 5. Structure and Coherence. They

[5] http://www1.cs.columbia.edu/~becky/DUC2006/2006-pyramid-guidelines.html

[6] This equals the sum of the weights of the Summary Content Units (SCUs) that a peer summary matches, normalized by the weight of an ideally informative summary consisting of the same number of contributors as the peer.

[7] *http://www1.cs.columbia.edu/~ani/DUC2005/Tool.html*

also assigned a content responsiveness score to each of the automatic summaries. The content score is an integer between 1 (very poor) and 5 (very good) and is based on the amount of information in the summary that helps to satisfy the information need expressed in the topic narrative. These measures were used at DUC 2007. Table 4.9 presents the average linguistic quality and overall responsive scores of all our systems. The same baseline system's scores are given for meaningful comparison. From these results, we can see that all our systems perform better than the baseline system in terms of overall responsiveness score whereas in terms of linguistic quality they perform close to each other. The results also show that the ESSK system can perform better than the TF*IDF system in terms of linguistic quality.

Systems	Linguistic Quality	Overall Responsiveness
Baseline	4.24	1.80
TF*IDF	4.16	3.00
SYN	4.15	2.75
SEM	4.12	2.80
SYNSEM	4.10	2.50
ESSK	4.20	2.75

Table 4.9: Linguistic quality and responsive scores for all systems

4.5 Aspect-driven Random Walk Model

To generate summaries as answers to the complex questions, we focus on a deeper semantic analysis of the source documents instead of relying only on document word frequencies to select important concepts. We use a predefined list of important aspects to direct our search for the most relevant sentences, and generate topic-focused summaries that cover all these aspects. For example, a topic about *Natural Disasters* might consider the aspects: *what happened; date; location; reasons for the disaster; casualties; damages; rescue efforts* *etc.* while generating the summary. We propose a novel topic-focused multi-document summarization framework that operates on a Markov chain model and follows a random walk paradigm in order to generate possible summary sentences (Chali, Hasan, and Imam, 2011b).

We build three alternative systems for summary generation that are based on important aspects, random walk model, and a combination of both. We run our experiments on the TAC[8]-2010, and DUC[9]-2006 data and based on the evaluation results we argue that augmenting important aspects with a random walk model often outperforms the other two alternatives. Contributions of this work are: a) constructing an aspect-based summarization model that generates summaries based on given important aspects about the topics, b) building a novel summarization model based on a random walk paradigm that operates on a Markov chain exploiting topic signature (Lin and Hovy, 2000) and Rhetorical Structure Theory (RST) (Mann and Thompson, 1988) as node weights and WordNet[10]-based sentence similarities as edge weights, and c) generating a hybrid summarization model combining the aspect-based model with the random walk approach. Extensive automatic evaluations suggest that the combined model can raise the performance up to 19.22% while manual evaluations further confirm this improvement. In the next subsections we present our three alternative summary generation models, and show the evaluation results.

4.5.1 Models

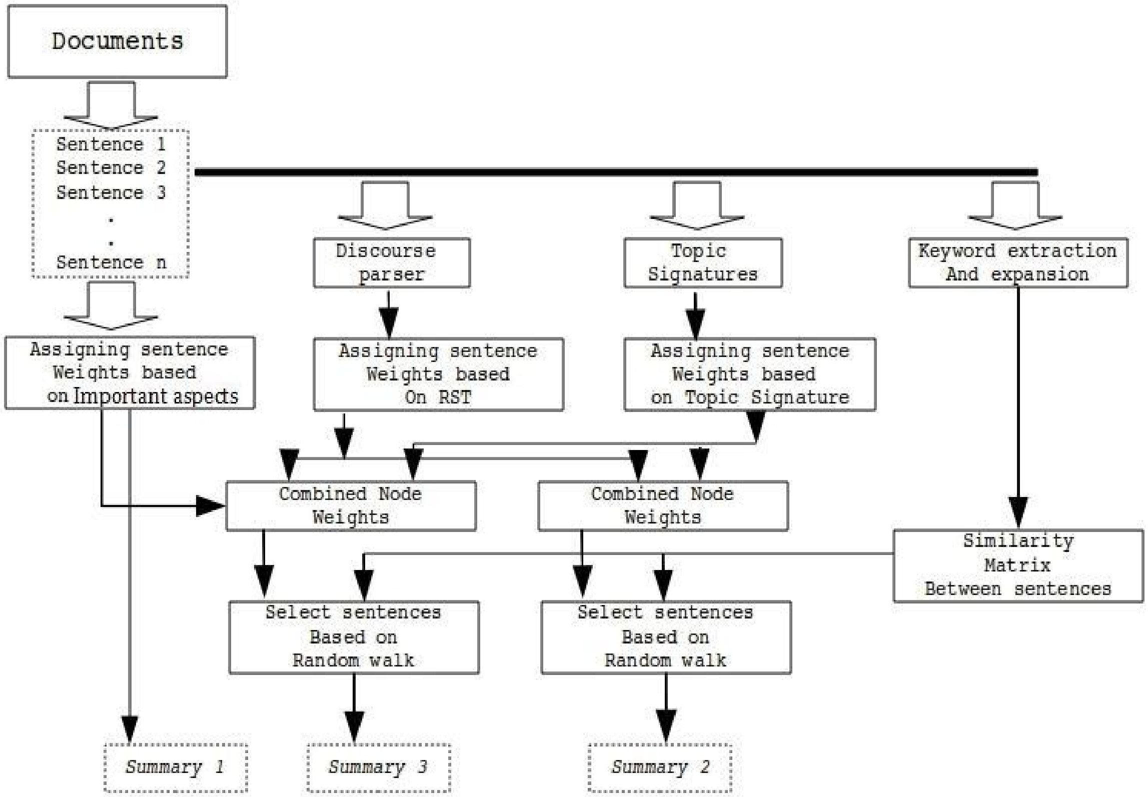
Our first model is solely based on aspect information, while the second follows a novel random walk framework, and the third model is the aspect-driven random walk approach that combines the intuitions of the first two models. We get a candidate summary from each

[8] http://www.nist.gov/tac/2010/

[9] http://duc.nist.gov/

[10] http://wordnet.princeton.edu/

of the model at the end of the summary generation procedure. Therefore, three models give us three candidate summaries for the same given topic. Figure 4.6 presents the overall architecture of our systems.



Documents
Sentence 1
Sentence 2
Sentence 3
.
.
Sentence n
Discourse parser
Topic Signatures
Keyword extraction And expansion
Assigning sentence Weights based on Important aspects
Assigning sentence Weights based On RST
Assigning sentence Weights based on Topic Signature
Combined Node Weights
Combined Node Weights
Similarity Matrix Between sentences
Select sentences Based on Random walk
Select sentences Based on Random walk
Summary 1
Summary 3
Summary 2

Figure 4.6: The overall architecture of our approaches

Aspect–based Model

Our first approach exploits the predefined list of important aspects to find the most relevant sentences from the document collection for creating the summaries. For each question (i.e., aspect) of a topic, we did keyword expansion using WordNet[11] (Fellbaum, 1998). For example, the word "happen" being a keyword in the given *aspect*: "What happened?"

[11] For simplicity, we consider the synsets up to level 1 in this research.

returns the words: *occur, pass, fall out, come about, take place* from WordNet. On the other hand, for each document sentence in the collection we perform Named Entity (NE) tagging using the *OAK* system (Sekine, 2002). Named Entities (NE) are defined as terms that refer to a certain entity. For instance, *USA* refers to a certain *country*, and *$200* refers to a certain quantity of money. OAK system has 150 named entities (such as PERSON, LOCATION, ORGANIZATION, GPE (Geo-Political Entity), FACILITY, DATE, MONEY, PERCENT, TIME etc.) that can be tagged. They are included in a hierarchy. We weight each sentence based on the presence of one or more Named Entity classes. We rank the document sentences based on the following two criteria:

1. Similarity of each sentence with the expanded aspect (in terms of word matching).

2. weight assigned to each sentence by the NE tagging procedure[12].

Finally, we select the top-ranked sentences to be included in the candidate summary (Summary 1 in Figure 4.6).

Random Walk Model

To include into our second candidate summary, we select the most relevant sentences by following a random walk on a graph where each node is a document sentence and the edges represent similarity between sentences. The whole procedure operates on a Markov chain (MC). A Markov chain is a process that consists of a finite number of states and some known probabilities p_{ij}, where p_{ij} is the probability of moving from state j to state i. For each node (i.e., sentence) and each edge in the graph, we calculate *"node weight"* and *"edge weight"*, respectively. Once we find all the node weights and edge weights, we perform a random walk on the graph following a Markov chain model in order to select

[12] For example, for an aspect like *"When did the accident happened?"*, we search for $< Time >$ tag in the NE tagged sentences and give them higher weights if found.

the most important sentences. The initial sentence is chosen simply based on the node (sentence) weights using the following formula:

$$InitialSentence = \arg\max_{i=1}^{N} \left(weight\left(S_i\right)\right) \tag{4.11}$$

where N is the total number of nodes in the graph. After finding the initial best sentence, in each step of the random walk we calculate the probability (transition probability) of choosing the next relevant sentence based on the following equation:

$$P(S_j|S_i) = \frac{1}{\alpha} \arg\max_{j=1}^{Z} \left(weight\left(S_j\right) * similarity\left(S_i, S_j\right)\right) \tag{4.12}$$

where S_i is the sentence chosen early, S_j is the next sentence to be chosen, Z is the set of sentence indexes that does not contain i, the $similarity(S_i, S_j)$ function returns a similarity score between the already selected sentence and a new sentence under consideration, and α is the normalization factor that is determined as follows:

$$\alpha = \sum_{j=1}^{Z} \left(weight\left(S_j\right) * similarity\left(S_i, S_j\right)\right) \tag{4.13}$$

Node Weight: We associate to each node (sentence) in the graph a weight that indicates the importance of the node with respect to the document collection. Node weights are calculated based on a Topic Signature (TS) model (Lin and Hovy, 2000) and Rhetorical Structure Theory (RST) (Mann and Thompson, 1988). We combine the weights of TS and RST, and normalize it to get the final weights of the sentences/nodes.

Using Topic Signature: Topic signatures are typically used to identify the presence of a complex concept–a concept that consists of several related components in fixed relationships (Lin and Hovy, 2000). Inspired by the idea presented in (Lin and Hovy, 2000), for each topic present in the data set, we calculate its topic signature defined as below:

$$\begin{aligned} TS &= \{topic, signature\} \\ &= \{topic, \langle (t_1, w_1), \cdots, (t_n, w_n) \rangle\} \end{aligned} \tag{4.14}$$

where *topic* is the target concept and signature is a vector of related terms. Each t_i is a term highly correlated to the *topic* with association weight, w_i. We use the following log-likelihood ratio to calculate the weights associated with each term (i.e., word) of a sentence:

$$w_i = log \frac{occurrences\,of\,t_i\,in\,topic\,j\,sentences}{occurrences\,of\,t_i\,in\,all\,topics'\,sentences} \tag{4.15}$$

To calculate the topic signature weight for each sentence, we sum up the weights of the words in that sentence and then, normalized the weights. Thus, a sentence gets a high score if it has a set of terms that are highly correlated with a target concept (topic).

Exploiting Rhetorical Structure Theory (RST): Rhetorical Structure Theory provides a framework to analyze text coherence by defining a set of structural relations to composing units ("spans") of text. The most frequent structural pattern in RST is that two spans of text are related such that one of them has a specific role relative to the other. A paradigm case is a claim followed by evidence for the claim. RST assumes an "Evidence" relation between the two spans that is denoted by calling the claim span a *nucleus* and the evidence span a *satellite*[13]. We parse each document sentence within the framework of Rhetorical Structure Theory (RST) using a Support Vector Machine (SVM)-based discourse parser described in (duVerle and Prendinger, 2009) that was shown 5% to 12% more accurate than current state-of-the-art parsers. We observe that in a relation the nucleus often contains the main information while the satellite provides some additional information. Therefore, we assign

[13] http://www.sfu.ca/rst/01intro/intro.html

a weight to each sentence that is a nucleus of a relation and normalize the weights at the end.

Edge Weight: Edge weight is determined by measuring similarity between the sentences. Initially, we remove the stopwords from the sentences using a stopword list. Then, we use the *OAK* system (Sekine, 2002) to get the stemmed words of a sentence. We expand the remaining keywords of the sentence using WordNet. Finally, we find the similar words between each pair of sentences that denote the edge weight between the two sentences. We build a similarity matrix by populating into it the edge weights between sentences.

Aspect-driven Random Walk Model

The third model that we construct to generate a candidate summary is based on augmentation of the predefined important aspects into the random walk framework. Motivated by Harabagiu, Lacatusu, and Hickl (2006), where they describe how a random walk can be used to populate a network with potential decompositions of a complex question, we propose to use the list of aspects (given in TAC-2010) in the random walk model as a guided way to provide a better coverage to satisfy a wide range of information need on a given topic. We term this model as a *Combined Model* since it combines the important aspects with the random walk paradigm. The whole procedure can be again formulated according to a Markov Chain principle described in Section 4.5.1 except the fact that the node(sentence) weights will also include the weights obtained by using the list of aspects' information as defined in Section 4.5.1. Figure 4.7 shows a part of the graph with node and edge weights (after applying the combined model) for the top ranking sentences that were chosen by the random walk. This is an example of a DUC-2006 topic outlined below.

```
<topic id = "D0626H" category = "2">
<title> bombing of US embassies in Africa  </title>
```

S1: Among them is Saudi dissident Osama bin Laden, who allegedly runs al Qaida, a radical Islamic network accused of planning the bombings.

S2: In an interview Tuesday, Home Affairs Minister Ali Ameir Mohamed likened Ahmed to a chameleon.

S3: It said Khalid, who can not speak English or Kiswahili but only Arabic, was identified by a guard and a civilian worker at the embassy and a third witness.

S4: Although no details were released in court, local media said traces of chemicals that could have been used to make the bomb had been found in Saleh's home and car.

S5: The action contrasted markedly to a decision by Kenya, where the American Embassy was bombed on the same day.

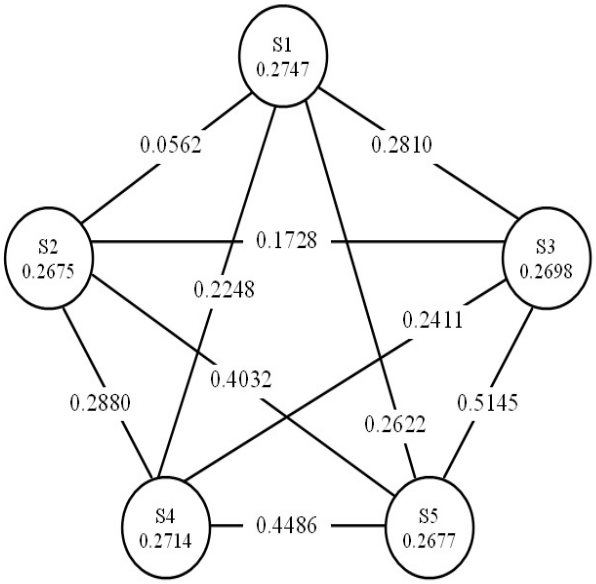


Figure 4.7: Important aspects with random walk model

From Figure 4.7, we get to the fact that initially, sentence *S1* is chosen into the candidate summary as it has the highest node (sentence) weight, then, by performing a random walk based on the transition probabilities of the Markov chain model, we find *S2* as the next candidate sentence, then, *S3*, *S4*, *S5* and so on. The random walk stops after the *k* steps which is related to reaching the summary-length of 250 words.

4.5.2 Experiments

Task Description: TAC-2010 provides a new research direction for multi-document summarization by the means of predefined supervision or guide (the category and its aspects) that defines what information the reader is actually looking for. The task of DUC-2006 models the real-world complex question answering in terms of multi-document summarization. That is: *"Given a complex question (topic description) and a collection of relevant documents, the task is to synthesize a fluent, well-organized 250-word summary of the documents that answers the question(s) in the topic"*. We consider a modified task description that induces the guided concept of TAC-2010 in order to automatically generate 250-word summaries like DUC-2006. Our summarization task can be defined as:

"To write a 250-word summary of a set of given newswire articles for a given topic, where the topic falls into a predefined category."

Data: In this research, we run our experiments using the TAC-2010 and DUC-2006 data applying three different models to generate three candidate summaries for each topic. The test dataset in TAC-2010 is composed of 44 topics, divided into five categories: Accidents and Natural Disasters, Attacks, Health and Safety, Endangered Resources, Investigations and Trials. In this research, we consider only the first two categories[14]. As DUC-2006 data were not categorized, we manually categorize them to put into our chosen categories: Accidents and Natural Disasters, and Attacks. Since human-generated abstract summaries are not publicly available, we perform an extensive manual evaluation on the TAC-2010 data to report comparisons based on linguistic quality and responsiveness of the summaries. For DUC-2006 data, we obtain both an automatic[15] and a manual evaluation.

[14] TAC provides already categorized data.

[15] Abstract summaries are available for comparisons.

Evaluation Results: For the DUC-2006 data, we carried out automatic evaluation of our candidate summaries using ROUGE (Lin, 2004) toolkit. For all our systems, we report the widely accepted important metrics: ROUGE-2 and ROUGE-SU. We also present the ROUGE-1 scores since they provide a better correlation with the human judgement. We show the 95% confidence intervals of the important evaluation metrics for our systems to report significance for doing meaningful comparison. Table 4.10 to Table 4.12 show the ROUGE-1, ROUGE-2, and ROUGE-SU scores of our three different summary generation models.

Scores	Aspects	Random Walk	Combined
Recall	0.3488	0.3344	0.3624
Precision	0.3415	0.3604	0.3556
F-score	0.3444	0.3460	0.3587

Table 4.10: ROUGE-1 measures

Scores	Aspects	Random Walk	Combined
Recall	0.0711	0.0500	0.0633
Precision	0.0693	0.0545	0.0609
F-score	0.0701	0.0520	0.0620

Table 4.11: ROUGE-2 measures

Scores	Aspects	Random Walk	Combined
Recall	0.1159	0.1029	0.1211
Precision	0.1109	0.1182	0.1156
F-score	0.1123	0.1090	0.1179

Table 4.12: ROUGE-SU measures

For all the three systems, Table 4.13 shows the F-scores of the reported ROUGE measures while Table 4.14 reports the 95% confidence intervals of the important ROUGE measures. Table 4.13 clearly shows that the **Combined** system improves the ROUGE-

1, ROUGE-2, and ROUGE-SU scores over the **Random walk** system by **3.67%**, **19.22%**, and **8.21%**, respectively, whereas, it could not beat the ROUGE-2 score of **Aspect**–based system but improves the ROUGE-1, and ROUGE-SU scores by **4.15%**, and **4.97%**, respectively. These results suggest that augmenting important aspects with the random walk model provides a better content coverage to satisfy the information need. The proposed methods are also compared with a *Baseline* system. The *Baseline* is the official baseline system established in DUC-2006 that generated the summaries by returning all the leading sentences (up to 250 words) in the $\langle TEXT \rangle$ field of the most recent document(s). We also list the average ROUGE scores of all the participating systems of DUC-2006 (i.e., *DUC-Average*). Table 4.15 presents this comparison which denotes that the **Combined** system improves the ROUGE-1, and ROUGE-2 scores over the **Baseline** system by **11.77%**, and **17.78%**, respectively, whereas, it performs closely to the average DUC-2006 systems.

Systems	ROUGE-1	ROUGE-2	ROUGE-SU
Aspects	0.3444	0.0701	0.1123
Random walk	0.3460	0.0520	0.1090
Combined	0.3587	0.0620	0.1179

Table 4.13: ROUGE F-scores for different systems

Systems	ROUGE-2	ROUGE-SU
Aspects	0.0569 - 0.0844	0.1053 - 0.1190
Random walk	0.0373 - 0.0682	0.0894 - 0.1262
Combined	0.0364 - 0.0879	0.0989 - 0.1363

Table 4.14: 95% confidence intervals for different systems

We conduct an extensive manual evaluation in order to analyze the effectiveness of our approach. We judged the summaries for linguistic quality and overall responsiveness according to the DUC evaluation guidelines[16]. Table 4.16 and Table 4.17 presents the av-

[16] http://www-nlpir.nist.gov/projects/duc/duc2007/quality-questions.txt

Systems	ROUGE-1	ROUGE-2
Aspects	0.3444	0.0701
Random walk	0.3460	0.0520
Combined	0.3587	0.0620
Baseline	0.3209	0.0526
DUC-Average	0.3778	0.0748

Table 4.15: Comparison with DUC-2006 systems

erage linguistic quality and overall responsive scores of all the systems on TAC-2010 data and DUC-2006 data, respectively. To compare the proposed models' performance with the state-of-the-art systems, in Table 4.17 we also list the scores of *LCC's GISTexter*[17] system that participated in the DUC-2006 competition and was ranked as one of the best systems. Analyzing these results yields the fact that augmenting important aspects with the random walk model often outperforms the random walk model alone in terms of linguistic quality and responsiveness scores. Table 4.17 shows that the proposed aspect-driven random walk model (i.e., *Combined*) performs very close to LCC's system in terms of linguistic quality while considerably outperforming it in terms of overall responsiveness scores. This confirms that the use of the aspect information enhances the coverage of the information that is necessary to satisfy the quest of the users.

Systems	Lin. Quality	Responsiveness
Aspects	4.00	4.00
Random walk	3.60	3.00
Combined	4.00	3.00

Table 4.16: Linguistic quality and responsiveness scores (TAC-2010 data)

[17] *http://duc.nist.gov/pubs/2006papers/lcc2006.pdf*

Systems	Lin. Quality	Responsiveness
Aspects	3.72	3.00
Random walk	3.52	3.00
Combined	3.76	3.20
LCC	4.10	2.84

Table 4.17: Linguistic quality and responsiveness scores (DUC-2006 data)

4.6 Conclusion

In this chapter, we have proposed the use of syntactic, shallow semantic and extended string subsequence kernels for measuring the similarity between the sentences in the traditional graph-based random walk framework for answering complex questions. We have analyzed the impact of exploiting syntactic and semantic information on the performance of the random walk model. Experiments demonstrate the effectiveness of the proposed approach. We have also presented a novel methodology that uses a predefined list of important aspects in a random walk framework by performing a deeper semantic analysis of the source documents.

Chapter 5

Learning Good Decompositions of Complex Questions

5.1 Introduction

Complex questions address an issue that often relates to multiple entities, events and their complex relations. A complex question might ask about events, biographies, definitions, descriptions, or reasons (Chali, Joty, and Hasan, 2009). However, it is not always understandable, to which direction one should move to search for the answer to a complex question. For example, a complex question like *"Describe the tsunami disaster in Japan."* has a wider focus without a single or well-defined information need. To narrow down the focus, this question can be decomposed into a series of simple questions such as *"How many people had died by the tsunami?"*, *"How many people became homeless?"*, *"Which cities were mostly damaged?"* etc. Decomposing a complex question automatically into simpler questions in this manner such that each of them can be answered individually by using the state-of-the-art Question Answering (QA) systems, and then combining the individual answers to form a single answer to the original complex question has been proved effective to deal with the complex question answering problem (Harabagiu, Lacatusu, and Hickl, 2006; Hickl and Harabagiu, 2006). However, issues like judging the significance of the decomposed questions remained beyond the scope of all the researches done so far. Enhancing the quality of the decomposed questions can provide more accurate answers to the complex questions (Chali, Hasan, and Imam, 2012b). So, it is important to judge the quality of the decomposed questions and then, investigate more methods to enhance their quality. In this research, we address this challenging task and come up with a supervised model to automatically learn good decompositions of complex questions.

To find answers to complex questions, it is important at first to know which information is relevant to an event, biography, definition, description or reason. We assume that a set of relevant data set is given for each complex question that certainly possesses the potential answer to the complex question. However, it is still necessary to identify the most important sentences from the given data set (that are mostly relevant to contain potential answers) since the data set may have a huge number of sentences and therefore, not suitable to act as a reasonably straightforward answer to the complex question. For this reason, during training data generation, we initially determine the most important sentences from the given set of relevant documents, and then, simplify these sentences in the second step. In the third step, questions are generated from the simplified sentences. These questions, considered as candidate decompositions of the complex question, are manually annotated (as good or bad candidates) and used to train a Support Vector Machines (SVM) classifier. In the testing phase, the SVM-learned model is used to identify good candidate decompositions of the previously unseen complex questions automatically. Experiments on the DUC data sets prove the effectiveness of our approach[1]. The next sections include discussions about our motivation and related work, the overview of our approach, the training data generation phase, our supervised model, and finally the evaluation results.

5.2 Motivation and Related Work

Earlier studies have shown that the state-of-the-art QA systems can handle simple questions in a straight forward manner (Moldovan, Clark, and Bowden, 2007) whereas complex questions often need sophisticated treatment such as question decomposition and multi-document summarization (Harabagiu, Lacatusu, and Hickl, 2006). As it is often difficult to predict to which direction one should move to search for the complete answer to a complex question, efficient question decomposition could act as a guide for the search. A similar

[1] The proposed method for question decomposition is tested on the domain of newswire articles. However, our approach is generic and applicable across other domains.

task has drawn considerable attention in the recent Text Analysis Conferences (TAC-2010 and TAC-2011, Guided Summarization Task[2]). Here, the main objective is to guide the search based on a number of predefined (human-made) aspects. For example, in the "Accidents and Natural Disasters" category, the given aspects are: a) WHAT: what happened, b) WHEN: date, time, other temporal placement markers, c) WHERE: physical location etc. As these "aspects" are created by humans manually, it is both difficult and time consuming to address all the possible topic categories of the world. This motivated us to propose a supervised approach for automatically learning good decompositions of complex questions that can in turn act as automatically generated aspects for any unseen topic category. Extensive experiments on the DUC benchmark data sets showed the effectiveness of our proposed approach.

Question decomposition has been proved effective to deal with the complex question answering problem in many studies. For example, in Harabagiu, Lacatusu, and Hickl (2006), they introduce a new paradigm for processing complex questions that relies on a combination of (a) question decompositions; (b) factoid QA techniques; and (c) Multi-Document Summarization (MDS) techniques. Necessity and positive impact of question decomposition have been also shown in many studies in the complex question answering domain (Lacatusu, Hickl, and Harabagiu, 2006; Hickl and Harabagiu, 2006). However, there is no direct research on the problem of automatically learning good decompositions of complex questions rather several paraphrasing and textual entailment methods can be considered the most relevant tasks that have been researched. For example, in QA systems for document collections a question could be phrased differently than in a document that contains its answer (Pasca, 2003). Studies have shown that the system performance can be improved significantly by taking such variations into account (Harabagiu and Hickl, 2006). Techniques to measure similarities between texts can be an effective way of judging the

[2] http://www.nist.gov/tac/2011/Summarization/Guided-Summ.2011.guidelines.html

importance of a sentence. Semantic roles typically offer a significant first step towards deeper text understanding (Moschitti et al., 2007). There are approaches in "Recognizing Textual Entailment", "Sentence Alignment" and "Question Answering" that use semantic information in order to measure the similarity between two textual units (MacCartney et al., 2006). This indeed motivates us to find semantic similarity between a document sentence and a complex question while selecting the most important sentences. Simplifying a sentence can lead to more accurate question generation (Heilman and Smith, 2010a), so we simplify the complex sentences in this research. Different methods have been proposed so far to accomplish the task of Question Generation (QG) (Wang, Tianyong, and Wenyin, 2008; Chen, Aist, and Mostow, 2009). Some QG models utilize question generation as an intermediate step in the question answering process (Hickl et al., 2005). Generated questions can be ranked using different approaches such as statistical ranking methods, dependency parsing, identifying the presence of pronouns and named entities, and topic scoring (Heilman and Smith, 2010a; McConnell et al., 2011).

5.3 Overview of Our Approach

The main contribution of this research is to develop a classification system that given a complex question and a list of simple questions can decide whether questions in the latter are to be considered a decomposition of the former, viz. they ask about part of the information asked by the complex question. To accomplish this task, simple questions are generated from the documents containing possible answers to the complex question and used to train the classifier[3]. There is a pipeline starting with a complex question and ending with a set of (presumably) one or more simple questions. We assume that a set of relevant documents is given along with each complex question that certainly possesses potential answers to the

[3] The same procedure is applied to generate the test data set.

complex question. However, it is still necessary to identify the most important sentences due to the presence of a huge number of sentences in the data set. We conduct a shallow and a deep semantic analysis between the complex question and the given document sentences to perform this task. Sentences that are found during this process can be long and complex to deal with. Hence, we pass the selected sentences through a sentence simplification module. Once we find the simple sentences, we use a sentence-to-question generation approach in order to generate corresponding questions. We claim that these questions are the potential candidate decompositions of the complex question. We judge the quality of the generated decompositions manually and annotate them into two classes: good candidate and bad candidate, considering their correctness at the question level and verifying whether they can actually satisfy the information need stated in the original complex question partially. We employ a well-known supervised learning technique: SVM, that is trained on this annotated data set and then, the learned model is used to identify good decompositions of the unseen complex questions automatically. Learning good decompositions of complex questions is unique and to the best of our knowledge, no other study has investigated this challenge before in our setting. Figure 5.1 shows the overview of our work. In the next sections, we elaborate on our approach in more detail.

5.4 Training Data Generation

5.4.1 Filtering Important Sentences

We use two methods to extract the most important sentences related to the complex question from the given document collection. Firstly, we parse the document sentences (and the question) semantically using the Semantic Role Labeling (SRL) system, ASSERT[4]. We

[4] Available at http://cemantix.org/assert

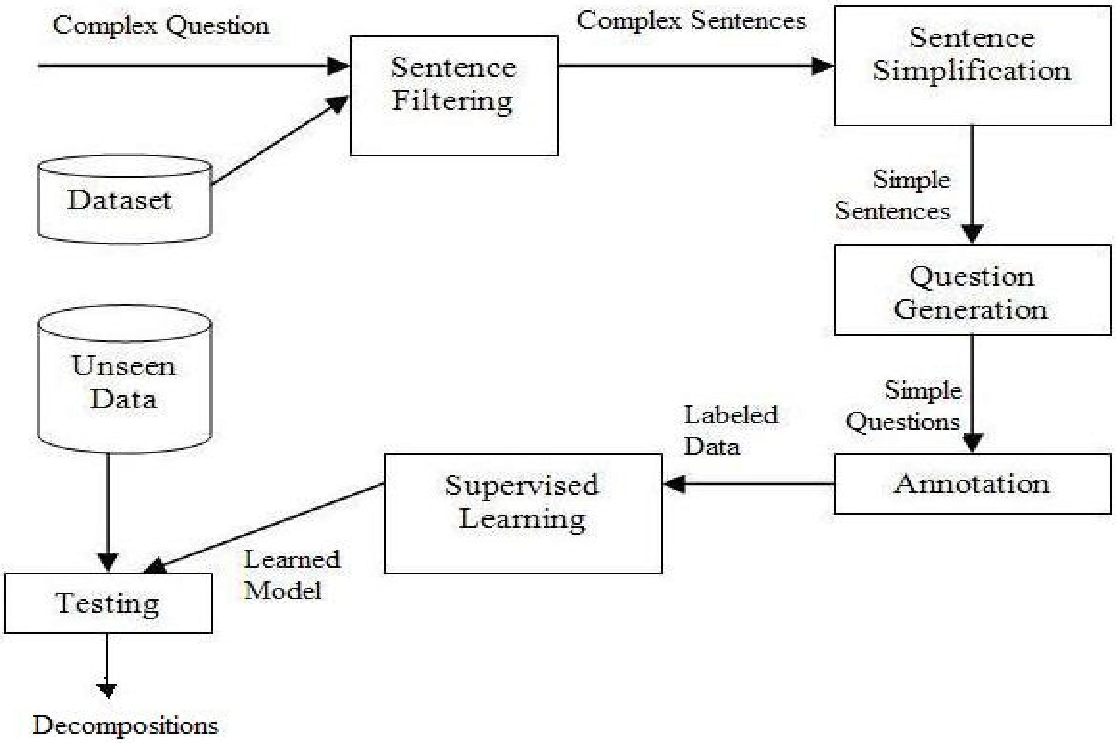


Figure 5.1: Overview of our work

use the Shallow Semantic Tree Kernel (SSTK) (Moschitti et al., 2007) to get a similarity score between a document sentence and the complex question based on their underlying semantic structures. Secondly, after getting stemmed words by using the *OAK* system (Sekine, 2002) for each sentence and the complex question, we perform keyword expansion using WordNet[5] (Fellbaum, 1998). For example, the word "happen" being a keyword in the question "What happened?" returns the words: *occur, pass, fall out, come about, take place* from WordNet. Then, we find out the similar words between the sentence-question pair that gives us a similarity score. We combine this score with the score obtained from the shallow semantic analysis and select the top-scored sentences from the documents. For example[6], *"With economic opportunities on reservations lagging behind those available*

[5] http://wordnet.princeton.edu/

[6] We look into a running example through out the rest of this chapter considering the complex question: "Discuss conditions on American Indian reservations or among Native American communities."

in big cities, and with the unemployment rate among Native Americans at three times the national average, thousands of poor, often unskilled Native Americans are rushing off their reservations." is selected as an important sentence.

5.4.2 Simplifying the Sentences

Sentences that we select in the earlier stage may have complex grammatical structure with multiple embedded clauses. Therefore, we simplify the complex sentences with the intention to generate more accurate questions. We call the generated simple sentences, *elementary sentences* since they are the individual constituents that combinedly possess the overall meaning of the complex sentence. We use the simplified factual statement extractor model[7] (Heilman and Smith, 2010a). Their model extracts the simpler forms of the complex source sentence by altering lexical items, syntactic structure, and semantics and by removing phrase types such as leading conjunctions, sentence-level modifying phrases, and appositives. For example, for the complex sentence that was selected as important in Section 5.4.1, we get one of the possible elementary sentences as, *"Thousands of poor, often unskilled Native Americans are rushing off their reservations"*.

5.4.3 Sentence-to-Question Generation

Once we get the elementary sentences, our next task is to produce a set of possible questions from them according to Ali, Chali, and Hasan (2010). We claim these questions to be the candidate decompositions of the original complex question. In this research, we work on generating six simple types of questions: *who, what, where, when, whom* and *how much*. We use the OAK system (Sekine, 2002) to produce Part-Of-Speech (POS) tagged

[7] Available at http://www.ark.cs.cmu.edu/mheilman/

and Named Entity (NE) tagged sentences. We use these information to classify the sentences by using a sequence of two simple classifiers. The first classifies the sentences into fine classes (Fine Classifier) and the second into coarse classes (Coarse Classifier). This is a similar but opposite approach to the one described in (Li and Roth, 2002). We define the five coarse classes as: 1) Human, 2) Entity, 3) Location, 4) Time, and 5) Count. Based on the coarse classification, we consider the relationship between the words in the sentence. For example, if the sentence has the structure "Human Verb Human", it will be classified as "whom and who" question types. We define a set of ninety basic word-to-word interaction rules to check the coarse classes. We use the POS information to decompose the main verb and perform necessary subject-auxiliary inversion and finally, insert the question word (with a question mark at the end) to generate suitable questions from the given elementary sentence. For example, for the elementary sentence generated in Section 5.4.2, we get a simple question as: *"Who are rushing off their reservations?"*.

5.5 Supervised Model

For supervised learning techniques, annotated or labeled data is required as a precondition. We manually annotate the generated simple questions (i.e. candidate decomposed questions) into two classes[8]: good candidate and bad candidate, employ the Support Vector Machines (SVM) that is trained on this annotated data set and then, use the learned model to predict good decompositions from the unlabeled candidate set of decompositions (i.e. test data set) automatically. We describe our feature space, learning and testing modules in the following subsections.

[8] We inspect each question to measure whether they are lexically, syntactically and semantically correct or not. We also judge each decomposed question against the original complex question and analyze further to find out whether it can ask for any information that can be found in the given data. This analysis guides us to label each question as $+1$ (good candidate) or -1 (bad candidate).

5.5.1 Feature Space

For our SVM classifier, we use a total of thirteen features that are divided into two major categories: one that considers the correctness at the question level and other is a coverage component that measures whether a decomposed question can actually satisfy the information need stated in the original complex question partially. We automatically extract these features from the questions (for both training and testing data) in order to feed them to the supervised models for learning and then, for prediction. These features are related to the original important sentence (that is selected in Section 5.4.1), the input sentence (elementary sentence), the generated simple question, and the original complex question. Correctness of the questions can be measured using a composition of the following features:

Grammaticality: We count the number of proper nouns, pronouns, adjectives, adverbs, conjunctions, numbers, noun phrases, prepositional phrases, and subordinate clauses in the syntactic structures of the question and the input sentence. We set a certain threshold[9] to denote the limit up to which a candidate can be termed as good. We also include some boolean features to encode the tense information of the main verb.

Length: We calculate the number of tokens in the question, the original source sentence, the input elementary sentence, and the answer term (that is replaced by a question type). We set a threshold on this value, too.

Presence of Question Word: We consider some boolean features to identify the presence or absence of a certain question type: *who, what, where, when, whom* and *how much*.

Presence of Pronouns: If a question has one or more pronouns, we understand that the question is asking about something that has limited reference and hence, we consider the question as *vague*. To identify whether a question includes pronouns or not, we employ a boolean feature.

[9] The thresholds are set after inspecting the questions and the input sentences manually.

The coverage component of our feature extraction module tells whether a decomposed question can satisfy the requested information need partially. To automatically encode this feature for each question, we conduct an extensive linguistic analysis and WordNet-based semantic similarity measure between the decomposed question and the original complex question.

Linguistic Analysis: We use ROUGE (Recall-Oriented Understudy for Gisting Evaluation) to automatically determine the quality of a question by comparing it to the original complex question using a collection of measures (Lin, 2004).

WordNet-based semantic similarity measure: We conduct a similarity measure between the decomposed question and the original complex question (according to the procedure discussed in Section 5.4.1) that outputs a similarity score as the feature value.

5.5.2 Learning and Testing

Once we get the feature values for all the decomposed questions along with the associated annotation (good or bad candidate), we feed this data to the supervised learner so that a learned model is established. We use a set of 523 questions for the training purpose. Later, this model is used to predict the labels for the new set of simple questions automatically during the testing phase. Our test data set includes 350 questions. In this work, we use SVM (Cortes and Vapnik, 1995) as the classifier. We use the SVM^{light} (Joachims, 1999) package[10] for training and testing in this work. SVM^{light} consists of a learning module and a classification module. The learning module takes an input file containing the feature values with corresponding labels and produces a model file. The classification module is used to apply the learned model to new samples. We use $g(x)$, the normalized distance from the hyperplane to each sample point, x to rank the questions.

[10] http://svmlight.joachims.org/

5.6 Evaluation and Analysis

5.6.1 Corpus

We consider the task of DUC-2006 which asks to synthesize a fluent, well-organized 250-word summary of the documents that answers the question(s) in the topic statement. We use a subset of 10 topics[11] from the DUC-2006 data to run our experiments.

5.6.2 Cross-validation

Cross-validation is an effective technique for estimating the performance of a predictive model. We apply a 3-fold cross-validation on the annotated questions (i.e. training data set) to estimate how accurately our SVM model would perform on the unlabeled set of candidate decompositions (test data set). To allow some flexibility in separating the classes (i.e. good and bad decompositions in our case), SVM models have a cost parameter, C, which controls the trade-off between allowing training errors and forcing rigid margins. The main idea is to create a soft margin that can allow some misclassifications. Estimating the optimal value of C is a difficult task. We use a randomized local-grid search according to Hsu, Chang, and Lin (2008) for estimating the value of C. We try the value of C in 2^i by following heuristics, where $i \in \{-5, -4, \cdots, 4, 5\}$ and set C as the best performed value of 0.0625 for the linear kernel[12]. In Figure 5.2, we show the effect of C on the testing accuracy of the SVM classifier. We can see that the accuracy is largely dependent on the value of the parameter C. The accuracy rises with the increase of the C value and reaches the peak when $C = 0.0625$. However, after that, an opposite trend is visible with the increase of the C value.

[11] We use 6 topics for training and 4 topics for testing.

[12] We found linear kernel as the best performer among all kernels.

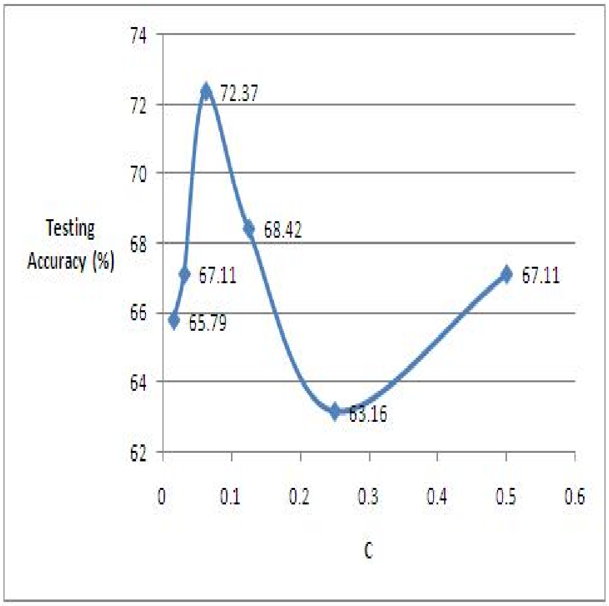


Figure 5.2: Effect of *C* on SVM's testing accuracy

5.6.3 Intrinsic Evaluation

Using the learned model, the supervised SVM classifier automatically predicts the labels (good or bad candidate) of the new set of decomposed questions that are generated for each new complex question. To evaluate the performance of our approach, we manually assess the quality of these questions. Two university graduate students judged the questions for linguistic quality and overall responsiveness following a similar setting to the DUC-2007 evaluation guidelines[13]. The given score is an integer between 1 (very poor) and 5 (very good) and is guided by consideration of the following factors: 1. Grammaticality, 2. Correct question type, 3. Referential clarity (Presence of pronoun), and 4. Meaningfulness. They also assigned a content responsiveness score to each question. This score is also an integer between 1 (very poor) and 5 (very good) and is based on the factor whether the question helps to satisfy the information need expressed in the original complex question. This task was performed intuitively by investigating the original complex question, the cor-

[13] http://www-nlpir.nist.gov/projects/duc/duc2007/quality-questions.txt

responding simple question at hand, and the given document collection. We compare the top-ranked good questions with the performance of a set of randomly picked (good/bad mixed) questions. For each topic, we also judge the performance of the bad questions alone[14]. Table 5.1 shows the evaluation results for SVM. Analyzing Table 5.1, we see that SVM predicted *Good Questions* improve the linguistic quality and responsiveness scores over *Mixed (Random) Questions* by 74.06%, and 100.00%, respectively whereas they outperform the Linguistic Quality and Responsiveness scores over *Bad Questions* by 20.58%, and 12.50%, respectively. These results suggest that the SVM classifier performed well to rank the decomposed questions accurately. We also see that *Bad Questions* outperform the *Mixed (Random) Questions* in terms of linguistic quality because they were small in length and had good grammatical structure. However, they could not beat the responsiveness scores meaning that smaller questions have limited coverage over the requested information need. We show some examples of good and bad decompositions generated by our system in Table 5.2.

Categories	Linguistic Quality	Responsiveness
Good Questions	4.10	3.60
Mixed (Random)	2.35	1.80
Bad Questions	3.40	1.60

Table 5.1: Linguistic quality and responsiveness scores (average) for SVM

In another evaluation setting, the two annotators judge the questions for their overall acceptability as a good or a bad candidate decomposition and assign a score (an integer between 1 (very poor) and 5 (very good)) to each of them. The outcome of this evaluation scale is then converted into a binary (+1/-1) rating scale, deciding that a score above 3 is positive. Then, the accuracy of the SVM's binary prediction is computed with respect to

[14] *Good questions* refer to the top 50% questions with high scores predicted by the SVM classifier, whereas *bad questions* refer to the bottom 50% questions.

Categories	Decompositions
Good	Who are rushing off their reservations? Where are native Americans relocating at a dizzying rate? Who left the reservation as a teen-ager in the mid-1970s?
Bad	Who was 43? Where is no funding available to help us? What said, “We got the worst of everything”?

Table 5.2: Examples of good and bad decompositions

the manual annotation using the following formula:

$$Accuracy = \frac{number\,of\,Correctly\,Classified\,Questions}{Total\,number\,of\,Test\,Questions} \tag{5.1}$$

We experimented with a total of 226 questions[15] for this evaluation and found 159 of them to be correctly classified by the SVM classifier showing an accuracy of 70.35%. An inter-annotator agreement of Cohen's $\kappa = 0.2835$ (Cohen, 1960) was computed that denotes a fair agreement (Landis and Koch, 1977) between the raters.

5.6.4 Extrinsic Evaluation

To determine the quality of the decomposed questions based on some other task such as summarization can be another effective means of evaluation. We call it as extrinsic evaluation. We pass the top-ranked decomposed questions to the Indri search engine (i.e. a component of the Lemur toolkit[16]). For all decomposed questions, Indri returns ranked sentences from the given data set that are used to generate a 250-word summary according to the DUC guidelines. We evaluate these summaries against four human-generated “reference summaries” (given in DUC-2006) using ROUGE (Lin, 2004) which has been widely

[15] We consider only the questions for which both annotators agreed on rating as positive or negative.

[16] Available at http://www.lemurproject.org/

adopted by DUC. We consider the widely used evaluation measures Precision (P), Recall (R) and F-measure for our evaluation task. Table 5.3 shows the ROUGE-2 and ROUGE-SU scores of our proposed SVM system as they are used as the official ROUGE metrics in the recent DUC evaluations.

Measures	Recall	Precision	F-score
ROUGE-2	0.0675	0.0570	0.0618
ROUGE-SU	0.1027	0.0734	0.0856

Table 5.3: ROUGE measures for SVM

Statistical Significance: To show a meaningful comparison, in Table 5.4, we present the average ROUGE-2 scores (Recall) of our SVM system and the two state-of-the-art systems, *Trimmer* and *HMM Hedge* that use a Multi-Candidate Reduction (MCR) framework for multi-document summarization (Zajic et al., 2007). These well-known systems do not perform question decomposition, so, we treat them as the *non-decomposed baselines* to perform system comparisons. An approximate result to identify which differences in the competing systems' scores are significant can be achieved by comparing the 95% confidence intervals for each mean. So, we include the 95% confidence intervals to report the statistical significance of our system. Table 5.4 yields that the SVM system outperforms the two baseline systems. We can also see that the confidence intervals of all the systems overlap with each other meaning the fact that there is no significant difference between our system and the baseline systems.

Systems	ROUGE-2
Trimmer	0.0671 [0.06332–0.07111]
HMM Hedge	0.0625 [0.05873–0.06620]
SVM	0.0675 [0.06548–0.07221]

Table 5.4: Comparison of different systems

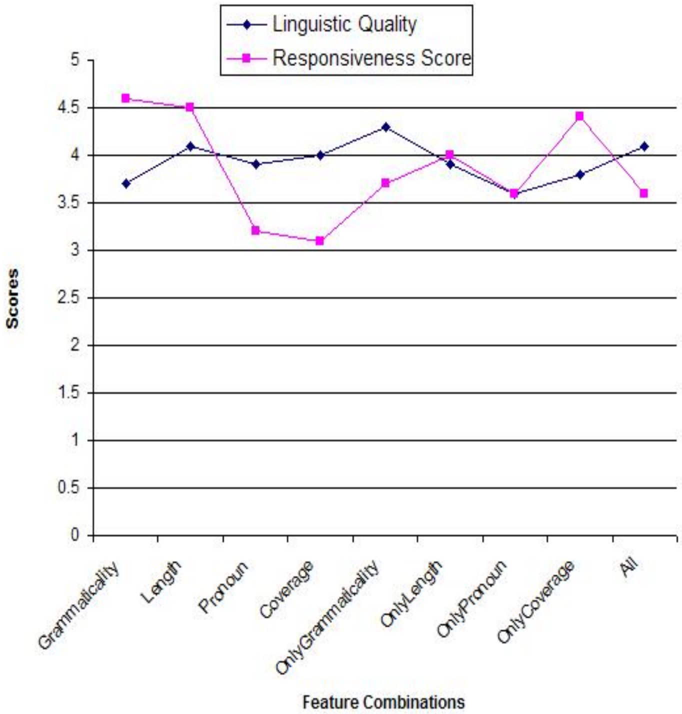


Figure 5.3: Graph for different feature combinations

5.6.5 Feature Engineering

To analyze the impact of different features, we run another experiment considering the SVM classifier generated top-ranked questions. In Figure 5.3, we plot a graph to show the performance of different systems considering several variants of the feature space during experiments. In the figure, **Grammaticality**, **Length**, **Pronoun**, and **Coverage** indicate that the corresponding feature is not considered during experiments, whereas **OnlyGrammaticality**, **OnlyLength**, **OnlyPronoun**, and **OnlyCoverage** denote the presence of that particular feature only. **All** denotes the inclusion of all features. From the figure, we understand that if we exclude the *Grammatically* feature, the responsiveness score improves quite a lot, whereas exclusion of the *Length* feature produces good scores for both linguistic quality and responsiveness. On the other hand, if we do not consider the *Pronoun* feature, the scores have a negative impact. Again, omitting the *Coverage* feature decreases the responsiveness score. On the other hand, if we consider *OnlyGrammaticality* feature, we have better linguistic quality and worse responsiveness score. *OnlyLength* feature yields good scores for both linguistic quality and responsiveness scores whereas *OnlyPronoun* feature provides bad scores for both denoting its lower impact on the SVM learner. *Only-Coverage* feature shows a higher responsiveness score with a moderate linguistic quality score and finally, considering *All* features yields a good linguistic quality while showing a decent performance in terms of responsiveness score. From this comparison of feature combinations, we can conclude that the inclusion of the *Pronoun*, and *Coverage* features helps to achieve the best performance for the considered task. This comparison also suggests that the chosen features are appropriate by themselves, but in combination are not able to produce a qualitative jump in performance.

5.7 Conclusion

In this chapter, we have proposed a supervised model for learning good decompositions of complex questions with the intention to improve the effectiveness of the question answering systems. During the training and the testing phases, we have used a collection of documents that contains the answers to the complex questions. We have extracted the most important sentences from these documents that could be the potential answer, then simplified the sentences and generated simple questions from these sentences. An SVM-based classification system has been developed to assess the quality of these simple questions. This is the first work to assess the quality of decompositions of complex questions. A series of evaluations have been carried out, which demonstrates the effectiveness of the proposed approach.

Chapter 6

Topic to Question Generation

6.1 Introduction

A complex question often asks about a topic of user's interest. Therefore, the problem of complex question decomposition (discussed in Chapter 5) closely relates to the problem of topic to question generation. This was our first motivation to address the challenge of automatically generating questions from topics (Chali and Hasan, 2012c) in order to enhance the scope of our problem domain (Chali and Hasan, 2012c).

The problem of topic to question generation is also important from another point of view. When a user is served with a ranked list of relevant documents by the standard document retrieval systems (i.e. search engines), his/her search task is usually not over (Chali, Joty, and Hasan, 2009). The next step for him/her is to look into the documents themselves and search for the precise piece of information he/she was looking for. This method is time consuming, and a correct answer could easily be missed, by either an incorrect query resulting in missing documents or by careless reading. This is why, QA-research has received immense attention from the information retrieval, information extraction, machine learning, and natural language processing communities (Kotov and Zhai, 2010). One of the main requirements of a QA system is that it must receive a well-formed question as input in order to come up with the best possible correct answer as output. Available studies revealed that humans are not very skilled in asking good questions about a topic of their interest. They are forgetful in nature which often restricts them to properly express whatever that is peeking in their mind. Therefore, they would benefit from automated Question Generation (QG) systems that can assist in meeting their inquiry needs (Olney, Graesser, and Person, 2012; Ali, Chali, and Hasan, 2010; Kotov and Zhai, 2010; Rus and Graesser, 2009; Lauer, Peacock, and Graesser, 1992; Graesser et al., 2001). Question asking and Question Generation are important components in advanced learning technologies such as intelligent tutoring systems, and inquiry-based environments (Graesser et al., 2001). A QG system would be useful for building better question asking facilities in intelligent tutoring systems. Another benefit of QG is that it can be a good tool to help improve the quality of the Question Answering (QA) systems (Graesser et al., 2001; Rus and Graesser, 2009).

The main motivation of this work is to generate all possible questions about a given topic. For example, given the topic *"Apple Inc. Logos"*, we can generate questions such as *"What is Apple Inc.?"*, *"Where is Apple Inc. located?"*, *"Who designed Apple's Logo?"* etc. We consider this task of automatically generating questions from topics and assume that each topic is associated with a body of texts having useful information about the topic. Our main goal is to generate fact-based questions[1] about a given topic from its associated content information. We generate questions by exploiting the named entity information and the predicate argument structures of the sentences (along with semantic roles) present in the given body of texts. The named entities and the semantic role labels are used to identify relevant parts of a sentence in order to form relevant questions over them. The importance of the generated questions is measured in two steps. In the first step, we identify whether the question is asking something about the topic or something that is very closely related to the topic. We call this the measure of *topic relevance*. For this purpose, we use Latent Dirichlet Allocation (LDA) (Blei, Ng, and Jordan, 2003) to identify the sub-topics (which are closely related to the original topic) in the given body of texts and apply the Extended String Subsequence Kernel (ESSK) (Hirao et al., 2003) to calculate their similarity with the questions. In the second step, we judge the syntactic correctness of each generated question. We apply the tree kernel functions (Collins and Duffy, 2001) and re-implement the syntactic tree kernel model according to Moschitti et al. (2007) for computing the syntactic

[1] We mainly focus on generating *Who, What, Where, Which, When, Why and How* questions in this research.

similarity of each question with the associated content information. We rank the questions by considering their topic relevance and syntactic correctness scores. Experimental results show the effectiveness of our approach for automatically generating topical questions. The next sections are organized as follows. Section 6.2 describes the related work followed by Section 6.3 that presents the description of our QG system. Section 3.4 explains the experiments and shows evaluation results.

6.2 Related Work

Recently, question generation has got immense attention from the researchers and hence, different methods have been proposed to accomplish the task in different relevant fields (Andrenucci and Sneiders, 2005). McGough et al. (2001) proposed an approach to build a web-based testing system with the facility of dynamic question generation. Wang, Tianyong, and Wenyin (2008) showed a method to automatically generate questions based on question templates (which are created from training on medical articles). Brown, Frishkoff, and Eskenazi (2005) described an approach to automatically generate questions to assess the user's vocabulary knowledge. To mimic the reader's self-questioning strategy during reading, Chen, Aist, and Mostow (2009) developed a method to generate questions automatically from informational text. On the other hand, Agarwal, Shah, and Mannem (2011) considered the question generation problem beyond sentence level and proposed an approach that uses discourse connectives to generate questions from a given text. Several other QG models have been proposed over the years that deal with transforming answers to questions and utilizing question generation as an intermediate step in the question answering process (Echihabi et al., 2003; Hickl et al., 2005). There are some other researchers who have approached the task of generating questions for educational purposes (Mitkov and Ha, 2003; Heilman and Smith, 2010b).

The Natural Language Processing (NLP), Natural Language Generation (NLG), Intelligent Tutoring System, and Information Retrieval (IR) communities have currently identified the Text-to-Question generation task as promising candidates for shared tasks[2] (Rus and Graesser, 2009; Boyer and Piwek, 2010). In the Text-to-Question generation task, a QG system is given a text, and the goal is to generate a set of questions for which the text contains answers. The task of generating a question about a given text can be typically decomposed into three subtasks. First, given the source text, a content selection step is necessary to select a target to ask about, such as the desired answer. Second, given a target answer, an appropriate question type is selected, i.e., the form of question to ask is determined. Third, given the content, and question type, the actual question is constructed. Based on this principle, several approaches have been described in Boyer and Piwek (2010) that use named entity information, syntactic knowledge and semantic structures of the sentences to perform the task of generating questions from sentences and paragraphs (Heilman and Smith, 2010a; Mannem, Prasad, and Joshi., 2010). Inspired by these works, we perform the task of topic to question generation using named entity information and semantic structures of the sentences. A task that is similar to ours is the task of keywords to question generation that has been addressed recently in Zheng et al. (2011). They propose a user model for jointly generating keywords and questions. However, their approach is based on generating question templates from existing questions which requires a large set of English questions as training data. In recent years, some other related researches have proposed the tasks of high quality question generation (Ignatova, Bernhard, and Gurevych, 2008) and generating questions from queries (Lin, Weng, and Keerthi, 2008). Fact-based question generation has been accomplished previously by Rus, Cai, and Graesser (2007), and Heilman and Smith (2010b). We also focus on generating fact-based questions in this research.

[2] http://www.questiongeneration.org/QGSTEC2010

Besides grammaticality, an effective QG system should focus deeply on the importance of the generated questions (Vanderwende, 2008). This motivates the use of a question ranking module in a typical QG system. Over-generated questions can be ranked using different approaches such as statistical ranking methods, dependency parsing, identifying the presence of pronouns and named entities, and topic scoring (Heilman and Smith, 2010a; Mannem, Prasad, and Joshi., 2010; McConnell et al., 2011). However, most of these automatic ranking approaches ignore the aspects of complex paraphrasing by not considering lexical semantic variations (e.g. synonymy) while measuring the importance of the questions. In our work, we use Latent Dirichlet Allocation (LDA) (Blei, Ng, and Jordan, 2003) to identify the sub-topics (which are closely related to the original topic) in the given body of texts. In recent years, LDA has become one of the most popular topic modeling techniques and has been shown to be effective in several text-related tasks such as document classification, information retrieval, and question answering (Misra, Cappé, and Yvon, 2008; Wei and Croft, 2006; Celikyilmaz, Hakkani-Tur, and Tur, 2010). Hirao et al. (2003) introduced ESSK considering all possible senses to each word to perform their summarization task. Their method is effective. However, the fact that they do not disambiguate word senses cannot be disregarded. In our task, we apply ESSK to calculate the similarity between important topics (discovered using LDA) and the generated questions in order to measure the importance of each question. We use disambiguated word senses for this purpose.

Syntactic information has been used successfully in *question answering* previously (Chali, Hasan, and Joty, 2009; Chali, Hasan, and Joty, 2011; Zhang and Lee, 2003; Moschitti et al., 2007; Moschitti and Basili, 2006). Pasca and Harabagiu (2001) argued that with the syntactic form of a sentence one can see which words depend on other words. We also feel that there should be a similarity between the words which are dependent in the sentences present in the associated body of texts and the dependency between words of the generated question. This motivates us to propose the use of syntactic kernels in judging the syntactic correctness of the generated questions automatically.

The main goal of our work is to generate as many questions as possible related to the topic. We use NE information and the predicate argument structures of the sentences to accomplish this goal. Our approach is different from the setup in shared tasks (Rus and Graesser, 2009; Boyer and Piwek, 2010) as we generate a set of basic questions which are useful to add variety in the question space. A paragraph associated with each topic is used as the source of relevant information about the topic. We evaluate our systems in terms of topic relevance which is different from the prior works (Heilman and Smith, 2010a; Mannem, Prasad, and Joshi., 2010). Syntactic correctness is also an important property of a good question. For this reason, we evaluate our system in terms of syntactic correctness as well. The proposed system will be useful to generate topic related questions from the associated content information which can be used to incorporate a "question suggestions for a certain topic" facility in the search systems. For example, if a user searches for some information related to a certain topic, the search system could generate all possible topic-relevant questions from a preexistent related body of texts to provide suggestions. Kotov and Zhai (2010) approached a similar task by proposing a technique to augment the standard ranked list presentation of search results with a question based interface to refine user given queries.

The major contributions of this research can be summarized as follows:

- We perform the task of topic to question generation which can help users in expressing their information needs. Questions are generated using a set of general-purpose rules based on named entity information and the predicate argument structures of the sentences (along with semantic roles) present in the associated body of texts.

- We use LDA to identify the sub-topics (which are closely related to the original topic) in the given body of texts and apply ESSK (with disambiguated word senses) to calculate their similarity with the questions. This helps us to measure the importance of each question.

- We apply the tree kernel functions and re-implement the syntactic tree kernel model for computing the syntactic similarity of each question with the associated content information. In this way, we judge the syntactic correctness of each generated question automatically.
- The ESSK similarity scores and the syntactic similarity scores are used to rank the generated questions. In doing so, we show that the use of ESSK and syntactic kernels improve the relevance and the syntactic correctness of the top-ranked questions, respectively.
- We also run experiments by narrowing down the topic focus. Experiments with the topics about persons (biographical focus) reveal improvements in the overall results.

6.3 Topic to Question Generation

Our QG approach mainly builds on four steps. In the first step, complex sentences (from the given body of texts) related to a topic are simplified as it is easier to generate questions from simple sentences. In the next step, named entity information and predicate argument structures of the sentences are extracted and then, questions are generated using them. In the third step, LDA is used to identify important sub-topics from the given body of texts and then ESSK is applied to find their similarity with the generated questions. In the final step, syntactic tree kernel is employed and syntactic similarity between the generated questions and the sentences present in the body of texts determines the syntactic correctness of the questions. Questions are then ranked by considering the ESSK similarity scores and the syntactic similarity scores. We describe the overall procedure in the following subsections.

6.3.1 Sentence Simplification

Sentences may have complex grammatical structure with multiple embedded clauses. Therefore, we simplify the complex sentences with the intention to generate more accurate questions. We use the simplified factual statement extractor model[3] of Heilman and Smith (2010a). Their model extracts the simpler forms of the complex source sentence by altering lexical items, syntactic structure, and semantics and by removing phrase types such as leading conjunctions, sentence-level modifying phrases, and appositives. For example, given a complex sentence *s*, we get a corresponding simple sentence as follows:

Complex Sentence (s): Apple's first logo, designed by Jobs and Wayne, depicts Sir Isaac Newton sitting under an apple tree.

Simple Sentence: Apple's first logo is designed by Jobs and Wayne.

6.3.2 Named Entity (NE) Information and Semantic Role Labeling (SRL) for QG

We use the Illinois Named Entity Tagger[4], a state of the art NE tagger that tags a plain text with named entities (people, organizations, locations, miscellaneous) (Ratinov and Roth, 2009). Once we tag the topic in consideration and its associated body of texts, we use some general purpose rules to create some basic questions even though the answer is not present in the body of texts. For example, "Apple Inc." is tagged as an organization, so we generate a question: "Where is Apple Inc. located?". The main motivation behind generating such questions is to add variety to the generated question space. Table 6.1 shows some example rules for basic questions generated in this work.

[3] Available at http://www.ark.cs.cmu.edu/mheilman/

[4] Available at http://cogcomp.cs.illinois.edu/

Tag	Example Question
person	Who is *person*?
organization	Where is *organization* located?
location	Where is *location*?
misc.	What do you know about *misc.*?

Table 6.1: Example basic question rules

Our next task is to generate specific questions from the sentences present in the given body of texts. For this purpose, we parse the sentences semantically using a Semantic Role Labeling (SRL) system (Kingsbury and Palmer, 2002; Hacioglu et al., 2003), ASSERT[5]. ASSERT is an automatic statistical semantic role tagger, that can annotate naturally occuring text with semantic arguments. When presented with a sentence, it performs a full syntactic analysis of the sentence, automatically identifies all the verb predicates in that sentence, extracts features for all constituents in the parse tree relative to the predicate, and identifies and tags the constituents with the appropriate semantic arguments. For example, the output of the SRL system for the sentence “Apple’s first logo is designed by Jobs and Wayne.” is: [ARG1 Apple ’s first logo] is [TARGET designed] [ARG0 by Jobs and Wayne]. The output contains one verb (predicate) with its arguments (i.e. semantic roles). These arguments are used to generate specific questions from the sentences. For example, we can replace *[ARG1 ..]* with *What* and generate a question as: “What is designed by Jobs and Wayne?”. Similarly, *[ARG0 ..]* can be replaced and the question: “Who designed Apple’s first logo?” can be generated. The semantic roles *ARG0...ARG5* are called *mandatory arguments*. There are some additional arguments or semantic roles that can be tagged by ASSERT. They are called *optional arguments* and they start with the prefix *ARGM*. These are defined by the annotation guidelines set in (Palmer, Gildea, and Kingsbury, 2005). A set of about 350 general purpose rules are used to transform the semantic-role labeled sentences into the questions. The rules were set up in a way that we could use the semantic role

[5] Available at http://cemantix.org/assert.html

information to find the potential answer words in a sentence which would be replaced by suitable question words. In case of a mandatory argument, the choice of question word depends on the argument's named entity tag (e.g. "Who" for a person, "Where" for a location etc.). Table 6.2 shows how different semantic roles can be replaced by possible question words in order to generate a question.

Arguments	Question Words
ARG0...ARG5	Who, Where, What, Which
ARGM-ADV	In what circumstances
ARGM-CAU	Why
ARGM-DIS	How
AGRM-EXT	To what extent
ARGM-LOC	Where
ARGM-MNR	How
ARGM-PNC	Why
ARGM-TMP	When

Table 6.2: Semantic roles with possible question words

6.3.3 Importance of Generated Questions

Latent Dirichlet Allocation (LDA): To measure the importance of the generated questions, we use LDA (Blei, Ng, and Jordan, 2003) to identify the important sub-topics from the given body of texts. LDA is a probabilistic topic modeling technique where the main principle is to view each document as a mixture of various topics. Here each topic is a probability distribution over words. LDA assumes that documents are made up of words and word ordering is not important ("bag-of-words" assumption) (Misra, Cappé, and Yvon, 2008). The main idea is to choose a distribution over topics while generating a new document. For each word in the new document, a topic is randomly chosen according to this distribution and a word is drawn from that topic. LDA uses a generative topic modeling approach to specify the following distribution over words within a document:

$$P(w_i) = \sum_{j=1}^{K} P(w_i|z_i = j)P(z_i = j) \qquad (6.1)$$

where K is the number of topics, $P(w_i|z_i = j)$ is the probability of word w_i under topic j and $P(z_i = j)$ is the sampling probability of topic j for the i^{th} word. The multinomial distributions $\phi^{(j)} = P(w|z_i = j)$ and $\theta^{(d)} = P(z)$ are termed as topic-word distribution and document-topic distribution, respectively (Blei, Ng, and Jordan, 2003). A Dirichlet (α) prior is placed on θ and a Dirichlet (β) prior is set on ϕ to refine this basic model (Blei, Ng, and Jordan, 2003; Griffiths and Steyvers, 2002). Now the main goal is to estimate the two parameters: θ and ϕ. We apply this framework directly to solve our problem by considering each topic-related body of texts as a document. We use a GUI-based toolkit for topic modeling[6] that uses the popular MALLET (McCallum, 2002) toolkit for the back-end. The process starts by removing a list of "stop words" from the document and runs 200 iterations of Gibbs sampling (Geman and Geman, 1984) to estimate the parameters: θ and ϕ. From each body of texts, we discover K topics and choose the most frequent words from the most likely unigrams as the desired sub-topics. For example, from the associated body of texts of the topic *Apple Inc. Logos*, we get these sub-topics: *janoff, themes, logo, color, apple*.

Extended String Subsequence Kernel (ESSK): Once we identify the sub-topics, we apply ESSK to measure their similarity with the generated questions. ESSK is used to measure the similarity between all possible subsequences of the question words/senses and topic words/senses. We calculate the similarity score Sim(T_i, Q_j) using ESSK where T_i denotes a topic/sub-topic word sequence and Q_j stands for a generated question. Formal definition of ESSK is presented in Section 3.3.2.

[6] Available at http://code.google.com/p/topic-modeling-tool/

6.3.4 Judging Syntactic Correctness

The generated questions might be syntactically incorrect due to the process of automatic question generation. It is time consuming and a lot of human intervention is necessary to check for the syntactically incorrect questions manually. We strongly believe that a question should have a similar syntactic structure to a sentence from which it is generated. For example, the sentence “Apple’s first logo is designed by Jobs and Wayne.”, and the generated question “What is designed by Jobs and Wayne?” are syntactically similar. Hence, to judge the syntactic correctness of each generated question automatically, we apply the tree kernel functions and re-implement the syntactic tree kernel model for computing the syntactic similarity of each question with the associated content information. We first parse the sentences and the questions into syntactic trees using the Charniak parser[7] (Charniak, 1999). Then we calculate the similarity between the two corresponding trees using the *tree kernel* method (Collins and Duffy, 2001). We convert each parenthetic representation generated by the Charniak parser into its corresponding tree and give the trees as input to the tree kernel functions for measuring the syntactic similarity. Details of this process is described in Section 4.4.2.

The tree kernel (TK) function computes the number of common subtrees between two trees. Such subtrees are subject to the constraint that their nodes are taken with all or none of the children they have in the original tree. The *TK* function gives the similarity score between each sentence in the given body of texts and the generated question based on the syntactic structure. Each sentence[8] contributes a score to the questions and then the questions are ranked by considering the average of similarity scores.

[7] Available at ftp://ftp.cs.brown.edu/pub/nlparser/

[8] We consider that a question is syntactically fluent as well as relevant to the topic if it has similar syntactic sub-trees as those of the most sentences in the body of texts.

6.4 Experiments

6.4.1 System Description

We consider the task of automatically generating questions from topics where each topic is associated with a body of texts having a useful description about the topic. The proposed QG system ranks the questions by combining the topic relevance scores and the syntactic similarity scores of Section 3.3.1 and Section 6.3.4 using the formula as follows:

$$w * ESSK_{score} + (1 - w) * SYN_{score} \tag{6.2}$$

Here w is the importance parameter which holds the value in $[0, 1]$. We kept $w = 0.5$ to give equal importance[9] to topic relevance and syntactic correctness.

6.4.2 Corpus

To run our experiments, we use the dataset provided in the Question Generation Shared Task and Evaluation Challenge[10] (QGSTEC, 2010) for the task of question generation from paragraphs. This dataset consists of 60 paragraphs about 60 topics that were originally collected from several Wikipedia, OpenLearn, and Yahoo!Answers articles. The paragraphs contain around $5 - 7$ sentences for a total of $100 - 200$ tokens (including punctuation). This dataset includes a diversity of topics of general interest. We consider these topics and treat the paragraphs as their associated useful content information in order to generate a set of questions using our proposed QG approach. We use 10 topics and their associated

[9] A syntactically incorrect question is not useful even if it is relevant to the topic. This motivated us to give equal importance to topic relevance and syntactic correctness. The parameter w can be tuned to investigate its impact on the system performance.

[10] http://www.questiongeneration.org/mediawiki

paragraphs as the development data[11]. A total of 2186 questions are generated from the remaining 50 topics (test data) to be ranked.

6.4.3 Evaluation Setup

Methodology: We use a methodology derived from Boyer and Piwek (2010), and Heilman and Smith (2010b) to evaluate the performance of our QG systems. Three native English-speaking university graduate students judge[12] the quality of the top-ranked 20% questions using two criteria: topic relevance and syntactic correctness. For topic relevance, the given score is an integer between 1 (very poor) and 5 (very good) and is guided by the consideration of the following aspects: 1. Semantic correctness (i.e. the question is meaningful and related to the topic), 2. Correctness of question type (i.e. a correct question word is used), and 3. Referential clarity (i.e. it is clearly possible to understand what the question refers to). For syntactic correctness, the assigned score is also an integer between 1 (very poor) and 5 (very good). Whether a question is grammatically correct or not is checked here. For each question, we calculate the average of the judges' scores.

Systems for Comparison: We report the performance of the following systems in order to do a meaningful comparison with our proposed QG system:

(1) Baseline1: This is our QG system without any question-ranking method applied to it. Here, we randomly select 20% questions and rate them.

(2) Baseline2: For our second baseline, we build a QG system using an alternative topic modeling approach. Here we use a topic signature model (instead of using LDA as discussed in Section 6.3.3) (Lin and Hovy, 2000) to identify the important sub-topics from

[11] We use this data to build necessary general purpose rules for our QG model.

[12] The inter-annotator agreement of Fleiss' $\kappa = 0.41, 0.45, 0.62$, and 0.33 are computed for the three judges for the results in Table 6.3 to Table 6.6, indicating moderate (for the first two tables), substantial and fair agreement (Landis and Koch, 1977) between the raters, respectively.

the sentences present in the body of texts. The sub-topics are the important words in the context which are closely related to the topic and have significantly greater probability of occurring in the given text compared to that in a large background corpus. We use a topic signature computation tool[13] for this purpose. The background corpus that is used in this tool contains 5000 documents from the English GigaWord Corpus. For example, from the given body of texts of the topic *Apple Inc. Logos*, we get these sub-topics: *jobs, logo, themes, rainbow, monochromatic*. Then we use the same steps of Section 6.3.3 and Section 6.3.4, and use equation 6.2 to combine the scores. We evaluate the top-ranked 20% questions and show the results.

(3) State-of-the-art: We choose a publicly available state-of-the-art QG system[14] to generate questions from the sentences in the body of texts. This system was shown to achieve good performance in generating fact-based questions about the content of a given article (Heilman and Smith, 2010b). Their method ranks the questions automatically using a logistic regression model. Given a paragraph as input, this system processes each sentence and generates a set of ranked questions for the entire paragraph. We evaluate the top-ranked 20% questions[15] and report the results.

Results and Discussion: Table 6.3 shows the average topic relevance and syntactic correctness scores for all the systems. From these results we can see that the *proposed QG* system improves the topic relevance and syntactic correctness scores over the *Baseline1* system by 61.86%, and 34.98%, respectively, and improves the topic relevance and syntactic correctness scores over the *Baseline2* system by 7.40%, and 7.57%, respectively. On the other hand, the *proposed QG* system improves the topic relevance and syntactic correctness scores over the *state-of-the-art* system by 3.88%, and 2.89%, respectively. From these results, we can clearly observe the effectiveness of our proposed QG system. The

[13] Available at http://www.cis.upenn.edu/~lannie/topicS.html

[14] Available at http://www.ark.cs.cmu.edu/mheilman/questions/

[15] We ignore the yes-no questions for our task.

improvements in the results are statistically significant[16] ($p < 0.05$).

The main goal of this work was to generate as many questions as possible related to the topic. For this reason, we considered generating the basic questions. These questions were also useful to provide variety in the question space. We generated these questions using the NE information. As the performance of the NE-taggers were not that great, we had a few of these questions generated. In most cases, these questions were outranked by other important questions that included a combination of topics and sub-topics to show higher topic relevance score measured by ESSK. Therefore, they do not have a considerable impact on the evaluation statistics. We claim that the overall performance of our systems could be further improved if the accuracy of the NE-tagger and the semantic role labeler could be increased.

Systems	Topic Relevance	Syntactic Correctness
Baseline1 (No Ranking)	2.15	2.63
Baseline2 (Topic Signature)	3.24	3.30
State-of-the-art (Heilman and Smith, 2010b)	3.35	3.45
Proposed QG System	3.48	3.55

Table 6.3: Topic relevance and syntactic correctness scores

Acceptability Test: In another evaluation setting, the three annotators judge the questions for their overall acceptability as a good question. If a question shows no deficiency in terms of the criteria considered for topic relevance and syntactic correctness, it is termed as *acceptable*. We evaluate the top 15% and top 30% questions separately for each QG system and report the results indicating the percentage of questions rated as acceptable in Table 6.4. The results indicate that the percentage of the questions rated acceptable is reduced when we evaluate more number of questions which proves the effectiveness of our QG system.

[16] We tested statistical significance using Student's t-test.

Systems	Top 15%	Top 30%
Baseline1 (No Ranking)	35.2	32.6
Baseline2 (Topic Signature)	45.9	33.8
State-of-the-art (Heilman and Smith, 2010b)	44.7	38.5
Proposed QG System	46.5	40.6

Table 6.4: Acceptability of the questions (in %)

Systems	Topic Relevance	Syntactic Correctness
Baseline1 (No Ranking)	3.20	3.54
Baseline2 (Topic Signature)	3.80	3.92
State-of-the-art (Heilman and Smith, 2010b)	4.01	4.15
Proposed QG System	4.12	4.25

Table 6.5: Topic relevance and syntactic correctness scores (narrowed focus)

Systems	Top 15%	Top 30%
Baseline1 (No Ranking)	41.3	37.1
Baseline2 (Topic Signature)	53.5	43.6
State-of-the-art (Heilman and Smith, 2010b)	57.5	43.2
Proposed QG System	58.4	44.5

Table 6.6: Acceptability of the questions in % (narrowed focus)

Systems	Top-ranked questions
Baseline2	Who presented Jobs with several different monochromatic themes for the bitten logo? What were conceived to make the logo more accessible? Who liked the logo?
State-of-the-art	Whose first logo depicts Sir Isaac Newton sitting under an apple tree? What depicts Sir Isaac Newton sitting under an apple tree? What did Janoff present Jobs with?
Proposed QG System	Who designed Apple's first logo? What was replaced by Rob Janoff 's "rainbow Apple"? What were conceived to make the logo more accessible?

Table 6.7: System output

Narrowing Down the Focus: We run further experiments by narrowing down the topic focus. We consider only the topics about persons (biographical focus). We choose 10 persons as our topics from the list of the 20th century's 100 most influential people, published in Time magazine in 1999 and obtained the paragraphs containing their biographical information from Wikipedia articles[17]. We generate a total of 390 questions from the considered 10 topics and rank them using different ranking schemes as discussed before. We evaluate the top 20% questions using the similar evaluation methodologies and report the results in Table 6.5. Again, we evaluate the top 15% and top 30% questions separately for each QG system and report the results indicating the percentage of questions rated as acceptable in Table 6.6. From these tables, we can clearly see the improvements in all the scores for all the QG approaches. This is reasonable because the accuracy of the NE tagger and the semantic role labeler is increased for the biographical data. These results further demonstrate that the proposed system is significantly better (at $p < 0.05$) than the other considered systems.

An Input-Output Example: An input to our systems is for instance, the topic *"Apple Inc. Logos"* with the associated content information (body of texts): *"Apple's first logo, designed by Jobs and Wayne, depicts Sir Isaac Newton sitting under an apple tree. Almost immediately, though, this was replaced by Rob Janoff's "rainbow Apple", the now-familiar*

[17] http://en.wikipedia.org/wiki/Time_100

rainbow-colored silhouette of an apple with a bite taken out of it. Janoff presented Jobs with several different monochromatic themes for the "bitten" logo, and Jobs immediately took a liking to it. While Jobs liked the logo, he insisted it be in color to humanize the company. The Apple logo was designed with a bite so that it would be recognized as an apple rather than a cherry. The colored stripes were conceived to make the logo more accessible, and to represent the fact the monitor could reproduce images in color. In 1998, with the roll-out of the new iMac, Apple discontinued the rainbow theme and began to use monochromatic themes, nearly identical in shape to its previous rainbow incarnation."
The output of our systems is the ranked lists of questions. We show an example output in Table 6.7.

6.5 Conclusion

In this chapter, we have presented a novel method of automatically generating questions from topics where each topic is associated with a body of texts containing useful information. A central aspect of the proposed system is the use of LDA for topic modeling with the goal of exploiting topic information to improve and rank results. We have also proposed a methodology of using syntactic tree kernels to automatically compute the syntactic correctness of generated questions. Extensive evaluations have demonstrated that our system performs considerably better than an state-of-the-art system.

Chapter 7

Conclusion

In this thesis, we have presented novel ideas and frameworks for answering complex questions. The fundamental focus was to address several issues of the complex question answering problem and minimize the existing gaps in the literature. We define the complex questions as the kind of questions whose answers need to be obtained from pieces of information scattered in different documents. Our experiments and evaluations were mainly influenced by the specific scenario proposed by the DUC (2005-2007) tasks. In fact, DUC proposes a query-focused summarization task whose features have allowed us to simulate our experiments with complex question answering. Hence, the considered complex questions are the type of questions that request information such as an elaboration about a topic, description about an event or entity, illustration about an opinion, and definition or discussion about an aspect or term or procedure.

Effective complex question answering can aid to the improvement of the search systems. When a user searches for some information, the traditional search engines usually offer a listing of sources through which the user has to continue navigating until the desired information need is satisfied. Moreover, search engines lack a way of measuring the level of user satisfaction which could have been used to enhance the search policy in real time. A reinforcement learning methodology is suitable and appropriate to address these issues, as shown in Chapter 2. The main motivation behind mapping the complex question answering task to a reinforcement learning framework was to enable real-time learning by treating the task as an interactive problem where user feedback can be added as a reward. Initially, we have simplified this assumption by not interacting with the users directly rather we have employed the human-generated abstract summaries to provide a small amount of supervision using reward scores through textual similarity measurement. Later, we have extended the model by incorporating a user interaction component to guide the candidate sentence selection process during the reinforcement learning phase. We have compared our reinforcement system with a baseline, a supervised (SVM) system, and an unsupervised (K-means) system. Extensive evaluations on the DUC benchmark data sets have showed the effectiveness of the proposed approach. Furthermore, experiments with the user interaction component have revealed that the systems trained with user interaction perform better than that of having no interaction. The evaluations have also showed that the reinforcement system is able to learn automatically (i.e. without interaction) and effectively after a sufficient amount of user interaction is provided as the guide to candidate answer sentence selection.

The implications of the proposed reinforcement learning approach for the search systems can be observed in terms of the user modeling component, which helps the learning framework to refine the previous answers to complex questions based on user feedback. In Chapter 2, we have modeled user interaction during the candidate sentence selection process of the learning stage. However, another general scenario of modeling user interaction into a QA system could be as follows: a user would submit a question to the QA system and receive an extract summary of a list of relevant documents (obtained by a Web search engine) as the answer to the question. The user would then be asked to give a rating about his satisfaction which could be encoded as a reward in the reinforcement learning approach. The current answer to the complex question would be refined accordingly and the same process needs to be followed until the satisfaction level reaches to the maximum. This process is definitely time consuming, however, once the system receives a considerable amount of feedback about several complex questions, a reinforcement learning system could learn about the user's interests, and choices from this data. Then, the learned model could be used to answer unseen complex questions efficiently. We would like to accomplish this research in the future. In our proposed reinforcement learning formulation, we kept the value of ε static through out the weight learning phase to denote a fixed probability of exploration. In the future, we will experiment on tuning the value of ε where we will start with a high value to ensure a greater amount of exploration and less exploitation while gradually decreasing ε to reduce exploration as time passes. In our formulation, we have used ROUGE as a reward function to provide feedback to each chosen action. ROUGE is a measure to count word (or word sequence) overlap that can lead to inaccurate textual similarity computation. We believe that the performance of our reinforcement learning framework could improve if we can build a reward function to capture syntactic and semantic properties of the texts. We plan to explore this research by using different textual similarity measurement techniques such as Basic Element (BE) overlap (Hovy et al., 2006), syntactic similarity measure (Moschitti and Basili, 2006), semantic similarity measure (Moschitti et al., 2007), and Extended String Subsequence Kernel (ESSK) (Hirao et al., 2003) as the reward functions.

An ideal summary or answer to the complex question should satisfy the user requested information need as a whole. As noted before, extraction-based summarization needs to undertake complex analysis and synthesis of information as because important pieces of information might be scattered across multiple documents in the collection. This process might yield to redundancy of information, which is specially true for the newswire domain since frequent reference to a previously occurred event is a common practice among the journalists. Note that, in this thesis, we run our complex question answering experiments using the newswire corpora of DUC (2006-2007) and TAC 2010. Most of the extractive summarization approaches (such as our reinforcement learning framework of Chapter 2) try to judge individual sentences of the document using some criteria and do not focus on the overall quality of the summary. It is indeed a difficult task to conclude what combination of different salient sentences (to form a summary) of the document can satisfy the user the most. Essentially, what we are looking for is the optimum summary that was not possible to achieve in Chapter 2. Integer Linear Programming (ILP) techniques can provide an efficient solution to this problem, as discussed in Chapter 3. There we have formulated the complex question answering task in terms of integer linear programming. To address the problem of redundancy in the summary while allowing more important information, we have proposed to use ILP-based sentence compression models. As there was no previous research that analyzed the effectiveness of using different sentence compression models for the task of query-focused multi-document summarization inside an ILP framework, our experiments were designed to minimize this gap in the literature.

Our empirical evaluation on the DUC-2007 dataset have suggested that the semantically motivated sentence compression models can enhance the overall summarization performance in presence of the semantic redundancy constraint in the summarization model and this can be achieved irrespective of the compression and extraction order followed during the process. Our results have also demonstrated that a combined optimization framework of compression and extraction can achieve better performance than the pipeline-based approaches effectively. We have also found that the *SumFirst* approach shows superior performance to that of the *ComFirst* approach suggesting the fact that extracting the most important sentences before compression is a more effective way of summarization. In this research, we have used different textual similarity measurement techniques as the redundancy constraints of the ILP-based summarization framework and performed an extensive experimental evaluation to show their impact on the overall summarization performance. Experimental results have showed that the use of semantic similarity measure as the $Sim(i, j)$ function in the redundancy constraint yields the best performance. Overall, our global optimization frameworks have showed promising performance with respect to the state-of-the-art systems. In the future, we intend to apply our integer linear programming approach for answering complex questions to other available datasets of DUC-2005 and DUC-2006. The findings should hold for these datasets as well as for other genres of datasets since we believe that our ILP-based compression and summarization models could be tuned to fit them. We also plan to use other automatic measures (Saggion et al., 2010; Pitler, Louis, and Nenkova, 2010) to evaluate our approach.

The outcome of the research in Chapter 3 is the knowledge that ILP-based compression and summarization models can essentially help in generating better optimal summaries. This fact can be utilized by inducing the concept of optimality into our reinforcement learning framework (of Chapter 2), which is seemed to be a more appropriate method for the complex question answering task. We can construct a reward function like Ryang and Abekawa (2012) that would compute the overall reward of a candidate summary after a set of sentences has been selected according to a predefined limit (i.e. 250 words for our experiments). In that case, we need to form a feature space that would capture the properties of the summary as a whole. We can also modify the action space of the reinforcement learning framework by exploiting an ILP-based sentence compression model, which would provide the best reduction of a selected sentence (i.e. optimal) before its inclusion into the summary space.

Extraction-based summarization deeply relies on methods to identify the most important sentences in the document collection. A sentence can be deemed important if it is found relevant to the given complex question. This task could be accomplished by considering a feature space (or a relevance function) as shown in Chapter 2 and Chapter 3. However, there might be a case where a sentence is not found to be relevant to the question rather it is mostly relevant to other important sentences in the document. Therefore, similarity computation among the sentences of the document collection is considered a key to effective extraction-based summarization. Two sentences in the document collection might not necessarily have common words, but, there might be still some latent relationship between them. Such asymmetry and transitivity among the sentences are resolved by using a graph-based random walk model for summarization (Erkan, 2007). The traditional graph-based random walk models use basic similarity functions such as cosine measure in order to assess similarity between the sentences, where syntactic and semantic properties of the texts are ignored. This has motivated us in Chapter 4 to propose the use of syntactic and semantic structures in measuring the similarity between the sentences in the graph-based random walk framework for answering complex questions. Our experiments have suggested that: (a) similarity measures based on the syntactic tree and/or shallow semantic tree and Extended String Subsequence Kernel (ESSK) outperform the similarity measures based on the cosine measures i.e. TF*IDF and (b) similarity measures based on the shallow semantic tree perform the best for this problem. In the future, we plan to experiment with a shallow syntactic sentence similarity measurement approach like Basic Elements (BE) to check how it performs on the graph-based random walk model for the task of complex question answering.

In Chapter 4, we have also proposed a novel methodology that uses a predefined list of important aspects (corresponding to the complex question) in a random walk framework by performing a deeper semantic analysis of the source documents instead of relying only on document word frequencies to select important concepts. Evaluations on the DUC-2006 and TAC-2010 data have indicated that augmenting the important aspects into the random walk model considerably outperforms the random walk model if used alone. Experiments have also suggested the fact that the aspects can provide a certain amount of supervision to cover all the relevant perspectives of a topic (or a complex question) and hence, the use of it with any sophisticated model such as random walk can enhance the model's performance substantially in comparison to the model if used alone. The proposed methods in Chapter 4 are very general and can be used effectively in our reinforcement learning framework and ILP model in order to rank the sentences in the document collection based on a given complex question or to provide some useful guidance with a list of predefined aspects to enable a better understanding of the main focus of the question. We would like to explore this research in the future.

As stated before, the research described in this thesis concerns the answering of complex questions that are in reality broader information requests about a certain topic. Chapter 4 has shown how we can utilize a human-made predefined list of aspects to guide the important sentence selection process. However, it is not a feasible task to create a representative list of aspects for all possible topics of the world. We relate this problem with complex question decomposition in Chapter 5, where we have presented a Support Vector Machine (SVM)-based supervised model for automatically learning good decompositions of complex questions that can in turn act as a useful guide for summary sentence selection. The main idea that has been proposed is to look at a training set of documents that are known to possess the answer to the complex question, to extract key sentences for the question, to simplify these sentences and to rephrase them as questions. We have performed a rigorous evaluation and analysis to show the effectiveness of our approach. Furthermore, we have analyzed the impact of different features on the performance of the SVM classifier and concluded that the combination of the *Pronoun*, and *Coverage* features yields the best performance. We plan to use more sophisticated features in the future in order to learn good decompositions of complex questions and investigate the potentials of other supervised machine learning techniques such as Conditional Random Fields (CRF), and Hidden Markov Models (HMM) for the learning task.

Our evaluation framework for learning good decompositions of complex questions looks at the fact if one simple question is a good decomposition of one complex question. It would be interesting to use a modified evaluation methodology where we could find whether a set of simple questions is a proper decomposition of the considered complex question. Another drawback of the proposed approach is that the supervised approach might result to overfitting. For example, before the tsunami disaster in Japan, nuclear leak was not yet included in the previous tsunami. In this case, the previous learned model might not deal with Japan's tsunami well. This issue can be resolved if we incorporate this model into our reinforcement learning framework, where real time user feedback can essentially aid in upgrading the search policy of the learner. The proposed method in Chapter 5 can automatically generate important aspects of a topic, which can be encoded into our other frameworks in order to facilitate narrowing down the focus of the complex question.

The problem of complex question decomposition can be generalized to the problem of topic to question generation, as shown in Chapter 6. The benefit of this generalization is to enable generating more topic relevant questions that would aid in satisfying wider information requests. We have considered the task of automatically generating questions from topics where each topic is associated with a body of texts containing useful information. The proposed method has exploited the named entity and semantic role labeling information to accomplish the task. A key aspect of our approach was the use of latent Dirichlet allocation (LDA) to automatically discover the hidden sub-topics from the sentences.

We have proposed a novel method to rank the generated questions by considering: 1) sub-topical similarity determined using ESSK algorithm in combination with word sense disambiguation, and 2) syntactic similarity determined using the syntactic tree kernel based method. We have compared the proposed question generation (QG) system with two baseline systems and one state-of-the-art system. The evaluation results have showed that the proposed QG system significantly outperforms all other considered systems as our system generated top-ranked questions were found to be better in topic-relevance and syntactic correctness than those of the other systems. We have also conducted another experiment by narrowing down the topic focus. Experiments have further demonstrated the effectiveness of the proposed topic to question generation approach. In the future, we hope to carry on the related ideas of topic to question generation and develop further mechanisms to question generation based on the dependency features of the answers and answer finding (Li and Roth, 2006; Pinchak and Lin, 2006). The outcome of this research is an efficient method of automatically generating topical questions that can be incorporated into our complex question answering models of Chapter 2 to Chapter 4 to provide better direction of search for the most important sentences relevant to the complex question in consideration.

It is important to note that although the reported research in this thesis has mainly focused on addressing the pitfalls of the existing methodologies of answering complex questions, the individual components of the proposed approaches can be effectively connected into a coherent framework as discussed above. Our proposed methods and systems have largely depended on complex preprocessing steps that involved the use of different off-the-shelf tools and techniques. We firmly believe that the reported results would further improve if the tools and techniques used in this thesis could be enhanced to show higher degree of accuracy during the preprocessing stage.

References

Agarwal, M., R. Shah, and P. Mannem. 2011. Automatic Question Generation Using Discourse Cues. In *Proceedings of the 6th Workshop on Innovative Use of NLP for Building Educational Applications*, pages 1–9. ACL.

Al-Maskari, A., M. Sanderson, and P. Clough. 2007. The Relationship between IR Effectiveness Measures and User Satisfaction. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '07, pages 773–774. ACM.

Ali, H., Y. Chali, and S. A. Hasan. 2010. Automation of Question Generation from Sentences. In *Proceedings of QG2010: The Third Workshop on Question Generation*, pages 58–67, Pittsburgh, USA.

Andrenucci, A. and E. Sneiders. 2005. Automated Question Answering: Review of the Main Approaches. In *Proceedings of the 3rd International Conference on Information Technology and Applications (ICITA'05)*, Sydney, Australia.

Barzilay, R. and M. Elhadad. 1997. Using Lexical Chains for Text Summarization. In *Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics and the 8th European Chapter Meeting of the Association for Computational Linguistics, Workshop on Intelligent Scalable Test Summarization*, pages 10–17, Madrid.

Berg-Kirkpatrick, T., D. Gillick, and D. Klein. 2011. Jointly Learning to Extract and Compress. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, HLT '11, pages 481–490. ACL.

Blei, D. M., A. Y. Ng, and M. I. Jordan. 2003. Latent Dirichlet Allocation. *Journal of Machine Learning Research*, 3:993–1022.

Bouayad-Agha, N., A. Gil, O. Valentin, and V. Pascual. 2006. A Sentence Compression Module for Machine-Assisted Subtitling. In *Computational Linguistics and Intelligent Text Processing*, pages 490–501. Springer Berlin Heidelberg.

Boyer, K. E. and P. Piwek. 2010. Proceedings of QG2010: The Third Workshop on Question Generation. Pittsburgh: questiongeneration.org.

Branavan, S. R. K., H. Chen, L. S. Zettlemoyer, and R. Barzilay. 2009. Reinforcement Learning for Mapping Instructions to Actions. In *Proceedings of the Joint conference of the 47th Annual Meeting of the Association for Computational Linguistics (ACL-IJCNLP 2009)*, pages 329–332, Suntec, Singapore.

Brown, J. C., G. A. Frishkoff, and M. Eskenazi. 2005. Automatic Question Generation for Vocabulary Assessment. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, Vancouver, British Columbia, Canada.

Cancedda, N., E. Gaussier, C. Goutte, and J. M. Renders. 2003. Word Sequence Kernels. *Journal of Machine Learning Research*, 3:1059–1082.

Carbonell, J. and J. Goldstein. 1998. The Use of MMR, Diversity-based Reranking for Reordering Documents and Producing Summaries. In *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 1998)*, pages 335–336, Melbourne, Australia.

Carbonell, J., D. Harman, E. Hovy, S. Maiorano, J. Prange, and K. Sparck-Jones. 2000. Vision Statement to Guide Research in Question & Answering (Q&A) and Text Summarization. *National Institute of Standards and Technology (NIST) Draft Publication*.

Celikyilmaz, A., D. Hakkani-Tur, and G. Tur. 2010. LDA based Similarity Modeling for Question Answering. In *Proceedings of the NAACL HLT 2010 Workshop on Semantic Search*, SS '10, pages 1–9. ACL.

Chali, Y. and S. A. Hasan. 2012a. On the Effectiveness of Using Sentence Compression Models for Query-Focused Multi-Document Summarization. In *Proceedings of the 24th International Conference on Computational Linguistics (COLING 2012)*, pages 457–474.

Chali, Y. and S. A. Hasan. 2012b. Query-focused Multi-document Summarization: Automatic Data Annotations and Supervised Learning Approaches. *Journal of Natural Language Engineering*, 18(1):109–145.

Chali, Y. and S. A. Hasan. 2012c. Towards Automatic Topical Question Generation. In *Proceedings of the 24th International Conference on Computational Linguistics (COLING 2012)*, pages 475–492.

Chali, Y., S. A. Hasan, and K. Imam. 2011a. A Reinforcement Learning Framework for Answering Complex Questions. In *Proceedings of the 16th International Conference on Intelligent User Interfaces*, IUI '11, pages 307–310. ACM.

Chali, Y., S. A. Hasan, and K. Imam. 2011b. An Aspect-driven Random Walk Model for Topic-Focused Multi-Document Summarization. In *Proceedings of the 7th Asian Information Retrieval Societies Conference (AIRS 2011)*, pages 386–397. Springer-Verlag.

Chali, Y., S. A. Hasan, and K. Imam. 2011c. Using Semantic Information to Answer Complex Questions. In *Proceedings of the 24th Canadian Conference on Artificial Intelligence (CAI 2011)*, pages 68–73. Springer-Verlag.

Chali, Y., S. A. Hasan, and K. Imam. 2012a. Improving the Performance of the Reinforcement Learning Model for Answering Complex Questions. In *Proceedings of the 21st ACM Conference on Information and Knowledge Management (CIKM 2012)*, pages 2499–2502. ACM.

Chali, Y., S. A. Hasan, and K. Imam. 2012b. Learning Good Decompositions of Complex Questions. In *Proceedings of the 17th International Conference on Applications of Natural Language Processing to Information Systems (NLDB 2012)*, pages 104–115. Springer-Verlag.

Chali, Y., S. A. Hasan, and S. R. Joty. 2009. Do Automatic Annotation Techniques Have Any Impact on Supervised Complex Question Answering? In *Proceedings of the Joint conference of the 47th Annual Meeting of the Association for Computational Linguistics (ACL-IJCNLP 2009)*, pages 329–332, Suntec, Singapore.

Chali, Y., S. A. Hasan, and S. R. Joty. 2011. Improving Graph-based Random Walks for Complex Question Answering Using Syntactic, Shallow Semantic and Extended String Subsequence Kernels. *Information Processing and Management (IPM), Special Issue on Question Answering*, 47(6):843–855.

Chali, Y. and S. R. Joty. 2008. Improving the Performance of the Random Walk Model for Answering Complex Questions. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 9–12, OH, USA. ACL.

Chali, Y., S. R. Joty, and S. A. Hasan. 2009. Complex Question Answering: Unsupervised Learning Approaches and Experiments. *Journal of Artificial Intelligence Research*, 35:1–47.

Charniak, E. 1999. A Maximum-Entropy-Inspired Parser. In *Technical Report CS-99-12*, Brown University, Computer Science Department.

Chen, W., G. Aist, and J. Mostow. 2009. Generating Questions Automatically from Informational Text. In *Proceedings of the 2nd Workshop on Question Generation (AIED 2009)*, pages 17–24.

Clarke, J. and M. Lapata. 2008. Global Inference for Sentence Compression An Integer Linear Programming Approach. *Journal of Artificial Intelligence Research*, 31(1):399–429.

Cohen, J. 1960. A Coefficient of Agreement for Nominal Scales. *Educational and Psychological Measurement*, 20(1):37–46.

Cohn, T. and M. Lapata. 2008. Sentence Compression Beyond Word Deletion. In *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008)*, pages 137–144, Manchester, UK, August.

Collins, M. and N. Duffy. 2001. Convolution Kernels for Natural Language. In *Proceedings of Neural Information Processing Systems*, pages 625–632, Vancouver, Canada.

Conroy, J. M. and D. P. O'Leary. 2001. Text Summarization Via Hidden Markov Models. In *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2001)*, pages 406–407, New Orleans, LA, USA.

Conroy, J. M., J. D. Schlesinger, and D. P. O'Leary. 2006. Topic-Focused Multi-Document Summarization Using An Approximate Oracle Score. In *Proceedings of the COLING/ACL on Main conference poster sessions*, pages 152–159.

Cortes, C. and V. N. Vapnik. 1995. Support Vector Networks. *Machine Learning*, 20:273–297.

Das, D. and A. F. T. Martins. 2007. *A Survey on Automatic Text Summarization*. Language Technologies Institute, Carnegie Mellon University.

Daumé III, H. and D. Marcu. 2005. Bayesian Multi-Document Summarization at MSE. In *Proceedings of the Workshop on Multilingual Summarization Evaluation (MSE).*

Daumé III, H. and D. Marcu. 2006. Bayesian Query-Focused Summarization. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, pages 305–312.

Deerwester, S., S. Dumais, G. Furnas, T. Landauer, and R. Harshman. 1990. Indexing by Latent Semantic Analysis. *Journal of the American Society for Information Science*, 41(6):391–407.

Denis, P. and J. Baldridge. 2007. Joint Determination of Anaphoricity and Coreference Resolution using Integer Programming. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics;*, pages 236–243. ACL.

Dethlefs, N., H. W. Hastie, V. Rieser, and O. Lemon. 2012. Optimising Incremental Dialogue Decisions Using Information Density for Interactive Systems. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, EMNLP-CoNLL 2012, Jeju Island, Korea*, pages 82–93.

duVerle, D. A. and H. Prendinger. 2009. A Novel Discourse Parser Based on Support Vector Machine Classification. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP (ACL '09): Volume 2*, pages 665–673.

Echihabi, A., U. Hermjakob, E. Hovy, D. Marcu, E. Melz, and D. Ravichandran. 2003. Multiple-Engine Question Answering in TextMap. In *Proceedings of the Twelfth Text REtreival Conference*, pages 772–781, Gaithersburg, Maryland.

Edmundson, H. P. 1969. New Methods in Automatic Extracting. *Journal of the Association for Computing Machinery (ACM)*, 16(2):264–285.

Efron, B. and R. J. Tibshirani. 1994. *An Introduction to the Bootstrap*. CRC Press.

Erkan, G. 2007. *Using Graphs and Random Walks for Discovering Latent Semantic Relationships in Text*. Ph.D. thesis, Department of Computer Science and Engineering, University of Michigan.

Erkan, G. and D. R. Radev. 2004. LexRank: Graph-based Lexical Centrality as Salience in Text Summarization. *Journal of Artificial Intelligence Research*, 22:457–479.

Fellbaum, C. 1998. WordNet - An Electronic Lexical Database. Cambridge, MA. MIT Press.

Ferrier, L., 2001. *A Maximum Entropy Approach to Text Summarization*. M.Sc. thesis, School of Artificial Intelligence, Division of Informatics, University of Edinburgh.

Filatova, E., V. Hatzivassiloglou, and K. McKeown. 2006. Automatic Creation of Domain Templates. In *Proceedings of the COLING/ACL on Main conference poster sessions*, COLING-ACL '06, pages 207–214.

Filippova, K. 2010. Multi-Sentence Compression: Finding Shortest Paths in Word Graphs. In *Proceedings of the 23rd International Conference on Computational Linguistics*, pages 322–330. ACL.

Galanis, D. and I. Androutsopoulos. 2010. An Extractive Supervised Two-Stage Method for Sentence Compression. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, HLT '10, pages 885–893. ACL.

Galanis, D., G. Lampouras, and I. Androutsopoulos. 2012. Extractive multi-document summarization with integer linear programming and support vector regression. In *COLING*, pages 911–926.

Geman, S. and D. Geman. 1984. Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images. *IEEE Trans. Pattern Anal. Mach. Intell.*, 6:721–741.

Genest, P. and G. Lapalme. 2012. Fully abstractive approach to guided summarization. In *ACL (2)*, pages 354–358.

Gillick, D. and B. Favre. 2009. A Scalable Global Model for Summarization. In *Proceedings of the Workshop on Integer Linear Programming for Natural Langauge Processing*, ILP '09, pages 10–18. ACL.

Giménez, J. and L. Màrquez. 2003. Fast and accurate part-of-speech tagging: The svm approach revisited. In *RANLP*, pages 153–163.

Goldstein, J., M. Kantrowitz, V. Mittal, and J. Carbonell. 1999. Summarizing Text Documents: Sentence Selection and Evaluation Metrics. In *Proceedings of the 22nd International ACM Conference on Research and Development in Information Retrieval (SIGIR 1999)*, pages 121–128, Berkeley, CA, USA.

Goldstein, J., V. Mittal, J. Carbonell, and M. Kantrowitz. 2000. Multi-document Summarization by Sentence Extraction. In *Proceedings of the 2000 NAACL-ANLPWorkshop on Automatic summarization - Volume 4*, pages 40–48. ACL.

Gong, Y. and X. Liu. 2001. Generic Text Summarization Using Relevance Measure and Latent Semantic Analysis. In *Proceedings of the 24th International ACM Conference on Research and Development in Information Retrieval (SIGIR 2001)*, pages 19–25, New Orleans, LA, USA.

Graesser, A. C., K. VanLehn, C. P. Rose, P. W. Jordan, and D. Harter. 2001. Intelligent Tutoring Systems with Conversational Dialogue. *AI Magazine*, 22(4):39–52.

Greenwood, M. A. 2005. *Open-Domain Question Answering*. Ph.D. thesis, Department of Computer Science, University of Sheffield.

Griffiths, T. L. and M. Steyvers. 2002. Prediction and Semantic Association. In *NIPS'02*, pages 11–18.

Hacioglu, K., S. Pradhan, W. Ward, J. H. Martin, and D. Jurafsky. 2003. Shallow Semantic Parsing Using Support Vector Machines. In *Technical Report TR-CSLR-2003-03*, University of Colorado.

Harabagiu, S. and A. Hickl. 2006. Methods for using textual entailment in open-domain question answering. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, pages 905–912. ACL.

Harabagiu, S., A. Hickl, J. Lehmann, and D. Moldovan. 2005. Experiments with interactive question-answering. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, ACL '05, pages 205–214. ACL.

Harabagiu, S., F. Lacatusu, and A. Hickl. 2006. Answering Complex Questions with Random Walk Models. In *Proceedings of the 29th Annual International ACM SIGIR conference on Research and Development in Information Retrieval (SIGIR 2006)*, pages 220 – 227.

Hartigan, J. A. and M. A. Wong. 1979. A K-means Clustering Algorithm. *Applied Statistics*, 28:100–108.

Hasan, S. A., 2010. *Answering Complex Questions: Supervised Approaches*. M.Sc. Thesis, Department of Computer Science, University of Lethbridge, Canada.

Heilman, M. and N. A. Smith. 2010a. Extracting simplified statements for factual question generation. In *Proceedings of the Third Workshop on Question Generation*.

Heilman, M. and N. A. Smith. 2010b. Good Question! Statistical Ranking for Question Generation. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 609–617.

Hickl, A. and S. Harabagiu. 2006. Enhanced interactive question-answering with conditional random fields. In *Proceedings of the Interactive Question Answering Workshop at HLT-NAACL 2006*, IQA '06, pages 25–32. ACL.

Hickl, A., J. Lehmann, J. Williams, and A. Harabagiu. 2005. Experiments with interactive question-answering. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL05*, pages 60–69.

Hirao, T., H. Isozaki, E. Maeda, and Y. Matsumoto. 2002a. Extracting Important Sentences with Support Vector Machines. In *Proceedings of the 19th International Conference on Computational Linguistics*, pages 1–7, Taipei, Taiwan.

Hirao, T., H. Isozaki, E. Maeda, and Y. Matsumoto. 2002b. Extracting Important Sentences with Support Vector Machines. In *Proceedings of the 19th International Conference on Computational Linguistics*, pages 1–7, Taipei, Taiwan.

Hirao, T., Y. Sasaki, H. Isozaki, and E. Maeda. 2002c. NTT's Text Summarization System for DUC 2002. In *Proceedings of the Document Understanding Conference*, pages 104–107, Philadelphia, Pennsylvania, USA.

Hirao, T., Y. Sasaki, H. Isozaki, and E. Maeda. 2002d. NTT's Text Summarization System for DUC 2002. In *Proceedings of the Document Understanding Conference*, pages 104–107, Philadelphia, Pennsylvania, USA.

Hirao, T., J. Suzuki, H. Isozaki, and E. Maeda. 2003. NTT's Multiple Document Summarization System for DUC 2003. In *Proceedings of the Document Understanding Conference*, Edmonton, Canada.

Hirao, T., J. Suzuki, H. Isozaki, and E. Maeda. 2004. Dependency-based Sentence Alignment for Multiple Document Summarization. In *Proceedings of the 20th International Conference on Computational Linguistics*, pages 446–452, Geneva, Switzerland.

Hirschman, L. and R. Gaizauskas. 2001. Natural language question answering: the view from here. *Natural Language Engineering*, 7(4):275–300.

Hovy, E. and C. Y. Lin. 1998. Automated text summarization and the summarist system. In *Proceedings of a workshop on held at Baltimore, Maryland: October 13-15, 1998*, TIPSTER '98, pages 197–214. ACL.

Hovy, E., C. Y. Lin, L. Zhou, and J. Fukumoto. 2006. Automated Summarization Evaluation with Basic Elements. In *Proceedings of the 5th Conference on Language Resources and Evaluation*, Genoa, Italy.

Hsu, C., C. Chang, and C. Lin. 2008. A Practical Guide to Support Vector Classification, National Taiwan University, Taipei 106, Taiwan, http://www.csie.ntu.edu.tw/ cjlin.

Ignatova, K., D. Bernhard, and I. Gurevych. 2008. Generating High Quality Questions from Low Quality Questions. In *Proceedings of the Workshop on the Question Generation Shared Task and Evaluation Challenge*, Arlington, VA. NSF.

Jezek, K. and J. Steinberger. 2008. Automatic Text Summarization (The state of the art 2007 and new challenges). In *Znalosti*, pages 1–12.

Jing, H. 2000. Sentence Reduction for Automatic Text Summarization. In *Proceedings of the Sixth Conference on Applied Natural Language Processing*, pages 310–315. ACL.

Joachims, T. 1998. Text Categorization with Support Vector Machines: Learning with Many Relevant Features. In *European Conference on Machine Learning (ECML).*

Joachims, T. 1999. Making large-Scale SVM Learning Practical. In *Advances in Kernel Methods - Support Vector Learning*.

Joty, S. R., 2008. *Answer Extraction for Simple and Complex Questions*. M.Sc. Thesis, Department of Computer Science, University of Lethbridge, Canada.

Kingsbury, P. and M. Palmer. 2002. From Treebank to PropBank. In *Proceedings of the International Conference on Language Resources and Evaluation*, Las Palmas, Spain.

Knight, K. and D. Marcu. 2000. Statistics-Based Summarization - Step One: Sentence Compression. In *Proceedings of the Seventeenth National Conference on Artificial Intelligence and Twelfth Conference on Innovative Applications of Artificial Intelligence*, pages 703–710. AAAI Press.

Knight, K. and D. Marcu. 2002. Summarization Beyond Sentence Extraction: A Probabilistic Approach to Sentence Compression. *Artificial Intelligence*, 139(1):91–107.

Kolla, M., 2004. *Automatic Text Summarization using Lexical Chains: Algorithms and Experiments*. M.Sc. Thesis, Department of Computer Science, University of Lethbridge, Canada.

Kotov, A. and C. Zhai. 2010. Towards Natural Question Guided Search. In *Proceedings of the 19th international conference on World wide web*, WWW '10, pages 541–550. ACM.

Kouylekov, M. and B. Magnini. 2005. Recognizing Textual Entailment with Tree Edit Distance Algorithms. In *Proceedings of the PASCAL Challenges Workshop: Recognising Textual Entailment Challenge*.

Kupiec, J. 1993. MURAX: A Robust Linguistic Approach for Question Answering Using an On-Line Encyclopedia. In *SIGIR*, pages 181–190.

Kupiec, J., J. Pedersen, and F. Chen. 1995. A Trainable Document Summarizer. In *Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 1995)*, pages 68–73, Seattle, Washington, USA.

Lacatusu, F., A. Hickl, and S. Harabagiu. 2006. Impact of question decomposition on the quality of answer summaries. In *Proceedings of the fifth international conference on Language Resources and Evaluation, (LREC 2006)*.

Landis, J. R. and G. G. Koch. 1977. The Measurement of Observer Agreement for Categorical Data. *Biometrics*, 33(1):159–174.

Lauer, T. W., E. Peacock, and A. C. Graesser. 1992. Questions and Information Systems.

Lee, Ju-Hong, Sun Park, Chan-Min Ahn, and Daeho Kim. 2009. Automatic generic document summarization based on non-negative matrix factorization. *Information Processing and Management*, 45(1):20–34.

Li, J., L. Sun, C. Kit, and J. Webster. 2007. A Query-Focused Multi-Document Summarizer Based on Lexical Chains. In *Proceedings of the Document Understanding Conference*, Rochester, USA. NIST.

Li, X. and D. Roth. 2002. Learning Question Classifiers. In *Proceedings of the 19th International Conference on Computational Linguistics*, pages 1–7, Taipei,Taiwan.

Li, X. and D. Roth. 2006. Learning Question Classifiers: The Role of Semantic Information. *Journal of Natural Language Engineering*, 12(3):229–249.

Lin, C. 2004. ROUGE: A Package for Automatic Evaluation of Summaries. In *Proceedings of Workshop on Text Summarization Branches Out, Post-Conference Workshop of Association for Computational Linguistics*, pages 74–81, Barcelona, Spain.

Lin, C., R. C. Weng, and S. S. Keerthi. 2008. Trust Region Newton Method for Large-Scale Logistic Regression. *Journal of Machine Learning Research*, 9:627–650.

Lin, C. Y. 1999. Training a Selection Function for Extraction. In *Proceedings of the Eighth International Conference on Information and Knowledge Management*, pages 55–62. ACM.

Lin, C. Y. 2003. Improving Summarization Performance by Sentence compression: A Pilot Study. In *Proceedings of the sixth international workshop on Information retrieval with Asian languages - Volume 11*, pages 1–8. ACL.

Lin, C. Y. and E. Hovy. 1997. Identifying Topics by Position. In *Proceedings of the Fifth Conference on Applied Natural Language Processing*, pages 283–290. ACL.

Lin, C. Y. and E. H. Hovy. 2000. The Automated Acquisition of Topic Signatures for Text Summarization. In *Proceedings of the 18th conference on Computational linguistics*, pages 495–501.

Lin, J., A. Fernandes, B. Katz, G. Marton, and S. Tellex. 2003. Extracting Answers from the Web Using Knowledge Annotation and Knowledge Mining Techniques. In *Proceedings of the Eleventh Text REtreival Conference*, Gaithersburg, Maryland.

Lin, J., N. Madnani, and B. J. Dorr. 2010. Putting the user in the loop: interactive maximal marginal relevance for query-focused summarization. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, HLT '10, pages 305–308. ACL.

Litman, D. J., M. S. Kearns, S. Singh, and M. A. Walker. 2000. Automatic Optimization of Dialogue Management. In *Proceedings of COLING*.

Litvak, M., M. Last, and M. Friedman. 2010. A New Approach to Improving Multilingual Summarization using a Genetic Algorithm. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 927–936. ACL.

Lodhi, H., C. Saunders, J. Shawe-Taylor, N. Cristianini, and C. Watkins. 2002. Text Classification Using String Kernels. *Journal of Machine Learning Research*, 2:419–444.

Luhn, H. P. 1958. The Automatic Creation of Literature Abstracts. *IBM Journal of Research and Development*, 2:159–165.

MacCartney, B., T. Grenager, M. C. de Marneffe, D. Cer, and C. D. Manning. 2006. Learning to Recognize Features of Valid Textual Entailments. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the ACL*, page 4148, New York, USA.

Madnani, N., D. Zajic, B. Dorr, N. F. Ayan, and J. Lin. 2007. Multiple Alternative Sentence Compressions for Automatic Text Summarization. In *Proceedings of the 2007 Document Understanding Conference (DUC-2007) at NLT/NAACL 2007*.

Mani, I., 2001. *Automatic Summarization*. John Benjamins Co, Amsterdam/Philadelphia.

Mani, I. and M. T. Maybury, 1999. *Advances in Automatic Text Summarization*. MIT Press.

Mann, W. C. and S. A. Thompson. 1988. Rhetorical Structure Theory: Toward a Functional Theory of Text Organization. In *Text*, pages 8(3): 243–281.

Mannem, P., R. Prasad, and A. Joshi. 2010. Question Generation from Paragraphs at Upenn. In *Proceedings of the Third Workshop on Question Generation*.

Manning, C. D. and H. Schutze, 2000. *Foundations of Statistical Natural Language Processing*. The MIT Press.

Marcu, D. 1998a. Improving Summarization Through Rhetorical Parsing Tuning. In *The Sixth Workshop on Very Large Corpora*, pages 206–215, Montreal, Canada.

Marcu, D. C. 1998b. *The Rhetorical Parsing, Summarization, and Generation of Natural Language Texts*. Ph.D. thesis, Department of Computer Science, University of Toronto, Canada.

Martins, A. F. T. and N. A. Smith. 2009. Summarization with a Joint Model for Sentence Extraction and Compression. In *Proceedings of the Workshop on Integer Linear Programming for Natural Langauge Processing*, ILP '09, pages 1–9. ACL.

McCallum, A. K. 2002. MALLET: A Machine Learning for Language Toolkit.

McConnell, C. C., P. Mannem, R. Prasad, and A. Joshi. 2011. A New Approach to Ranking Over-Generated Questions. In *Proceedings of the AAAI Fall Symposium on Question Generation*.

Mcdonald, R. 2006. Discriminative Sentence Compression with Soft Syntactic Constraints. In *Proceedings of the 11th Conference of the EACL*.

McDonald, R. 2007. A Study of Global Inference Algorithms in Multi-document Summarization. In *Proceedings of the 29th European conference on IR research*, ECIR'07, pages 557–564. Springer-Verlag.

McGough, J., J. Mortensen, J. Johnson, and S. Fadali. 2001. A Web-based Testing System with Dynamic Question Generation. In *ASEE/IEEE Frontiers in Education Conference*.

McKeown, K. and D. R. Radev. 1995. Generating Summaries of Multiple News Articles. In *Proceedings of the 18th annual international ACM SIGIR conference on Research and Development in Information Retrieval*, pages 74–82.

Mihalcea, R. and P. Tarau. 2004. TextRank: Bringing Order into Texts. In *Proceedings of the Conference of Empirical Methods in Natural Language Processing (EMNLP 2004)*, Barcelona, Spain.

Mirkin, B., 2005. *Clustering for Data Mining: A Data Recovery Approach*. Boca Raton F1, Chapman and Hall/CRC.

Misra, H., O. Cappé, and F. Yvon. 2008. Using LDA to Detect Semantically Incoherent Documents. In *Proceedings of the Twelfth Conference on Computational Natural Language Learning*, CoNLL '08, pages 41–48. ACL.

Mitkov, R. and L. A. Ha. 2003. Computer-aided Generation of Multiple-Choice Tests. In *Proceedings of the HLT-NAACL 03 workshop on Building educational applications using natural language processing - Volume 2*, pages 17–22.

Moldovan, D., C. Clark, and M. Bowden. 2007. Lymbas PowerAnswer 4 in TREC 2007. In *Proceedings of the 16th Text REtreival Conference*, Gaithersburg, Maryland.

Molina, A., J. Torres-Moreno, E. SanJuan, I. da Cunha, G. Sierra, and P. Velázquez-Morales. 2011. Discourse Segmentation for Sentence Compression. In *Proceedings of the 10th Mexican international conference on Advances in Artificial Intelligence - Volume Part I*, pages 316–327. Springer-Verlag.

Molla, D. and S. Wan. 2006. Macquarie University at DUC 2006: Question Answering for Summarisation. In *Proceedings of the Document Understanding Conference*. NIST.

Moschitti, A. and R. Basili. 2006. A Tree Kernel Approach to Question and Answer Classification in Question Answering Systems. In *Proceedings of the 5th International Conference on Language Resources and Evaluation*, Genoa, Italy.

Moschitti, A., S. Quarteroni, R. Basili, and S. Manandhar. 2007. Exploiting Syntactic and Shallow Semantic Kernels for Question/Answer Classificaion. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics (ACL 2007)*, pages 776–783, Prague, Czech Republic.

Nastase, V. 2008. Topic-Driven Multi-Document Summarization with Encyclopedic Knowledge and Spreading Activation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP-08)*, pages 763–772.

Olney, A. M., A. C. Graesser, and N. K. Person. 2012. Question Generation from Concept Maps. *Dialogue and Discourse*, 3(2):75–99.

Osborne, M. 2002. Using Maximum Entropy for Sentence Extraction. In *Proceedings of the ACL-02 Workshop on Automatic Summarization*, pages 1–8. ACL.

Otterbacher, J., G. Erkan, and D. R. Radev. 2005. Using Random Walks for Question-focused Sentence Retrieval. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 915–922, Vancouver, Canada.

Page, L., S. Brin, R. Motwani, and T. Winograd. 1999. The PageRank Citation Ranking: Bringing Order to the Web. Technical Report 1999-66, Stanford University.

Palmer, M., D. Gildea, and P. Kingsbury. 2005. The Proposition Bank: An Annotated Corpus of Semantic Roles. *Computational Linguistics*, 31:71–106.

Pasca, M., 2003. *Open-domain question answering from large text collections*. Center for the Study of Language and Information, 2nd edition.

Pasca, M. and S. M. Harabagiu. 2001. Answer Mining from On-Line Documents. In *Proceedings of the Association for Computational Linguistics 39th Annual Meeting and 10th Conference of the European Chapter Workshop on Open-Domain Question Answering*, pages 38–45, Toulouse, France.

Pinchak, C. and D. Lin. 2006. A Probabilistic Answer Type Model. In *Proceedings of the 11th Conference of the European Chapter of the Association for Computational Linguistics*, pages 393–400.

Pingali, P., Rahul K., and V. Varma. 2007. IIIT Hyderabad at DUC 2007. In *Proceedings of the Document Understanding Conference*, Rochester, USA. NIST.

Pitler, E., A. Louis, and A. Nenkova. 2010. Automatic Evaluation of Linguistic Quality in Multi-document Summarization. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 544–554. ACL.

Punyakanok, V., D. Roth, and W. Yih. 2004. Mapping Dependencies Trees: An Application to Question Answering. In *Proceedings of AI & Math*, Florida, USA.

Punyakanok, V., D. Roth, W. Yih, and D. Zimak. 2004. Semantic Role Labeling via Integer Linear Programming Inference. In *Proceedings of the 20th international conference on Computational Linguistics*, COLING '04. ACL.

Quarteroni, S. and S. Manandhar. 2009. Designing an interactive open-domain question answering system. *Natural Language Engineering*, 15(1):73–95.

Ratinov, L. and D. Roth. 2009. Design Challenges and Misconceptions in Named Entity Recognition. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning*, pages 147–155. ACL.

Riedel, S. and J. Clarke. 2006. Incremental Integer Linear Programming for Non-projective Dependency Parsing. In *Proceedings of EMNLP*, pages 129–137.

Roth, D. and W. Yih. 2004. A Linear Programming Formulation for Global Inference in Natural Language Tasks. In *Proceedings of CoNLL-2004*, pages 1–8.

Roy, N., J. Pineau, and S. Thrun. 2000. Spoken Dialogue Management Using Probabilistic Reasoning. In *Proceedings of ACL.*

Rus, V., Z. Cai, and A. C. Graesser. 2007. Experiments on Generating Questions About Facts. In *Proceedings of the 8th International Conference on Computational Linguistics and Intelligent Text Processing*, pages 444–455. Springer-Verlag.

Rus, V. and A. C. Graesser. 2009. The Question Generation Shared Task and Evaluation Challenge. In *Workshop on the Question Generation Shared Task and Evaluation Challenge, Final Report*, The University of Memphis. National Science Foundation.

Russell, S. and P. Norvig, 2003. *Artificial Intelligence A Modern Approach, 2nd Edition*. Prentice Hall.

Ryang, S. and T. Abekawa. 2012. Framework of Automatic Text Summarization Using Reinforcement Learning. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, EMNLP-CoNLL 2012, Jeju Island, Korea*, pages 256–265.

Saggion, H., J. Torres-Moreno, I. Cunha, and E. SanJuan. 2010. Multilingual Summarization Evaluation without Human Models. In *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*, pages 1059–1067. ACL.

Sakai, H. and S. Masuyama. 2004. A multiple-document summarization system with user interaction. In *Proceedings of the 20th international conference on Computational Linguistics*, COLING '04. ACL.

Scheffler, K. and S. Young. 2002. Automatic Learning of Dialogue Strategy Using Dialogue Simulation and Reinforcement Learning. In *Proceedings of HLT*.

Schenker, N. and J. Gentleman. 2001. On judging the significance of differences by examining the overlap between confidence intervals. *The American Statistician*, 55:182–186.

Schilder, F. and R. Kondadadi. 2008. FastSum: Fast and Accurate Query-based Multi-document Summarization. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics on Human Language Technologies: Short Papers*, pages 205–208. ACL.

Scholkopf, B. and A. Smola, 2002. *Learning with Kernels*. MIT Press, Cambridge, MA.

Sekine, S. 2002. Proteus Project OAK System (English Sentence Analyzer), http://nlp.nyu.edu/oak.

Sekine, S. and C. Nobata. 2004. Definition, Dictionaries and Tagger for Extended Named Entity Hierarchy. In *Proceedings of the Forth International Conference on Language Resources and Evaluation*.

Sekine, S. and C. A. Nobata. 2001. Sentence Extraction with Information Extraction Technique. In *Proceedings of the Document Understanding Conference (DUC 2001)*, New Orleans, Louisiana, USA.

Shen, D., J. Sun, H. Li, Q. Yang, and Z. Chen. 2007. Document Summarization Using Conditional Random Fields. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI 2007)*, pages 2862–2867, Hyderabad, India.

Shin, K. and T. Kuboyama. 2008. A Generalization of Hausslers Convolution Kernel: Mapping Kernel. In *ICML*, pages 944–951.

Simmons, R. F. 1965. Answering English Questions by Computer: A Survey. *Communications of the ACM*, 8(1):53–70.

Singh, S. P., M. J. Kearns, D. J. Litman, and M. A. Walker. 1999. Reinforcement Learning for Spoken Dialogue Systems. In *Advances in NIPS*.

Sjöbergh, J. 2007. Older Versions of the ROUGEeval Summarization Evaluation System Were Easier to Fool. *Information Processing and Management*, 43:1500–1505.

Strzalkowski, T. and S. Harabagiu, 2008. *Advances in Open Domain Question Answering*. Springer.

Sutton, R. S. and A. G. Barto. 1998. *Reinforcement Learning: An Introduction* . The MIT Press, Cambridge, Massachusetts, London, England.

Vanderwende, L. 2008. The Importance of Being Important: Question Generation. In *Proceedings of the Workshop on the Question Generation Shared Task and Evaluation Challenge*, Arlington, VA. NSF.

Voorhees, E. M. 1999. Overview of the TREC 1999 Question Answering Track. In *Proceedings of the 8th Text REtreival Conference*, Gaithersburg, Maryland.

Wan, X. and J. Xiao. 2009. Graph-Based Multi-Modality Learning for Topic-Focused Multi-Document Summarization. In *Proceedings of the 21st international jont conference on Artifical intelligence (IJCAI-09)*, pages 1586–1591.

Wan, X., J. Yang, and J. Xiao. 2007a. Manifold-Ranking Based Topic-Focused Multi-Document Summarization. In *Proceedings of the 20th international joint conference on Artifical intelligence (IJCAI-07)*, pages 2903–2908.

Wan, X., J. Yang, and J. Xiao. 2007b. Towards an Iterative Reinforcement Approach for Simultaneous Document Summarization and Keyword Extraction. In *Proceedings of the Joint conference of the 47th Annual Meeting of the Association for Computational Linguistics (ACL-2007)*, pages 552–559, Prague, Czech Republic.

Wang, M. 2006. A Survey of Answer Extraction Techniques in Factoid Question Answering. In *CMU 11-762 Language and Statistics II, literature review project*.

Wang, W., H. Tianyong, and L. Wenyin. 2008. Automatic Question Generation for Learning Evaluation in Medicine. In *LNCS Volume 4823*.

Wang, Y., M. Huber, V. N. Papudesi, and D. J. Cook. 2003. User-guided reinforcement learning of robot assistive tasks for an intelligent environment. In *Proceedings of the IEEE/RJS International Conference on Intelligent Robots and Systems (IROS), Las Vegas, NV*. IEEE.

Webb, N. and T. Strzalkowski. 2006. Proceedings of the HLT-NAACL Workshop on Interactive Question Answering. ACL.

Wei, X. and W. B. Croft. 2006. LDA-based Document Models for Ad-hoc Retrieval. In *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '06, pages 178–185. ACM.

White, M., T. Korelsky, C. Cardie, V. Ng, D. Pierce, and K. Wagstaff. 2001. Multidocument Summarization via Information Extraction. In *Proceedings of the First International Conference on Human Language Technology Research*, HLT '01, pages 1–7.

Wu, M., F. Scholer, and A. Turpin. 2008. User preference choices for complex question answering. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '08, pages 717–718. ACM.

Yan, R., J. Nie, and X. Li. 2011. Summarize what you are interested in: An optimization framework for interactive personalized summarization. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 1342–1351, Edinburgh, Scotland, UK. ACL.

Yoshikawa, K., R. Iida, T. Hirao, and M. Okumura. 2012. Sentence Compression with Semantic Role Constraints. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 349–353, Jeju Island, Korea. ACL.

Zajic, D., B. J. Dorr, J. Lin, and R. Schwartz. 2007. Multi-candidate Reduction: Sentence Compression as a Tool for Document Summarization Tasks. *Information Processing and Management*, 43(6):1549–1570.

Zaragoza, H., B. B. Cambazoglu, and R. Baeza-Yates. 2010. Web Search Solved?: All Result Rankings the Same? In *Proceedings of the 19th ACM international conference on Information and knowledge management*, CIKM '10, pages 529–538. ACM.

Zha, H. 2002. Generic summarization and keyphrase extraction using mutual reinforcement principle and sentence clustering. In *Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 113–120. ACM.

Zhang, A. and W. Lee. 2003. Question Classification Using Support Vector Machines. In *Proceedings of the Special Interest Group on Information Retrieval*, pages 26–32, Toronto, Canada. ACM.

Zheng, Z., X. Si, E. Y. Chang, and X. Zhu. 2011. K2Q: Generating Natural Language Questions from Keywords with User Refinements. In *Proceedings of the 5th International Joint Conference on Natural Language Processing*, pages 947–955.

Zhou, L., M. Ticrea, and E. H. Hovy. 2005. Multi-document Biography Summarization. *CoRR*, abs/cs/0501078.

Appendix A
Reference Cue Words and Stop Words

Cue Words

indeed	further	as well
as this	either	neither
not only	but also	the reason is
as well as	also	moreover
what is more	as a matter of fact	furthermore
in addition	besides	to tell you the truth
in fact	actually	amazingly
to say nothing of	too	let alone
much less	additionally	nor
alternatively	on the other hand	not to mention
such as	this time	at this time
this also	several years ago	long ago
during	eventually	meanwhile
essentially	enormously	majority of the
absolutely	necessary	especially
specially	after	before
at least	at most	most
therefore	this is	that is
reasonable	according to	throughout
at this point	along with	previously
as	particularly	including
as an illustration	for example	like
in particular	for one thing	to illustrate
for instance	notably	by way of example
speaking about	considering	regarding
with regards to	as for	concerning
on the subject of	the fact that	similarly
in the same way	by the same token	in a like manner
equally	likewise	namely
specifically	thus	I mean
put another way	in other words	but
by way of contrast	while	on the other hand
however	yet	whereas
though	in contrast	when in fact
conversely	still	even more
above all	more importantly	but even so
nevertheless	even though	admittedly
nonetheless	despite	notwithstanding

albeit	although	in spite of
regardless	granted	be that as it may
either way	whichever happens	in either event
in any case	at any rate	in either case
whatever happens	all the same	in any event
rather	instead	for the reason that
being that	in view of	inasmuch as
because	seeing that	owing to
due to	in that	since
forasmuch as	for this reason	on condition
provided that	in case	in the event that
as long as	so long as	unless
given that	granting	providing that
even if	only if	as a result
consequently	hence	in consequence
so that	accordingly	as a consequence
so much that	for the purpose of	in the hope that
for fear that	with this intention	to the end that
in order to	lest	with this in mind
in order that	so as to	under those circumstances
then	in that case	if not
that being the case	if so	otherwise
initially	to start with	first of all
to begin with	at first	for a start
secondly	subsequently	previously
next	afterwards	to conclude
as a final point	at last	in the end
finally	lastly	to change the topic
incidentally	by the way	to get back to the point
to resume	anyhow	anyway
at any rate	to return to the subject	as was previously stated
in summary	all in all	to make a long story short
as I have said	to sum up	overall
as has been mentioned	to summarize	to be brief
briefly	given these points	in all
on the whole	as has been noted	hence
in conclusion	in a word	to put it briefly
in sum	altogether	in short

Stop Words

reuters	ap	jan	feb	mar	apr
may	jun	jul	aug	sep	oct
nov	dec	tech	news	index	mon
tue	wed	thu	fri	sat	’s
a	a’s	able	about	above	according
accordingly	across	actually	after	afterwards	again
against	ain’t	all	allow	allows	almost
alone	along	already	also	although	always
am	amid	among	amongst	an	and
another	any	anybody	anyhow	anyone	anything
anyway	anyways	anywhere	apart	appear	appreciate
appropriate	are	aren’t	around	as	aside
ask	asking	associated	at	available	away
awfully	b	be	became	because	become
becomes	becoming	been	before	beforehand	behind
being	believe	below	beside	besides	best
better	between	beyond	both	brief	but
by	c	c’mon	c’s	came	can
can’t	cannot	cant	cause	causes	certain
certainly	changes	clearly	co	com	come
comes	concerning	consequently	consider	considering	contain
containing	contains	corresponding	could	couldn’t	course
currently	d	definitely	described	despite	did
didn’t	different	do	does	doesn’t	doing
don’t	done	down	downwards	during	e
each	edu	eg	e.g.	eight	either
else	elsewhere	enough	entirely	especially	et
etc	etc.	even	ever	every	everybody
everyone	everything	everywhere	ex	exactly	example
except	f	far	few	fifth	five
followed	following	follows	for	former	formerly
forth	four	from	further	furthermore	g
get	gets	getting	given	gives	go
goes	going	gone	got	gotten	greetings
h	had	hadn’t	happens	hardly	has
hasn’t	have	haven’t	having	he	he’s
hello	help	hence	her	here	here’s
hereafter	hereby	herein	hereupon	hers	herself
hi	him	himself	his	hither	hopefully

how	howbeit	however	i	i’d	i’ll
i’m	i’ve	ie	i.e.	if	ignored
immediate	in	inasmuch	inc	indeed	indicate
indicated	indicates	inner	insofar	instead	into
inward	is	isn’t	it	it’d	it’ll
it’s	its	itself	j	just	k
keep	keeps	kept	know	knows	known
l	lately	later	latter	latterly	least
less	lest	let	let’s	like	liked
likely	little	look	looking	looks	ltd
m	mainly	many	may	maybe	me
mean	meanwhile	merely	might	more	moreover
most	mostly	mr.	ms.	much	must
my	myself	n	namely	nd	near
nearly	necessary	need	needs	neither	never
nevertheless	new	next	nine	no	nobody
non	none	noone	nor	normally	not
nothing	novel	now	nowhere	o	obviously
of	off	often	oh	ok	okay
old	on	once	one	ones	only
onto	or	other	others	otherwise	ought
our	ours	ourselves	out	outside	over
overall	own	p	particular	particularly	per
perhaps	placed	please	plus	possible	presumably
probably	provides	q	que	quite	qv
r	rather	rd	re	really	reasonably
regarding	regardless	regards	relatively	respectively	right
s	said	same	saw	say	saying
says	second	secondly	see	seeing	seem
seemed	seeming	seems	seen	self	selves
sensible	sent	serious	seriously	seven	several
shall	she	should	shouldn’t	since	six
so	some	somebody	somehow	someone	something

sometime	sometimes	somewhat	somewhere	soon	sorry
specified	specify	specifying	still	sub	such
sup	sure	t	t’s	take	taken
tell	tends	th	than	thank	thanks
thanx	that	that’s	thats	the	their
theirs	them	themselves	then	thence	there
there’s	thereafter	thereby	therefore	therein	theres
thereupon	these	they	they’d	they’ll	they’re
they’ve	think	third	this	thorough	thoroughly
those	though	three	through	throughout	thru
thus	to	together	too	took	toward
towards	tried	tries	truly	try	trying
twice	two	u	un	under	unfortunately
unless	unlikely	until	unto	up	upon
us	use	used	useful	uses	using
usually	uucp	v	value	various	very
via	viz	vs	w	want	wants
was	wasn’t	way	we	we’d	we’ll
we’re	we’ve	welcome	well	went	were
weren’t	what	what’s	whatever	when	whence
whenever	where	where’s	whereafter	whereas	whereby
wherein	whereupon	wherever	whether	which	while
whither	who	who’s	whoever	whole	whom
whose	why	will	willing	wish	with
within	without	won’t	wonder	would	would
wouldn’t	x	y	yes	yet	you
you’d	you’ll	you’re	you’ve	your	yours
yourself	yourselves	z	zero		

Appendix B

Example of User Interaction Experiment

User Interaction Experiment

Interaction Example

Human interaction starts:

Topic No.: D0620B

Topic: school violence prevention measures

Question: Discuss measures that schools and school districts have taken to prevent violent occurrences and shootings, such as those in Littleton, Colorado and Jonesboro, Arkansas.

Current Summary:

Top five candidates:

1:In Virginia , where 14-year-old Quinshawn Booker wounded a teacher and a 74-year-old school volunteer last June 15 , Gov. Jim Gilmore has released $1.5 million to put some 45 additional police officers in schools .

2:Shamrock is not a seedy , dangerous school far from it .

3:“ They ’ll report anything from a burglary in progress to students being disruptive or being where they ’re not supposed to be . ”

4:Connecting with students who hurt and healing them is more likely to occur when psychologists are a visible force in the schools , says Dwyer .

5:Her school system ’s low-key response to the shootings at Littleton ’s Columbine High School on April 20 reflects a national pattern , according to students , school officials , researchers and safety advocates interviewed in the past two weeks .

Select a sentence to add to current summary: 1

End of human interaction in iteration 1

Current Summary: *In Virginia , where 14-year-old Quinshawn Booker wounded a teacher and a 74-year-old school volunteer last June 15 , Gov. Jim Gilmore has released $1.5 million to put some 45 additional police officers in schools . “ There ’s a fine line , ” he said.*

Top five candidates:

1:But as another school shooting tragedy unfolded in Littleton , Colo. , on Tuesday , officials questioned whether any school could protect against such deadly events .

2:There were 40 shooting deaths in schools in the 1997-1998 school year , the report said .

3:After it became known that the Littleton Police Department had been unfamiliar with the layout of Columbine High School , where 12 students and a teacher were killed by two young gunmen on April 20 , many districts began distributing floor plans and computerized blueprints of their schools to their police departments , which in turn began touring the schools over the summer .

4:“ In my view , the schools are safer than they were a year ago because we ’ve had a heightened need to be vigilant , ” he says .

5:(AP) – New software that helps highlight school trouble spots may help educators trying to avert violent episodes such as the Columbine High School shootings earlier this year .

Select a sentence to add to current summary: 3

End of human interaction in iteration 2

Current Summary: *In Virginia , where 14-year-old Quinshawn Booker wounded a teacher and a 74-year-old school volunteer last June 15 , Gov. Jim Gilmore has released $ 1.5 million to put some 45 additional police officers in schools . “ After it became known that the Littleton Police Department had been unfamiliar with the layout of Columbine High School , where 12 students and a teacher were killed by two young gunmen on April 20 , many districts began distributing floor plans and computerized blueprints of their schools to their police departments , which in turn began touring the schools over the summer.*

Top five candidates:

1:Story Filed By Cox Newspapers For Use By Clients of the New York Times News Service

2:Suzann Wilson , who lost her 11-year-old daughter , Britthney , knows of the changes at Westside but says more should be done to make students feel safe .

3:The parent councils Cellucci mentioned also include school principals , teachers , and , in some cases , community leaders and students .

4:“ Unfortunately , I do n’t hear that at all , ” said Scott Poland of Cypress , Texas , author of “ Coping with Crisis ” and a school psychologist who has been part of teams dealing with multiple shootings in several states .

5:“ We ’re running a school district not a prison system , ” said Ferrar , Seattle ’s security director .

Select a sentence to add to current summary: 2

End of human interaction in iteration 3

Current Summary: *In Virginia , where 14-year-old Quinshawn Booker wounded a teacher and a 74-year-old school volunteer last June 15 , Gov. Jim Gilmore has released $ 1.5 million to put some 45 additional police officers in schools . “ There ’s a fine line , ” he said . After it became known that the Littleton Police Department had been unfamiliar with the layout of Columbine High School , where 12 students and a teacher were killed by two young gunmen on April 20 , many districts began distributing floor plans and computerized blueprints of their schools to their police departments , which in turn began touring the schools over the summer . Suzann Wilson , who lost her 11-year-old daughter , Britthney , knows of the changes at Westside but says more should be done to make students feel safe.*

Automatic learning starts for remaining iterations..

Automatic learning ends..

Final summary: *In Virginia , where 14-year-old Quinshawn Booker wounded a teacher and a 74-year-old school volunteer last June 15 , Gov. Jim Gilmore has released $ 1.5 million to put some 45 additional police officers in schools . " After it became known that the Littleton Police Department had been unfamiliar with the layout of Columbine High School , where 12 students and a teacher were killed by two young gunmen on April 20 , many districts began distributing floor plans and computerized blueprints of their schools to their police departments , which in turn began touring the schools over the summer . Suzann Wilson , who lost her 11-year-old daughter , Britthney , knows of the changes at Westside but says more should be done to make students feel safe . But Cmdr. Patrick Fitzgerald , a city police officer who coordinates school safety , said he 's heard more from reporters writing about Thursday 's anniversary than from parents and students who fear it may precipitate copycat violence. " Things have been very quiet out here , " he said. " When we take attendance on Thursday , we 'll find out . " I feel accepted and wanted . Shamrock is not a seedy , dangerous school far from it . In Lawrence and Boston , if parents protest the uniforms , their children can usually wear what they want . There 's just one problem : While plenty shocking , this statistic is n't entirely accurate.*

Fully Automatic Learning Example

Automatic learning starts..
Automatic learning ends..

Final summary: *Schools have phased in innovations that range from more visible security procedures and creation of more local chapters of SAVE (Students Against Violence Everywhere) to a more efficient use of school psychologists and counselors . Graduation awaits next Friday . After it became known that the Littleton Police Department had been unfamiliar with the layout of Columbine High School , where 12 students and a teacher were killed by two young gunmen on April 20 , many districts began distributing floor plans and computerized blueprints of their schools to their police departments , which in turn began touring the schools over the summer . Suzann Wilson , who lost her 11-year-old daughter , Britthney , knows of the changes at Westside but says more should be done to make students feel safe . (Several students at Heritage High School have said they knew that the teen-age gunman there had talked of emulating the youths who opened fire at Columbine High a month earlier .) I feel accepted and wanted . Shamrock is not a seedy , dangerous school far from it . There 's just one problem : While plenty shocking , this statistic is n't entirely accurate . On April 20 , 1999 , two students at Columbine High in Littleton , Colo. , fatally shot 12 students and a teacher before killing themselves . They were angry that a month was set aside to honor black history but only a day to celebrate Cinco de Mayo.*

Appendix C
Sample Summaries

Sample Summaries for Topic-D0703A (DUC-2007)

Topic Title

steps toward introduction of the Euro

Complex Question

Describe steps taken and worldwide reaction prior to the introduction of the Euro on January 1, 1999. Include predictions and expectations reported in the press.

Human Generated Summary

Most predictions prior to introduction of the euro on Jan. 1, 1999 were positive. By early 1996 a majority of Europeans accepted the idea and France's Prime Minister was strongly supportive. By 1997 most British commentary was favorable. In Germany the Bundesbank predicted that private investors would profit. Bank officials as far away as Zambia saw benefits. By 1998 the Chinese government had officially welcomed coming of the euro and the European Central Bank President predicted that the euro would eventually rival the dollar. Design of the euro note began in 1996. In 1998 the test printing of banknotes began and Germany minted some euro coins. By the end of the year Germany was still experimenting with euro coins but France was in full production followed by Finland, Belgium and Spain. In May 1998 two major banks began quoting prices in euros. In October Bulgaria linked its currency, the lev, to the German mark, followed in December by fixing the value of the lev to the euro. In September key euro indicators were offered daily on the internet and in October Thailand announced that it was considering use of the euro in its foreign reserves. In November the Reserve Bank of India permitted bank transactions in euros followed in December by Romania's posting the euro in its rates of exchange. Small independent countries such as Monaco still faced some currency problems as Jan. 1, 1999 approached, but special arrangements were in process by European Union authorities.

SVM Generated Summary

Despite skepticism about the actual realization of a single European currency as scheduled on January 1 , 1999 , preparations for the design of the Euro note have already begun . European Union 's single-currency plan will take effect in 11 of its member states on January 1 , 1999 , when the euro will become a currency in the financial sense , but will first be used only in cashless tradings . The European single currency euro will go ahead on schedule on January 1 , 1999 with a broad membership , according to a survey of some prominent British economists . German Economics Minister Guenter Rexrodt said today that most German companies are not prepared for the shift to the single European currency , the Euro , which is due to be launched on January 1 , 1999 . Germany is in " an advanced stage " of experimenting striking the euro , while other European nations which *will join euro are due to begin producing the coin next year . Thailand is considering using the European single currency , the euro , in the country 's foreign reserves , the Nation reported Tuesday . Besides France , only Finland , Belgium and Spain have begun their production of the euros , but not as much as France , said AFP quoting official sources . Only 20 percent of German companies have taken concrete steps to get prepared for the Euro , Rexrodt said , adding that the country 's state institutions were also not well prepared . BNR indicative exchange rates will be modified to include euro instead of the current European single currency , the ECU .*

K-means Generated Summary

About the future European currency , 20 percent feel " well informed " , 79 percent feel to be not well informed . The RBI has set up a working group to study the implications of the Euro launch . The Euro will be launched on January 1 , 1999 . The Council of Economic Ministers of Thailand has ordered the Finance Ministry and the Bank of Thailand to study the possibility as the single European currency is set to go into circulation on January 1 next year . Local press reports today said that the building societies have produced blueprints for euro savings accounts , passbooks and cash-dispensers despite the government 's increasingly Eurosceptical tone . The report also said Duisenberg expects the future relationship between the dollar and the euro , which officially goes into effect on Jan. 12 , to be stable . He pointed out that such a political plan concerns the future of France and Germany , as well as the future role which the European countries will play in the world . It has n't been decided whether the Euro should include country emblems . Despite skepticism about the actual realization of a single European currency as scheduled on January 1 , 1999 , preparations for the design of the Euro note have already begun . The dollar has been weakening against European currencies in recent months . Stressing that the introduction of a single currency will be a great contribution to the unity of an expanded European Union (EU) , Juppe reiterated France 's commitment to the timetable and criteria of the single currency system set in the Maastricht treaty , under which the single European currency , recently named Euro , will be realized by January 1 , 1999 .

Reinforcement System Generated Summary

Speculation has surrounded the euro 's launch , with some economists predicting countries may switch the foreign reserves from the U.S. dollar to the euro , which represents some of the strongest economies of the world. Citibank , the world 's largest foreign exchange operator , and ABN-AMRO said that they had planned to quote prices in the euro for their clients starting next Monday. The local newspaper today also quoted senior economist with the bank Jacob Lushinga as saying that the idea of a single currency is to help facilitate trade and enhance exchange of goods that could be beneficial to Zambia. "Romania 's Central Bank (BNR) and the country 's commercial banks fear little about the introduction of the euro and will easily adapt to it", BNR Foreign Currency Director George Mucibabici told a recent colloquium on euro in Bucharest. Meanwhile the bank also decided to start *this Autumn a test printing of the euro banknotes , which have a total of seven face values. With regard to the two French territories , the ministers decided that the euro would be their currency two and that France would continue to grant legal status to their banknotes and coins. " Vatican City will have the right to continue issuing collector coins and the Republic of San Marino to continue using gold coins denominated in scudi", the ministers said. Its chief executive Geoffrey Lister said , "We believe euro mortgages will be popular because they will offer lower interest rates than sterling mortgages. Bradford and Bingley 's plans for a euro mortgage are already at an advanced stage , and will be launched ahead of the UK 's participation in the single currency.*

Combined Optimization (bigram+sem+SEM) System Generated Summary

The code is intended for use in any application trade , based on uniform , internationally agreed representations. A spokesman said , our view is single currency will happen the UK will in two or three years . investors are set to benefit from transaction costs after the advent of the euro, the said . Romanian bankers and economists could not be indifferent to future Euro in international payment. or Euro 's effect on Romanian commercial and payments balance , Isarescu said. The European summit confirmed that Belgium , Germany , Spain , France , Ireland , Italy , Luxembourg , the Netherlands , Austria , Portugal and Finland will be founding members of Euro organization . European stocks and bonds are denominated in euros , investors will make use of cross-border opportunities in the euro area become the second capital market in world . The Vatican and the Republic of San Marino are microstates located within Italy and use the Italian lira as their currency . Under the system , German mark is equivalent to 1,000 Bulgarian. The local quoted senior economist with the bank Jacob the idea of a single currency is exchange of goods Zambia . The answers represent the precondition for identifying the monetary and currency in the shortest delay the Governor added . Vasilescu said central bank had decided rates of exchange posted by hard currency operators should include the single currency unit . people worry that the budgetary criteria for the single currency will cause unemployment and social welfare .

*TF*IDF System Generated Summary*

Stressing that the introduction of a single currency will be a great contribution to the unity of an expanded European Union (EU) , Juppe reiterated France 's commitment to the timetable and criteria of the single currency system set in the Maastricht treaty , under which the single European currency , recently named Euro , will be realized by January 1 , 1999 . This new code has been issued to allow progress with the technical preparations for the European single currency , scheduled to be launched on January 1 , 1999 , said a press release issued here today . The Administration Board of Romania 's central bank (BNR) has had made a number of decisions in connection with the introduction of the single currency in the European Union 's member states beginning January 1 , 1999 , *BNR director Adrian Vasilescu said on Monday . In response to a question concerning the issue at today 's news conference , Zhu said the introduction of the Euro is expected to stimulate European integration , help establish a more balanced world financial system and exert influence on the world 's economy and politics . The introduction of the single European currency , the Euro , would pose strategical as well as tactical problems to the economy of Romania , an associate EU country seeking admission , the Rompres news agency Thursday quoted Romania 's Central Bank Governor Mugur Isarescu as saying . The Euro will be launched on January 1 , 1999 . The European Union member states are required to completely replace their own national currencies with the Euro from January 1 , 2002 .*

SEM System Generated Summary

BNR and Romanian commercial banks that want to perform euro services will have to open accounts in euro by January 4 , 1999 , the first scheduled day of euro use , and inform their foreign partners . Local press reports today said that the building societies have produced blueprints for euro savings accounts , passbooks and cash-dispensers despite the government 's increasingly Eurosceptical tone . At present , qualms about a single currency still exist in France and many other EU member countries as some economists fear that a single currency means a country 's loss of financial sovereignty . The actual banknotes and coins will be introduced over a six-month period starting January 1 , 2002 , after which the participants ' present national currencies will cease to be legal tender . A combined European Union market is expected to be helpful for Thai exporters to gain more excess into the market . The currency board was established to help reduce inflation by preventing the central bank from lending to the government or local banks . The Council of Economic Ministers of Thailand has ordered the Finance Ministry and the Bank of Thailand to study the possibility as the single European currency is set to go into circulation on January 1 next year . The Euro will be launched on January 1 , 1999 . In view of these confusions , Juppe gave a clear-cut answer by saying in the speech that only a single currency can guarantee currency stability , which means lower interest rates , sustained economic growth and good prospects of employment.