Learning to Rank and Order Answers to Definition Questions

Shailesh Pandey

Submitted for the degree of Doctor of Philosophy

University of York

Computer Science

February 2012

Abstract

The task of ordering a set of ranked result returned by an online search engine or an offline information retrieval engine is termed as reranking. It is called reranking for the reason that the candidate answer snippets are extracted by the information retrieval systems using some strategy for scoring, for example, based on occurrence of query words. We therefore assume the results to be already ranked and therefore the subsequent ranking is termed as reranking. Ranking drastically reduces the number of documents that will be processed further. Reranking usually involves deeper linguistic analysis and use of expert knowledge resources to get an even better understanding. The first task this thesis explores is regarding reranking of answers to definition questions. The answers are sentences returned by the google search engine in response to the definition questions. This step is relevant to definition questions because the questions tend to be short and therefore the information need of the user is difficult to assess. This means the final result is not a single piece of information but a ordered set of relevant sentences. In this thesis we explore two approaches to reranking that uses dependency tree statistics in a probabilistic setting. One of them is based on calculating edit distance between trees and tree statistics from the corpus and other one uses a tree kernel function and involves using the output from trained classifiers directly.

The second task this thesis explores is the task of sentence ordering for definition questions. The reranking part of the definition question answering pipeline is able to identify the sentences that are relevant to a given question. However, answer to a definition question is a collection of sentences that has some coherent ordering between them. In a way this is not far away from the characteristics observed in a good summary. We believe that by moving sentences around to form a more coherent chunk we will be able to better meet the expectation

of a user by improving his reading experience. We present an approach that finds an ordering for the sentences based on the knowledge extracted from observing the order of sentences in Wikipedia articles. Due to the popularity and acceptability of Wikipedia, proven by the fact that wikipedia results are ranked high by all major commercial search engines, it was chosen as the standard to be learnt from and compared against. We present a framework that uses the order of sentences extracted from Wikipedia articles to construct a single big graph of connected sentences. As a mechanism to select a node in the graph, we define a scoring function based on the relative position of candidate sentences.

Contents

List of Figures

List of Tables

List of Algorithms

Acknowledgements

My journey would not have been possible without the support and guidance from my supervisor Suresh Manandhar. He supported me financially, was extremely patient with me and was always there whenever I needed advice. I would like to thank Suresh for supporting me since 2007.

I would like to thank my assessor, Daniel Kudenko, for his interest towards my work. His feedback and comments on the reports in the earlier stages of the PhD and later as the internal examiner of the thesis were valuable. Sincere thanks goes to Mark Stevenson, my external examiner, for reading my thesis and suggesting improvements. Their suggestions have specifically helped me to improve the latter chapters of the thesis.

I started on my PhD alongside Shuguang Li, Azniah Ismail and Burcu Can. In the old computer science building, with the open floor plan, they were the faces and the voices that were familiar to me. Another friendly face was Ahmad Shahid. In the new building I shared the room with Ioannis Klapaftis and he was always encouraging and supportive. They all have been wonderful.

I have to thank Suresh again but this time for being a guardian, a brother and a friend. I have so many wonderful memories of the times I have spent in his house. For me, his house was my home and he was my family. I always enjoyed being there and in the company of Am, Milan and Dristi. I cherish the times I have spent with Milan.

Shiromani and Manju helped me settle down in York and provided companionship throughout the years. They have cooked a lot of meals, provided a getaway place and took good care of me. I could always depend on their help. I thank them for treating me like a family member. Bishnu has been a good friend. He and Saraswoti welcomed me into their family and I have enjoyed the moments we have shared

together.

Manas Patra was another person whose company I enjoyed a lot. There are far too many good moments to recollect. His company was extremely valuable when I was going through a bad period in the final stages of my PhD. He looked after me really well. I thank him for his friendship.

Without Santa and Suraj it would have been extremely difficult for me. They helped me financially, supported me in bad times, fed me, entertained me and looked after me very well. When I was back in Nepal, Suraj was the person I would think of when I had to get something done at York. And he did everything I asked of him. I can be really miserable at times and I thank them for standing by me. I am forever grateful for their friendship. Many wonderful people came into my life during this period. There are so many memories and moments that I can recall associated with Ankur, Siva, Spandana, Sonia and Tasawer. I thank you all for the wonderful times and your friendship.

On my return to Nepal, Birodh, Dinesh, Rupesh, Samjhana and Trishna were the voices of encouragement. They believed in me more than I had belief in myself. They included me in their moments of happiness and I always felt part of the group. This made the days leading up to my thesis defence and corrections bearable.

Special thanks goes to Matt. I regard his friendship as the most valuable achievement during the stay in England. Without his voice and friendship, it would have been extremely difficult for me.

Computer science meant programming and the word research was not part of my vocabulary. This was before I met Yogesh Raj. He is my mentor, a voice of reason and a source of genuine affection. I would be lost in the wilderness without him. I thank him for his trust and belief in me.

In this long journey, the only expectation from my family was the answer to the question “Are you happy?”. This thesis is dedicated to them.

Declaration

This thesis has not previously been accepted in substance for any degree and is not being concurrently submitted in candidature for any degree other than Doctor of Philosophy of the University of York. This thesis is the result of my own investigations, except where otherwise stated. Other sources are acknowledged by explicit references.

Chapter 1

Background

1.1 Introduction

Search engines have become an important tool in the modern age. With more and more information being put online, the ever growing role of a search engine cannot be stressed enough. The mere thought of navigating through a vast ocean of web without a good search engine is dizzying. Even with such a tool, the user has to do the hard bit. In the business of search, relevant search terms uncover relevant information.

Searching is an information retrieval process where the user presents the system with a set of words that best describe his/her information need. In turn, the system returns a ranked list of relevant documents. It is now up to the user to search for the needle in this relatively smaller haystack. The haystack maybe a fraction of the information out there but it still is huge (if you are lucky only few 1000 documents) from a user's perspective. Searching can be a really frustrating experience. Most of us might have have experienced a frustrated search session sometime or another.

It seems obvious that a more focused information retrieval process is needed. When we are looking for a specific information like the question "How tall is Mt. Everest?", we don't want to plough through a large list of relevant documents. We will be happy with the figure 29,029 ft or 8848 m. Actually this can already be achieved in a search engine such as google by giving the query "mt Everest height" (without the quotes). The first response is something like "Best guess for Mount

Everest Elevation is 8,848 m ...". But we do not get the exact same result when the query is changed to "how tall is mt Everest". The first few documents returned by the search engine does contain the correct answer (and are highlighted in bold). Many of them contain the exact query as well but the expected answer is not explicitly presented as in the first case. If the query is changed to "why is mt Everest tall", 6 out of 10 results (documents) in the first page is exactly the same as the "how tall" query. None of the snippets answers the question (either partially or completely). The answer might be in some of the documents but the user has to do the hard bit. S/he has to open it and search for it. Even in cases where we are not looking for a specific information, such as "What is a poem?", we would probably prefer a set of relevant paragraphs compared to a list of documents.

When the web exploded in the late 80s and early 90s, search engines played a vital role. But with the current rate of information explosion, availability of faster machines and faster and better linguistic methods/tools, it is an eventuality that in the future we would be using a focused information retrieval tool, Question Answering (QA) systems.

In the subsequent sections we briefly introduce the area of question answering and the stages in a question answering process. We conclude the chapter with the objectives and aims of this thesis.

1.2 Question Answering

Question answering (QA) can be defined as a process of finding concise information in response to a user's query. Instead of returning a list of documents, a QA system returns relevant sections of text (few hundred characters) from within the documents.

Research in QA started with the development of systems that responded to questions by fetching the knowledge stored in structured databases. The QA system BASEBALL (Green et al., 1961) could answer questions about baseball games played in the American league. But such systems could provide an answer only if it was present in the database. Most of the early systems could be viewed as providing a natural language interface to databases.

Research into question answering gained momentum when it was

introduced as a track in the text retrieval conference TREC[1] in 1999 (Voorhees, 1999). The data comprised of mainly newspaper articles. The questions asked were short and fact seeking such as "Where is Taj Mahal?" and "Who was the 16th President of the United States?" Participants had to return a ranked list of five [document-id, answer-string] pairs, where answer-string is the answer candidate and document-id is the document where it was pulled from. There has been a surge of experiments and evaluation forums like CLEF, NTCIR, including advancement of TREC experiments since. The task in these conferences is to find answers which are present in a large collection of documents. Over the years, there has been a gradual move to introduce slightly more difficult questions like the *other* questions (analogous to definition questions) and complex interactive questions in TREC.

Question answering systems can be broadly classified as being closed domain or open domain. Closed domain systems limit themselves to a specific domain/topic. This specialisation is often achieved by utilising the structured data in the form of domain specific ontologies, rules and heuristics. LUNAR (Woods, 1977) is an early example of such a system. LUNAR allowed questions to be asked about moon rocks and used shallow parsing. Basically it provided a natural language front-end to a database. Open domain systems cannot rely on domain specific heuristics and rules. From a macro perspective open domain systems are pattern seeking in general. With the large amount of organised and unorganised data available, looking for reoccurring interesting patterns is a key characteristic of this line of research. Another characteristic of an open domain system is the use of information retrieval as a backbone for document retrieval. This is necessary because processing is expensive and document collections are extremely large. For instance, using google as an information retrieval engine, a smaller set of query specific web documents is obtained.

1.3 QA System Architecture

The general architecture of a QA system has not changed much over time. Most of the current systems have an architecture similar to Lasso

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

(Moldovan et al., 1999). The user of the system has an information need expressed as a question. The question is analysed at various levels and information such as the question type is extracted. The question is converted into a search engine specific query, fed to a search engine which searches over the corpus to find relevant documents. The relevant documents returned by the search engine are mined to extract the potential answers. The exact answer or a set of answers, depending on the type of question asked, is presented to the user. The basic architecture of a QA system is shown in Figure 1-1. A general QA system has the following stages: (1) Question processing, (2) Document Retrieval, (3) Passage Retrieval, (4) Answer Extraction and (5) Answer Reranking.

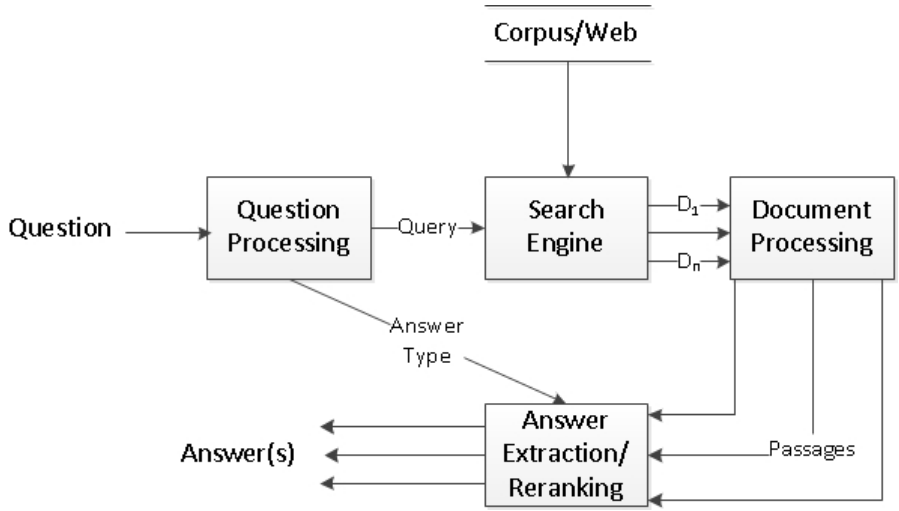


Figure 1-1: *A Typical Question Answering Pipeline*

1.3.1 Question Processing

One of the important tasks at this stage is to find out the expected answer type based on lexical and syntactic clues from the question. For example, in the question "When was Buddha born?", the question word "when" signals that the expected answer will be of the type TIME/DATE. By figuring out this piece of information, we are able to eliminate lot of unnecessary processing. This could also result in the trigger of an answer type specific strategy. The strategy might involve using a specific set of knowledge bases and information sources

(e.g. biography.com for definition questions). Another strategy could be to use a different module. For example, a different set of patterns could be used or reranking could be performed. Complex machine learning is being used for the task of "Question Classification". The main advantages of a learning approach is that it overcomes the limited coverage of the hand crafted rules and are able to handle unseen question types. The question types can range from coarse grained to fine grained. Li and Roth (2002) built classifiers for learning six coarse grained and fifty fine grained question types. These learning methods can utilise lexical features such as the sequence of words and n-grams in the question (Zhang and Lee, 2003), syntactic features like part of speech tags and n-grams over them and semantic relations such as hypernymy from knowledge resources like WordNet [2] (Fellbaum, 1998) to select an answer type from a fixed taxonomy (Loni et al., 2011).

1.3.2 Document Retrieval

The next step is to transform the question into a search engine friendly query and retrieve documents. The transformation can be as simple as removal of unimportant words such as determiners and punctuations which are known as stopwords. Usually one maintains a list of such words and tokens. Another commonly used method is query expansion. A simple strategy can be to include all morphological variants of the non-stopwords (Bilotti et al., 2004). The terms in itself can have different weights associated with them. Relevance feedback is another approach to query expansion. The original query is modified by adding terms extracted from the documents returned by the query. Stemming is also commonly seen in practise at this stage. Usually the corpus from where the information is pulled is also stemmed. The Porter stemming (Porter, 1980) is one of the most commonly used algorithms. However the orthographic algorithm conflates "organization" and "organ". It also fails to transform some words as well. So the performance gain from using a stemmer and query expansion is not guaranteed. Instead of querying the web using a popular search engine, another approach

[2]WordNet: http://wordnet.princeton.edu/

is to use information retrieval (IR) engines like Lucene[3] and Terrier[4] to index and query documents offline from a collection. IR basically is used to coarsely identify a small set of relevant documents and exclude a larger set of irrelevant documents.

1.3.3 Passage Retrieval

From the list of documents returned by the IR engines, top ranked documents are selected and processed. Processing usually involves splitting of sentences and may also involve coreference resolution. Coreference resolution is the process in which we identify the noun phrases that are referring to the same object in the real world (Ng, 2008). For example in the sentence "John promised that he is going to buy Mary a diamond necklace", "John" and "he" refer to the same entity. Usually determining the entity referred to by a pronoun or a noun phrase can be useful. In a question answering system the simplest use could be to determine if the pronoun refers to the target entity in the question. The aim of the passage retrieval step is to identify most promising passages that are likely to be the answers or contain them. Therefore coreference resolution can be useful to reject false positives and identify target specific passages. Although performance improvement might be achieved by using coreference resolution, it comes with an expense in computation. Sliding window based scoring is commonly seen in systems participating in TREC experiments. One of the favoured approaches is to select passages with high frequency of query words.

1.3.4 Answer Extraction

In the case of question seeking a single piece of information, answer extraction is the task of selecting the highest scoring answer. For this type of question the answer type can be extremely useful. The task might be as simple as collecting entities that match the answer type (Abney et al., 2000). Predefined simple patterns can also be used to retrieve answer candidates (Magnini et al., 2002). Data driven methods exploit the redundant nature of the web (or a very large corpora) to

[3]http://lucene.apache.org/java/docs/index.html
[4]http://terrier.org/

single out promising snippets. In case of questions like the definition seeking questions, a list of snippets (e.g. sentences) is gathered.

1.3.5 Answer Reranking

Narrowing down the answer candidates from documents to a small ranked list provides the opportunity to further examine them. In a way we are trying to order the already ordered (ranked) list of answer candidates in terms of its relevance to the question. Short questions tend to give less information and this step can be seen as an attempt to examine if an answer candidate correctly answers a question. Answer reranking can be as simple as a dictionary lookup. Consulting WordNet glosses to compile a dictionary of words and reranking based on the occurrences of these words is one such approach (Lin, 2002). Similarly, the web can be used to compile a dictionary of important words. However, deeper analysis could also be employed as this stage. Quarteroni et al. (2007) defined tree structures to represent the text being reranked. Support Vector Machine (SVM) classifiers using tree kernels to operate on these trees have to be trained. One of the tree representations is able to represent the predicate-argument relations.

1.4 Question Answering at TREC

Research in question answering gained momentum after the National Institute of Science and Technology (NIST) started showing interest in the area specifically since it was added as a track in 1999 (Voorhees, 1999) as part of the Text REtrieval Conference (TREC). The question answering track in the 1999 TREC or TREC-8, only had fact seeking questions or factoid questions as they are more commonly known. Since then every year questions have got tougher and the scope has widened with additions of definition, list and relationship questions.

At TRECs 8 and 9, the task was to return up to five [doc-id, answer-string] pairs for each question, where the answer can be any text string containing at most 50 characters. Here doc-id refers to the document from which the answer was extracted. The nature of the task produced a general pipelined approach for question answering systems. The

process involved question analysis, locating relevant documents and extracting answers. Complexities in the algorithm of the approaches were the result of addition of new requirements. For example, search for a single and short piece of text, as an answer, in place of returning five text chunks. Due to the addition of the list and definition questions, the requirements were different from factoid questions. At TREC 2003 (Voorhees, 2003) definition questions were introduced which in TREC 2004 (Voorhees, 2004) were grouped along with the factoid and list questions according to a common topic or 'target'. TREC 2005 (Voorhees and Dang, 2005) introduced the relationship question track. The task was to retrieve short passages describing relations between two entities. However, most of the approaches were similar to the ones used for other questions.

In the question classification stage the methods used for understanding the questions included simple keyword matching, pattern searching or parsing of the text. Most of the approaches in question processing use named entity recognisers. GuruQA (Prager et al., 2000) uses a simple template based approach. Webclopedia (Hovy et al., 2000) learns patterns from the search engine result. However, the best performance is achieved by using supervised machine learning such as Sparse Network of Winnows (SNoW) and Support Vector Machine (SVM) (Zhang and Lee, 2003). In these, various syntactic and semantic features are extracted to represent the questions such as bag-of-words, bag-of-terms and syntactic tree of the question.

The document retrieval stage saw the use of query expansion by using lexical resources such as WordNet. A simple strategy for expansion could be the addition of synonyms. Greenwood (2004) used pertainyms to improve passage retrieval for questions looking for information about a location. Furthermore, the expansion and retrieval steps can be put in a loop known as a pseudo-relevance feedback loop. Techniques such as query reformulation were also seen. For example, a query "When was Buddha born" could be reformulated into "Buddha was born". The task for the search engine is then to search the new text. Stemming is another one of the commonly used techniques in TREC in the document retrieval stage. The Porter stemmer (Porter, 1980) is probably the most popularly used algorithm. The task of a stemmer is to reduce a word

which can have many morphological variants into its stem or root. For example walks, walking and walked are all reduced to walk. But query expansion and pseudo-relevance feedback largely have not contributed positively overall.

Passage retrieval can be as simple as counting number of terms a passage has in common with the query (Light et al., 2001) or a density based approach (Clarke et al., 2000). Okapi (Robertson et al., 1994, 1995), a popularly used weighting scheme in document retrieval for question answering, has also been used for passage retrieval (Tellex et al., 2003). One way to use Okapi is to treat a document as a collection of passages and score each of them. A passage can be a paragraph or it can be obtained by sliding a window over the document (Llopis and GonzÃ¡lez, 2001) at sentence intervals.

Answer extraction might involve searching for named entities that correspond to the answer type obtained from question classification (Abney et al., 2000). One could also exploit the redundancy characteristics of a very large corpora (such as the web) and use predefined patterns to retrieve short text snippets (Clarke et al., 2000). Redundancy based approaches exploit the simple idea that in a very large document collection, repeated occurrences of an answer might be observed in multiple documents. There have been efforts to perform deep analysis on the question in order to fully understand it as well. Natural language processing tools for tasks such as recognising syntactic alterations, resolving anaphora and abductive proofs were part of the mechanisms in LCCs approach (Moldovan et al., 2002).

Questions like the definition questions require answer reranking. Lin (2002) utilised WordNet glosses to assign scores to answer candidates. WordNet gloss (the definition) for *cancer* is "any malignant growth or tumor caused by abnormal and uncontrolled cell division; it may spread to other parts of the body through the lymphatic system or the blood stream." For each word in the gloss, a gloss weight based on its occurrence in the entire WordNet database is computed. The score of a candidate answer for reranking is the sum of the weights of the matching words between WordNet glosses and answer sentences.

1.5 Motivation

Chapter 2 offers a detailed description on the TRECs view and the general view on definition questions and the answers we expect in return to a definition question. It is accepted that a single piece of information cannot fully answer a definition question. We take a view that a definition question such as "What is X?" can be interpreted as "give definition about X". As Han et al. (2006) puts it, "The definition about X consists of conceptual facts or principal events that are worth being registered in a dictionary or an encyclopedia for explaining X". In this thesis we do not try to outline what facts and events are considered to be "worthy" of an entry in encyclopedias or dictionaries. For our work any snippet which mentions the question target (entity) can be considered as one of the potential answers to a definition question. Due to this nature of definition questions, we end up with many answer candidates. However some definitions (candidates) are richer than others. We make an argument that a worthy definition sentence should be rich in style. Dictionaries and encyclopedias are valuable resources to study the richness expected from a good definition. If we look at Wikipedia articles dedicated to a specific person, it is very likely that the first few sentences (or the first paragraph) will present information regarding his date of birth, country of birth and his/her major achievements. This is consistently observed across introductory paragraph of articles devoted to a person. It is fair to assume that sentences (snippets) carrying important dates (birth,death) and important achievements (awards) are richer definition candidates. For a person trying to find out who Albert Einstein is, snippet mentioning list of countries Einstein visited is likely to be of lesser importance in comparison to his birth date and scientific achievements. However it would be incorrect to assume that such a snippet has no importance. It is just very unlikely. A question such as "Who is Albert Einstein?" does not shed much light on the intent of the user. Therefore people doing research in definition question answering rely on statistics(e.g. repetitions of an information on the web) and popular encyclopedias (e.g. Wikipedia) to learn about the richness of a snippet.

When we give the question "Who is Albert Einstein?" as a query

to a search engine, it returns a list of documents that the algorithm considers to be important. At the basic level the ranking of documents are based on criteria such as the credibility of the source, viewing statistics and number of links pointing to the documents. For example the top three results are Wikipedia page on Einstein, biography page from nobelprize website (`http://www.nobelprize.org/`) and article from biography.com website (`http://www.biography.com/`). If the question was posed to a definition question answering system a popular strategy is to extract the top few hundred documents from the result of a search engine. From this collection of documents promising snippets (sentences, paragraphs) are extracted. One of the common approaches is to use patterns locate promising snippets. However patterns alone are not able to determine the richness of an answer. This is the reason why reranking is performed. Since the number of snippets tend to be very small, deeper processing becomes feasible. Hence the output of an answer reranking step is a list of snippets ordered on their richness. Reranking has been shown to improve performance. Lin (2002) reported a 25% improvement in mean reciprocal rank (MRR) and a 14% improvement in finding answers in the top 5.

In this thesis richness refers to the style of the definition sentences. For example, the sentence "Albert Einstein was born in 14 March 1879." is richer in style in comparison to "14 March 1879 Albert Einstein born was." Although both of these sentences contain same information, the first sentence looks more like a definition sentence. Similarly the sentence "Barack Obama, President of the United States ..." is also a sentence rich in style. Traditionally common surface level linguistic constructs are used to extract definition sentences. Instead of manually crafting such patterns we would like to learn patterns that define the style of a definition sentence. Furthermore we would like to give a higher score to "Albert Einstein was born on 14 March 1879." in comparison to "14 March 1879 Albert Einstein born was." To be able to recognise that the first sentence is better in style than the second, we have to move to a deeper analysis and look at more structured representation such as the syntactic trees of these sentences. In our work we try to learn subtree patterns from a dependency parse tree representation of a sentence. Intuitively this should be able to capture a richer set of

patterns.

In experiments such as TREC, an answer to a definition question is a list of information bearing snippets. As such the positions of the snippets do not affect the score. However retrieving definition snippets and ranking them on the basis of richness is only one facet of the problem. Since an answer is a collection of sentences, ordering becomes an important parameter for readability. We can think of the introductory paragraph of a wikipedia article and other encyclopedias as a summary of important facts and events related to the target of the question. In the area of text summarisation sentence ordering is a necessary component to achieve better readability. Significant improvement in readability can be obtained with proper ordering (Barzilay et al., 2002). This also holds true in case of an answer to a definition question. For example, a definition paragraph about a person tends to start with the birth date. A paragraph of ill ordered sentences can create confusion and degrade the reading experience of the user. If the definition sentences were all extracted from a single document then the task of ordering would be to mimic the original document. However, the reranked sentences are extracted from multiple documents returned by the search engine. Therefore a definition question answering system should have a sentence ordering component as well. The final output of a definition question answering system should try to be coherent similar to summaries. It has been observed that biographical Wikipedia articles follow a general presentation template (Biadsy et al., 2008). For example the birth information most of the time (if not always) precedes the death information. The authors of Wikipedia pages follow guidelines so as to produce consistent looking pages[5]. In our work we look at Wikipedia articles to help compose a coherent definition chunk. Order of sentences in the introductory paragraph of Wikipedia articles is used as a template in the model. We take a view that a good answer to a definition question should resemble the structure of a Wikipedia article.

[5]The page `http://en.wikipedia.org/wiki/Wikipedia:Manual_of_Style/Biographies` sets out the guidelines for English Wikipedia. It clearly mentions what content should go into the introductory section of an article.

1.6 Main Research Objective

In this thesis the main objective is to investigate and address two tasks related to definition question answering: reranking and sentence ordering. The objective is not to build a fully-fledged question answering system which would require considerable resources in terms of time and tailoring. The objective is to explore the challenges put forward by these two tasks, understand the effect of our assumptions and solutions related to them and provide systems that can be baseline for future experiments.

First, we aim at developing an approach to reranking answer candidates for definition questions. More precisely we will investigate how structural features, specifically dependency parse trees, can be utilised in a probabilistic setting. We restrict definition questions to be of the form "Who is X?" and "What is X?". We restrict the definition questions to four entity types: *person*, *company*, *disease* and *rule*.

The second aim is to develop an approach for ordering sentences to produce coherent text for definition questions. We investigate how we can learn ordering from Wikipedia articles. We propose an approach that constructs a graph from the Wikipedia articles and introduce a scoring scheme on it. We stay with the same four entity types as in the reranking experiments.

1.7 Research Question and Outline of the Thesis

Based on the first aim, the main question to be answered in this thesis:
Main Question How can structural features such as the dependency parse tree be used in a probabilistic setting?

In Chapter 4 we formulate a probabilistic framework for reranking candidate answer sentences related to definition questions. We are interested in capturing the style of the definition sentences like the definition language model of Han et al. (2006). These two approaches are similar in their aspiration. The contribution of this thesis is the utilisation of structural features, the dependency parse tree, in a probabilistic setting. By the introduction of a tree, we need to find a

replacement for counting word occurrences to compute the various probability estimates.

RQ I: How to compute probability estimates on trees in a statistical scheme? As an answer to this question, we introduce the notion of similarity in a probabilistic setting. We hypothesise that by introducing the concept of similarity we will further be able to overcome the problem of data sparsity that can be present in strict matching of trees.

In Chapter 5 we investigate the benefit of using the edit distance to compute similarity of trees. We explore how we can compute the various probability estimates based around the calculation of edit distance and its use for reranking. Edit distance has been used previously by Punyakanok et al. (2004) in question answering to measure similarity between the question representation and the candidate answer. However, in our work we use edit distance to compute similarity between a candidate sentence and the training instance (positive example). The reason why we do not measure similarity between a question sentence and the candidate answer is because we concentrate on definition questions which tend to be very short (a few words). Lack of terms cross out the option of comparing question and candidate answer sentences. We utilise a dataset constructed from the web specifically for the answer reranking task. The dataset and tools used in this part of the experiment are described in Chapter 3.

RQ II: How can we use kernel functions and SVMs to gather better probability estimates?

In Chapter 6 we present a machine learning framework for definition answer reranking. We get rid of manually set thresholds in the edit distance approach by using probability estimate from the SVM classifier directly in our scoring mechanism. We use the result of the learnt classifier and fit a parametrised sigmoid function to get the posterior probability. The parameters of the sigmoid function are computed from a held out set and using cross validation. Use of a tree kernel should provide a better intuitive approach to measure similarity in terms of number of common sub-structures. The dataset and tools used in this part of the experiment is described in Chapter 3.

Based on the second aim, the main question to be answered in this

thesis:

Main Question How can we learn to order definition sentences to produce a coherent chunk?

In Chapter 7 we investigate how we can learn to order a set of sentences by analysing Wikipedia articles. We describe our attempt to define a graph framework that utilises sentence ordering information from Wikipedia articles to guide the ordering process. We further define a scoring metric that is introduced as part of the framework. Scoring is based on the notion of relative distance between the nodes and reachability statistics. The dataset and tools used in this part of the experiment is described in Chapter 3.

Chapter 8 concludes the thesis with a summary of Chapter 4 to 7 and the limitations of these works. We explore possible future work at the end of the thesis.

Chapter 2

Definition Question Answering

2.1 Introduction

Definition questions are questions about a target 'X'. Definition questions are important as query logs of various search engines shows that 25% of queries are definition seeking queries (Rose and Levinson, 2004).

The obvious difference from a single fact seeking question is that a single piece of information is unlikely to fulfil the need (information need) of the user. An answer to a definition question is likely to be a short paragraph that defines 'X'. In case the 'X' is a person, a good answer to the definition question would be similar to an entry in an encyclopedia such as Britannica[1]. Following is the introductory paragraph from the biography of Albert Einstein taken from Britannica[2].

"Albert Einstein, (born March 14, 1879, Ulm, Württemberg, Germanyâdied April 18, 1955, Princeton, New Jersey, U.S.), German-born physicist who developed the special and general theories of relativity and won the Nobel Prize for Physics in 1921 for his explanation of the photoelectric effect. Einstein is generally considered the most influential physicist of the 20th century."

The passage contains important dates (birth, death), important achievements (awards) and other notable information. We would expect to see this information in almost all pages dedicated to a famous person. Another highly popular encyclopedia is Wikipedia[3]. The introductory

[1]http://www.britannica.com/
[2]http://www.britannica.com/EBchecked/topic/181349/Albert-Einstein
[3]http://en.wikipedia.org/

paragraph in Wikipedia is longer than a typical encyclopedia entry but it is a collection of similar notable information. If we look at the definition of a company, such as Microsoft, the information contained in the passage is different from a person's description. It contains information such as dates (when it was founded), location (headquarters) and products which we normally only associate with a company.

An obvious difficulty in answering definition questions is that the question sentence gives very little information about what the user wants to know. If the intent of the user is to find a precise piece of information but the motive is not clear from the question then a dialogue would be necessary. This thesis does not address the issue of user-system interaction. In this thesis we assume that a good answer to a definition question should resemble the introduction passages of these carefully crafted encyclopedias. For example, in a definition of a person we could include information regarding his/her education, work, hobbies and achievements. We hold a view that a good definition would include several of such information carrying text snippets.

2.2 TRECs view on Definition QA

TREC 2003 (Voorhees, 2003) introduced the definition question as a task in their evaluation. Among the 50 definition questions selected, 30 were related to *person*, 10 were related to *organisation* and 10 were related to other targets. The task was to find text snippets that provide extended definitions to the definiendum (target) in the AQUAINT corpus. In TREC 2004 (Voorhees, 2004) definition questions were labelled as "other" questions and included as the last question in a series. The task was to find *vital* nuggets (snippets) that did not convey the information found for earlier questions in the series. Systems were also penalised for retrieving nuggets not on the assessor's list. The system was not penalised for retrieving nuggets labelled as *okay*. The objective of the experiment was to find as many *vital* nuggets as possible and as short as possible.

The following scenario guideline was given to the participants of the experiment. It describes the nature of answer expected by the assessors (Voorhees, 2003). This is based on the idea that a good answer to a

definition question has to address the users' need and therefore needs to be adaptive.

> The questioner is an adult, a native speaker of English, and an "average" reader of US newspapers. In reading an article, the user has come across a term that they would like to find out more about. They may have some basic idea of what the term means either from the context of the article (for example, a bandicoot must be a type of animal) or basic background knowledge (Ulysses S. Grant was a US president). They are not experts in the domain of the target, and therefore are not seeking esoteric details (e.g., not a zoologist looking to distinguish the different species in genus Perameles).

ID	Type	Question
1	FACTOID	What division (weight) did he win?
2	FACTOID	When did he win the title?
3	FACTOID	How old was he when he won the title?
4	FACTOID	Who did he beat to win the title?
5	FACTOID	Who beat him to take the title away?
6	LIST	List the names of boxers he fought.
7	OTHER	Other

Table 2.1: *TREC 2004 Question Set for the topic 'Floyd Patterson'*

Table 2.1 gives an example of a set of questions on the topic 'Floyd Patterson'. Table 2.2 gives an example of vital and okay nuggets on the topic 'Floyd Patterson' for the 'other' question. As we can see in Table 2.2, nuggets and the answers to the previous *list* and *factoid* questions do not overlap.

2.2.1 Evaluation Framework

To evaluate the results, human assessors prepared a list of "information nuggets" from the document collection and the list of snippets returned

ID	Category	Nugget
1	vital	had 64 fights
2	vital	wrote book 'Victory Over Myself'
3	okay	Was Golden Glove champ
4	okay	won 13 million dollars
5	okay	managed by D'Amato
6	okay	won Olympic gold in 1952
7	vital	won 40 by knockouts
8	okay	has memory problems
9	okay	described as punch-drunk

Table 2.2: *TREC 2004 Nuggets list for the topic 'Floyd Patterson'*

by the systems participating in the evaluation. Systems are evaluated on the basis of the number of *vital* nuggets identified by the assessors. Recall was defined as the ratio of the number of nuggets retrieved to the total number of nuggets identified by the assessors. Precision was defined on the basis that the user would prefer a shorter answer over a longer one given that they contain the same piece of information. Precision was set to one if the answer snippet was within the predefined *allowance* and scored down using Equation (2.1) for other cases.

$$Precision(P) = 1 - \frac{length - allowance}{length} \tag{2.1}$$

Where,
$length$ = amount of non-whitespace characters in the entire output
$allowance = 100 \times (T_{vital} + T_{okay})$
T_{vital} = Number of vital nuggets returned in the response
T_{okay} = Number of okay nuggets returned in the response
T_{gold} = Number of vital nuggets in the gold standard

Final score assigned to a response was the F-measure given by Equation (2.2).

$$F = \frac{26 * P * R}{25 * P + R} \tag{2.2}$$

Here, recall (R) is weighted as five times more important than precision (P). In TREC 2007, instead of a single assessor providing judgement on whether a nugget is vital or okay, multiple assessors were assigned

External Resource Names	Coverage of Topics (out of 85)
Biography.com (http://www.biography.com/)	19
S9 (http://s9.com/biography/index.html)	15
Wikipedia (http://en.wikipedia.org/wiki/Main_Page)	63
Bartleby.com (http://www.bartleby.com/)	37
Google Glossary (search by "$define :< term >$" in Google)	25
WordNet Glossary	13

Table 2.3: *List of external knowledge sources and their coverage.*

the task. The intuition behind the concept is the observation that the importance of a fact is directly related to the number of people that recognise it.

In TREC-2007, a Nugget Pyramid method was used for evaluating other questions (Lin and Demner-Fushman, 2006). In this, multiple assessors provide judgements on whether a nugget is vital or okay. Then a weight is assigned to each nugget based on the number of assessors who labelled it as vital (Dang et al., 2006). A nugget labelled "vital" by most of the assessors (not necessarily all) would receive a weight of one. Nugget recall was taken as the ratio of the sum of weights of matched nuggets to the sum of weights of all nuggets in the list.

2.2.2 Definition QA at TREC

TREC-2004

In TREC 2004, Cui et al. (2004) extracted documents for the question target (here referred to as sch_term) from TREC corpus as well as six from external sources. Table 2.3 lists the sources that were utilised and the topic coverage.

Sentences are weighed using both the data sets. The weighing method used the collected centroid words. In input sentences, centroid words are words that co-occur frequently with the target. To measure the centroid weight for a word, Mutual Information (MI) was computed. Equation 2.3 was the actual equation used to compute weight for a word

'w'.

$$weight_{centroid}(w) = \frac{\log(co(w, sch\_term) + 1)}{\log(sf(w) + 1) + log(sf(sch\_term) + 1)} \times idf(w) \tag{2.3}$$

Here,
$co(w, sch\_term)$ = Number of sentences where w co-occurs with sch_term
$sf(w)$ = Number of sentences containing w
$idf(w)$ = Inverse document frequency of w

Words whose weight exceeded the average plus a standard deviation were selected as centroid words. Sentences were ranked based on the centroid vector. Their model accounts for the observation that not all of the ranked sentences can be considered as definition sentences although they might be related to the target. Manually coded patterns were used to determine if a sentence was a definition sentence. But this presents an obvious limitation. The main limitation is that it cannot deal with linguistic variations both at the vocabulary level and at the syntactic level. As a solution they proposed a soft pattern matching approach. Given a set of training instances (for a target) a vector representing the soft pattern P_a is generated by aligning the training instances according to the relative position from the target (sch_term). Such a vector representation of a pattern P_a looks like $slot_{-w}, ..., slot_{-1}, sch\_term, slot_1, ..., slot_w$. Here, $slot_i$ is a vector of pairs of tokens with their probabilities.

For a test sentence S with a vector representation $< token_{-w}, ..., token_{-1}, sch\_termtoken_1, token_w >$, the degree of match between S and P_a is based around matching for individual slots as well as sequences of slots. First assuming slot independence,

$$P_a\_weight_{slots} = P(S|P_a) = \prod_{i=-w}^{w} P(token_i|slot_i)$$

To model the sequence of token, tokens to the left and right of the sch_term are considered separately. For the sequence of tokens to the right,

$$\begin{aligned} P(right\_seq|P_a) &= P(token_1, ..., token_w|P_a) \\ &= P(token_1)P(token_2|token_1)...P(token_w|token_{w-1}) \end{aligned}$$

Here, $P(token_w|token_{w-1}) = \frac{count(token_{w-1}token_w)}{count(token_{w-1})}$.

Similarly, $P(left\_seq|P_a)$ is computed for left sequence of tokens. Then overall sequence weight for a vector is calculated as

$$P_a\_weight_{seq} = 0.3 * P(left\_seq|P_a) + 0.7 * P(right\_seq|P_a)$$

As we can see by using 0.7, the right sequence is considered more important. So, the weight of the pattern P_a with length normalisation (fragment_length) is

$$pattern\_weight = \frac{P_a\_weight_{slots} * P_a\_weight_{seq}}{fragment\_length}$$

So the score for a sentence S is

$$score(S, P_a) = 0.4 * weight_{centroid}(S) + 0.6 * pattern\_weight(S, P_a)$$

The value of 0.6 shows the model favours pattern rules and that patterns are important for definition question answering. Their best performance was when patterns were considered more important (0.6) than just word statistics.

TREC-2005

To answer the definition questions, a cascade of filters (CFA) approach was employed by Schone et al. (2005) on the top documents retrieved by the information retrieval engine. The sentence extraction filter is applied on sentences that contain the question noun phrase or its synset synonyms (obtained from WordNet) and a numeric value. Mainly this was useful for "How many ..." type questions. However, this could be useful for other question types as well. They used a modified version of this filter for *other* questions. Basically, sentences containing the target

were saved. A filter that accepted only those candidate sentences with length in between 10 and 250 words was applied next. The number of sentences per topic was set to 50. Another filter, the template matcher filter, was applied on the shallow parse of the question. For the question "How many hexagons are on a soccer ball" the templates generated were (a) "< # > hexagons are on a soccer ball", (b) "soccer ball has < # > hexagons" and (c) "soccer ball contains < # > hexagons". The semantic rules filter tries to eliminate answer candidates using semantic rules. For example, removing sentences whose main verb is not a synonym of the verb in the question. The final filter, the trigram shallow parsing filter, extracts trigrams from the question and the candidate answers. If there is a trigram match between the question and any candidate answer, the rest of the sentences are removed. In the next step, the $< directobject, verb, value >$ triples are extracted from the question and the candidate answers. Matching candidates are kept.

TREC-2006

The method used by Kaisser et al. (2006) for answering definition questions is very simple. Different strategies (in total 8) for creating a query depending on the type of the target were devised. For a target of type *person*, search queries using the targets and quoted targets were created. For example, the queries for the target "Bill Clinton" would be Bill Clinton and "Bill Clinton". For complex target such as "John William King convicted of murder", which is an event, its main noun phrase (NP) was extracted. In this case the queries would be John William King convicted of murder and "John William King". In case the target ended with a preposition phrase (PP), an additional quoted query excluding that PP would be formed. For example, the queries for the target "Great Wall of China" would be Great Wall of China, "Great Wall of China" and "Great Wall".

The queries are fed to a search engine and the top 50 results are analysed. The frequency count of all non-stop words within the results are computed. Now, the AQUAINT corpus is searched for the presence of the target at the sentence level. A manually defined limit of 200 was set for the number of allowable non-space characters in the sentence. If

this limit was exceeded, it was split at punctuation marks. Sentences are then assigned a score which is the sum of all weights of all the web words occurring in it and divided by the length (non-white characters) of the sentence. The highest scoring sentence was then chosen. The weight for each word that is present in this sentence is adjusted. Weights of all non-target words were divided by 5 and by 2 for the target words. Again, all sentences are scored and the highest scoring sentence is selected. The process continues until the length threshold for answer sentences is exceeded.

A definition question answering system defined in Zhou et al. (2006) consists of four modules viz. document processing, web knowledge acquisition, relative terms extraction and definition generation.

The document processing module generates a candidate set of answers based on the target term. It has three steps: document retrieval, candidate sentence extraction and initial score calculation. After a candidate set of sentences has been identified, it is passed onto the initial score calculation module. For a sentence s_i the score is defined as

$$init\_score(s_i) = \theta \times target\_score(s_i) + (1 - \theta) \times doc\_score(s_i) \quad (2.4)$$

The target score(target_score) is based on the occurrence of the target word, phrases and entities. It is computed as

$$target\_score(s_i) = \alpha \frac{c(w)}{n_w} + \beta \frac{c_p}{n_p} + \gamma \frac{c_e}{n_e}$$

Here n_w, n_p and n_e represent the number of words, phrases and Name Entities contained in s_i respectively. $c(w)$, $c(p)$ and $c(e)$ represents the number of the words, phrases and Name Entities that is present in both s_i and the target. α, β and γ are set as 0.3, 0.3 and 0.4 respectively.

The document score (doc_score) is calculated as

$$doc\_score(s_i) = maxdocw(s_i) \times 2 - \frac{2 \times docn(s_i)}{docn(s_i)^2 + 1}$$

Here $docn(s_i)$ is the number of documents returned containing the sentence s_i. $maxdocw(s_i)$ is the maximum score of the documents.

Theta in Equation 2.4 is set to 0.8.

The web knowledge acquisition module collects the definitions for the target using online knowledge bases such as WordNet glosses and encyclopedia.com. Let $s_D = D_1, D_2, ..., D_k$ be the set containing definition sentences collected from these resources. For a candidate answer A_j and a definition D_i, S_{ij} represents the similarity of D_i and A_j based on term frequency and inverse document frequency, *tf-idf*. The score for the candidate answer is then calculated as

$$web\_score(A_i) = \sum_{j=1}^{n} w_j S_{ij} \quad (Note : \sum_{j=1}^{n} w_j = 1)$$

To extract more reliable information than those extracted by using target alone, the relative terms extraction module tries to collect words, phrases and entities closely related to the target. Let $T = t_1, t_2, ..., t_n$ be the words, phrases, and entities in the set of candidate answer sentences $S = s_1, s_2, ..., s_n$. The relativity score $r(t_i)$ for the target and t_i is

$$r(t_i) = \sum_{j=1}^{n} E(t_i, s_j) \times init\_score(s_j)$$

Where,

$$\begin{aligned} E(t_i, s_j) &= 1 \text{ if } t_i \in s_j \\ &= 0 \text{ otherwise} \end{aligned}$$

Based on the score, the top 15 relative words, phrases and entities were selected. For a candidate sentence s_i with n_w words, n_p phrases and n_e entities out of which r_w, r_p and r_e are relative words, phrases and entities respectively, the relative term score for the candidate sentence is

$$relative\_score(s_i) = \alpha * (\sum_{i=1}^{r_w} \frac{r(w_i)}{n_w}) + \beta * (\sum_{j=1}^{r_p} \frac{r(p_j)}{n_p}) + \gamma * (\sum_{k=1}^{r_e} \frac{r(e_k)}{n_e}) \tag{2.5}$$

α, β and γ was set to 0.3, 0.3 and 0.4 respectively.

Finally the definition generation module ranks the candidate sentences based on the linear combination of init_score, web_score and

relative_score.

TREC-2007

The best system of TREC 2007 was Qiu et al. (2007). It used language models and syntactic features to rank candidate answers. First, a question target was fed as a query to the search engine. At most 200 relevant documents were retrieved. Sentences in these documents were checked to see if any noun word found in the sentence appears in the target and if the sentences have more than 70% word overlap with one of the sentences already extracted. If no noun words are found and if there is more than 70% overlap, the sentence is not considered further and ignored.

In the training phase, these retrieved sentences are used as training instances. In testing, the sentences retrieved are split into short snippets using the following regular expression "$(,|-|)$" and the snippet length was restricted to 40. All combination of continuous snippets are taken as candidate answer sentences. Finally, the candidate answer sentences are checked for redundancies and chosen if they fulfil the redundancy conditions.

To obtain a feature based on language models a sentence can be viewed as a sequence of words i.e. $S = w_1 w_2 ... w_n$. $\log P(S|C)$ was computed for the sentence S in a corpus C. In total four corpora were used: AQUAINT, AQUAINT*, definition corpus (DC) and target corpus (TC). To get the AQUAINT* corpus, the named entities (person, location and organisation) are replaced by PRN, LCN and ORG respectively. Furthermore, all numbers are replaced by the token CD. The definition corpus is formed by collecting articles for the target from Wikipedia. The same processing applied to AQUAINT in order to get AQUAINT* is also applied. To get the target corpus, the target was fed as a query to google search. The first 100 result snippets were considered as the target corpus.

By using the definition corpus, the probability that a candidate sentence is a definition can be estimated. Similarly, from the target corpus relatedness between the candidate sentence and the corpus is measured.

To obtain a feature based on syntax of a sentence, minipar (Lin,

1998) is applied to each sentence and a set of $< w_1, relation, w_2 >$ triples is extracted. For any relation rel if there is a triple $< w_1, rel, w_2 >$ such that one of w_1 and w_2 is not a stop word and does appears in the target, $rel(s) = 1$ otherwise it is taken as zero. However, not all relations are useful in finding the correct answer. Chi-square test was performed to select four features, "punc", the appositive "appo", the complement clause of prepositional phrase "pcomp-n" and the grammatical subject "s".

The modelling framework is not clearly mentioned in the paper. However, in the latter part of this thesis we explain one such language modelling framework that they have also used.

As we look at the scenario guidelines, evaluation of definition questions in TREC is subjective. To achieve a good performance in real life with the given scenario a system would require personalisation capabilities. This is beyond the scope of this thesis. Furthermore, TREC evaluation does not take coherence into account. One of the focus of this thesis is to produce coherent definition text. It is also observed that statistical methods have done well in TREC experiments. These systems exploit various online knowledge bases such as the WordNet gloss and online encyclopedias to learn a model for scoring definition snippets. The main work in this thesis is to explore the hypothesis that patterns can be found in definition sentences which are specific to the type of the entities they are definitions of. Below we introduce the approach of a domain specific view to definition question answering.

2.3 Domain Specific View to Definition Question Answering

In this thesis the entity type of the question target is taken to be the domain of the definition question. Domain specific data is utilised to learn the characteristics that we seek in a answer to a definition question. For a single fact seeking question there are criteria to determine what makes a good answer. The answer type is usually a reliable guide that points to the correct section in the text. For definition questions we do not have a standard interpretation about what constitutes a good

answer. In this thesis we take the criteria outlined by Han et al. (2006) to judge whether a sentence qualifies as a good answer. We take a stand that a good definition answer sentence not only has the content (a piece of information related to the question target) but it also adheres to a definition style. A good definition is a collection of several such good definition sentences.

2.3.1 Distributional Hypothesis in Definitional QA

The distributional hypothesis (Harris, 1954) states that words that occur in similar contexts tend to be similar in meaning. To put it in terms of Firth (1957), "You shall know a word by the company it keeps.". Harris (1954) illustrates the concept of relating meaning to the context as follows:

> The fact that, for example, not every adjective occurs with every noun can be used as a measure of meaning difference. For it is not merely that different members of the one class have different selections of members of the other class with which they are actually found. More than that: if we consider words or morphemes A and B to be more different in meaning than A and C, then we will often find that the distributions of A and B are more different than the distributions of A and C. In other words, difference of meaning correlates with difference of distribution.

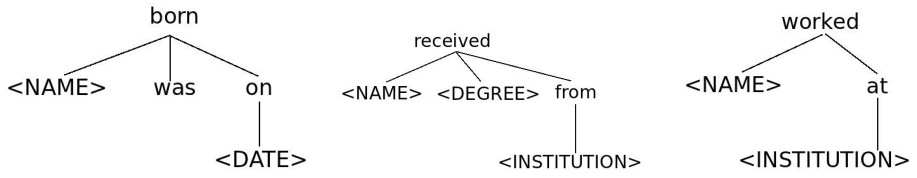
Basically what it means is that words that have similar meaning tend to occur in similar contexts in a large corpus. Harris also proposed the view that the similarity/dissimilarity of meaning can be quantified and determined from the difference in their context. It is stated as follows:

> If A and B have some environments (contexts) in common and some not we say that they have different meanings, the amount of meaning difference corresponding roughly to the amount of difference in their environments (contexts) ...If A and B never have the same environment, we say that they are members of two different grammatical classes ...

This statement proposes that words that occur in similar contexts in a large corpus tend to have similar meaning. The whole area of vector space representation of meaning is based around this distributional hypothesis. One way to construct a vector for an expression is to label in which documents in the collection it can be found and not be found. Another commonly used technique to construct a vector would be to look within a certain fixed window from the expression being investigated. In case of latent semantic analysis (LSA) (Landauer and Dumais, 1997) typically a term-document matrix is constructed. The row belonging to a term (e.g. word) is the vector representation of that term. The value in the cell is the frequency of occurrence of that term in that column (document). Metrics such as cosine similarity are then used to get a numeric value for the degree of similarity.

Previous work by Han et al. (2006) analyses the distribution of words within the TREC corpus. The results showed that the distributional hypothesis for definition question answering is valid. For example we would expect to see words such as "born", "educated" and "elected" frequently occurring in the description of a person whereas we associate words like "group", "profit", "subsidiary" and "commercial" with the description of the companies and businesses. Similarly we use words like "law", "court", "parliament" and "regulations" to describe laws or treaties and "vaccination", "treatment", "medicine" and "symptoms" to describe a disease or illness. Presence of such words strongly signals the entity type that the sentence is talking about.

Lin and Pantel (2001) extends the distributional hypothesis commonly applied to words (to measure word similarity) to paths in dependency trees. They hypothesised that if two paths tend to link the same set of words, the meanings of the paths are likely be similar. The paths in their algorithms consisted of slots to be filled in by words. These words forms the context for the corresponding path. Instead of paths we apply the distributional hypothesis to the trees. We further hypothesise that this distributional hypothesis is true in case of trees and we can expect to find domain specific patterns (sub-structures) from syntactic trees of sentences. For the task of answering definition questions, the domain refers to one of the four entity types. We hypothesise that trees with same set of sub-trees should convey similar



(a) (b) (c)

Figure 2-1: *Dependency tree to capture the patterns: (a)⟨NAME⟩was born on ⟨DATE⟩; (b)⟨NAME⟩received ⟨DEGREE⟩from ⟨INSTITUTION⟩and (c) ⟨NAME⟩worked at ⟨INSTITUTION⟩*

information about the target. We expect to see some subtrees that occur frequently in definition sentences of a specific entity type and less often for other entity types. This is an attempt to identify definition bearing sentences that are popularly captured by using surface text patterns. For example, Ravichandran and Hovy (2002) learnt patterns such as "born in ⟨ANSWER⟩, ⟨NAME⟩", "⟨NAME⟩was born on ⟨ANSWER⟩","⟨NAME⟩(⟨ANSWER⟩" and "⟨NAME⟩(⟨ANSWER -)" for the question looking for birth date as an answer. Here the tokens ⟨ANSWER⟩and ⟨NAME⟩are slots to be filled in. However a strict surface text pattern matcher is not able to overcome the distance between the tokens. Instead of surface patterns, we capture the syntactic information in the sentence by using the dependency tree of the sentence. We look to capture dependencies between words and not rely on the ordering of words in the sentence. Figure 2-1 shows dependency tree patterns to capture the sentences "Alan Turing was born on June 23, 1912 ...", "Turing received PhD from Princeton University" and "Turing worked at Bletchley Park" respectively from left to right. These three patterns are commonly seen in biographies (*person* entity type). The presence of such patterns (sub-trees) is a strong indicator that the entity (denoted by the ⟨NAME⟩) is a person. These patterns form the context for the entity type (*person* in this example).

2.3.2 Types and Characteristics of Definition Questions

The minimal form of a definition seeking question would be "Who is X?", "What is X?", "Describe X" or "Define X". The approaches explored in this thesis look at such short questions.No information apart from the target 'X' is available in the question. This is similar to the TREC evaluation framework. The important processing involved in answering these questions is to determine the type of 'X' (person, organisation etc.). We use a named entity recogniser for this task.

It is also assumed that the question mentions only a single entity for which a definition is being searched. However, this is not always the case. It is not uncommon to see questions such as "Who are X and Y?". The straightforward solution in this case would be to break this question into two, "Who is X?" and "Who is Y?" and treat them independently. However, the ambiguity is present because the user might be interested in 'X' and 'Y' as a single entity like in "Laurel and Hardy" and not definitions of separate entities.

It is also common to see definition questions with user hints. This is usually the case when the user thinks that there might be potential ambiguity. For example television series Friends query log[4] contains the question "Who is Denise, the roommate that Phoebe mentioned in episode 603 & 605?" We can also see questions such as "What are the recent developments on the war on terror?". The cue word "recent" signals that the search should be restricted to recently updated documents. This was very evident from the FAQ dataset constructed by Jijkoun and de Rijke (2005) as well.

Spelling errors and ungrammatical constructs are quite frequent in online documents. Ungrammatical text would result in a wrong parse tree in our case. The reason could be the users' familiarity to searching algorithms and thereby assuming grammar is not much useful. It could also be the case that the language is not the first language for the user. Definitely such kind of queries cannot be taken to be simple and could be quite difficult to interpret.

Figueroa (2010) mentions ten issues that are related to definition

[4]http://www.friends-tv.org/faq.html

questions. These include the observations made by us. We tend to see multiple target questions such as "What does ⟨target_1⟩ and ⟨target_2⟩ mean?" and "What is the definition of/for ⟨target_1⟩ and ⟨target_2⟩?" because it is a convenient way to ask for definitions on several targets. In the case of a compound target such as "Laurel and Hardy" where it refers to a single entity, user might enclose them in quotes making the job easier. Unless there are no results to a compound query, in most cases it is difficult to interpret the goal. Another issue is regarding definition questions with abbreviations such as "What does the abbreviation of ⟨target⟩ stand for?" Although a sentence that expands on the acronym could be retrieved, it is usually the case the user is looking for a longer explanation.

2.3.3 Characteristics of a Good Definition

Many definition question answering systems in TREC have employed hand crafted patterns to locate definition bearing sentences. Hildebrandt et al. (2004) use common linguistic constructs to identify such sentences. For example, appositives such as "Barack Obama, President of the United States ..." and construct like "Rambo also known as John Rambo ..." are mostly used to introduce a person. In this case as well, the underlying intuition is that the rules are domain specific. Han et al. (2006) takes a view that an answer to a definition question should not only have relevant content, but it should also have a good definition style.

With availability of manually crafted large scale encyclopedias such as Wikipedia, statistical approaches to learning such domain specific words or patterns are popular. The aim is to learn the underlying linguistic phenomena by learning from positive examples provided by these resources. In Han et al. (2006), one characteristic of a good answer is that it talks about the target or "topic" we are interested. Second characteristic is that a good answer should be similar to a descriptive sentence.

Suppose for the question "What is Google Inc.?" we obtain the potential answers:
A1: Google Inc. is an American multinational Internet and software

corporation.
A2: Google was first incorporated as a privately held company on September 4, 1998.
A3: Google was founded by Larry Page and Sergey Brin while they were both attending Stanford University.
A4: Google announced major new changes
A5: Page who founded google is responsible for PageRank

Although A4 talks about Google, it is not a good answer as it is not in a good descriptive form and does not explicitly present at least a piece of information. Sentence A5 though it has the topic Google, the information is about a person Page who founded Google. So A4 does not model the definition well where as A5 does not model the topic well. The possible answer candidates that have both qualities are A1, A2, A3.

So the task is to find the answer that maximises the joint probability $P(T, D|S)$ i.e. finding a sentence 'S' that best represents the definition 'D' and best describes the topic 'T'. The sentences are thus ranked using the following score:

$$DS(S) = P(w_{1,n}|T) \times P(w_{1,n}|D) \times P(w_{1,n})^{-2} \tag{2.6}$$

Where,
$P(w_{1,n})$ = Language model
$P(w_1, n|D)$ = Definition model
$P(w_{1,n}|T)$ = Topic model

The language model is a bag-of-words model obtained by assuming that occurrence of words are independent of each other.

$$P_{uni}(w_{1,n}) = \prod_{i=1}^{n} P(w_i) \tag{2.7}$$

So the overall probability of a sequence of words is the product of the individual words' probability. The probability of a word is the ratio of its occurrence in the entire corpus divided by the total number of words in the corpus.

The score from a topic model is obtained from Equation 2.8.

$$P_{uni}(w_{1,n}|T) = \prod_{i=1}^{n} P(w_i|T) \tag{2.8}$$

Like the language model, it is also computed using unigrams. Furthermore, it is a linear combination estimated from three evidences. Here, α, β and γ are empirically set interpolation parameters.

$$P_{uni}(w_{1,n}|T) = \prod_{i=1}^{n} \alpha P(w_i|R) + \beta P(w_i|E) + \gamma P(w_i|W) \tag{2.9}$$

$P(w_i|R)$ is the probability that a word is generated from the top ranked documents retrieved by an information retrieval engine in response to the query 'X' where 'X' is the target of the question. $P(w_i|E)$ is the probability obtained from external resources such as Wikipedia and Biography.com. $P(w_i|W)$ is obtained from top pages retrieved from the web. Each of them are estimated similarly as

$$P(w_i|R) = \frac{C_R(w_i) + \mu P(w_i)}{\sum_j C_R(w_j) + \mu} \tag{2.10}$$

The probability in Equation 2.10 is a smoothed estimated. $C_R(w_i)$ is the number of times w_i occurs in the resource R. μ is a empirically set smoothing parameter.

The definition model is computed similarly to the topic model. The difference is in the resources used for definition modelling. The author gathered definitions for arbitrary targets from online resources that were used in the topic model as well. The definition language model in their QA pipeline models the characteristics of a sentence from their definition corpus (composed of definitions from sources like Biography.com, Columbia Encyclopedia and Wikipedia). The key intuition behind the model is based on the differences in word distribution depending on the entity type of the target. For example we would expect to see words such as "born", "educated" and "elected" frequently occurring in the description of a person whereas we associate words like "group", "profit", "subsidiary" and "commercial" with the description of the companies and businesses. Similarly we use words like "law", "court", "parliament" and "regulations" to describe laws or

treaties and "vaccination", "treatment", "medicine" and "symptoms" to describe a disease or illness.

To capture this hypothesis, the collection was also split into target specific sub-collection. The target type chosen were *person*, *organisation* and *term*. The overall score is a interpolation of the target type specific model (domain model) and the overall model (general model).

$$P_{uni}(w_{1,n}|D) = \prod_{i=1}^{n} \lambda P(w_i|D_t) + (1-\lambda)P(w_i|D_{all}) \tag{2.11}$$

Here, $P(w_i|D_t)$ and $P(w_i|D_{all})$ are the likelihood of locating a word in the corresponding collections.

$$P(w_i|D_t) = \frac{C_{D_t}(w_i) + \mu P(w_i)}{\sum_j C_{D_t}(w_j) + \mu} \tag{2.12}$$

Aligning with the view taken by Han et al. (2006), we agree with the criteria that a good definition sentence should focus on a topic while having a good definition style. Furthermore, we also put an extra condition that a definition sentence should present at least a single piece of information about the target (entity). This was the criteria based on which we collected the training set. In the chapter 3 we describe the dataset used in the experiments that follow. We also believe that the distributional assumption made by Han et al. (2006) should hold true for definition questions. We take one step further and hypothesise that we can see similar behaviour when we look at the structural level of definition sentences. In Chapter 4 we introduce a probabilistic framework that allows for the use of a dependency tree representation of sentences. In Chapter 5 and 6 we explore the consequences of this assumption. We describe the dataset and tools that have been used throughout the experiments in Chapter 3. But before that we describe the evaluation framework followed in the thesis and how and why it differs from the TREC's experiments.

2.3.4 Proposed Evaluation Framework

The evaluation of definition questions in TREC is subjective. The assessors are responsible for determining which nuggets are "vital" and if two nuggets are conceptually the same (or not). It is quite likely that two assessors (and any two persons for that matter) may have different opinions on their interpretation of quality of definition. As Figueroa (2010) points out, one of the major limitations of the TREC evaluation is the lack of clarity in comparing the performance for instances of the same entity types. The same piece of information might make it on to the list of an instance whereas it might not be considered for another (even though it should have been). It is difficult for a system to decide what facts to include or exclude in what seems like an arbitrary decision. The nature of task in this thesis is different to that of TREC. First, we have a domain specific evaluation. Test questions have been grouped into four categories (entity types) viz. *person*, *company*, *disease* and *rule*. Another difference in the proposed evaluation is that a good nugget in our case is the one that resembles part of a definition. In order for a sentence to qualify as a definition, it must present at least one information about the target explicitly. Sentences that only hinted at the information is treated as a non-definition sentence. For example, for the question "Who is Jack Welch?", "Mr Jack Welch, the chairman of GE, has dismissed ..." was labelled as correct where as "Mr Jack Welch, chairman, said the company's ... contributed to the improvement" was labelled as incorrect as it provides incomplete information. In the latter sentence name of the company is missing and therefore it is labelled as a non-definition sentence. We rely on the statistics of the pattern found in the definition to assign importance to that definition. A definition containing frequently occurring pattern (in our case part of a syntactic tree) is assigned a higher score. Evidence from the domain is considered by the scoring mechanism. For example, a pattern might occur frequently in definitions of a particular entity type (*person*, *company*, *disease* or *rule*) and not much in the remaining entity types. This intuition forms the basis for judging the importance of a definition for the entity type in the question (e.g. 'Floyd Patterson' is of *person* type). The thesis therefore focuses on the style of the definition and is not able to judge the quality of a definition apart from

what is given by the statistical evidence. It is therefore not meaningful to compare the proposed system in a TREC style experiment because the judgement of what is "vital" is not the same as if a definition has a good style. A definition with a high scoring style (pattern) could well be non-important (presenting uninteresting information) as per the TREC guidelines. It is our view that a complete definition question answering system (and the evaluation framework) should combine a way to measure both the style and quality ("vital").

There are other important issues in TREC evaluation that make it difficult to be used for definition questions in isolation. For example the goal of the "other" task in the TREC, "fetch vital information not obtained by earlier factoid questions in the series", requires implementation of a filtering mechanism. However it is likely that the previous "vital" information are also pieces of information worthy of being a definition. It also makes the task of comparing systems at TREC tricky. Furthermore, the key task in this thesis is to check if the distributional hypothesis holds true for definitions of a particular entity. As a consequence we thought it would be relevant to have at least four different entity types compared to traditional two in TREC (third category includes all other types). The entity types *disease* and *rule* are not part of the standard TREC experiments. The necessary data (training and testing) has been collected using the Google search engine and the entity instances have been collected from Wikipedia pages. We have chosen MAP (Mean Average Precision) as the main metric for performance evaluation as the F-score does not assess the importance of the ranking order.

Chapter 3

Experimental Setup

3.1 Document Collection for Answer Reranking

3.1.1 Collection used for Experiments

In the first stage of creation of the training and test dataset for an entity type, we manually mined approximately 100 entities from Wikipedia for each of the four entity types viz. *person*, *company*, *disease* and *rule*. Each entity was then fed as a query to the google search engine. The top 50 sentences were stored for further analysis. Each sentence was labelled as a definition sentence or a non-definition sentence based on our criteria. In order for a sentence to qualify as a definition sentence, it must present at least one piece of information about the target explicitly. Sentences that only hinted at the information were treated as a non-definition sentence. This is in line with the evaluation in Moschitti et al. (2007). For example, for the question "Who is Jack Welch?", "Mr Jack Welch, the chairman of GE, has dismissed ..."was labelled as correct where as "Mr Jack Welch, chairman, said the company's ... contributed to the improvement" was labelled as incorrect as it provides incomplete information. We also made sure that the sentences were topic aligned as described by Han et al. (2006). We also constructed a separate held-out set for determining the parameters of a sigmoid function. The sigmoid function is used to estimate the posterior probability. The held-out set contains 51, 60, 47 and 53 sentences for

person, *company*, *disease* and *rule* respectively.

To collect the test dataset we fed the 25 entities to the Google search engine and retained the top twenty sentences. This is then fed as an input to our reranking module. Sentences in the test dataset are also labelled according to the same criteria used for labelling the training data.

In case of edit distance approach, the training dataset was a combined set of training and held-out set.

3.1.2 Data Labelling

The training and test sentences were labelled as either definition sentences (label "1") or non-definition sentences (label "0") by two annotators. In order for a sentence to qualify as a definition sentence, it must present at least one piece of information about the target explicitly. Sentences that only hinted at the information were treated as non-definition sentences. For example the sentence "Millions of people use Microsoft Office to create as many documents every day." is labelled as a non-definition sentence for the questions "What is Microsoft?". From this sentence it is easy to infer that Microsoft has a product named "Office" but it is not explicitly stated. Hence, this sentence is labelled as "0". The sentence "Larry page is the founder of a company" is also labelled "0"for the question "Who is Larry Page?". The sentence does not mention the name of the company where Larry Page is the founder and is regarded as an incomplete sentence. This is the case because this sentence is unlikely to satisfy the need of the person who posed this question. Since we are looking at each sentence in isolation, we do not take into consideration that the next sentence could be "The company is Google". The two sentences taken together explicitly provides a complete information about Larry Page.

The following instruction was provided to the annotators and is same as described in Moschitti et al. (2007).

Sentences are to be labelled as "1"â answered the question either concisely or with noise; the rest â labelled as "0"â are either irrelevant to the question or contained hints relating to the question but could not be judged as valid answers. In order for a sentence to qualify as a

"answer", it must present at least one piece of information about the target explicitly. Sentences that only hinted at the information should be treated as non-definition sentences.

For instance, given the question "What is AIDS?", the sentence "AIDS is everywhere."was labelled "0", while "Genetic research indicates that HIV originated in west-central Africa during the early twentieth century"was labelled "1". Here, the first sentence could not be judged as a valid answer while the latter one is a concise answer (definition).

Similarly, given the question "Who is Jack Welch?", the sentence "Mr Jack Welch, the chairman of GE, has dismissed ..."was labelled as "1"where as "Mr Jack Welch, chairman, said the companyâs ... contributed to the improvement"was labelled as "0"as it provides incomplete information. Without the mention of the company where Jack Welch is a chairman, the sentence is labelled as incomplete.

Only those definition sentences (label 1) that were agreed upon by both the annotators were used for training. We ended up with 558, 572, 473 and 492 definition sentences for *person*, *company*, *disease* and *rule* respectively. Out of the total training data only fourteen instances were found to have been incorrectly labelled by the first annotator. Twelve of the mistakes were processing errors and incorrect sentences were labelled as valid definition sentences. The error resulted from processing of fairly large amount of data and thus human errors were to be expected. Second annotator was able to spot these errors and were fixed. Few of these errors were that the entity in question was different from the entity in the candidate sentence. Another source of error came from the observation that the candidate sentence contained abbreviation of the entity in question. For example an incorrectly labelled sentence for the question "What is British Broadcasting Corporation?" was "The BBC is a semi-autonomous public service broadcaster that operates under a Royal Charter and a Licence and Agreement from the Home Secretary". Although we know BBC is an abbreviation used by the "British Broadcasting Corporation" and therefore the sentence should be a definition sentence. But according to out labelling criteria this information is not explicit in the sentence and the sentence is seen as unrelated to the question and thus labelled incorrect. Two of the

sentences were very noisy and the validity of the labelling could not be resolved by discussion among the annotators and were removed. Similarly in the test data, the sentences were either labelled as a definition sentence or a non-definition sentence. Sentences whose validity could not be resolved by discussion among the annotators were removed. Three such instances were found and the source of error was the noise contained in the sentence. Some of the sentences were extremely noisy because the sentences are collected from the Internet and are not edited. For example "JFE Holdings, JFE ???????????? Jeiefu? H?rudingusu Kabushiki- gaisha , TYO : 5411 is a corporation headquartered in Tokyo , Japan ." was removed from the training set after discussion among the annotators. This problem is usually not seen in popularly used datasets which mainly comprises of newspaper articles.

3.2 Document Collection for Sentence Ordering

3.2.1 Corpus For Learning Text Similarity

The English pages in Wikipedia[1] were indexed using lucene as the first step in creating a vector space model. Lucene[2] is an off the shelf text retrieval engine written in Java. Before indexing, stopwords removal and stemming using porter stemming (Porter, 1980) were performed. The list of stopwords used can be found in Appendix A.

3.2.2 Dataset used for Graph Construction

For the learning phase of the approach, we have used 354, 250, 93 and 101 Wikipedia articles for *person*, *company*, *disease* and *rule* respectively. Only the introduction section of the article is used in the experiments. We consider the introductory section to be a better approximation of a good definition when compared to a shorter encyclopedia entries.

[1]The download date was 2011/07/22. It is available at http://dumps.wikimedia.org/enwiki/20110722/

[2]See http://lucene.apache.org/java/docs/index.html

3.2.3 Dataset used for Evaluation

For evaluation we take articles related to 50 entities for each of the four entity types. Similar to the training phase, only the introduction section of the article is used. There is no overlap between the training and test entities.

3.3 Resources Collected but not Used

We mention the availability of this resource in case it might be useful for other researchers. We have downloaded the collection of frequently asked question and answer pairs collected by Jijkoun and de Rijke (2005). The collection consists of 2,824,179 question/answer pairs in XML format automatically extracted from the Frequently Asked Questions (FAQ) pages[3]. We wrote a script to extract question/answer pairs are store it in a relational database (MySQL). All the questions and answers have been split into sentences and dependency-parsed using the MaltParser[4]. Before dependency parsing was performed, sentence splitting, tokenisation, part-of-speech tagging and named entity recognition was performed using the OAK system[5]. The dependency parsed data is available in MySQL as well as plain text files in the malttab format.

The question table has the following format. qcount is the question id and scount is the sentence number within that question id. word, pos, wordorder, entity store the word, its part-of-speech, its original position in the sentence and entity type (if an entity) respectively. dep is the dependency relation and node is a number indicating the position of the head node in the relationship. The answer table looks similar. The only change is in the name of the field qcount (acount is used).

One of the reason we had to abandon this dataset was because quite a few definition questions also had an associated question. For example Who was Douglas_B._Gardner '_83 and why is the Gardner_Center named for him ?

[3]It is available from: http://ilps.science.uva.nl/resources/webfaq

[4]MaltParser can be downloaded from http://maltparser.org/. We have used version 0.4 written in C for parsing the text.

[5]Tool available upon request. See http://nlp.cs.nyu.edu/oak/

Field	Type	NULL	Key	Default	Extra
qcount	int(11)	YES	MUL	NULL	
scount	int(11)	YES		NULL	
word	varchar(1024)	YES		NULL	
pos	varchar(255)	YES		NULL	
dep	varchar(255)	YES		NULL	
node	int(11)	YES		NULL	
wordorder	int(11)	YES		NULL	
entity	varchar(255)	YES	MUL	NULL	

Table 3.1: *Column description of the question table in the FAQ dataset*

Who is Ayn_Rand ? What else did she write ?
What is Microsoft_Publisher , and Why do I Need It ?
What does TPC stand for ? Who is Arlington Hewes ?

This meant that we would have to go through all the answer sentences and separate out sentences which were an answer to the first part and not the second part. It is a manually time consuming task so we instead tried using the following strict regular expression "^(what)\s+(is|was)+" to only retrieve those questions of the form "what is X?" and "who is/was X?".

We also noticed that the named entity tagger performed poorly for the question as it tried to produced fine grained hierarchy for short definition questions. As a result the manual scanning, selection and correction of a question/answer pair to be used for our experiments was time consuming. Another source of error was in the linguistic analysis of the question. Since the question is posted by a human, questions can have lot of spelling errors, grammatical mistakes and compound information all leading to incorrect linguistic analysis. Following is such an erroneous sentence.

Who is the " Anthony " in Movin ' Out (Anthony 's Song) ——————————————— [Thanks to Zach Pendleton (yellowsnowman @ utah-inter.net) for this info .]

Although we decided not to use the FAQ dataset in the end, they are very useful resources as they specify the real userâs need. We are

only looking at definition questions but the FAQ dataset is full of other complex questions that has not been fully explored by the research community.

3.4 Tools and Libraries Utilised

3.4.1 Linguistic pre-processing

The pre-processing step mainly involves stopwords removal and stemming. Stopwords are identified using a manually crafted list. Appendix A shows the stopword list we used in sentence ordering task. We use the Porter's algorithm for stemming. The tool for stemming can be downloaded from Martin Porter's page[6].

3.4.2 Linguistic processing

By linguistic processing we refer to the common task of sentence splitting, tokenizing, part-of-speech tagging and named entity recognition. We have used the toolkit provided by Sekine (2008) for all of the mentioned tasks. The major reason for using this tool was the named entity tagger. Their tool can identify approximately 150 kinds of named entities (Sekine, 2008). The tool is part of their research to develop an extended named entity hierarchy of more than 200 named entities. For the definition of the entities in their extended hierarchy see `http://nlp.cs.nyu.edu/ene/version6_1_0eng.html`.

3.4.3 Dependency Parsing

We have used MaltParser (Nivre et al., 2006) for dependency parsing of sentences in our reranking task. We have use the version 0.4 which is in C[7]. We have used the pre-trained parsing model for English available from their website.

[6]See http://tartarus.org/martin/PorterStemmer/

[7]It can be downloaded from `http://w3.msi.vxu.se/~nivre/research/MaltParser.html`

3.4.4 Edit Distance

Tree edit distance is calculated using the tool provided by Stephen Wan[8]. It implements the Zhang and Sasha dynamic programming based algorithm (Zhang and Shasha, 1989) for ordered labelled trees.

3.4.5 Support Vector Machine

We have used the tree kernel package by Alessandro Moschitti (Moschitti, 2006b) built on top of SVM-light (Joachims, 1999) in the reranking task. The tool can be downloaded from `http://disi.unitn.it/moschitti/Tree-Kernel.htm`.

3.4.6 Vector Space Representation

We have used the semanticvectors package[9] (Widdows and Ferraro, 2008) to build the vector space representation from the Wikipedia text. Their library includes implementation of various algorithms but we use the random projection technique only.

[8]See `http://web.science.mq.edu.au/~swan/howtos/treedistance/package.html`

[9]Available at: http://code.google.com/p/semanticvectors/

Chapter 4

Probabilistic Framework for Reranking Definition Sentences

The work on question answering has come a long way from solely relying on heuristics to using sophisticated statistical methods. This is also evident from the systems we see in TREC experiments. Instead of relying on experts to spell out the heuristics (the surface level patterns) the field now relies on statistical methodologies to capture the underlying linguistic phenomena. In the subsequent two chapters we investigate two ways to utilise and learn patterns over trees. In this chapter we outline the probabilistic framework for reranking answers to a definition question which forms the basis for those approaches.

4.1 Probabilistic Framework for Reranking Sentences

The sentence reranking task is to model the probability of an answer candidate sentence 'S' given an entity type T_s for a question 'Q'. Sentence reranking is the task of estimating $P(S|T_s)$ for a given definitional question on the entity type T_s. To estimate this model, we could represent a candidate sentence as a sequence of words. This is a popular way to calculate the probability estimate for a sentence using word statistics. This representation overcomes the problem of data sparsity. A sentence might not be observed enough in the corpus to reliably compute the estimate whereas a word is likely to be observed more

EntityType	Description	Example
Person	Name of a person including legendary or fictional characters. Nicknames are also included.	George W. Bush, Bush, George, Edgar Allan Poe, J.Lo, Jennifer, Jen
Organisation	Names of organisations that consist of more than one person	United Airlines, Citibank, Verizon, Coldwell Banker, US Air Force, BBC, GHQ
Disease	Names of diseases and injuries	myocardial infraction, stroke, aphasia, cold, facial neuralgia, heart failure
Rule	rule, constitution, treaty, law, bill	Constitution of Japan, U.S.-Japan Security Treaty, Anglo-Russian Entente

Table 4.1: *Description of Entity Types*

frequently. In our model we deviate from this path and represent the sentence by its dependency parse tree (instead of a sequence of words). We explain what a dependency parse tree is and how we count such trees in subsequent sections.

Apart from the sentence S there is another term in the model, the entity type T_S. In a general setting T_s could be described as the domain where the sentence S is likely to be observed. In this thesis T_s can refer to one of the four entity types viz. *person*, *organisation*, *disease* and *rule*. A named entity tagger developed by Sekine (2008) was used to determine the entity type of the question target 'X'. Table 4.1 lists the short description of the four entities as defined bySekine et al. (2002) [1].

Assuming that a single feature f_s is used to represent the candidate sentence 'S', sentence ranking can be done based on the estimate of $P(f_s|T_s)$. As mentioned earlier, f_s in our model is a dependency tree representation of the sentence S. To estimate this conditional prob-

[1] Details can be found at `http://nlp.cs.nyu.edu/ene/version6_1_0eng.html`

ability we utilise $C = \{f_1, f_2, ..., f_{|C|}\}$ which is a set of features from the collection of positive examples belonging to the entity type T_s. Therefore $P(f_s|T_s)$ can be estimated as

$$P(f_s|T_s) = \sum_{i=1}^{|C|} P(f_s, f_i|T_s) \tag{4.1}$$

Assuming f_s is conditionally independent of T_s given f_i we get,

$$P(f_s|T_s) = \sum_{i=1}^{|C|} P(f_i|T_s) \times P(f_s|f_i) \tag{4.2}$$

We now apply Bayes' rule to invert the conditional probability $P(f_i|T_s)$,

$$P(f_i|T_s) = \frac{P(T_s|f_i)}{P(T_s)} \times P(f_i) \tag{4.3}$$

The expression to be used to rank a candidate sentence S, therefore is,

$$P(f_s|T_s) = \sum_{i=1}^{|C|} \frac{P(T_s|f_i) \times P(f_i)}{P(T_s)} \times P(f_s|f_i) \tag{4.4}$$

4.2 Computing Estimates for Definition Sentences

4.2.1 Working with Trees

A popular approach is to consider a sequence of words as the appropriate representation of a sentence. There are few important reasons for doing this. A large corpus of words is readily available and by adding a strong independence assumption we almost ensure that the probability of a sentence is never zero. When we work with trees we cannot be certain that we will get a non-zero estimate. It is unlikely that a random sentence will have an exact matching sentence in the collection. However, it is plausible that we will find at least one sentence

that has certain sub-section of the tree in common. This is why we get a similarity function to approximate the probability estimate. This is explained in the subsequent sections. Throughout our experiments, the feature representing a sentence is a dependency parse tree. In this section we define what a dependency tree is and how the framework can be adjusted and various statistics can be approximated to allow to work with trees.

Dependency Tree

From a syntactic analysis point of view, we can describe the structure of a sentence from two directions. The first approach is to break a sentence into constituents and further into smaller constituents. Such a way of analysing sentences is called a phrase structure grammar (Chomsky, 1985) as depicted in Figure 4-1. Table 4.2 lists the meaning of node labels. Here, NP, VP and PP are phrase labels and the rest are word labels (part-of-speech).

Another approach is based on binary links (known as dependencies) between lexical elements. Such a way of representing sentence structure is called dependency grammar (Nivre, 2005). The main idea behind this is that the syntactic structure contains binary asymmetrical relation between a word (head) and another word(modifier). Figure 4-2 shows the equivalent dependency representation of Figure 4-1. Table 4.3 lists the meaning of node labels.

Label	Description	Tree Node
NP	Noun Phrase	NP1, NP2, NP3, NP4
VP	Verb Phrase	VP1, VP2
PP	Prepositional Phrase	PP1, PP3
VBD	Verb, past tense	VBD
VBN	Verb, past participle	VBN
IN	subordinating conjunction	IN1, IN2
NNS	Noun, plural	NNS
NNP	Proper noun, singular	NNP1, NNP2
DT	Determiner	DT
CC	Coordinating conjunction	CC

Table 4.2: *Description of node labels in Tree 4-1*

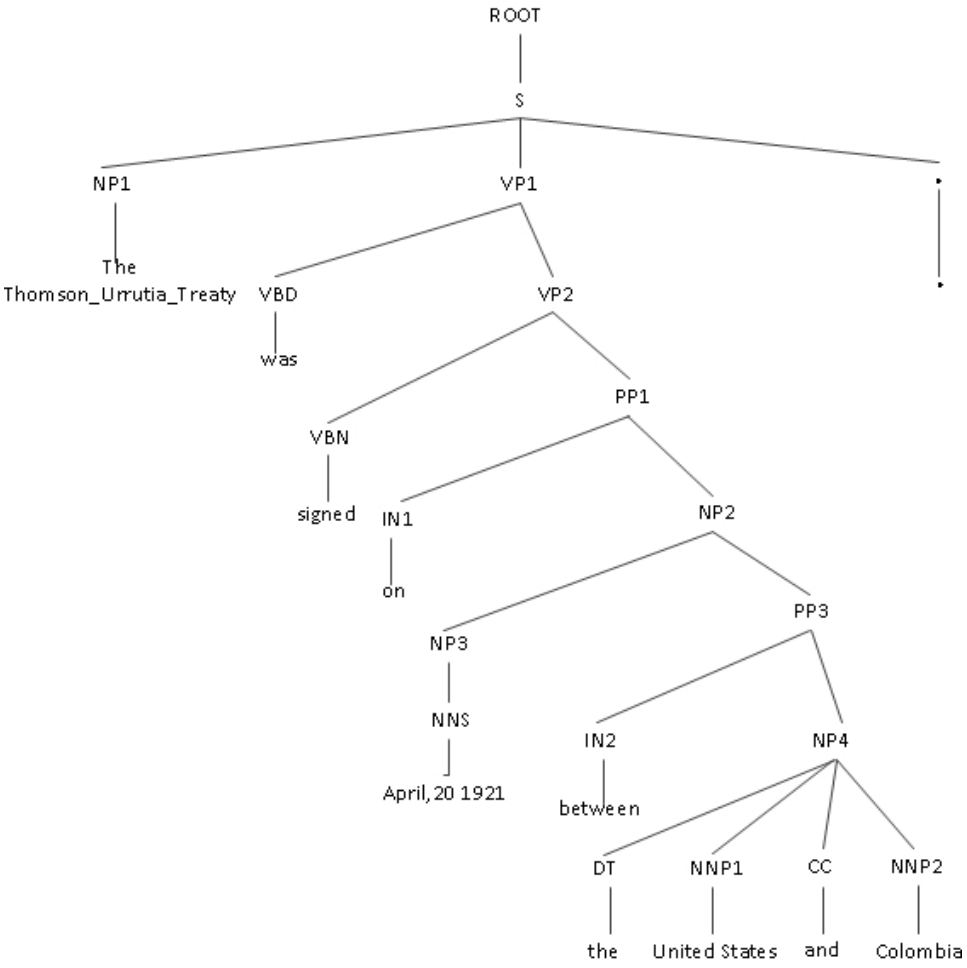


Figure 4-1: *Parse Tree Representation of a Sentence*

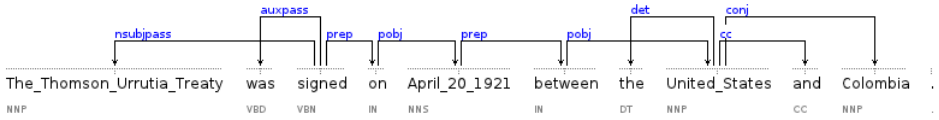


Figure 4-2: *Dependency Tree Representation of the Sentence*

Relation	Description
nsubjpass(signed-3, The_Thomson_Urrutia_Treaty-1)	passive nominal subject
auxpass(signed-3, was-2)	passive auxiliary
root(ROOT-0, signed-3)	ROOT
prep(signed-3, on-4)	prepositional modifier
pobj(on-4, April_20_1921-5)	object of preposition
prep(April_20_1921-5, between-6)	prepositional modifier
det(United_States-8, the-7)	determiner
pobj(between-6, United_States-8)	object of preposition
cc(United_States-8, and-9)	coordination
conj(United_States-8, Colombia-10)	conjunct

Table 4.3: *Description of node labels in Tree 4-2*

The short description of dependency relations listed in the Table 4.3 taken from de Marneffe and Manning (2008) are:

- A passive nominal subject is a noun phrase which is the syntactic subject of a passive clause.
- A passive auxiliary of a clause is a non-main verb of the clause which contains the passive information.
- The root grammatical relation points to the root of the sentence. A fake node âROOTâ is used as the governor.
- A prepositional modifier of a verb, adjective, or noun is any prepositional phrase that serves to modify the meaning of the verb, adjective, noun, or even another preposition.
- The object of a preposition is the head of a noun phrase following the preposition, or the adverbs “here” and “there”.

- A determiner is the relation between the head of an NP and its determiner.
- A coordination is the relation between an element of a conjunct and the coordinating conjunction word of the conjunct.
- A conjunct is the relation between two elements connected by a coordinating conjunction, such as âandâ, âorâ, etc

The main attraction of dependency representation from an application point of view is that the dependency links are closer to the semantic relationships which are the next level up in processing terms. Specifically dependency parsing can be seen as a natural step towards obtaining the predicate argument structure. To add to this, dependency structures are popular in question answering due to the fact that it overcomes some of the limitations of bag-of-words matching implicitly.

Exploitation of Dependency Trees in Question Answering

By learning over dependency trees we should be able to overcome the limitations of a bag-of-words approach. For a question "What is Adobe systems?", a bag-of-words approach can rank either of the following two sentences at the top.

1. "Analysts said the tone for the session was set overnight by word from Adobe Systems, a computer software company, that ..."
2. "The Cupertino, Calif., firm said the year-ago results included $48 million from the sale of stock in Adobe Systems Inc."

Here, only the first sentence contains an actual definition snippet ("Adobe Systems, a computer software company"). We expect bag-of-words model to fail at recognising that the first one is better than the second because both sentences mention the target named entity "Adobe Systems" and both of them have domain specific terms such as "software", "company", "firm" and "stock" which we normally associate with *company* entity type. The idea behind learning from trees is that the proposed model will be able to learn discriminating features (sub-structures) from dependency tree for an entity type, *company* in this case.

In the PIQASso question answering system (Attardi et al., 2001) the dependency parsed information of a candidate answer sentence is checked for the presence of a relation extracted from its question. For example in the question "Who killed John F. Kennedy", a good candidate sentence will contain the answer bearing lexical element as a subject of the verb "kill" and "John F. Kennedy" in the object relation.

Punyakanok et al. (2004) viewed the problem of selecting a candidate answer for a given question as the task of measuring the distance between the dependency trees of the question and the candidate answer. To measure the similarity between trees they use edit-distance with manually tuned parameters as the cost function. Edit distance is basically the cost of transforming one tree into another. Three operations are permitted in their implementation of edit distance viz. deleting, inserting and changing of a node. The idea in this approach is to incorporate syntactic information by the use of dependency trees and add semantic information (named entity, synonyms) to the nodes. The task of finding the final answer requires matching of nodes in the question and answer. The chosen answer is the one with the smallest tree distance. It is evident from their performance improvement that the head-modifier relationship provides an additional level of disambiguation which bag-of-words model would not detect.

By using dependency relations one can overcome the variation that a bag-of-word approach will not catch. However the same semantic relations can be expressed by many different dependency patterns. For example, "X wrote Y" is equivalent to "X is the author of Y". Bouma et al. (2005) overcome this variation by incorporating a set of equivalence relations over dependency patterns for Dutch QA. Dependency trees can also be viewed as a set of tuples of the form $\langle Head, Rel, Dep \rangle$, where Head is the root form of the head of the relation and Dep is the head of the constituent that is the dependent. Their QA system, Joost, works first by dependency parsing the question to identify the question class. For each question class, one or more syntactic patterns are defined. The answer identification process typically involved extracting relevant snippets from the relevant documents, dependency parsing it and matching against set of syntactic patterns. In case there were multiple answers, ranking was done by considering features like

proportion of dependency relation match, proportion of proper names, nouns and adjectives, syntactic context of answer, frequency of the answer and search engine score for the answer.

It is observed that a strict matching of relations may not be always successful. For example an equivalent relations can be specified differently. Cui et al. (2005) proposes a fuzzy relation matching that is based on statistical models. The matching process involves extracting paired corresponding paths from the dependency tree. The paths are paired by matching their nodes at the ends. The matching score is then computed using the IBM translation model 1 (Brown et al., 1993). Matching score of a relation path from a candidate answer can be seen as the probability of translating to it from its corresponding path in the question. This approach depends on matching paths based on the question terms. In case there are few matched question terms or due to paraphrasing relation paths cannot be paired. The solution proposed by the authors is to use query expansion.

4.2.2 Getting Similarity in the Mix

Data sparsity is a problem associated with statistical methods in natural language processing. It is not uncommon to find a sentence that has never been seen in the training corpus. As a result statistical methods based around relative frequency counts will not work. One of the ways around the problem is to model the relationship between features by looking at features that are similar in some way to the ones being examined. The conditional probability $P(f_s|f_i)$ in our model suffers from the problem of data sparsity. In a maximum likelihood setting the probability $P(f_s|f_i)$ would be

$$P(f_s|f_i) = \frac{c(f_i, f_s)}{c(f_i)} \tag{4.5}$$

Here $c(f_i, f_s)$ is the frequency of (f_i, f_s) and $c(f_i)$ the frequency of f_i in the training corpus. For an unseen pair of features, the estimate is zero. However in our formalism, we approximate this conditional probability $P(f_s|f_i)$ as

$$P(f_s|f_i) \approx sim(f_s, f_i) \tag{4.6}$$

Here $sim(f_s, f_i)$ is the similarity score of (f_s, f_i). When $sim(f_s, f_i) \approx 0$ it means that the tree representations of the two sentences are completely different. In this case $P(f_s|f_i)$ should go to $P(f_s)$ as f_i is irrelevant to the estimation of f_s. Similarly if $sim(f_s, f_i) \approx 1$ then $P(f_s|f_i) \approx P(f_i|f_i) = 1$. Therefore this approximation of probability by using a similarity function although not precise is able to capture the spirit for the task of ranking. It is obvious that the estimation becomes a score rather than a probability in our case. Since we are concentrating on a ranking task, we do not require exact probabilities as long as the differences are maintained. In a way what this expression states is that if two objects are similar the chances of finding them together (in a common entity type) is also high and vice versa. It is not uncommon to see similarity function being used for probability estimation. Lee (1999) uses similarity function in the calculation to estimate the probability to account for unseen occurrences. This is similar to what we are trying to achieve. We are using the sentences in a specific domain (entity type) to estimate the probability that a given sentence could be found in that domain. For a interesting discussion on how conditional probabilities can be derived from absolute probabilities and similarity functions, readers are encouraged to see Blok et al. (2003).

Since we are only interested in ranking, we have approximated the estimates $P(f_s|f_i)$ and $P(f_i|T_s)$. The nature of the task allows us to introduce the notion of similarity in measuring $P(f_s|f_i)$. There are two key advantages with this formulation: (1) it helps overcome data sparsity and (2) it acts as a smoothing mechanism. Similarity values range from 0 to 1, no match to a perfect match. It is highly likely that in a collection of trees, we will find a tree that has a fragment in common with the tree being examined. This ensures that the probability estimate is almost always non-zero. Depending on the task, the type of function chosen to model similarity can be plugged in the model. Another advantage of this relaxation allows us the flexibility to choose any kind of feature as long as a similarity function can be defined on them.

To get the similarity function in the range of [0,1], an approximate probability measure, we compute the normalised measure using Equation (4.7).

$$P(f_s|f_i) = \frac{sim(f_s, f_i)}{\sqrt{sim(f_s, f_s) * sim(f_i, f_i)}} \tag{4.7}$$

The answer ranking model by Ko et al. (2010) incorporated similarity functions to measure answer correctness in a statistical setting. The similarity function is a scoring function that calculates similarity between candidate answers. Their main objective is to exploit the redundancy in the candidate answer set. This is different from our usage where similarity is directly taken to be an approximate estimate of the conditional probability. In this thesis the score for a sentence determines the likelihood of it being a definition sentence in the given named entity class. Also the similarity measurement is between a candidate answer and positive examples belonging to the entity type in the question. Furthermore they do not look at structural features. They use metrics such as Levenshtein distance, cosine similarity and Jaccard similarity for strings. In addition to these, two strings are deemed similar if the former is a synonym of the latter. These metrics were appropriate as they only look at list and factoid questions whose answers are short text. However for complex questions such as definition questions where the answers are long sentences, these features are not enough by themselves.

4.2.3 Smoothing

Statistical modelling approaches suffer from a sparse data problem (a lot of zero frequency counts). In a maximum likelihood estimation setting there can be many events where the count is zero and hence the estimate will be zero as well. This is usually not desirable and for that reason smoothing is necessary. Basically we are trying to make probabilities that are zero to be non-zero. The resulting probabilities will not be very high but are better than zero.

Add One Smoothing

The basic idea is to assume that every seen or unseen datum/event occurred once more than it did in the training corpus. For example, in case of a corpus with N tokens and a vocabulary (word types) of V,

equation (4.8) is the standard maximum likelihood estimate without smoothing and (4.9) is the case when smoothing is applied.

$$P(w_i) = \frac{C(w_i)}{N} \tag{4.8}$$

$$P(w_i) = \frac{C(w_i) + 1}{N + V} \tag{4.9}$$

The advantage of add one smoothing is it is simple to implement. The main disadvantage is that it moves too much mass from seen events to unseen events.

Dirichlet Smoothing

Consider the case of the document retrieval task. The smoothing used in typical language models are length independent. The problem is serious in the case of a collection that has big variation in length of documents. Maximum likelihood estimates will favour shorter documents. It might be favourable to use smoothing that allows document dependent parameters. Dirichlet smoothing (Zhai and Lafferty, 2004) is an example of such a strategy. Since a language model is a multinomial distribution, the conjugate prior for Bayesian analysis is the Dirichlet distribution (MacKay and Peto, 1994). The parameters of the Dirichlet are $(\mu p(w_1, C), \mu p(w_2, C), ..., \mu p(w_n, C))$. Therefore the estimate of $P(w_i|d)$ is given as

$$P(w_i|d) = \frac{c(w_i; d) + \mu P(w_i|C)}{\sum_{w_i \in d} c(w_i; d) + \mu} \tag{4.10}$$

4.3 Related Work

A language modelling approach to estimate $P(S|T_s)$ was taken by Han et al. (2006). This is known as the definition language model. By considering a sentence 'S' to be a sequence of words $w_{1,n}$ they instead compute $P(w_{1,n}|T_s)$. Furthermore they assume the word occurrences to be independent of each other. This is a strong assumption that ignores the word order. Using this assumption and applying the chain rule they

rewrite the definition language model as:

$$P(w_{1,n}|T_s) = \prod_{i=1}^{n} P(w_i|T_s) \tag{4.11}$$

The probability $P(w_i|T_s)$ is obtained from external knowledge resources such as Wikipedia and Biography.com as well as from results of the search engines. Further explanation on this framework was explored in Section 2.3.3.

The probabilistic framework we present here is trying to capture the exact same behaviour that Han's definition language model tries to. It is trying to score a sentence (answer candidate) by comparing it with positive examples in T_s. It also disregards the relationship between the question entity and the entity in the sentence. Like Han's definition language model we are interested in the style of the sentence. A candidate sentence (S) will be judged to be similar to a definition sentence in the domain (T_s) if their dependency parse tree matches. The difference from Han's model begins with the choice of feature used to represent the sentence. They represent a sentence by its sequence of words whereas we represent a sentence by its dependency tree. Although it might seem as a trivial replacement, the tree structure forces us to find a replacement for counting word occurrences to compute the probability estimates. As we will see in Chapters 5 and 6, we introduce the notion of similarity to estimate the conditional probability distributions. Similar is the difference from the work of Whittaker et al. (2005) and Heie et al. (2010) where they refer to $P(S|T_s)$ as the retrieval model. Their model is essentially a language model.

4.4 Conclusion

The approximate probability estimate formulation for a candidate sentence introduced in this chapter is a relaxed estimate useful for the ranking task. The nature of the task allows us to introduce similarity into the mix as well as make independence assumptions. The ranking scheme can be used for other question types, apart from the definition questions. The requirements are that (1) it must support the distributional hypothesis and (2) similarity function can be defined over the

features. The main advantage of this approach is the flexibility in the use of features. Although we look at only syntactic trees, deeper understanding can be potentially achieved in this statistical framework by using richer structured features. In the next chapter we introduce edit distance between dependency trees as one way to compute tree similarity. Various conditional probabilities introduced in this model are estimated based on the edit distance between syntactic trees.

Chapter 5

Incorporating Tree Edit distance in the Framework

In this chapter we investigate the performance of using the edit distance as a measure of tree similarity in a probabilistic setting. Tree edit distance is an extended take on edit distance between two strings. The estimates for various conditional probabilities introduced in the probabilistic framework are calculated on the basis of edit distance scores.

First we introduce the algorithm behind the tree edit distance and follow with its incorporation in our proposed framework.

5.1 Edit Distance

Edit distance or Levenshtein distance (Levenshtein, 1966) is well defined in the case of strings where the metric measures the differences between two strings. The difference is measured as the number of transformations needed to transform one string into the other. In a usual setting for strings, the allowable transformations include insertion, deletion or substitution of a character. Depending on the application, the cost associated with each of the three transformations can be different. Edit distance therefore is the total cost of all the applied transformations.

For example, the edit distance for the transformation from "walkin" to "walking" is 1. The transformation involves addition of a single char-

acter 'g' at the end. Similarly, the edit distance for the transformation from "walking" to "walkin" is also 1. However, the transformation here is the deletion of a character. Although the number of transformations are same in both the cases, the cost associated with the transformations can be different.

In our ranking scheme, we define edit distance over a pair of trees.

5.1.1 Tree Edit Distance

Edit distance for trees is simply the cost involved in transforming a tree into another. Similar to the case of a string, various transformations are defined each with their own cost. Commonly three operations are permitted on the nodes: relabelling (substitution), insertion and deletion. In our experiment, we also consider these three transformations. The three operations are defined as following:

Relabelling : Change the node label l_1 to l_2 in tree T. Figure 5-1 depicts the result of the relabel operation on a tree.

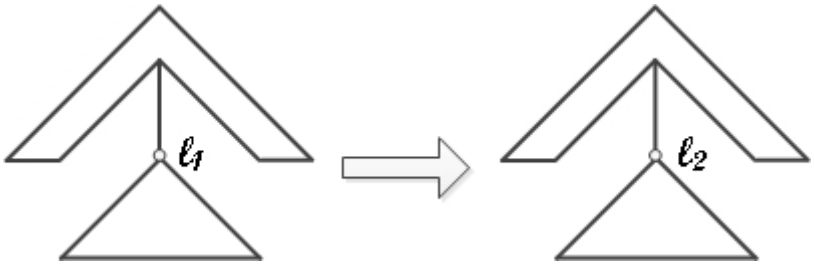


Figure 5-1: *Relabelling label from l_1 to l_2*

Insertion : Inserting a node l_2 as a child of l_1. Figure 5-2 depicts the result of the insert operation on a tree.

Deletion : Deletion of a non-root node l_2 in T. Deletion of node l_2 results in the children of l_2 being adjusted such that they are now the children of the parent of l_2. Figure 5-3 depicts the result of the delete operation on a tree.

In our work, the distance between the trees is based on the dynamic programming based algorithm proposed by Zhang and Shasha (1989) for ordered labelled trees. Ordered labelled trees are trees where the nodes are labelled and left to right order among the siblings is significant. More formally, if T is a rooted tree we call T a labelled tree if

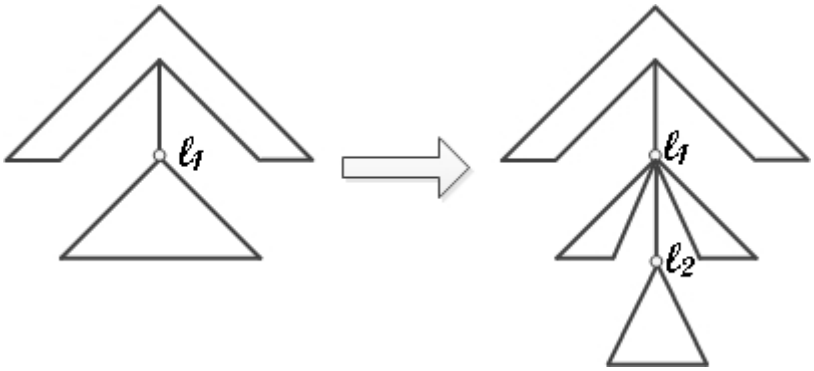


Figure 5-2: *Inserting a node l_2 as a child of a node labelled l_1*

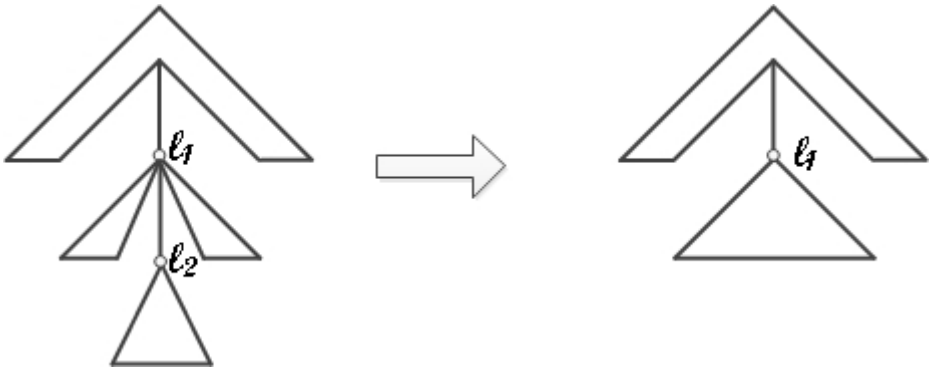


Figure 5-3: *Deletion of the node l_2*

each node is assigned a symbol from a fixed finite alphabet Σ. We call T an ordered tree if a left to right order among siblings in T is given. In this thesis the ordered labelled trees are dependency parse trees.

To explain the edit distance of trees, we use the notation from Tai (1979). An edit operation is represented by $b \rightarrow c$ where b and c are nodes or a null node (Λ). The edit operations as illustrated in the figures can be formally defined as following. $b \rightarrow c$ is a relabelling operation if $b \neq \Lambda$ and $c \neq \Lambda$. It is a delete operation if $b \neq \Lambda = c$ and it is an insert operation of $b = \Lambda \neq c$. Let $S =< s_1, s_2, ..., s_m >$ be a sequence of edit operations that transforms a tree T_1 to T_2 and $\gamma(b \rightarrow c)$ the cost function. The cost of sequence of operations is therefore

$$\gamma(S) = \sum_{i=1}^{m} \gamma(s_i) \tag{5.1}$$

The distance $\delta(T_1, T_2)$ from tree T_1 to T_2 is defined to be the mini-

mum cost of all sequences of edit operations that transform T_1 to T_2.

$$\delta(T_1, T_2) = min\{\gamma(S)|S\} \tag{5.2}$$

The problem here is that the number of different sequences of edit operations that can transform a tree to the other can be infinitely large. Hence it is not possible to enumerate all valid sequences and find the cost. However, if the cost function satisfies the triangularity property, that is $\delta(b \to c) \leq \gamma(b \to a) + \gamma(a \to c)$ then Tai (1979) showed that the minimum cost is equal to the minimum cost of a mapping.

A mapping is a triple (M, T_1, T_2) where M is any set of pairs of integers satisfying

1. $i_1 = i_2$ iff $j_1 = j_2$
2. $i_1 < i_2$ iff $j_1 < j_2$
3. $T_1[i_1]$ is an ancestor of $T_1[i_2]$ iff $T_2[j_1]$ is an ancestor of $T_2[j_2]$

The cost of mapping M is

$$\begin{aligned}\gamma(M) = &\sum\nolimits_{(i,j)\in M}\gamma(T_1[i] \to T_2[j]) + \sum\nolimits_{(i,j)\in I}\gamma(T_1[i] \to \Sigma) \\ &+ \sum\nolimits_{(i,j)\in J}\gamma(\Sigma \to T_2[j])\end{aligned} \tag{5.3}$$

Here I is the set of index of nodes in T_1 not mapped by M and J is the set of index of nodes in T_2 not mapped by M. The idea is clearer from an example. Consider the trees T_1 and T_2 in Figure 5-4. A dotted line from $T_1[i]$ to $T_2[j]$ indicates that either $T_1[i]$ should be changed to $T_2[j]$ if $T_1[i] \neq T_2[j]$ or $T_1[i]$ is changed to $T_2[j]$ if $T_1[i] = T_2[j]$.

Nodes of T_1 not touched are to be deleted and nodes of T_2 not touched are to be inserted. We then get the trees as in Figure 5-5.

For our experiments, the cost of inserting and deleting a node was set as 1. The cost of relabelling is 0 if the nodes have same label and 1 otherwise.

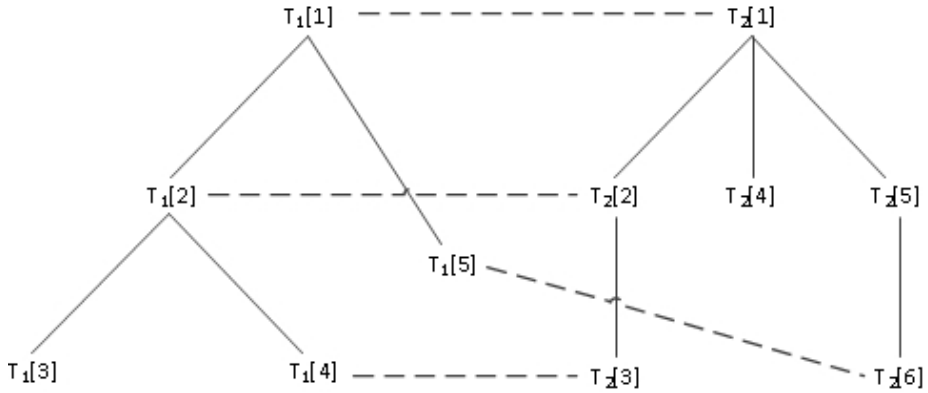


Figure 5-4: *Original trees T_1 and T_2*

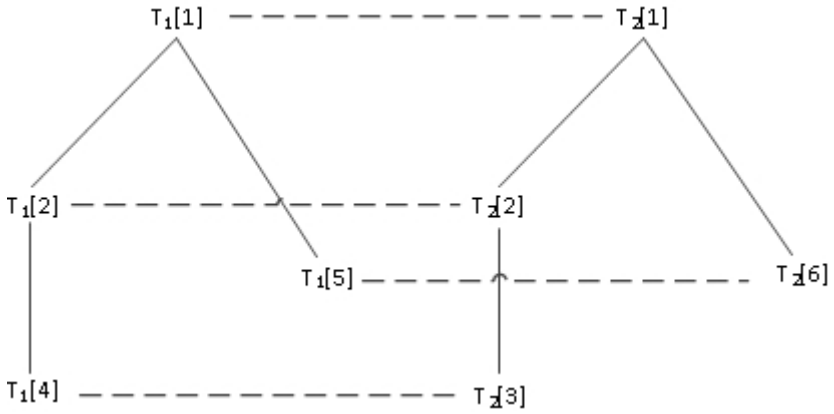


Figure 5-5: *Trees T_1 and T_2 after deleting untouched nodes*

5.2 Statistical Model around Tree Edit Distance

5.2.1 Approximating Probability Estimates

$$P(f_s \mid f_i) \simeq sim(f_s, f_i) \tag{5.4}$$

$$sim(f_s, f_i) = \frac{1}{editdistance(f_s, f_i)} \tag{5.5}$$

The term $P(f_s|f_i)$ is computed based on the notion of similarity of trees. We define the similarity between two trees by Equation (5.5) as the inverse of the edit distance. The lower the cost of the transformations required to convert one tree to another, the more similar they are, and vice versa. If no transformations are required then the similarity

is one.

To enable counting for trees we introduce a manually defined threshold to judge if two trees are similar. We say a tree T_1 is similar to another tree T_2 if $editdistance(T_1, T_2) <= threshold$. Put another way, if the edit distance is within the specified threshold, we consider the two trees to be similar. This is a mechanism introduced to facilitate counting of trees similar to the case of words in a language model. We have used the tool provided by Stephen Wan[1] to measure edit distance for trees.

Data sparsity is always a problem in a maximum likelihood setting. Encountering a tree from the candidate sentence that is not present in the training collection is a possibility. We therefore use smoothing on the probability to get the estimates.

5.2.2 Smoothing Estimates

Dirichlet smoothing is applied to gather the estimate $P(f_i|T_s)$. The probability estimate counts the number of trees compared to words and it is even more likely that we will require smoothing. In our experiments we have experimented with different values of μ, from 10 to 4000. Usually in information retrieval μ is around 500 and 10000 for the best performance (Zhai and Lafferty, 2004).

$$P(f_i|T_s) = \frac{count_{Ts}(f_i) + \mu P(f_i \mid C)}{\sum_m count_{Ts}(f_m) + \mu} \tag{5.6}$$

Finally, we use add-one smoothing to compute $P(f_i|C)$.

$$P(f_i|C) = \frac{count_C(f_i) + 1}{\sum_l count_C(f_l) + l} \tag{5.7}$$

Here the term $count_{T_s}(f_i)$ is the number of times the feature f_i occurs in the domain T_s. $P(f_i|C)$ computes the probability of observing the feature f_i in the entire collection 'C'. $count_C(f_i)$ is the number of times the feature f_i occurs in the entire collection. The counting process involves computing the edit distance for the pair of trees and comparing

[1]Available for download from `http://web.science.mq.edu.au/~swan/howtos/treedistance/`

with the *threshold*. If edit distance is within the threshold, we consider it to be a match.

5.3 Experimental Setup

5.3.1 Evaluation Metrics

We have used *success at n*, *precision at n* and *Mean Reciprocal Rank* as the evaluation metrics. In all cases we refer to the definition bearing sentence as the relevant sentence.

- success at n (S@n): For a test question, S@n is 1 if a relevant sentence is found in the first 'n' rows (results), 0 otherwise. This thesis evaluates S@1, S@5 and S@10.
- precision at n (P@n): For a test question, precision is the percentage of retrieved sentences which are relevant. Therefore, P@n is the precision after 'n' sentences have been retrieved. This evaluates P@1, P@5 and P@10.
- Mean Reciprocal Rank (MRR): For a test question, reciprocal rank (RR) is $\frac{1}{r}$ where 'r' is the rank of the first row for which a relevant sentence is found, or zero if a relevant sentence was not found. MRR is the mean of the reciprocal ranks over all the test questions.

5.3.2 Training and Testing Dataset

The dataset for this experiment and the labelling guidelines are described in section 3.1.1. The training and test sentences were labelled as either definition sentences (label "1") or non-definition sentences (label "0") by two annotators. In order for a sentence to qualify as a definition sentence, it must present at least one piece of information about the target explicitly. Sentences that only hinted at the information were treated as non-definition sentences. We ended up with 558, 572, 473 and 492 definition sentences for *person*, *company*, *disease* and *rule* respectively. To collect the test dataset we fed the 25 entities to the

Google search engine and retained the top twenty sentences for each of the entity types.

The preprocessing step involves converting all the dependency trees into an ordered tree notation format required by the tool to compute edit distance. We experiment with both word and part-of-speech tag as a node label. Tree edit distance is calculated using the tool provided by Stephen Wan[2]. It implements the Zhang and Sasha dynamic programming based algorithm (Zhang and Shasha, 1989) for ordered labelled trees. An example of the ordered tree notation is given below:

```
Root:0-VBD:2;VBD:2-PUNCTPUNCT:14;VBD:2-NNP:1;VBD:2-IN:9;
VBD:2-IN:7;VBD:2-NN:4;VBD:2-VBG:15;VBD:2-PUNCTPUNCT:21;VBD:
2-IN:12;NN:4-IN:5;NN:4-JJ:3;IN:5-NN:6;IN:7-NNP:8;IN:9-NNP:1
1;NNP:11-RB:10;IN:12-CD:13;VBG:15-IN:19;VBG:15-IN:16;IN:16-
NNP:18;NNP:18-PRP$:17;IN:19-CD:20;
```

Here, `Root:0-VBD:2` denotes that there is a link from a node labelled `Root` to a node labelled `VBD`. The labels represent part-of-speech tags of the corresponding word. As you can see each node is assigned a node id (the integers) and the ordering is from left to right as we move down the tree.

There is no training phase in the edit distance approach. When a sentence is to be ranked, it is compared with all the positive instances belonging to a specific entity type. For instance, an answer candidate belonging to a *person* entity type will be compared with all of the 558 positive instances.

5.4 Results

5.4.1 Edit Distance Approach

We first experimented by setting $\mu = 4000$ and *threshold* to 10. Basically, if the cost of the edit operation is less than 10 then we consider two trees to be similar. Table 5.1 shows the result for using part-of-speech tags as node labels. Table 5.2 shows the result for the case of word as

[2]See `http://web.science.mq.edu.au/~swan/howtos/treedistance/package.html`

node labels. Table 5.3 presents the result for $\mu = 4000$ and *threshold* set to 20.

EntityType	S@1	S@5	S@10	MRR	P@1	P@5	P@10
Person	0.00	0.28	0.64	0.15	0.00	0.064	0.11
Company	0.00	0.29	0.54	0.16	0.00	0.075	0.09
Disease	0.08	0.33	0.83	0.25	0.08	0.10	0.12
Rule	0.08	0.48	0.84	0.25	0.08	0.12	0.16

Table 5.1: *Results with MU=4000 Thres=10, part-of-speech tag as a node label*

EntityType	S@1	S@5	S@10	MRR	P@1	P@5	P@10
Person	0.04	0.24	0.64	0.18	0.00	0.06	0.11
Company	0.00	0.25	0.58	0.15	0.00	0.07	0.09
Disease	0.08	0.42	0.75	0.24	0.08	0.12	0.14
Rule	0.12	0.48	0.84	0.27	0.12	0.11	0.16

Table 5.2: *Results with MU=4000 Thres=10 and word as a node label*

EntityType	S@1	S@5	S@10	MRR	P@1	P@5	P@10
Person	0.00	0.28	0.64	0.15	0.00	0.07	0.12
Company	0.00	0.29	0.54	0.18	0.00	0.08	0.08
Disease	0.08	0.37	0.83	0.26	0.08	0.11	0.14
Rule	0.12	0.40	0.76	0.26	0.12	0.11	0.15

Table 5.3: *Results with MU=4000 Thres=20, part-of-speech tag as a node label*

5.4.2 Baseline

As a baseline we have taken the original order of candidate sentences. The order of the sentences reflect their position in the results returned by the google search engine.

The baseline approach significantly outperforms the edit distance approach in all metrics and for all the entity types. This does not come

EntityType	S@1	S@5	S@10	MRR	P@1	P@5	P@10
Person	0.56	0.96	1	0.74	0.56	0.27	0.19
Company	0.33	0.83	0.87	0.51	0.33	0.25	0.17
Disease	0.45	0.83	0.96	0.59	0.45	0.26	0.21
Rule	0.44	0.88	0.96	0.51	0.44	0.28	0.19

Table 5.4: *Results from the baseline*

as a surprise considering the quality of results from the search engine and the limitations of the edit distance approach. For example, in case of *person* questions, the first result returned by the search engine is usually a Wikipedia entry. *P@1* metric clearly demonstrates this behaviour. The analysis of the results from the edit distance approach is presented in the next section.

5.5 Analysis

Since we are comparing entire trees, it was expected that the scoring would be biased towards smaller trees. This was indeed the case in our experiments. Table 5.5 shows some of the high scoring snippets for the *person* entity type. Results are from the setting $\mu = 4000$ and *threshold* set to 20 and using word as a node label. There is a good chance of finding these small trees as a subtree in the positive examples we have collected for each entity type. As a result the cost of transforming a smaller tree has a good chance of being less compared to trees with greater depth and branching factor. Since it is less likely to find many exact matching trees (unless we have a very large dataset) we see that shorter sentences hold an advantage. Unlike the results in Han et al. (2006), the use of tree edit distance is not able to improve on the baseline. Actually the performance is very bad.

The *QuestionNo.* column is the question id and *AnswerNo.* is the answer number as stored in our database. Answer number is the position of the text snippet in the result obtained from google search engine.

A similar case can be seen for other entity types. Table B.1, B.2 and B.3 shows some of the high scoring snippets for *company*,*disease* and

Question No.	Answer No.	Sentence
27	13	Nationally-acclaimed Shakespeare theatre .
60	1	Actor : Star_Wars .
118	1	Actor : Top Gun .
177	4	Actor : Superhero Movie .
108	13	Jordan , Michael J.

Table 5.5: *Top scoring five sentences for person entity type*

rule entity types respectively. The corresponding tables can be found in Appendix B.

Table 5.6 lists the correctly ranked sentences in top 1. We can see there are only results from the *disease* and *rule* entity types. From the table showing the top ranked sentences we could see that, especially in the case of the *rule* entity type, longer sentences (greater depth) were being retrieved. This is usually due to the nature of answers belonging to this entity class. Answers about a rule, policy and treaty tend to be composed of long sentences. Due to this we see fewer number of short sentences in the domain dataset as well.

Entity Type	Question No.	Answer No.	Sentence
DISEASE	1125	14	No specific type of person has GERD .
DISEASE	1236	1	Apr_1_,_2007 ... Premenstrual syndrome (PMS) is a group of symptoms linked to the menstrual cycle .
RULE	1533	9	Treaty of Batum signed between Ottoman Turkey and Armenia .
RULE	1605	2	The_Paris_Peace_Conference (July_29 to October_15_,_1946) resulted in the Paris_Peace Treaties signed on February_10_,_1947 .
RULE	1647	16	The Test Ban_Treaty of 1963 prohibits nuclear weapons tests " or any other nuclear explosion " in the atmosphere , in outer space , and under water .

Table 5.6: *Top ranked sentences across all entity types*

Table 5.7 lists the correctly ranked top five sentences for the *person* entity type. The rest of the tables can be found in the Appendix

B. Table B.4 lists the correctly ranked top sentences in the 5 for the *company* entity type. Table B.5 lists the correctly ranked top sentences in top 5 for the *disease* entity type. Table B.6 lists the correctly ranked top sentences in top 5 for the *rule* entity type. We can see that the method is able to find correct answers that are longer in length. A part of the good performance is dependent on the quality and quantity of the positive example dataset.

Question No.	Answer No.	Sentence
3	11	The beatification of Mother_Teresa was conducted on Oct._19_,_2003 .
22	5	Vincent_van_Gogh (March_30_,_1853 - July_29_,_1890) is generally considered the greatest Dutch painter after Rembrandt , though he had little success during ...
31	1	Terry Paxton Bradshaw (born September_2_,_1948) , also known by the nickname " Mr .
35	1	Whoopi Goldberg (pronounced _???pi_ ; born Caryn Elaine Johnson ; November_13_,_1955) is an American comedienne , actress , singer-songwriter and Emmy ...
46	2	Neil Leslie Diamond (born January_24_,_1941) is one of America 's most enduring and successful singer-songwriters .

Table 5.7: *Sentences Correctly Ranked in top five for person entity type*

We also experimented by removing the $P(f_i|T_s)$ part and only using the similarity score between trees for ranking. Table 5.8 shows the result from that experiment.

EntityType	S@1	S@5	S@10	MRR	P@1	P@5	P@10
Person	0.00	0.28	0.68	0.15	0.00	0.07	0.12
Company	0.04	0.25	0.50	0.20	0.04	0.08	0.08
Disease	0.08	0.33	0.79	0.24	0.08	0.10	0.14
Rule	0.16	0.44	0.76	0.28	0.16	0.12	0.15

Table 5.8: *Results with MU=4000 Thres=10, part-of-speech tag as a node label and no $P(f_i|T_s)$*

It was interesting to observe that the weight part of the scoring function, $P(f_i|T_s)$ plays no role in this approach. The performance does not degrade when we assign a uniform estimate. The similarly between the features clearly plays a major role in the ranking. One of the reasons we see such a non-effect of the weight term could be because many of the sentences contain lot of noise. The sentences we have used for training and testing have not been altered in any way. We can see noisy fragments such as "Apr_1_,_2007 ... " attached to the correct answer sentences. This additional text clearly increases the number of transformations required, hence the edit distance and therefore the similarity score. One way round this problem would be to identify the important subtrees in the dependency tree and ignore the rest of it.

Another factor that can affect the performance is the grammatical errors which we see often in online documents. An ungrammatical text will be interpreted incorrectly by the dependency parser. This is one of the reasons we can see good performance on carefully crafted dataset such as newspaper texts but not so good on the user contributed text. Modelling tasks for definition questions have therefore restricted themselves mostly to carefully crafted encyclopedias.

Another problem with a maximum likelihood approach is that it requires very large training data. We expect $P(f_i|T_s)$ to play a significant role if large enough dataset was available. This inefficiency due to size of the dataset is felt because we compare an entire tree with another. This problem cannot be resolved by increasing the value of *threshold*. Although by increasing *threshold* we allow more trees to be counted as being similar, this increase is equally observed for all the features thereby negating the effect.

After observing this result we put forward an intuitive hypothesis that a similarity metric at a subtree level should perform well for small and medium sized datasets. For example, patterns such as "NP_1, NP_2" and "NP_1, also known as NP_2" are two of the popular ways of introducing a person. The use of patterns has been successful in TREC experiments for both fact seeking questions (Brill et al., 2001) as well as definition questions (Hildebrandt et al., 2004; Xu et al., 2003). Here, NP_1 and NP_2 denote noun phrases. These approaches construct surface level patterns but their retrieval performance points at possible better

performance by looking for shorter patterns. We therefore reward performance by counting short structural patterns (subtrees). In the next chapter we investigate if this is the case by using a subtree kernel.

5.6 Conclusion

In this chapter we presented results from the experiments which attempted to incorporate structural features in a statistical framework. Using edit distance proved to be inefficient while calculating the probability estimates. This was mainly down to (1) using the entire length of the tree and (2) manually set thresholds. Based on the results and non-role played by $P(f_i|T_s)$, we look for a better way to compute this estimate. In the next chapter we explore the use of tree kernel that computes similarity between the trees by counting common sub-structures. From the relative success of surface pattern approaches it gives us enough evidence to make an observed guess that searching for common subtrees should perform better than trying to match an entire tree. The next chapter outlines an approach where the system learns discriminating features for each of the entity types using a support vector machine (SVM) classifier. This should be able to handle $P(f_i|T_s)$ much better than in the current scenario. The two reasons why it should work are (i) similarity is based on counting common substructures rather than edit distance (ii) SVM should perform better at unseen cases as well as estimating probability estimates from seen data. This should overcome the limitation of manually determining thresholds and smoothing constants. The subsequent approach relies on the posterior estimate from the trained SVM classifier directly removing these dependencies.

Chapter 6

Learning Framework for Reranking Definition Sentences

After observing the results from the edit distance technique we wanted to find (i) a more intuitive measure of tree similarity (ii) get rid of manually set thresholds and (iii) get $p(f_i|T_s)$ to play a role. We look to resolve these problems by using tree kernel that counts the number of common sub-structures to assign a similarity score. The posterior probability estimate from an SVM classifier is taken to be the value for $p(f_i|T_s)$. By using a held-out set and cross validation, we remove the need to manually specify any threshold values or constants.

6.1 System Introduction

Figure 6-1 depicts our pipeline structure for reranking candidate sentences. When a user question is submitted, the definition target (named entity) is extracted from it. For example, in the question "Who is Buddha?" the definition target is "Buddha" which is of type *person*. We then submit the definition target as a query term to the google search engine and extract the top twenty results. These top twenty sentences are then moved on to the reranking phase in the pipeline where each of them are assigned a score by the model. The score is determined by comparing the candidate sentence with the positive examples belonging to the entity type *person* and from the posterior probability estimate of a classifier. Reranking is then the task of sorting

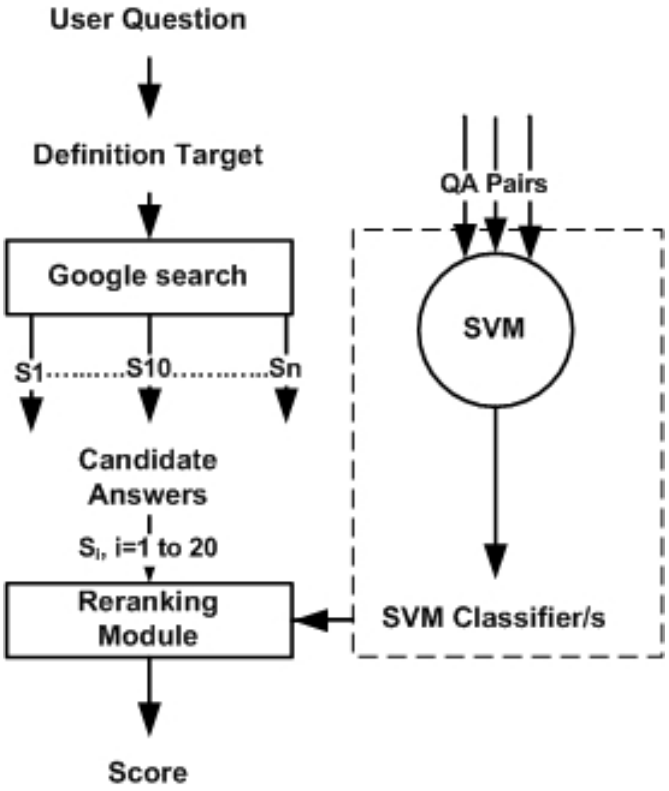


Figure 6-1: *Sentence Reranking Pipeline*

the sentences in descending order of the score.

6.2 Building a Classifier for an Entity Type

6.2.1 Support Vector Machine Classifier

Support Vector Machines (SVM) (Vapnik, 1982; Cortes and Vapnik, 1995)are an example of a supervised learning algorithm that works by identifying patterns in the data. SVM belongs to the class of maximum margin linear classifiers. The task in a linear classification (two class setting) is to separate data which can belong to one of the two classes by a linear function such that it maximises the distance between this function and the nearest data point of each class. This distance is what is known as the margin. Support vectors are basically the data points on the margin.

In a typical classification setting, we are given training data or features $x_1, x_2, ..., x_n$ which are vectors in some space with dimension

d. We are also given a label for each training instance. In a two class setting, let us assume that the label is $y \in +, -$. In Figure 6-2 we can see examples belonging to either of the two classes (+ or -) are separated in such a way that a line can be drawn which divides the plane such that examples belonging to one class lie on one side and the examples of the other class on the other side. This is what a SVM classifier essentially tries to achieve in the training phase. In the testing or classification phase, a previously unseen datum is assigned one of the two labels depending on its relative location to the separating line.

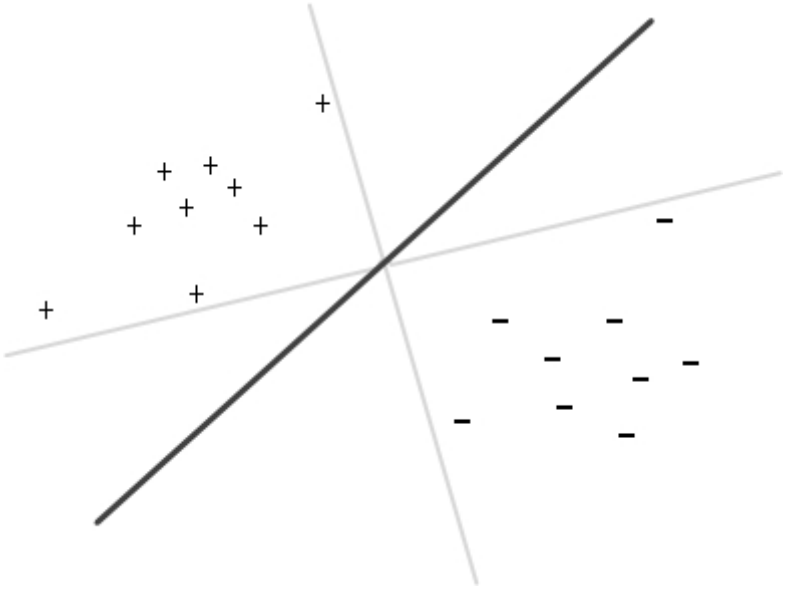


Figure 6-2: *SVM margin*

In a higher n-dimensional space, this separating line would be a (n-1) dimensional hyperplane. Even in this case of two dimensions, there are infinitely many possibilities to draw a line. A SVM classifier is looking for a line that can generalise as much as possible while accounting for new unseen data. The problem here is to learn complex pattern while excluding the exceptions i.e. the overfitting problem. SVMs achieve this by selecting a line which is located in the middle between the nearest positive (+) and and negative (-) examples. This is what we refer to as finding a line with maximum margin. However, it is not always possible to draw a line that can clearly separate both set of data. This is where the concept of soft-margin is introduced. SVMs are basically allowed to perform misclassification. There is generally a

trade off between the margin and the permissible training error.

SVMs can be generalised to handle the non-linear separability case by applying the linear approach to to the transformed data $\phi(x_1), \phi(x_2), ..., \phi(x_n)$. Here ϕ represents the mapping from input space to a higher dimension feature space where dot product is defined.

Mathematically, given a set of training data $x_1, x_2, ..., x_n$ where $x_i \in R^n$ and $y \in +, -$, the task in SVM is to find the solution for the following optimisation problem

$$\min_{w,b,\xi} \frac{1}{2} w^T w + C \sum_{i=1}^{n} \xi_i \tag{6.1}$$

subject to

$$y_i(w^T \phi(x_i) + b) \geq 1 - \xi_i, \xi_i \geq 0 \tag{6.2}$$

Here ξ is called the slack variable. The constant C controls the trade-off between the maximisation of the margin and the separation of the training set. Lowering C allows the classifier to include more misclassifications and have a larger margin. This is what was referred to as the soft-margin strategy. The weight vector $w \in R^D$ and the threshold b are the parameters.

6.2.2 Training a Classifier

SVM classifiers belong to the class of linear classifiers. A single classifier therefore can only handle a binary classification task well. For multi-classification we have to train several such classifiers. The strategy we have used is the one-vs-one, winner takes all strategy. This method constructs M binary classifiers. In our case the number of classes (entity types) is four (M=4). Since there are only positive examples of each class, the examples from other classes are taken as negative examples. When a test instance is evaluated, in case the type of the target is not known or not detected, the classifier with the largest posterior probability estimate is chosen. In case the target type is known we can directly use the result from the classifier trained on the positive examples for that entity class.

We use the dataset described in Section 3.1.1 to train SVM classifiers. The feature we use are the dependency trees. SVM allows the

use of any kind of features through the use of kernel functions.

6.2.3 Kernel Functions

Kernel functions basically can be seen as the inner product in some complex feature space. By using kernel functions in SVMs, we are introducing the concept of similarity of data. This is one of the reason why kernel methods are popular, being able to see a dot product in higher dimensions in terms of similarity. Being able to use kernel functions also allows us to use natural language features that are non-numeric such as bag-of-words. The main advantage of using a kernel method is that it allows us to work in a very large dimension, potentially infinite, without the need of explicit handling of the features. The relationship between the kernel function K and the high dimensional mapping $\phi(.)$ is $K(x, y) =< \phi(x), \phi(y) >$. Therefore by defining a kernel function we are indirectly specifying $\phi(.)$. This is called the "kernel trick" (Aizerman et al., 1964).
Some basic kernel definitions with kernel parameters γ, r and d are given below. Polynomial, RBF and sigmoid functions are used to deal with non-linear cases.
Linear: $K(x_i, x_j) = x_i^T x_j$
Polynomial: $K(x_i, x_j) = (\gamma x_i^T x_j + r)^d, \gamma > 0$
Radial basis function (RBF) : $K(x_i, x_j) = exp(- \parallel x_i - x_j \parallel^2), \gamma > 0$
Sigmoid : $K(x_i, x_j) = tanh(\gamma x_i^T x_j + r)$

The similarity view of kernels further allows us to use structural data such as trees and graphs. We are no longer restricted to define the data in the style of feature:value pairs. This is useful in natural language processing where in many cases we are not certain how to obtain value for a feature. Due to this freedom, the focus is more on designing better kernels and not on converting the data to a feature:value representation.

Tree Kernels

A tree kernel computes similarity between two trees in terms of their sub-structures (Collins and Duffy, 2002). Using the notation used in Collins and Duffy (2002), a tree kernel that is based around count-

ing substructures (tree fragments) can be mathematically defined as $K(T_1, T_2) = h(t_1).h(t_2)$. If, $h_i(t_1)$ is the number of occurrences of the i^{th} tree fragment in T then $h(T) = (h_1(T), h_2(T), ..., h_n(T))$.

Let $I_i(n)$ be the indicator function which is 1 if the i^{th} tree fragment is rooted at 'n' and 0 otherwise. Then we have, $h_i(T_1) = \sum_{n_1 \in N_1} I_i(n_1)$ and $h_i(T_2) = \sum_{n_2 \in N_2} I_i(n_2)$.

$$h(T_1).h(T_2) = \sum_i h_i(T_1)h_i(T_2) \tag{6.3}$$

$$= \sum_{n_1 \in N_1} \sum_{n_2 \in N_2} \sum_i I_i(n_1)I_i(n_2) \tag{6.4}$$

$$= \sum_{n_1 \in N_1} \sum_{n_2 \in N_2} C(n_1, n_2) \tag{6.5}$$

A polynomial time computation of $C(n_1, n_2)$ is possible due to these observations:

- $C(n_1, n_2) = 0$ if production at n_1 and n_2 are different
- $C(n_1, n_2) = 1$ if production at n_1 and n_2 are same and n_1 and n_2 are pre-terminals
- For cases different from above,
  $C(n_1, n_2) = \prod_{j=1}^{nc(n_1)} (1 + C(ch(n_1, j), ch(n_2, j)))$

Here, $nc(n_1)$ is the number of children of node n_1 and $ch(n_1, j)$ denotes the j^{th} child of node n_1. Below we discuss the commonly used two types of tree fragments over which the kernel can operate.

Subtree Kernel

A subtree rooted at a node will contain all its descendants all the way down and including the leaf nodes. Figure 6-3 shows the tree for the sentence "I enjoyed the lunch" and its subtrees.

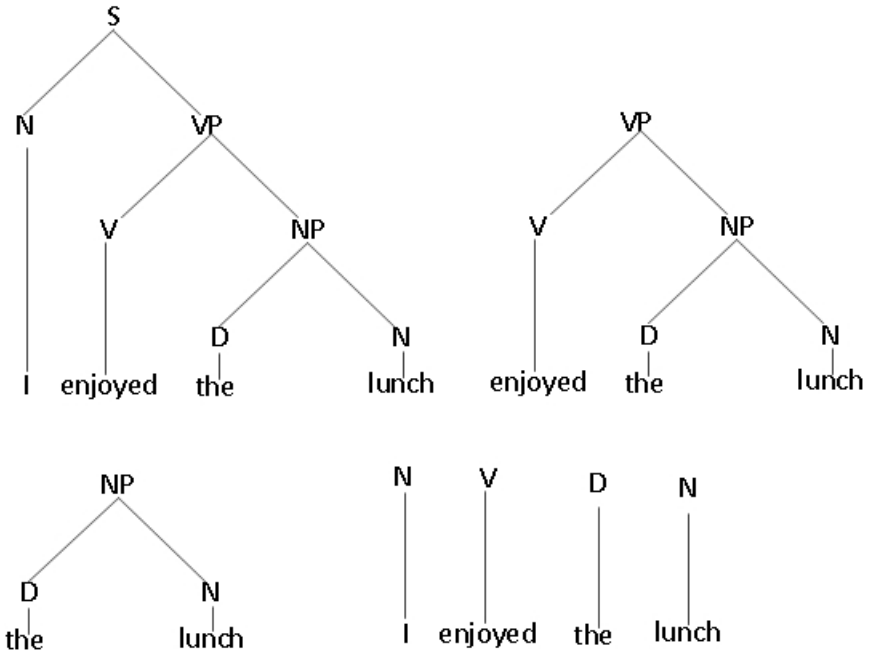


Figure 6-3: *A Parse Tree along with its Subtrees*

Subset Tree Kernel

A subset tree is a subtree having either all or no children of a node and is not a single node. In this setting, two nodes n_i and n_j match if (a) they have the same label (b) they have same number of children and (c) the corresponding child of n_i and n_j has the same label. Figure 6-4 shows the tree for the sentence "I enjoyed the lunch" and some of its subset trees.

6.3 Computing Probability Estimates

$$P(f_s \mid f_i) \simeq sim(f_s, f_i) \tag{6.6}$$

The term $P(f_s|f_i)$ is computed based on the notion of similarity similar to the case of the edit distance approach. The difference lies in how the similarity score of two tree representations f_s and f_i is calculated. Here, the similarity score is the value returned by the subset tree kernel function. Equation 6.7 is the normalised score thus obtained. Here $sim_k(x, y)$ is the number of common subset trees returned by the kernel function.

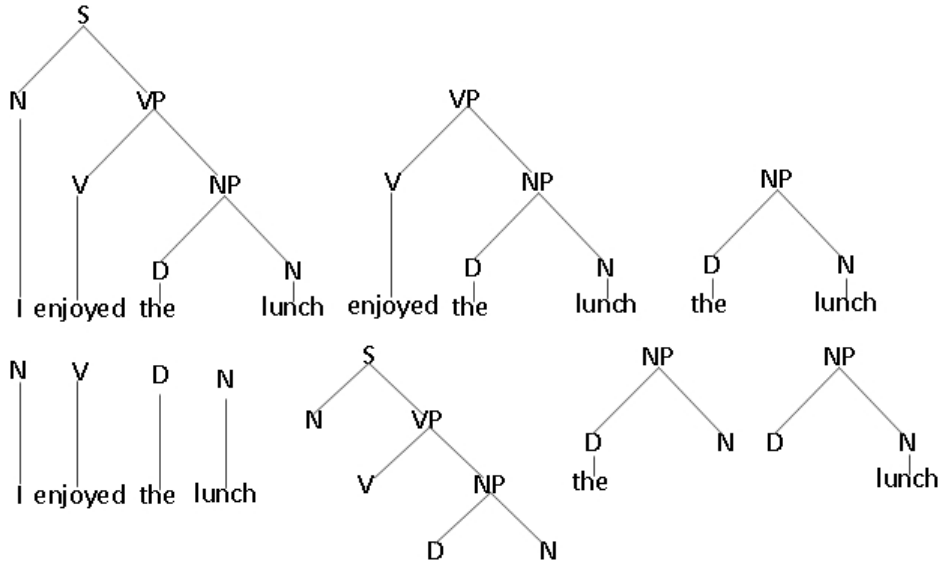


Figure 6-4: *A Parse Tree along with some of its Subset Trees*

$$sim(f_s, f_i) = \frac{sim_k(f_s, f_i)}{\sqrt{sim_k(f_s, f_s) \times sim_k(f_i, f_i)}} \tag{6.7}$$

The remaining task is to compute the estimate $P(f_i|T_s)$. In our probabilistic framework we computed the estimate as

$$P(f_i|T_s) = \frac{P(T_s|f_i)}{P(T_s)} \times P(f_i) \tag{6.8}$$

We ignore the term $P(T_s)$ in Equation 6.8 since it is same for all the candidate sentences being ranked. We make another simplifying assumption that $P(f_i)$ has an uniform distribution. Thus the only contribution to $P(f_i|T_s)$ is from the term $P(T_s|f_i)$. The value for this term is the task of obtaining posterior probability estimate from the classifier. The technique for getting the posterior probability is described next.

6.3.1 Posterior Probability Estimates from the Classifier

It has been shown that fitting a sigmoid function provides a good estimate of posterior probability (Platt, 1999). One can use the same training data and decision values $f(x_i)$ used to train the classifier and

can fit a parametric sigmoid function to approximate the posterior probability.

$$P(y = 1|x) = \frac{1}{1 + exp(Af + B)} \tag{6.9}$$

Here A and B are parameters of the sigmoid function that need to be determined. Fitting of a sigmoid function involves using maximum likelihood on the training set (f_i, y_i). In the case of a binary classification, the predicted class labels are either +1 or -1. Equation 6.10 gives the approximation of the posterior probability. Here y is the label, x the test instance and $f = f(x)$ is the decision function.

$$\begin{aligned} P(y = 1|x) \approx & P_{A,B}(f) \\ \equiv & \frac{1}{1 + exp(Af + B)} \end{aligned} \tag{6.10}$$

The best setting for A and B are estimated based on maximum likelihood estimation from a separate training set. This can be obtained using cross-validation. We use Lin's version[1] of Platt's algorithm in our experiments. The maximum likelihood problem solved to get A and B is:

$$\min_{(A,B)} -\sum_{i=1}^{l} (t_i \log(p_i) + (1 - t_i) \log(1 - p_i)) \tag{6.11}$$

Where,

$$p_i = P_{A,B}(f_i)$$

$$|t_i| = \begin{cases} \frac{N_+ + 1}{N_+ + 2} & \text{if } y_i = +1,\ i = 1, ..., l \\ \frac{1}{N_- + 2} & \text{if } y_i = -1 \end{cases}$$

Here, N_+ and N_- are the number of examples belonging to the positive and the negative class respectively in the training set. Newton's method with backtracking line search is used to solve the above optimisation problem to obtain the probability estimates in Lin's implementation (Lin et al., 2003). Our assumption is that this will provide a better estimation than we get from a maximum likelihood approach

[1]Pseudocode is provided in their paper.

as explored in the earlier chapter.

6.4 Experimental Setup

6.4.1 Training and Testing Dataset

For training purposes we use 558, 572, 473 and 492 sentences for *person*, *company*, *disease* and *rule* entity types respectively. We also constructed a separate held-out set for determining A and B in Equation 6.10. The held-out part is the one used to estimate the parameters of the sigmoid function. The held-out set contains 51, 60, 47 and 53 sentences for *person*, *company*, *disease* and *rule* entity types respectively. To collect the test dataset we fed a non-overlapping set containing 25 entities to google search engine and retained the top twenty sentences. This is then fed as an input (candidate answers as labelled in Figure 6-1) to our system and are reranked. Sentences in the test dataset is also labelled according to the same criteria used while filtering training data (see Chapter 3 for description of guidelines and dataset construction procedure).

We have used the same evaluation metrics *success at n*, *precision at n* and *Mean Reciprocal Rank* as used for the edit distance based approach. Success and precision is measured at n=1,5 and 10 respectively.

6.5 Results

		T_w^k	T_p^k	T_p^{k*}	B_w^k
PERSON	S@1	0.16	0.16	0.12	0.12
	S@5	0.60	0.56	0.60	0.64
	MRR	**0.36**	0.35	**0.36**	0.35
	P@1	0.16	0.16	0.12	0.12
	P@5	0.18	0.18	0.17	0.19
COMPANY	S@1	0.29	0.50	0.46	0.25
	S@5	0.91	0.83	0.80	0.87
	MRR	0.54	**0.63**	0.60	0.48
	P@1	0.29	0.50	0.46	0.25
	P@5	0.27	0.27	0.22	0.24
DISEASE	S@1	0.25	0.54	0.29	0.17
	S@5	0.87	0.95	0.83	0.79
	MRR	0.50	**0.68**	0.47	0.43
	P@1	0.25	0.54	0.29	0.17
	P@5	0.32	0.32	0.24	0.32
RULE	S@1	0.16	0.40	0.24	0.20
	S@5	0.48	0.72	0.64	0.64
	MRR	0.33	**0.55**	0.43	0.38
	P@1	0.16	0.40	0.24	0.20
	P@5	0.15	0.15	0.20	0.18

Table 6.1: *Summary of results from four separate experiments. T_w^k, T_p^k, T_p^{k*} and B_w^k are approaches using tree kernel (with word as node label), tree kernel (with part-of-speech as node label), without the use of the posterior estimate and bag-of-words kernel respectively.*

Table 6.1 shows the results from four separate experiments. T_w^k is the result from the utilisation of a tree kernel with word as the node label. Similarly, T_p^k is the result from the the use of tree kernel with part-of-speech tag as the node label. Table 6.2 shows the result from the experiment when word is used as the node label. Table 6.3 highlights the results from the experiment with part-of-speech-tag as the node label. We also ran an experiment to see if the posterior probability from the SVM had an effect on the performance of the approach. Column T_p^{k*} lists the performance of the system relying only on the similarity scores. A complete set of results is presented in Table 6.7 and analysed in the next section. We also used bag-of-words kernel as the baseline for the proposed tree kernel based approach. The column B_w^k lists the

EntityType	S@1	S@5	S@10	MRR	P@1	P@5	P@10
Person	0.16	0.60	0.96	0.36	0.16	0.18	0.19
Company	0.29	0.91	0.95	0.54	0.29	0.27	0.17
Disease	0.25	0.87	1.00	0.50	0.25	0.32	0.25
Rule	0.16	0.48	0.84	0.33	0.16	0.15	0.18

Table 6.2: *Results from the tree kernel approach with word as node labels*

EntityType	S@1	S@5	S@10	MRR	P@1	P@5	P@10
Person	0.16	0.56	0.88	0.35	0.16	0.18	0.19
Company	0.50	0.83	0.95	0.63	0.50	0.27	0.17
Disease	0.54	0.95	1.00	0.68	0.54	0.32	0.25
Rule	0.40	0.72	0.84	0.55	0.40	0.15	0.18

Table 6.3: *Results from the tree kernel approach with part of speech tags as node labels*

result from the bag-of-words approach. A complete set of results for the bag-of-words kernel is shown in Table 6.8.

It can be clearly seen that the tree kernel with part-of-speech tags as node labels outperforms all of the the other approaches for the three entity types in terms of Mean Reciprocal Rank (MRR). The MRR score is close to the best for the *person* entity type as well. Similar pattern is observed for the other performance metrics as well. The analysis section presents the results from significance tests on MRR to check if the improvement in performance is by chance or if the proposed tree kernel based approach is actually much better than the rest of the approaches.

6.6 Analysis

Table 6.4 and 6.5 shows some of the overall high scoring sentences. In the sentences we can clearly see the effect of noise on the scoring function. Noisy high scoring sentences usually have a large number of non-related named entities in them. Including large number of keywords in a documents is the most popular approach used to influence

search engine ranking algorithm.

Type	Question No.	Answer No.	Sentence
person	92	20	Jan 31 , 2010 ... Julia Roberts , Shirley MacLaine , Jennifer Garner ... Shirley MacLaine attendS Variety 's 1st Annual Power of Women Luncheon at the Beverly
company	592	12	Jason Wilkins , Head of Information Technology , Xstrata ... ture , Xstrata decided to streamline and optimize its IT by migrating to
person	20	18	Maewest , Mae West , Victoria_Mills , Look alike , Look-a-like , Impersonator .
rule	1647	5	Text of the 1963 treaty signed by the US , the UK , and the USSR banning atmospheric , oceanic , and extraterrestrial testing of nuclear weapons .
rule	1636	6	CERTAIN that the continuation of this process requires the utilization of the positive experience obtained from the application of the Montevideo_Treaty ,
person	3	4	Discusses the beautification , her early_years , and vocation .
rule	1501	1	In 1901 the United_States and the United_Kingdom signed the Hay-Pauncefote_Treaty .
company	188	19	Corbin Bleu Reivers , better known as Corbin Bleu , is an American actor , model , rapper , and singer .
company	135	7	Rock , Goth , Hard Rock , and Heavy_Metal label .

Table 6.4: *Top scoring sentences across all four entity types part A.*

Table 6.6 shows the top scoring ten sentences for *person* entity type. We see similar picture from these results as well. The noisy inclusion of named entities and proper nouns labels incorrect sentences as definition

Type	Question No.	Answer No.	Sentence
person	92	20	Jan 31 , 2010 ... Julia Roberts , Shirley MacLaine , Jennifer Garner ... Shirley MacLaine attendS Variety 's 1st Annual Power of Women Luncheon at the Beverly
rule	1501	16	Very little business beyond the consideration or the HayPaunce-fote_treaty will be transacted in the Senate before the adjournment for the holidays .
rule	1597	13	September_17_,_2005 marked the 60th anniversary of the Wanfried_agreement , also referred to as the Whisky-Vodka-Line_agreement , signed between the United ...
company	585	15	Michigan , Ohio , AXA_Advisors , Great Lakes , I-75 , Troy , Grand Rapids , Brighton , Detroit , Metro-Detroit , Dearborn , Livonia , Mt._Pleasant , Auburn Hills , ...
rule	1554	6	On July_6_,_1914 , he signed the Thomson-Urrutia_Treaty between the United_States and Colombia .
rule	1597	4	September_17_,_2005 marked the 60th anniversary of the Wanfried_agreement , also referred to as the Whisky-Vodka -Line agreement , signed between the United ...
rule	1618	15	ANZUS ANZUS joined the nations of Australia , New_Zealand and the United_States in a defence security pact for the Pacific region .

Table 6.5: *Top scoring sentences across all four entity types part B.*

sentences. Among the four entity types, *person* is the worst performing type in terms of all the metrics. This is mainly down to the noisy nature of sentences that are abundant for this type. We see far less of such noisy repetitions in other classes. It is to be noted that in training

no cleaning was performed to remove noise component included in the definition containing sentences. This was mainly down to the large amount of manual effort that would be required to do such a task.

Question No.	**Answer No.**	**Sentence**
92	20	Jan 31 , 2010 ... Julia Roberts , Shirley MacLaine , Jennifer Garner ... Shirley MacLaine attendS Variety 's 1st Annual Power of Women Luncheon at the Beverly
20	18	Maewest , Mae West , Victoria_Mills , Look alike , Look-a-like , Impersonator .
3	4	Discusses the beautification , her early_years , and vocation .
188	19	Corbin Bleu Reivers , better known as Corbin Bleu , is an American actor , model , rapper , and singer .
135	7	Rock , Goth , Hard Rock , and Heavy_Metal label .
65	2	Provides news , articles and details about the tennis player , perceived as a conscience leader , humanitarian , educator and athlete .
60	16	Harrison Ford Mercury Wellington Ohio Ford Mercury Dealership : prices , sales and specials on new cars , trucks , SUVs and Crossovers .
61	1	Picture , filmography , profile , television credits , and trivia .
188	1	This is the official site of actor , singer , popstar , producer Corbin Bleu .
108	20	Aug_14_,_2009 ... the entire directory , only in J_Jordan , _Michael ... ESPN.com : Michael_Jordan - Provides a variety of statistical data , player profile ,

Table 6.6: *Top ranked ten sentences for person entity type*

Table C.1 and C.2 in appendix C shows the top ranked 10 sentences for *company* entity type. This is the second best performing entity type after *disease*. The sentences contain less noise compared to *person* but more compared to *disease* and *rule* entity types. The best performing entity type *disease* shows significantly less noise in the test sentences as well as training sentences. Table C.3 and C.4 in appendix C shows the top ranked 10 sentences for *disease* entity type. One of

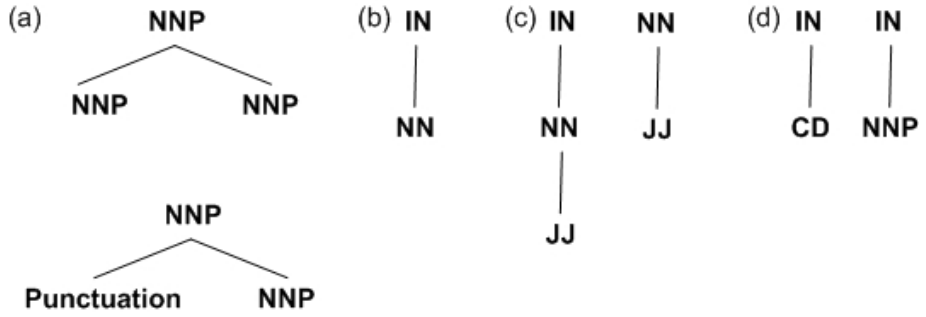


Figure 6-5: *Some frequent sub-structures from the correctly top ranked sentences using Tree kernel approach for (a) PERSON, (b) COMPANY, (c) DISEASE and (d) RULE entity types*

the characteristics of definition sentences in this entity type is that correct sentences tend to have a sequence of symptoms, diagnosis or prescriptions. For example we see sequences such as "shakes , slams , hits , or punches ". However the same characteristics can be seen in false positive sentences such as "symptoms , diagnosis , misdiagnosis , treatment , causes , patient stories , videos , forums , prevention ". Table C.5 and C.6 shows the top ranked 10 sentences for *rule* entity type. It has been observed that definition sentences in this entity class tend to be longer compared to other entity classes. This is down to the fact that most sentences are long explanations. Many sentences tend to be complex and carry more than a single piece of information.

Table C.7 and C.8 lists the correct sentences that received the highest scores thereby appearing at position one.Table C.9 and C.10 list the correct sentences that were ranked inside the top five results for the *person* entity type. Similarly Table C.11 and C.12 lists the top five results for the *company* entity type. Table C.13 and C.14 lists the top five results for the *disease* entity type. Finally, Table C.15 lists the top five results for the *rule* entity type.

Figure 6-5 shows some of the frequent sub-structures that we observed looking at the dependency trees of the test sentences that were ranked at the top for each of the four entity types.

In the case of correctly ranked sentences in the *person* entity type, these sub-structures capture the name of the person and place of birth of a person as shown by examples in Figure 6-5(a). Correctly ranked sentences for company shows frequent occurrences of text such as "by

EntityType	S@1	S@5	S@10	MRR	P@1	P@5	P@10
Person	0.12	0.60	0.92	0.36	0.12	0.17	0.18
Company	0.46	0.80	0.92	0.60	0.46	0.22	0.16
Disease	0.29	0.83	1.00	0.47	0.29	0.24	0.24
Rule	0.24	0.64	0.76	0.43	0.24	0.20	0.17

Table 6.7: *Results from the No-SVM tree kernel approach with part of speech tags as node labels*

- venture", "as - developer" and "in - software" which conform to the pattern in Figure 6-5(b). In the disease entity class we see frequent usage of adjectives. This is expected in description of medical conditions as shown by examples in Figure 6-5(c). Dates are common occurrences in sentences belonging to the rule entity type. Specifically, we can see the sub-structures shown in Figure 6-5(d) occurring frequently.

We also see sentences like "with Fred Rogers, Roger Trow, Johnny Casta, Norman Rockwell..." being ranked at the top in *person* entity type domain. In case a feature like "(NNP (NNP))" is learnt by the classifier, this sentence would receive high score. The posterior probability, $P(y = person|(NNP(NNP)))$, from the classifier is 0.99 which gives a clear indication that this is in fact the case. Similar is the case with other entity types where the discriminating feature learnt can lead to false positives. The results however justify our hypothesis that we postulated at the end of edit-distance based experiments that shorter patterns should lead to better performance. Even with a small set of training data, we significantly outperform the edit distance approach. This improvement is also down to letting SVM dictate the parameters for the sigmoid function and therefore the posterior probability thereby removing the disadvantages of a manually set constants.

To verify if the posterior probability estimate from the SVM classifier contributes to the scoring function, we performed an experiment leaving out the contribution of this part. We therefore only use contribution from similarity computation to obtain the overall score for a candidate sentence. Table 6.7 shows the results from this experiment. As we can see the results are worse across all three metrics. This clearly suggests that using posterior probability estimate is similar to

EntityType	S@1	S@5	S@10	MRR	P@1	P@5	P@10
Person	0.12	0.64	0.96	0.35	0.12	0.19	0.20
Company	0.25	0.87	0.96	0.48	0.25	0.24	0.17
Disease	0.17	0.79	1.00	0.43	0.17	0.32	0.25
Rule	0.20	0.64	0.84	0.38	0.20	0.18	0.18

Table 6.8: *Results obtained from the bag-of-words kernel experiment*

calculating the confidence of the classifier about a feature. Thus it acts as a weighing factor to control the effect of features in the training dataset on the overall score.

We also performed an experiment using the bag-of-words (BoW) kernel. A bag-of-words representation basically considers a sentence to be a collection of words and does not even consider the position of words. Mathematically a BoW kernel (Zhang et al., 2006) is defined as:

$$\hat{K}(s_1, s_2) =< \phi(s_1), \phi(s_2) >= \sum_{i=1}^{N} tf(t_i, s_1) tf(t_i, s_2) \tag{6.12}$$

Here s_1 and s_2 are the sentences being examined. $tf(t_i, s_j)$ denotes the frequency of term t_i in sentence s_j. The total number of terms N is the size of a dictionary formed by the union of words from both sentences. Stopwords are not considered. The kernel we use is obtained by normalising this kernel.

$$K(s_1, s_2) = \frac{\hat{K}(s_1, s_2)}{\sqrt{\hat{K}(s_1, s_1)\hat{K}(s_2, s_2)}} \tag{6.13}$$

Table 6.8 shows the result from the experiment using this normalised BoW kernel.

The result from BoW proves that it is a strong baseline for our task. This is in line with the results seen in the experiments by Han et al. (2006). The distributional hypothesis seems to holds at the word level. The tree kernel with word as node label achieve better performance except for *rule* entity type. The tree kernel with part-of-speech label outperforms the BoW kernel on all metric except for *person* on MRR

where the result is same.

6.6.1 Significance Tests

We have performed significance tests for the combined (all four entity types) mean reciprocal rank scores using approximate randomisation (Yeh, 2000) at 5% significance level. We use the sigf package (Padó, 2006) [2] for computing randomised statistics on mean reciprocal rank scores. The null hypothesis is that the two approaches are not different. The randomisation tests involves shuffling of the reciprocal rank scores and reassigning them to one of the two approaches. The impact of shuffling on performance is measured. This shuffling and re-measurement is performed 100000 times in our tests. The idea is that if the difference in performance is significant, random shuffling will only very infrequently result in a larger performance difference. The relative frequency of this event occurring can be interpreted as the significance level of the difference.

Table 6.9 shows the results of these significance tests.

System A	System B	p-value	Verdict
T_{pos}	T_{word}	0.002	Significant
T_{word}	T_{bow}	0.491	Not-Significant
T_{pos}	T_{bow}	0.008	Significant

Table 6.9: *Result from the approximate randomisation significance test*

The first row in Table 6.9 compares the result from the tree kernel with word labels and part-of-speech labels. The p-value from the 2-tailed test using approximate randomisation (100000 iterations) is 0.002 on reciprocal rank. Therefore we reject the null hypothesis. The second row compares the result from the tree kernel with word labels and bag-of-words kernel. The p-value from the 2-tailed test using approximate randomisation (100000 iterations) is 0.491 on reciprocal rank. Therefore we cannot reject the null hypothesis. The third row compares the result from the tree kernel with part-of-speech labels and bag-of-words kernel. The p-value from the 2-tailed test using

[2]The tool can be found at `http://www.nlpado.de/~sebastian/software/sigf.shtml`

approximate randomisation (100000 iterations) is 0.008 on reciprocal rank. Therefore we reject the null hypothesis. The results clearly show that tree kernel with part-of-speech labels perform significantly better than other kernel functions.

6.7 Conclusion

It is clear that noise is an important factor that needs to be addressed while training as well as testing. However, this is an inherent characteristics of data collected from the web. It is also reasonable to assume that the performance of the approach would improve if the noise was somehow removed or the learning and ranking was performed on a carefully crafted dataset. This is reasonable because the model we use relies mainly on the SVM posterior probability estimate. So cleaner data should equate to better learning. The best performing entity type was *disease* on all metrics. This was because it has the least noisiest of all the data. The results of significance tests show that the tree kernel significantly outperforms the BoW kernel. The results from significance test between word labels and part-of-speech labels show that better generalisation was obtained by using part-of-speech tags.

We have used the tree representation of the entire sentence for training. However, we should achieve better results if we could prune the tree. In case of fact seeking questions where expected answer type can be identified one way would be to only consider the sub-tree containing the mention of entity type and the answer term. However, for definition questions this is not possible. Considering that definition sentences are usually quite long, one could try to modify the kernel function so that sub-structures above a minimum depth or minimum branching factor are only considered. The task of designing a custom kernel for definition question answering is a big task on its own and we leave it as a future work.

Chapter 7

Learning Sentence Ordering

7.1 Introduction

One of the reasons to present a snippet rather than a single sentence arises from the difficulty in assessing the information need of the user (from the question alone). In case of definition questions, this is the characteristics of the expected answer. Having gathered a set of relevant definition sentences from the previous experiments, we would like to order it in a sequence that would be pleasing to the reader. Ideally we would like to present a coherent and cohesive definition snippet, similar to a nicely written summary. Not surprisingly, sentence ordering has been largely studied in the area of single and multi-document summarisation. It has been observed that even for questions whose information need can be determined, the reader prefers to get detailed information rather than an exact answer (Burger et al., 2001).

In this chapter we look at the task of sentence ordering and do not look at compression. We explore an approach for finding an order to a set of definition sentences by observing Wikipedia articles. More precisely, we attempt to organise the definition sentences in a way that it resembles the introductory section of a Wikipedia article. However we do not attempt to produce a cohesive text and also do not look into other aspects of producing a coherent text.

As in our earlier models, the experiments are performed for the four entity types: *person*, *company*, *disease* and *rule*. The models presented in this chapter gather statistics from the dataset belonging to each

of the entity types. The dataset is a collection of Wikipedia articles. Given a set of sentences to be ordered, we assume that we are provided with the information about the entity type. The models explored in this chapter exploit statistics and extract features from the respective dataset.

7.2 Ordering Sentences

Lapata (2003) presented a probabilistic approach to sentence ordering by learning the ordering constraints from a domain specific corpus. The probability of seeing a sequence of sentences $S_1...S_n$ can be written as

$$\begin{aligned} P(T) &= P(S_1...S_n) && (7.1)\\ &= P(S_1)P(S_2|S_1)P(S_3|S_1,S_2)...P(S_n|S_1...S_{n-1}) && (7.2)\\ &= \prod_{i=1}^{n} P(S_n|S_1...S_{n-i}) && (7.3) \end{aligned}$$

The calculation is simplified by assuming that the probability of occurrence of a sentence depends only on the previous sentence. This assumption can better handle the effect of data sparsity.

$$\begin{aligned} P(T) &= P(S_1)P(S_2|S_1)...P(S_n|S_{n-1}) && (7.4)\\ &= \prod_{i=1}^{n} P(S_i|S_{i-1}) && (7.5) \end{aligned}$$

Rather than computing $P(S_j|S_{j-1})$ directly, the estimates are based on looking at the features representing the sentences. Let $a_{i1}, a_{i2}...a_{in}$ be the features representing sentence S_i and $a_{(i-1)1}, a_{(i-1)2}...a_{(i-1)m}$ similarly for S_{i-1}. Assuming the features to be independent of each other, we have

$$\begin{aligned} P(S_i|S_{i-1}) &= P(a_{i1}|a_{(i-1)1})...P(a_{in}|a_{(i-1)m}) && (7.6)\\ &= \prod_{(a_{ij},a_{(i-1)k})\in S_i\times S_{i-1}} P(a_{ij}|a_{(i-1)k}) && (7.7) \end{aligned}$$

The approach starts by constructing a graph with all the N! orderings. A vertex in this graph represents a single sentence. Each

vertex is assigned a probability which is the product over the set of features representing this sentence. Now the node with the highest probability is ordered ahead of other nodes. The chosen node and its incident edges are deleted from the graph. $P(S_i|S_k)$ is computed for all of the remaining nodes given a selected node (S_k). This process continues until the graph is empty. The features they use are (a) verbs (lemmatized and non-lemmatized) (b) nouns (simple nouns, multi-word entities) (c) dependencies (triples relating to verbs, nouns and verb and nouns).

Biadsy et al. (2008) observed that the biographical pages in Wikipedia follow a general presentation template. They noted that birth information is mentioned before the death information. Current profession and institutional affiliations appear relatively early. Nuclear family members are generally mentioned before distant relations.

In their approach they use the position information of the sentences as they appear in the biographies and train an SVM regression model using class/lexical features of the sentence to its position. Class based features such as named entity tags (GPE (Geo-political Entity) and PER (Person)) are obtained from the output of a named entity tagger and corefferential resolver. Lexical features include unigrams and bigrams such as the tokens *born,became,was born, [TARGET_PER] died*. Here [TARGET_PER] is the label associated with the non-pronomial expression that corresponds to the target entity. The feature vector is an indicator of these features denoted by their counts.

Our model is similar to Lapata (2003). We also identify the next sentence to be ordered with the help of the last sentence that was picked. However our model is not defined in a strict probabilistic setting. The weight of a node in our model is based on the relative proximity and reachability statistics. Both of these statistics are defined later on. We also agree with the observations made by Biadsy et al. (2008) about the templated nature of Wikipedia articles. Our learning approach also relies on the original sentence location (position).

We have implemented Lapata(2003) and use it as the baseline model. The key difference in our implementation lies on how we estimate $P(S_i \mid S_k)$. We use the notion of similarity to compute the score for this conditional probability. We introduced this approximation in Chapter

4 and used it in the subsequent ranking models. We make the same approximation here i.e. $P(S_i \mid S_k) \simeq sim(S_i, S_k)$. By similarity we refer to cosine similarity of vectors.

In the next section we briefly describe how a sentence vector is built from a collection of texts. We discuss two popular approaches Latent Semantic Analysis (LSA) and Random Projection. We have used the *random projection* algorithm in the experiments. Subsequent sections will present the baseline model and our proposed model for sentence ordering.

7.3 Vector Based Similarity of Sentences

7.3.1 Introduction to Vector Space Model

A vector space model (VSM) represents every document in a collection as a vector. The component of that vector are terms present in the document. The total number of terms in the dictionary constitutes the dimension of the vectorial space (see figure 7-1). The angle θ between the vectors for documents d_1 and d_2 is used to determine the similarity of the two documents. The smaller the angle the more similar the documents. In section 7.3.1 we describe one of the most commonly used metric for similarity, the cosine similarity.

The weight of a component in information retrieval is usually the tf-idf metric.

$$tf\text{–}idf(t,d) = tf(t,d) \times idf(t,d) \tag{7.8}$$

Where,
$tf(t,d)$=number of times a term t appears in the document d
$idf(t,d) = \log \frac{N}{dc(t)}$

idf ensures that rare terms, terms that frequently occur in a small set of documents but only a few times in the whole collection, are assigned a greater weight than a term that occurs frequently in the whole collection. By using this weighting scheme, we are trying to locate terms that have discriminating properties meaning that the presence of such

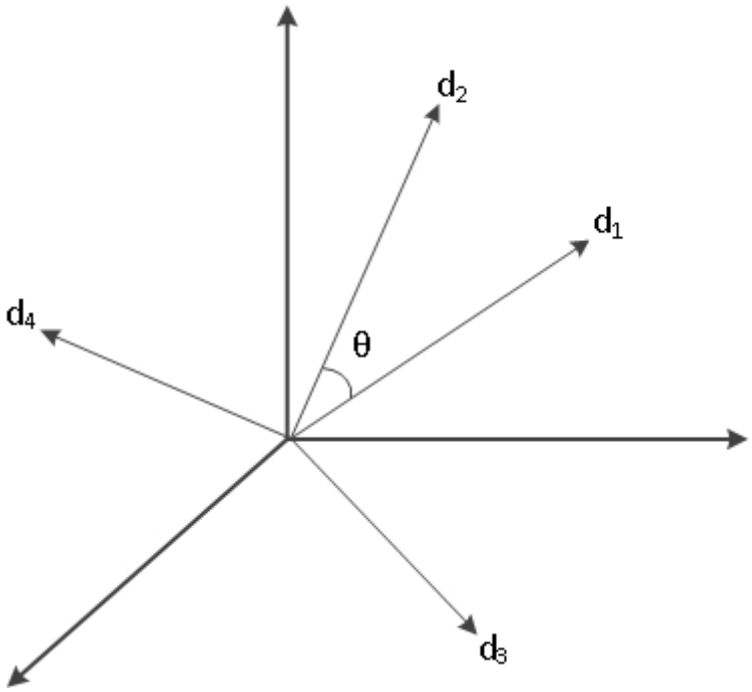


Figure 7-1: *Vector Space Representation*

terms in a document signals some specific nature about the document. For example, determiners occur very frequently in the entire collection. As such they have less discriminating properties. In an information retrieval task determiners are usually filtered out in the preprocessing stage. However, words like born and studied are words that we would expect to see in a document describing a person entity. Similarly we are likely to observe words such as founded and headquarters in a document describing some organisation. These words, by the virtue of appearing frequently in a small set of documents, gain a high discriminating power.

Here, $dc(t)$ is the total number of documents containing term t. In a normal scenario a vector in a VSM is very sparse meaning lot of its components have a zero weight. A collection therefore can be viewed as a matrix of size M by N. It is commonly known as a document-term matrix. Here, M is the total number of terms in the dictionary and N is the total number of documents. M is what defines the dimension of the vectorial space.

Efficient usage on vectors in a practical application is only possible after using some form of dimensionality reduction. One such way of

dimensionality reduction is singular value decomposition (SVD) and is explained further in chapter 7.3.2. Before that we describe how we can measure the similarity of two weighted vectors.

Cosine Similarity

Probably the most common metric used to compute similarity of two vectors is the cosine similarity. A cosine similarity measures the angle between two vectors in the M dimensional Euclidean space. Cosine similarity is defined in terms of dot product (also known as inner product) of the unit vectors. Formally, the cosine similarity $sim(t1, t2)$ is defined as

$$sim(d_1, d_2) = \frac{\vec{d_1} \cdot \vec{d_2}}{|\vec{d_1}||\vec{d_2}|} \tag{7.9}$$

Where,

$$\vec{d_1} = \vec{d}_{1,1}...\vec{d}_{1,M} \tag{7.10}$$
$$|\vec{d_1}| = \sqrt{\sum_{i=1}^{M} d_{1,i}^2}$$

The numerator is the dot product of the two vectors with M components. The denominator acts as a normalisation constant, cancelling out the large variation that might be observed due to the difference in the length of the documents for example. Usually we do not want our scoring function to be biased toward a longer document even when it might contain exactly the same terms as in the other shorter document. Because of normalisation we can compare a document vector with a query vector, for example.

7.3.2 Latent Semantic Analysis

LSA (Deerwester et al., 1990; Landauer et al., 1998) was originally developed to solve the problems of polysemy and synonymy in information retrieval. The basic idea behind LSA is the assumption that the semantic structure of a document can be captured and stored

in a term-context matrix representation. A term is a word and the context could be any meaningful unit such as sentence, paragraph or a document. The contexts in which a term does occur or does not occur determines the similarity between the terms. The cell contains the weighted frequency of terms in the context. Since we compare the similarity of two terms by comparing their contexts, it is not necessary that the two terms should ever occur together.

The original matrix representation is very large and sparse as a term usually occurs in a few contexts. LSA employs singular value decomposition (SVD) mainly for dimensionality reduction purposes. One way to perform LSA is:

1. First documents are collected and divided into contexts.

2. A co-occurrence matrix of term-context is created. The cell value contains the weighted frequency of a term in that particular context.

3. SVD is then performed. The dimension 'k' to be reduced to is determined empirically.

Singular Value Decomposition

Singular Value Decomposition (SVD) is based on the theorem from linear algebra which states that for a rectangular matrix A (m x n), there exists a factorisation such that $A = U\Sigma V^T$. The original matrix A is decomposed into three matrices U (m x r), Σ (r x r) and V (r x n). Here r is the rank of the matrix. U and V are orthogonal matrices and Σ is a diagonal matrix. Calculating the SVD of a matrix consists of finding the eigenvalues and eigenvectors of AA^T and A^TA. The eigenvectors of A^TA make up the columns of V. The eigenvectors of AA^T make up the columns of U. The singular values in Σ are square roots of eigenvalues from AA^T or A^TA. The values are ordered from largest to smallest along the main diagonal of the matrix. This is known as singular value decomposition because the factorisations generates eigenvalues that makes the Equation 7.11 true.

$$|A - \lambda I| = 0 \tag{7.11}$$

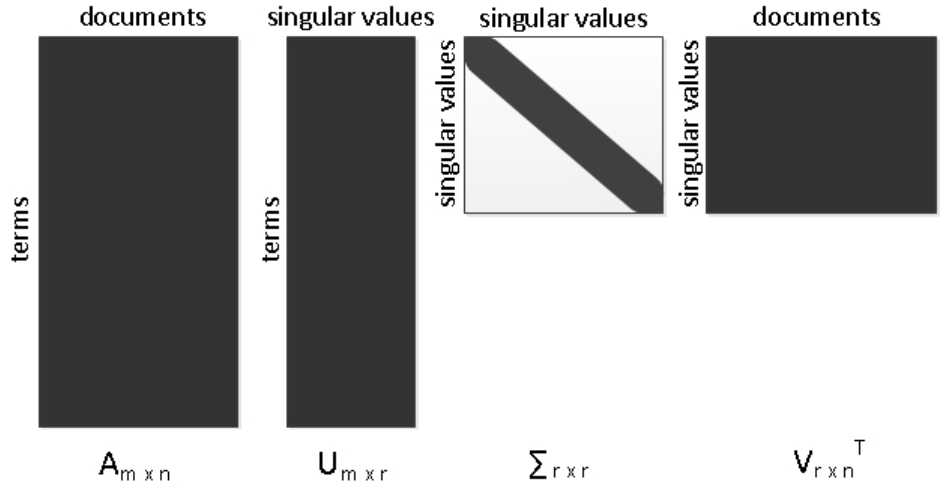


Figure 7-2: *SVD of a Matrix. Here r is the rank of the matrix*

Figure 7-2 shows the graphical representation of the matrices.

Furthermore, all but the 'k' highest singular values are set to zero. Usually 'k' is very, very small compared to the original dimension of the matrix. We compute similarities of terms in this reduced space. The discarded dimensions are assumed to be the result of noise or chance associations. Therefore the new product will result in an approximation of the original matrix A. As a result of this transformation, the cell values can change from the original values. Basically, some data points will move closer together and some will move further apart. Figure 7-3 shows the graphical representation of the matrices.

7.3.3 Random Projection

In random projection, the d-dimensional data is projected through the origin down to a k-dimensional subspace formed by a set of random vectors. Mathematically,

$$A_{k\times n} = R_{k\times m} \cdot X_{m\times n} \tag{7.12}$$

This reduction process is based on the Johnson-Lindenstrauss lemma (Johnson and Lindenstrauss, 1984). For any $0 < \varepsilon < 1$ and any integer n, let k be a positive integer such that

$$k \geq 4(\varepsilon^2/2 - \varepsilon^3/3)^{-k} \ln n \tag{7.13}$$

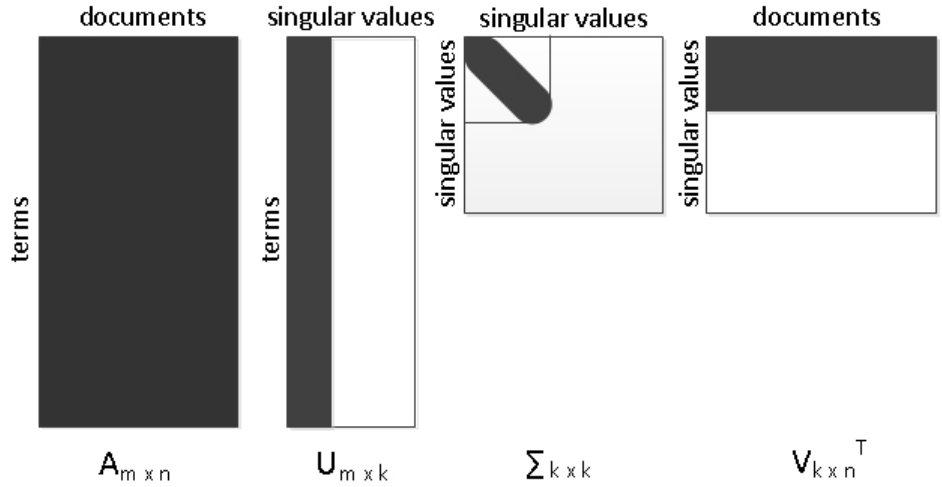


Figure 7-3: *SVD of a Matrix. Here $k \ll n$*

To put in words, a set of n points in a high-dimensional Euclidean space can be mapped down onto an $O(\log n/\varepsilon^2)$ dimensional subspace such that the distances between the points are approximately preserved, for any $0 < \varepsilon < 1$.

Typically the elements of the random matrix R are Gaussian distributed. Two such common distributions are:

$$r_{i,j} = \begin{cases} +1 & \text{with prob } \frac{1}{2} \\ -1 & \text{with prob } \frac{1}{2} \end{cases}$$

and

$$r_{i,j} = \sqrt{3}. \begin{cases} +1 & \text{with prob } \frac{1}{6} \\ 0 & \text{with prob } \frac{2}{3} \\ -1 & \text{with prob } \frac{1}{6} \end{cases}$$

If the random vectors are orthogonal, then the similarities between the original vectors are exactly preserved. However, the cost of orthogonalisation is expensive. The saviour is in the form of the observation that in a high-dimensional space, a much larger number of almost orthogonal vectors exist in comparison to the number of truly orthogonal vectors (Hecht-Nielsen, 1994).

We use the SemanticVectors[1] library (Widdows and Ferraro, 2008)

[1]See http://code.google.com/p/semanticvectors

that uses random projection for dimensionality reduction for calculating semantic similarity of terms, specifically sentences. The sentence vector is constructed by producing an aggregated word vector.

7.4 Probabilistic Sentence Ordering

We implemented the probabilistic sentence ordering approach of Lapata (2003) as the baseline approach. After making a simplifying assumption that the probability of the current sentence only depends on the previous sentence that was selected, the task eventually boils down to the calculation of the conditional probability $P(S_i \mid S_{i-1})$. Simply put this conditional probability can be read as "what is the probability that the sentence S_i will be seen after sentence S_{i-1}". A straightforward estimation would involve counting of the number of times the sentences co-occur in a corpus. It is likely that the exact sentences might not be present in the corpus and working at the sentence level becomes infeasible. The way she calculates this is by extracting features that are relevant to the sentences.

In our model that follows, we use the values from a similarity function to judge if two sentences are talking about the same thing. This is often described as calculating the semantic similarity of sentences. Here semantic similarity reflects the closeness of the sentences (vectors) in a vector space. To align the baseline model with our model we make an assumption that in a text, two sentences are located close to each other if they are similar and further away if they are dissimilar. In our case closeness is measured in terms of the relative position of the sentences in a wikipedia paragraph. Intuitively the similarity score provides an indication of the relative ordering of sentences in a text. Therefore the conditional probability $P(S_i \mid S_{i-1})$ can be approximated to be the value resulting from $sim(S_i, S_{i-1})$. Here $sim(S_i, S_{i-1})$ calculates the semantic similarity of sentences S_i and S_{i-1}. Sentences are treated as bag-of-words and the final vector representation is obtained by addition of word vectors.

Given a finite set of sentences $\{S_1, S_2, ..., S_n\}$ to be ordered, the method proceeds as follows. First we identify the most likely sentence to be placed in the first position of the ordered list. After that we

search for the most likely candidate sentence to follow the first sentence. This simply involves calculation of similarity scores between the first sentence and the remaining sentences. The sentence which is most similar (largest semantic similarity score) with the first sentence is then placed in the second position. Similarly we identify the sentence to fill the next slot based on its closeness with the sentence in the slot above it. The algorithm proceeds in the same manner until all sentences are consumed.

In this approach the key task is to identify the first sentence. The rest of the ordering just involves looking at the semantic similarity scores. To identify the first sentence, we compiled a list of first sentences from the introductory paragraph in the Wikipedia articles. Consider the case of the *person* entity type. From the dataset which consists of 354 articles we collected 354 sentences. For each of the sentences that are to be ordered, we calculate semantic similarity score with all 354 sentences. The unordered sentence with the highest score is taken to be the first sentence.

Before presenting the results of the experiment, we describe our approach in the next section. The main focus of this approach is to utilise the already existing ordering information implicitly present in the Wikipedia articles. In the baseline approach we utilise the first sentences in the Wikipedia articles. In our proposed approach we learn from the ordering of sentences in a Wikipedia summary. Learning involves construction of a directed graph from the content of the Wikipedia articles. A node in the graph represents a sentence. An edge in the graph connects two nodes(sentences). The presence of an edge between two nodes indicate that a node follows the other node in an ordering. The directed edge preserves the ordering information.

7.5 Learning to Order Sentences

7.5.1 Graph Construction

The proposed sentence ordering approach first builds a graph from the summary section of Wikipedia articles. A separate graph is built for each of the four entity types. An ordering experiment has an entity

type associated with it and only the appropriate graph will be used. In this graph a node represents a unique sentence from the summary collection. An edge (u, v) denotes that the sentence represented by node v follows the sentence represented by node u in some article. Later on in the algorithm, the edge list of the graph will be updated to reflect additional paths that were identified by locating similar nodes. This is to say that if there is an edge (u, t) and nodes v and u are similar, we add an edge (v, t) to the graph. Similarly if there is an edge (t, u) we add an edge (t, v) to the graph. By similar nodes we mean that the content of the nodes are semantically similar. The following algorithm shows how the graph is built.

Algorithm 1: Build a directed graph from Wikipedia articles

Input: A finite set $W = \{w_1, w_2, \ldots, w_n\}$ of Wikipedia articles
Output: A directed graph G

```
Construct a graph G with an empty node START for each w_i ∈ W do
    Add a node for each sentence in w_i
    if sentence represented by n follows m in the text then
        Add an directed edge (m, n)
    Add an edge (START, s_1). Here, s_1 is the first sentence in w_1
for addition of a new article to G do
    for each node v ∈ V(G) do
        for v_i similar to v in the neighbourhood do
            Add edges to connect v to terminal vertices of edges with v_i
            Add edges from all initial vertices of edges with v_i to v
return G
```

Viewing Article Summary as a Graph

Figure 7-4 shows a snippet taken from the summary section of Wikipedia entry for Barack Obama. Similarly Figure 7-5 shows a snippet taken from the summary section of Wikipedia entry for David Cameron.

To construct a graph we treat each sentence as a node in the graph. We then add a directed edge from node n_i to n_j if n_j follows n_i in the article. We do this for every Wikipedia article for each entity type. We then introduce an empty ROOT node in the graph. From the ROOT

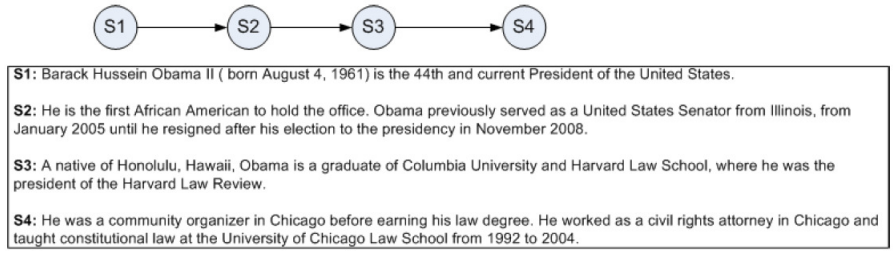


Figure 7-4: *Few introductory sentences from the Wikipedia article on Barack Obama represented as a graph*

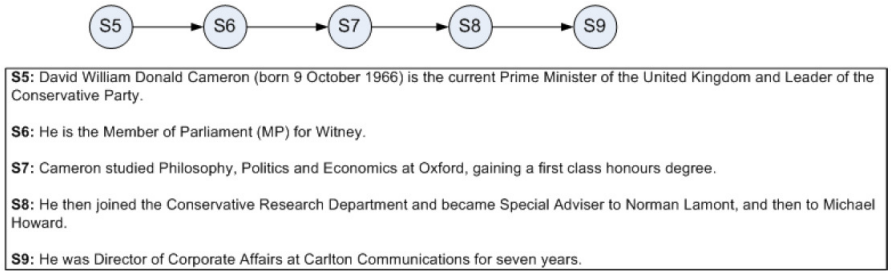


Figure 7-5: *Few introductory sentences from the Wikipedia article on David Cameron represented as a graph*

node we add an edge to the first node of the graph we had created earlier. We then end up with a graph as shown in Figure 7-6.

At this stage the total number of nodes is equal to the total number of sentences in our collection plus the ROOT node. We construct such a graph for each of the four entity types separately. The next step in learning is finding nodes that are similar.

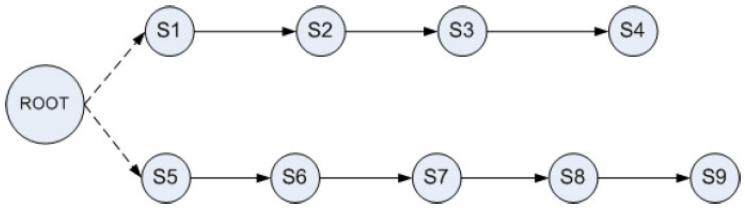


Figure 7-6: *Combining all articles into one single graph.*

Finding Similar Nodes

The similarity of two nodes is obtained by calculating the similarity between the sentences they hold. We have used the random projection approach available in SemanticVectors package (Widdows and Ferraro, 2008) to build the vector space. The vector representation of a sentence was composed by vector addition for the individual words in the sentence. The main reason for choosing a simple composition method was the due to execution time, both of training and testing.

Computing semantic similarity for a sentence is computationally expensive in the case of very large graphs. To reduce this cost, we make a simplifying assumption. This assumption is based on the idea that authors of Wikipedia articles tend to follow a standard guideline. We hypothesise that articles belonging to the same entity type will have similar looking articles. For example if we look at the two examples given above, the first sentence introduces the person. The second sentence gives information about the current job. So when searching for similar nodes, we only look at a set of neighbouring nodes. For example, a neighbourhood of less than or equal to one would mean looking at nodes lying one hop in front and behind the node being examined. We take the distance as the number of hops from the ROOT node to each node. Note that in our initial graph there is only one path from the ROOT node to any other node.

From this process we end up with a list of similar nodes for each node with the similarity score. A manually set threshold determines if the two nodes (sentences) are similar or not. We remove nodes with a similarity score less than the threshold from the similarity list.

Updating Edges of Similar Nodes

From each of the similar nodes in the similarity list, its outgoing and incoming edges and hence the nodes are collected. Outgoing edges from the node being examined to each of the outgoing nodes of the similar node are added. Similarly edges from the incoming nodes of the similar nodes are also added. Note that the incoming and outgoing sets are computed from the initial graph. As a result of this process, we end up with a graph with a large number of edges. For each input sentence we

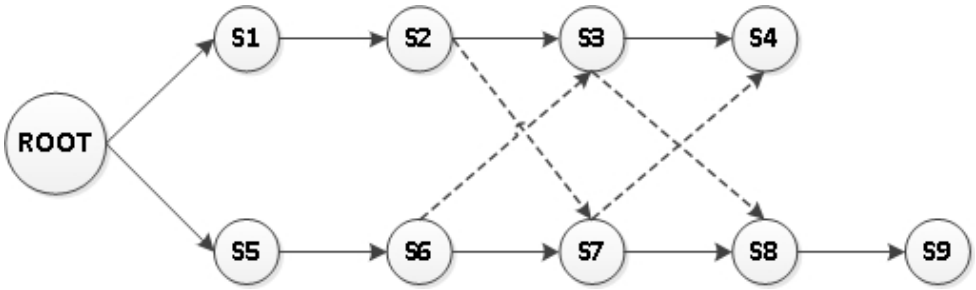


Figure 7-7: *Graph after the update process. The dotted edges denote newly added edges.*

find out the most similar node in the graph at the ordering stage.

Consider Figure 7-6 for illustration. Suppose node S3 is similar to S7. All the adjacent vertices of S3 now become adjacent to S7. Similarly all the adjacent vertices of S7 become adjacent to S3. The resulting graph is shown in Figure 7-7. The dashed lines indicate the newly added edges.

We started with a graph constructed from two documents. Let $D_1 = S1, S2, S3, S4$ and $D_2 = S5, S6, S7, S8, S9$ be the documents depicted in the figure. By the addition of these new edges, we can further generate new documents with these sequences of sentences: $D_3 = S1, S2, S7, S8, S9$, $D_4 = S1, S2, S7, S4$, $D_5 = S1, S2, S3, S8, S9$, $D_6 = S5, S6, S3, S4$, $D_7 = S5, S6, S7, S4$ and $D_8 = S5, S6, S3, S8, S9$. Basically by identifying similar nodes and adding new edges, we find alternative definition compositions. Although we do not look at definition text generation, if slots were to be identified from the sentences, by adding related filler content we could generate a definition chunk. As we keep on doing this for all pairs of similar nodes, we end up with massive number of edges.

7.6 Ordering Sentences

Finding an order to a set of sentences $S = \{S_1, S_2, \ldots, S_m\}$ uses the graph that was built earlier. The process is described in the Algorithm 2. The process involves mapping the unordered sentences on to the nodes in the graph. By mapping we mean finding a node (representing a sentence in the Wikipedia article) that is most similar to each of the

unordered sentences. Let these nodes be placed in a list called *Stamped*. The task of finding an order is reduced to selecting a node with greatest score from *Stamped* one at a time and placing it in the ordered list. The next section deals with the formulation of the score function for a node.

Algorithm 2: Find an ordering for a set of unordered sentences

Input: A finite set $S = \{S_1, S_2, \ldots, S_m\}$ of unordered sentences
Output: An ordered set O

```
for each s ∈ S do
    Find a node n ∈ G with content most similar to s
    Add this node to the list Stamped
Find node l ∈ Stamped with maximum score(l) from the START node
Add l to the Visited list
while |O| < |S| do
    Find node p in Stamped with largest score(p) not yet visited
    Add p to O
    Add p to Visited
return O
```

7.6.1 Score Function for a Node

The weight of a node is computed using Equation 7.14

$$score(n) = reachability(n) \times proximity(n) \tag{7.14}$$

Here, $reachability(n)$ is the visibility factor and $proximity(n)$ measures the relative proximity. The terms have very intuitive meanings which are described below. An ideal node would have high reachability and close proximity.

Reachability of a Node

For the input sentences $s_1, s_2, ..., s_k$ the first task is to find the most similar node for each of the sentences. Let nodes $n_1, n_2, ..., n_k$ be the nodes that were selected. We call these nodes the stamped nodes. In the first step we identify the stamped node closest from the ROOT node. Assume n_1 was selected. For each of the remaining nodes we compute the reachability term.

$$reachability(n_l) = \text{number of stamped nodes that can be reached from } n_l \tag{7.15}$$

A node n_l is reachable from node n_1 if there is a path from n_1 to n_l.

Relative Proximity

Staying with the same scenario as above, let us assume that n_1 was the first node to be chosen. For each of the nodes that remain to be ordered, denoted by $remain(N)$, we compute the shortest path from n_1. The shortest path denotes the number of hops from the start node to the end node obtained using Dijkstra's shortest path algorithm (Dijkstra, 1959). The intuition behind this factor is to choose the node that is closest to the last selected node (n_1).

Therefore, $proximity(n)$ is defined as

$$proximity(n) = \log \frac{\max\limits_{k \in remain(N)} dist_{n-1}(k)}{dist_{n-1}(n)} \tag{7.16}$$

7.7 Experimental Setup

The dataset used for training and testing are described in the section 3.2. A dataset was collected for each of the four entity types, *person*, *company*, *disease* and *rule*. The number of training files for *person*, *company*, *disease* and *rule* were 354, 250, 90 and 101 respectively. The number of test files for *person*, *company*, *disease* and *rule* were 48,50, 49 and 41 respectively. The ordering of the test sentences are reversed before presented for reranking. This is done so as to avoid cases where a correct ordering might be the result of chance. For example when two nodes attain exactly the same score, the first node would be chosen. By reversing the list, we actually penalise these cases.

Calculating the semantic similarly of a pair of sentences is computationally expensive. This usually involves similarity calculation between all pair of words. Ideally we would like to explore the entire graph to search for similar nodes during the learning process. As a workaround to this problem we make an assumption that the positions of similar

sentences in a piece of definition text do not vary by a large number. This is an acceptable enough assumption considering that Wikipedia tries to achieve a uniform looking introductory section across all the articles belonging to an entity type. In relation to our experiment, the position of a sentence (node) is measured as the number of hops it takes from the ROOT node to reach the node in question. For example in Figure 7-6, the distance from ROOT to sentence $S7$ is 3, from ROOT to $S4$ is 4 and so on.

7.7.1 Evaluation Metric

For the evaluation of the results, the ordering of sentences produced by our approach is compared with the gold standard ordering. An exact ordering produces a perfect score. We have used the Kendall's τ measure for evaluation. Kendall's τ has been used for evaluation of sentence ordering sub-task within the task of summarisation (Lapata, 2003; Barzilay and Lee, 2004) and concept-to-text generation (Karamanis, 2003; Karamanis and Mellish, 2005). It has been shown that this measure correlates reliably with human ratings (Lapata, 2006).

Let $S = \{s_1, s_2, ..., s_N\}$ be the set of sentences that we are going to find an ordering for. Let π and σ be two ordering for the set S. In our experiments one of them will be the gold standard ordering and the other the ordering produced by the experiment. Let $D(\pi, \sigma)$ denote the total number of interchanges of consecutive elements (adjacent transpositions) needed for π to look like σ. Therefore Kendall's τ can be defined as:

$$\tau = \frac{2D(\pi, \sigma)}{N(N-1)/2} \tag{7.17}$$

The metric ranges from -1 to +1. Minus one indicates the two orderings are complete reversals of each other and plus one indicates they are identical. Kendall's τ also penalises inverse rankings. We use this metric because it has a clear interpretation for our task.

Threshold	Person	Company	Disease	Rule
0.40	0.27	-0.06	-0.05	0.01
0.50	0.27	-0.04	-0.07	-0.02
0.60	0.21	-0.09	-0.11	-0.01
0.70	-0.07	-0.47	-0.66	-0.16

Table 7.1: *Results for $Neighbourhood = 0$. Cell values are the Kendall's Tau scores.*

Threshold	Person	Company	Disease	Rule
0.40	0.15	-0.15	-0.07	-0.04
0.50	0.18	-0.13	-0.04	-0.06
0.60	0.21	-0.03	-0.03	-0.03
0.70	0.19	-0.26	-0.57	-0.04

Table 7.2: *Results for $Neighbourhood \leq 1$. Cell values are the Kendall's Tau scores.*

7.8 Results And Analysis

7.8.1 Neighbourhood and Threshold

The best result was obtained when *Neighbourhood* was set to zero and *Threshold* was set to 0.40 or 0.50. Results are shown in Table 7.1. A *Neighbourhood* of zero greatly reduces the count of similar nodes to each node. As we increase the *Threshold*, this number is further reduces. What this means is that we end up with a graph that is not well connected. The set of nodes that can be reached from a particular node is far less when the *Threshold* is set to a lower value. This will also result in low reachability scores. As we can see from the results

Threshold	Person	Company	Disease	Rule
0.40	-0.48	-0.47	-0.39	-0.46
0.50	-0.49	-0.47	-0.42	-0.47
0.60	-0.38	-0.36	-0.13	-0.15
0.70	0.01	-0.31	-0.47	-0.14

Table 7.3: *Results for $Neighbourhood \leq 2$. Cell values are the Kendall's Tau scores.*

Threshold	Person	Company	Disease	Rule
0.40	-0.57	-0.56	-0.52	-0.55
0.50	-0.55	-0.51	-0.45	-0.56
0.60	-0.47	-0.37	-0.21	-0.38
0.70	0.06	-0.20	-0.27	-0.20

Table 7.4: *Results for $Neighbourhood \leq 3$. Cell values are the Kendall's Tau scores.*

presented in Table 7.1, the performance degrades as we increase the $Threshold$. However this does shed a positive light on the assumption that position of similar sentences across articles are similar.

However, this behaviour changes when we increase the scope of $Neighbourhood$. As we increase the value of $Neighbourhood$, the count of similar nodes are increased. By decreasing the value of $Threshold$, this count is further increased. But by increasing the $Neighbourhood$ the connectivity in the graph is increased to a degree where the reachability is good enough and higher value of $Threshold$ gives better performance. In case of $Neighbourhood \leq 3$, results shown in Table 7.4, the best results were obtained when $Threshold$ was set at 0.70. In the case of $Neighbourhood = 2$, results shown in Table 7.3, the best results were obtained when $Threshold$ was set at 0.70 as well. In case of $Neighbourhood \leq 1$, results shown in Table 7.2, the best results were obtained when $Threshold$ was set at 0.60.

It is obvious that when we use a lower $Threshold$ the number of false positives increases and the performance degrades. If we look at the results where $Neighbourhood \leq 3$, the performance is significantly worse at $Threshold = 0.40$ compared to $Threshold = 0.70$. This is true when $Neighbourhood \leq 2$ as well. Performance only degrades slightly in the case of $Neighbourhood \leq 1$. This could mean one or both of two things: (1) the position of similar sentences across articles are extremely similar (± 1) and (2) the advantage of connectivity outweighs the effect of selecting false positives.

The only entity type for which we achieve reasonable performance is *person*. This would suggest that the introductory sections of articles in this domain are much alike. If we look at articles belonging to this entity this does seem to back the results we have got. As we can see from

Threshold	Person	Company	Disease	Rule
0.40	0.33	-0.06	-0.06	0.06
0.50	0.32	-0.05	-0.14	0.06
0.60	0.13	-0.17	-0.26	0.04
0.70	-0.28	-0.67	-0.75	-0.38

Table 7.5: *Results for $Neighbourhood = 0$ with only reachability(n) used in scoring. Cell values are the Kendall's Tau scores.*

the two sample articles in Figure 7-4 and 7-5 the degree of similarity increases when we look at a more fine grained entity type resolution. Both these examples belong to the *politician* sub-type of *person*.

7.8.2 Scoring Function

While defining the scoring function for a node, one hypothesis was that the node closest to the last selected node has a better chance of being selected as the next node to be visited. To see if this *proximity* component contributes to the scoring function, we ran two experiments with only the *reachability* aspect. The fist experiment was with $Neighbourhood = 0$ and the second with $Neighbourhood = 3$.

Table 7.5 lists the results of $Neighbourhood = 0$ with only the *reachability* component. If we look at the result when $Threshold$ was set to 0.70, we see that the performance is significantly worse compared to the complete scoring function. With a high $Threshold$ value and a low $Neighbourhood$ lookout, the connectivity in the graph is very low. However performance at lower $Threshold$ values are comparable suggesting that *Reachability* is more important when the $Neighbourhood$ is set to a low value.

Table 7.6 lists the result from the $Neighbourhood \leq 3$ with only *reachability* component. The scores are bad compared to the case when the complete scoring function is used. In the case of $Threshold = 0.40$, the ordered lists are the complete opposite of the test dataset. The scores do not improve much when $Threshold$ is increased. This clearly indicates that the *Proximity* component plays a vital role when the graph is well connected.

The relation of the individual components in the scoring function

Threshold	Person	Company	Disease	Rule
0.40	-0.99	-1.00	-1.00	-1.00
0.50	-0.95	-0.99	-0.97	-1.00
0.60	-0.72	-0.88	-0.80	-0.87
0.70	-0.28	-0.38	-0.53	-0.67

Table 7.6: *Results for $Neighbourhood \leq 3$ with only reachability(n) used in scoring. Cell values are the Kendall's Tau scores.*

Approach	Threshold	Person	Company	Disease	Rule
$P_{N=0}$	0.40	0.27	-0.06	-0.05	0.01
$P_{N\leq 1}$	0.60	0.21	-0.03	-0.03	-0.03
$P_{N\leq 2}$	0.70	0.01	-0.31	-0.47	-0.14
$P_{N\leq 3}$	0.70	0.06	-0.20	-0.27	-0.27
$P^{R}_{N=0}$	0.40	0.33	-0.06	-0.06	0.06
L_b	0.40	0.22	0.11	0.19	0.15
L_b	0.50	0.22	0.13	0.14	0.17

Table 7.7: *Summary of the best results from the proposed approach (denoted by P with subscript N for Neighbourhood and superscript R for the use of reachability statistics only) and the baseline (L_b).*

therefore can be seen to be analogous to the behaviour of the components in the *tf-idf* metric. In the case of *tf-idf* terms that occur frequently but only in few documents have high discriminating power. Similarly, nodes with high *Reachability* and small *Proximity* are most likely to be chosen next (keeping *Neighbourhood* constant). By decreasing the value of *Threshold* we can increase the connectivity of nodes in the graph. This will result in an increased *Reachability* score but it also means that a lot of nodes may end up with the same degree of *Proximity*. A similar case is observed when we keep the value of the *Threshold* as a constant and increase the value of the *Neighbourhood* variable. It is clear that both of these components are necessary in order for the scoring function to be stable.

7.8.3 Baseline Approach

The best results from the proposed approach and the baseline are shown in Table 7.7. Compared to the result from the proposed approach ($P_{N=0}$

and $P^R_{N=0}$), the performance of the baseline (L_b) is slightly worse for the *person* entity type. The reason why the proposed approach performs better is because the introductory section of the Wikipedia articles for people look alike. The variation in the writing style and the information contained in the introductory section does not vary as much as for other entity types. It is evident that the the authors of the Wikipedia article have a clear understanding of what a article about a person should look like. Intuitively as well it is easier to figure out the contents for a *person* entity type and people have experience of writing such articles even before the Internet. For example the introductory section across such articles contains distinct paragraphs on personal information, education history, major achievements and career trajectory. Both the baseline and the proposed approach perform the best in the case of the *person* entity type. This reinforces the uniformity in the articles devoted to a person.

But not all the topics available in Wikipedia were widely written about and clearly understood before Wikipedia was popular (such as programming languages). However, it is a matter of time before the users' need on other topics are understood equally well and the articles will have uniform content. For the remaining three entity types (*company*, *disease* and *rule*) the baseline achieves the best scores. This is due to the variation in the Wikipedia articles of instances belonging to these entity types. This clearly suggests that the training set for these three entity types will have to be larger than the set for the *person* entity type. The baseline approach benefits from the fact that the test paragraphs are taken from Wikipedia articles and these sentences exhibit better coherence. The proposed approach relies on the graph constructed from articles that exhibit a lot more variations. The performance of the baseline would look very different if the sentences were compiled from the results of a search engine. In such a scenario the similarity scores between pairs of sentences would be significantly less. As a result the performance of both the approaches would not be significantly different.

The best results are achieved when the threshold is set in the mid-range (between 0.40 and 0.50). This is true for the baseline as well as the proposed approach. The performance degrades sharply when the threshold is increased beyond 0.50. This clearly shows that the

Threshold	Person	Company	Disease	Rule
0.40	0.22	0.11	0.19	0.15
0.50	0.22	0.13	0.14	0.17
0.60	0.17	0.06	-0.03	0.15
0.70	-0.004	-0.01	-0.21	0.17

Table 7.8: *Results from the baseline approach. Cell values are the Kendall's Tau scores.*

information contained in a sentence is not repeated. However the scores are pretty low for the baseline as well as the proposed approach. One of the major reasons is down to the use of a single feature, the semantic similarity. However, a random compilation of a set of features does not necessarily guarantee better results. Rather a careful selection of a set of features should produce a significant improvement. Work by Surdeanu et al. (2008) suggests that using an ensemble of features should perform better than a single feature in calculating the similarity. Even in this case an effective procedure is required to identify the features that would be part of such an ensemble. The accuracy of the similarity measure, specifically for the mapping scenario, has a large influence on the performance of the proposed approach. In the proposed approach the similarity function is used in two places. First it is used to detect nodes that are similar and secondly it is used in the mapping of the unordered sentences. Since the content of the articles belonging to the three entity types vary a lot in comparison to the *person* entity type, the performance suffers dramatically. A promising solution to this similarity bottleneck could be resolved by adopting the entity-aspect model (Li et al., 2010) which is outlined in section 8.3.2. However, that is a big undertaking on its own and we leave it as a future work.

Table 7.8 shows the complete result from the baseline approach of Lapata (2003). The results from the baseline reinforces the writing guidelines followed by the authors of the Wikipedia articles. The best performance is observed when the threshold is set at the lower end. This is similar to the observations from the proposed approach. Basically, the consecutive sentences in these articles contain distinct information. In the case of an article about a person, the introductory section is

divided into paragraphs that talk about a specific theme such as education, childhood, career and achievements. Both the approaches would perform better if the similarity metric is able to identify the theme from the sentences. The entity-aspect model (Li et al., 2010) would be one way to improve the similarity function and significantly improve upon the performance. However, the aim of this chapter is to introduce a model that is simple to interpret and easy to extend. Composing a similarity function that would capture the similarity between sentences is a large undertaking on its own.

7.9 Conclusion

The framework we have presented in this chapter is simple and fairly intuitive. The scoring function is very similar to the tf–idf metric in terms of interpretation and looks to be stable. The best performance was achieved for the *person* entity type. This shows that the authors of the Wikipedia follow a distinctive structure. The major limitation of the proposed approach is that the performance relies heavily on the accuracy of determining the similarity of nodes (sentences). By looking at the score for sets of similar sentences we can reliably say that semantic similarity of sentence is not enough on its own to judge if two sentences present the same piece of information. For entity types other than *person* a better result can be obtained when a good similarity metric is used on a larger set of training instances. It has been observed that larger variations in definitions can be seen for these three entity types and therefore they require a larger training set. However, the performance will not improve only by increasing the training dataset if the similarity metric is not good. The baseline model performs better than the proposed approach for three out of the four entity types. The simplicity of the baseline with no manually set parameters is a strong baseline. Significant performance improvements will be observed if we can find an accurate similarity function and a better mapping strategy. A promising solution to this similarity bottleneck could be found by using the entity-aspect model (Li et al., 2010). The main objective here is to recognise words that contribute in determining the nature of information being presented in the information. Our framework does

not have a method to find out which words in a sentence are important apart from selecting non-stopwords. This could be a potential area for exploration.

Chapter 8

Conclusion

8.1 Contributions

The main contributions of the thesis are in the reranking and ordering tasks related to definition question answering. The main contributions are:

1. A probabilistic framework for ranking definition sentences
2. Using tree similarity functions to estimate probabilities
3. Learning sentence ordering

8.1.1 A probabilistic framework for answer reranking

We started with a standard ranking model in a language modelling style, similar to the one defined by Han et al. (2006). One of the limitations of their approach was that their model was based around counting word unigrams. The problem we looked at was: how to incorporate richer information such as dependency trees and how to estimate probabilities for trees. The solution that we reached was on the observation that the task of reranking does not require probability estimates to be exact. This flexibility allowed the introduction of the notion of similarity directly in the probabilistic model. Although the focus of the thesis was on definition question answering, the framework

is flexible enough to handle any question types as long as (i) the distribution hypothesis is valid and (ii) a similarity function over the features can be defined.

8.1.2 Incorporating tree and learning in the framework

The first attempt to measure tree similarity was performed by computing the tree edit distance. The edit distance measure has been used before in a question answering task (Punyakanok et al., 2004). Although the work does not define a complete model like in this thesis, it does use edit distance between the question sentence and the candidate sentence to identify the correct answer. Basically the candidate answer with the lowest distance is chosen as the answer. In this thesis we define a formal way to compute score for a candidate sentence. The main problem was to compute $P(f_i|T_s)$. This estimate required counting similar trees. Two trees were considered similar if the edit distance was within the threshold. A manually set threshold was defined in this work. Although the results from the experiment were not good, the approach provides a way to incorporate structural features in a statistical setting. Improvements can be achieved by a better tree representation.

The edit distance based approach had a few limitations: (i) a manually set threshold was used (ii) the edit distance was computed for the entire sentence length (complete tree). The learning model we presented used the results from the learnt SVM classifier to estimate probabilities thereby removing the need of manually set threshold. The second limitation was removed by using a standard tree kernel that computes tree similarity by counting common sub-structures. This is a more reasonable estimate as two similar sentences are likely to share only part of their structure.

The results clearly showed that (i) using svm to estimate posterior probability was a better choice than using a set threshold and (ii) (sub)tree kernel functions are a more intuitive measure of similarity compared to edit distance. We also observed that a bag-of-words kernel is a strong baseline for our task as evident from the results of Han et al. (2006). This clearly points to the validity of the distributional

hypothesis. The tree kernel was able to produce significantly better results overall. This also sheds a positive light on the validity of our assumption. We did observe entity specific patterns although they also identified false negatives in some cases. A richer representation of nodes should be able to reduce the number of false negatives.

8.1.3 Learning sentence ordering

Another task in definition question answering that we explored was to present the candidate sentences in a coherent chunk. The sentence ordering approach we developed is a simple graph based approach. The first problem was to decide a source of definition texts to learn the ordering from. Wikipedia due to its acceptability and replicated article style was chosen as the desired source. We developed a strategy to visualise the Wikipedia introductory paragraph and capture the sentence ordering as seen in the original article. After similar nodes to the sentence to be ordered were located in the learnt graph, the problem was transformed to the problem of visiting nodes in order. The order in which the nodes were visited was the ordering that was enforced. A metric to compute a score for a node was also introduced which has two components: proximity and reachability.

8.2 Limitations of the Approaches

In the reranking task we considered definition questions of the simplest form. We did not consider questions with extra information such as user clues and noise. We also did not handle the case of compound definition questions such as "What is Wikipedia and Google?" or "Who are Laurel and Hardy?". In both cases we consider the compound target as a single entity. Most of these ambiguities raised by the lack of information in the question or additional noise could only be handled through a clarification dialogue with the user. This thesis does not deal with the cases where such kind of confusion arises. This is clearly a limitation that has to be handled if it is to be deployed in the real world.

The tree representations that we have used throughout the rerank-

ing tasks are the simplest form of generalisation that can be achieved. One of the strategies that we have used is to identify complex entities and merge them before passing on to the parser. For example "New York" is taken as a single token. Better generalisation is achieved by using part-of-speech tags as node labels. However, there has been a significant amount of work on encoding richer information in the vertices. Adding semantic classes to vertices by searching WordNet is one of the popular methods to introduce semantic information. Much richer representations such as PAS trees have been explored in Moschitti et al. (2007). Encoding semantic information could improve the performance of the classifier and the similarity measure.

One clear limitation comes from the distributional hypothesis assumption. Not all question types may adhere strictly or allow reformulation to stick to this assumption. 'Why' questions, as we discuss below, can be reformulated in a way that obeys the distributional hypothesis. We can even assume the topic and answer-type pair represent a class in a factoid setting. For example in the case of "How tall is Mt. Everest?" we can consider it to belong to the class type $< MOUNTAIN, HEIGHT >$. However, the validity of this reformulation has not been looked into.

A major limitation in the sentence ordering approach is that the performance relies heavily on the accuracy of sentence similarity. The model does not check for the case when a sentence is judged to be similar to an incorrect node. The effect of this is clearly reflected from the performance of the proposed approach. Also the model does not consider the case of redundant sentences in the input. However, it is likely that all similar sentences would map to the same node and therefore only one would be chosen. But extra words might affect the similarity measure as the model stands now. The solution is straightforward in the case where the sentences lexically resemble each other to a large extent. However if the sentences are lexically different but semantically similar and carry noise, the similarity measure could fail. We see the entity-aspect model as a possible alternative to measuring sentence similarity.

8.3 Future Work

8.3.1 Multiple Kernel Learning

A typical machine learning approach in a natural language processing (NLP) task usually defines a single kernel function over a rich representation of the input. Question classification is one of the areas where SVM and kernel methods have performed really well. From using a parse tree representation of the input sentence (Zhang and Lee, 2003), adding dependency relationships between words (Moschitti, 2006a) and using other syntactic and shallow semantic features (Moschitti et al., 2007), question classification is a well studied problem with a single kernel and highly polished features. However Multiple Kernel Learning (MKL) has not been extensively explored. In MKL the focus is to learn a kernel function which is a linear combination of some base kernels. A linear combination is only one way to combine multiple kernels and may not necessarily provide a richer representation. Taking the product of kernels, which is equivalent to tensor product of feature spaces, is another way. The application of MKL can be useful when there is more than one source of data. In this scenario there might be different kernels that give good performance on different sources. This is not usually seen in NLP tasks and MKL could be one of the ways to incorporate multiple sources of data. Generally only a single source of data is taken and a kernel that performs well on that dataset is defined. Chen et al. (2011) looked at MKL for question classification with state-of-the-art performance. The kernels they explore are the semantic tree kernel, a parse tree kernel with semantic features such as named entity information and WordNet classes incorporated into it, and a dependency tree kernel. To my knowledge this is the only exploration of MKL for question classification.

Similar to the task of question classification, the answer reranking task can use MKL to combine kernels over representations such as bag-of-words, dependency trees and shallow-semantic trees. If we look at the model proposed by Han et al. (2006) for definition question answering, they utilise different sources of data for estimating probabilities in the definition model and the topic model. The estimates in their model are maximum likelihood estimates on word unigram statistics.

However, if we are to use structural features then MKL could be useful because of the different sources of data.

8.3.2 Aspect Identification

The validity for the neighbourhood constraint in sentence ordering is only possible due to the style of Wikipedia articles. If the authors did not follow the guidelines then we could have seen a larger variation in the ordering of the sentences. In such cases it would be interesting to see what would happen when this constraint was removed and a node could see all other nodes. One of the obvious hindrances was due to the time for a large number of comparisons and expensive computation such as semantic similarity of sentences that would make it impractical. Furthermore, we have only looked at semantic similarity of sentences as the only feature representation for a sentence. The intuition behind it was that it would better capture the word distributional hypothesis compared to direct matching of tokens. One disadvantage is that it ignores the information about word position in the sentence. The use of structural features is a way to overcome this shortcoming. An ensemble of sentence similarity functions could then be utilised to improve the accuracy significantly.

The reason for using the introductory paragraphs of Wikipedia articles is from the observation that each sentence generally presents one piece of information about the entity (Li et al., 2010). The performance of the simple approach presented here largely depends on the accuracy of identifying similar fact expressing sentences. However results show that the similarity metric we have used is not good enough for this task. One of the potential solutions could come from the work of Li et al. (2010). In their entity-aspect model they make a similar observation to us about the nature of Wikipedia articles. Since a sentence only talks about a single piece of information, the key idea is to assign an aspect label to a sentence. They make an assumption that all words do not contribute the same amount to identify the aspect. Consider the following two sentences taken from Li et al. (2010).

- Venturi/D is/S a/S professor/A of/S physics/B at/S the/S University/A of/S Modena/D ./S

- He/S was/S a/S professor/A of/S physics/B at/S the/S University/A of/S Chicago/D until/S 1982/D ./S

Words labelled with 'S' are stopwords and they add no value to interpreting the aspect of a sentence. It is an universally accepted practise to remove stopwords in information retrieval tasks. Words labelled 'B' such as *physics* are background words. These words are commonly found in all aspects of the entity. Document words labelled by 'D' are seen as slots and vary from summary to summary. The words labelled 'A' are the aspect defining words. This should overcome one limitation in our work where we have assumed all the non-stopwords to be useful.

8.3.3 Other Question Types

This thesis only looks at definition questions. The probabilistic model allows the exploration of questions beyond definition questions as long as the distributional hypothesis holds and a way to compute similarity can be defined. One of the potential questions could be "Why" questions. Verberne et al. (2007) holds a view that the answer type to "why" questions (defined as "reason") should be broken down to its subtypes. The subtypes of *reason* are: cause, motivation, circumstance and purpose. The assumption is that there are lexical and syntactic cues that differentiate one answer type from the other. Their classification experiment clearly suggests that the distributional hypothesis should hold in this setting. Another interesting aspect of their work is using Rhetorical Structure Theory (RST) (Mann and Thompson, 1988; Carlson et al., 2003) for *why* questions. The hypothesis is that the question topic and answer text occur in different span of the text and a RST relation holds between these spans. But automatic discourse parsing is a difficult task, at least to produce the same quality as manual annotation. A potential research direction could be to design kernel functions that operate on RST trees and to explore if the discourse structure can be used to locate answers to "why" questions. At the moment most of the questions are open for exploration.

We can even assume the topic and answer-type pair to represent a definition class in a factoid question setting. For example, in the case

of "How tall is Mt. Everest?", we can consider it to belong to the class type $< MOUNTAIN, HEIGHT >$. Similarly, "Where was Buddha born?" is a question of the type $< PERSON, LOCATION >$. By reformulating factoid question in such a way allows the use of our approach. However the validity of this reformulation and what features would be required to capture the characteristics of the answer sentences is a big research question in its own right.

Appendices

Appendix A

List of Stopwords

Table A.1: *List of Stopwords*

punctpunct	-LRB-	-RRB-	's	're
$	0	000	1	2
3	4	5	6	7
8	9	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	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
eight	either	else	elsewhere	enough
entirely	especially	et	etc	even
ever	every	everybody	everyone	everything
everywhere	ex	exactly	example	except
f	far	few	fifth	first

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'd	he'll	he's	hello
help	hence	her	here	here's
hereafter	hereby	herein	hereupon	hers
herself	hi	him	himself	his
hither	hopefully	how	how's	howbeit
however	i	i'd	i'll	i'm
i've	ie	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	known	knows	l
last	lately	later	latter	latterly
least	less	lest	let	let's
like	liked	likely	little	look
looking	looks	ltd	m	mainly
make	many	may	maybe	me
mean	meanwhile	merely	might	more
moreover	most	mostly	much	must
mustn't	my	myself	n	name
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	shan't	she	she'd	she'll
she's	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	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	when's
whence	whenever	where	where's	whereafter
whereas	whereby	wherein	whereupon	wherever
whether	which	while	whither	who
who's	whoever	whole	whom	whose
why	why's	will	with	within
without	won't	wonder	would	wouldn't
x	y	yes	yet	you
you'd	you'll	you're	you've	your
yours	yourself	yourselves	z	zero

Appendix B

Results from Chapter 5 Experiments

Question No.	Answer No.	Sentence
511	8	Site Name , PETROBRAS .
661	20	Bouncypillar Paper Craft .
531	19	CISCO SYSTEMS .
608	18	SAP AKTIENGESELLSCHAFT .
646	2	Honda_Motor_Co. , Ltd .

Table B.1: *Top scoring five sentences for company entity type*

Question No.	Answer No.	Sentence
1095	18	Developmental Verbal .
1005	13	Acth Deficiency .
1256	4	is Child Abuse .
1125	14	No specific type of person has GERD .
1138	7	Hemorrhagic fever with renal syndrome .

Table B.2: *Top scoring five sentences for disease entity type*

Question No.	Answer No.	Sentence
1631	19	UK-US Mutual_Defense_Agreement .
1669	19	A BASIC "CONSTITUTIONAL" TREATY .
1501	14	THE HAY-PAUNCEFOTE TREATY .
1631	2	US-UK Nuclear Cooperation .
1679	13	Threshold Test Ban_Treaty .

Table B.3: *Top scoring five sentences for rule entity type*

Question No.	Answer No.	Sentence
511	18	Petroleo Brasileiro SA-Petrobras is an integrated oil and gas company .
545	3	PepsiCo , Incorporated (NYSE : PEP) is a Fortune 500 , American multinational corporation headquartered in Purchase , NY with interests in manufacturing and ...
549	2	American_International_Group , Inc. (AIG) (NYSE : AIG) is an American insurance corporation .
564	6	StatoilHydro - one of the world 's largest net sellers of crude oil .
592	2	Xstrata is a global diversified mining group , listed on the London and Swiss Stock Exchanges , ... Xstrata Plc.
646	6	American_Honda_Motor_Co. is based in Torrance , California .
652	4	Lloyds TSB_Bank_Plc is a retail bank in the United_Kingdom .

Table B.4: *Sentences correctly placed in top five for for company entity type*

Question No.	Answer No.	Sentence
1043	7	CADASIL is a microangiopathy mainly affecting the brain .
1069	1	Chronic Prostatitis_Chronic Pelvic Pain Syndrome (CP_CPPS) is a pelvic pain condition in men , and should be distinguished from other forms of prostatitis ...
1095	19	Developmental verbal dyspraxia (DVD) is a form of dyspraxia .
1098	1	Excessive daytime sleepiness (EDS) is characterized by persistent sleepiness , and often a general lack of energy , even after apparently adequate night time ...
1105	13	End-stage renal disease (ESRD) is the most feared consequence of kidney disease .
1107	15	May_24_,_2008 ... Fetal alcohol syndrome is among the most common known causes of mental retardation and as such , it is a major public health problem .
1125	14	No specific type of person has GERD .
1131	1	Guinea worm disease is a parasitic worm infection that occurs mainly in Africa .
1236	1	Apr_1_,_2007 ... Premenstrual syndrome (PMS) is a group of symptoms linked to the menstrual cycle .

Table B.5: *Sentences correctly placed in top five for for disease entity type*

Question No.	Answer No.	Sentence
1530	4	The_Lansing-Ishii_agreement is the crown of the high achievements of the Imperial Mission .
1533	9	Treaty of Batum signed between Ottoman Turkey and Armenia .
1536	4	The_Treaty of Saint-Germain was an integral part of World_War_II .
1553	3	The Peace of Riga , also known as the Treaty of Riga ; (Russian : (Rízhsky Mírny dogovór) , Latvian : R?gas miera l?gums and Polish :
1566	3	The_Treaty of Locarno was signed in October_1925 .
1575	2	The SovietâPolish Non-Aggression Pact was an international treaty .
1605	2	The_Paris_Peace_Conference (July_29 to October_15_,_1946) resulted in the Paris_Peace Treaties signed on February_10_,_1947 .
1642	3	The_Single_Convention on Narcotic Drugs is an international treaty to prohibit production and supply of specific (nominally narcotic) drugs and of drugs ...
1672	4	The_Convention_of_London of 1840 was a treaty with the formal title of Convention for the Pacification of the Levant , signed on 15_July_1840 between ...

Table B.6: *Sentences correctly placed in top five for rule entity type*

Appendix C

Results from Chapter 6 Experiments

Question No.	Answer No.	Sentence
592	12	Jason Wilkins , Head of Information Technology , Xstrata ... ture , Xstrata decided to streamline and optimize its IT by migrating to
592	16	Facts and figures about Xstrata , taken from Freebase , the world 's database .
585	15	Michigan , Ohio , AXA_Advisors , Great Lakes , I-75 , Troy , Grand Rapids , Brighton , Detroit , Metro-Detroit , Dearborn , Livonia , Mt._Pleasant , Auburn Hills , ...
646	2	Honda_Motor_Co. , Ltd .
574	17	For more than 30_years , Genentech has been at the forefront of the biotechnology industry , using human genetic information to discover , develop , ...

Table C.1: *Top ranked ten sentences for person entity type. Part I of II*

Question No.	Answer No.	Sentence
632	5	Fortis (Euronext : FORA , Euronext : FORB , LuxSE : FOR) is a company that was active in banking , insurance , and investment management .
541	6	Intel.com visitors , hardware and software developers , look to Intel 's Developer_Center for the resources and information they need .
576	11	ING DIRECT USA - offers one of the highest savings rates , competitive checking , CD , and mortgage rates , home equity loans and more .
646	15	Get the latest on HONDA_MOTOR_CO. , LTD .
574	3	Genentech_Inc. , a portmanteau of Genetic Engineering Technology , Inc. , is a biotechnology corporation , which was founded in 1976 by venture capitalist ...

Table C.2: *Top ranked ten sentences for person entity type. Part II of II*

Question No.	Answer No.	Sentence
1131	7	Guinea worm disease is a debilitating and painful infection caused by a large nematode ... At the beginning of the 20th century , guinea-worm disease , ...
1256	9	is a form of Abusive Head Trauma that occurs when a frustrated caregiver violently shakes , slams , hits , or punches a child 's head , ...
1008	11	Information on diagnosis , treatment , and research , and includes links to booklets and magazine articles .
1236	6	Premenstrual syndrome (known as PMS) involves a variety of physical , mental , and behavioral symptoms tied to a woman 's menstrual cycle .
1015	2	AIDS is caused by a virus called HIV , the Human Immunodeficiency Virus .

Table C.3: *Top ranked ten sentences for disease entity type. Part I of II*

Question No.	Answer No.	Sentence
1043	7	CADASIL is a microangiopathy mainly affecting the brain .
1105	2	Jan 25 , 2010 ... End-stage kidney disease is the complete , or almost complete failure of the kidneys to function .
1236	3	Premenstrual syndrome (PMS) (also called PMT or premenstrual tension) is a collection of physical , psychological , and emotional symptoms related to a
1124	5	Defines , explains its causes and diagnosis , and outlines treatments .
1170	5	Lymphocytic Choriomeningitis information including symptoms , diagnosis , misdiagnosis , treatment , causes , patient stories , videos , forums , prevention , ...

Table C.4: *Top ranked ten sentences for disease entity type. Part II of II*

Question No.	Answer No.	Sentence
1647	5	Text of the 1963 treaty signed by the US , the UK , and the USSR banning atmospheric , oceanic , and extraterrestrial testing of nuclear weapons .
1689	6	CERTAIN that the continuation of this process requires the utilization of the positive experience obtained from the application of the Montevideo_Treaty ,
1501	1	In 1901 the United_States and the United_Kingdom signed the Hay-Pauncefote_Treaty .
1501	17	In 1901 the United_States and the United_Kingdom signed the Hay-Pauncefote_Treaty .
1501	16	Very little business beyond the consideration or the HayPauncefote_treaty will be transacted in the Senate before the adjournment for the holidays .

Table C.5: *Top ranked ten sentences for rule entity type. Part I of II*

Question No.	Answer No.	Sentence
1597	13	September_17_,_2005 marked the 60th anniversary of the Wanfried_agreement , also referred to as the Whisky-Vodka-Line_agreement , signed between the United ...
1683	2	WHEREAS the Treaty of Amity and Cooperation in Southeast_Asia , which was signed on 24_February_1976 in Bali , Indonesia , was amended by the First and Second ...
1554	6	On July_6_,_1914 , he signed the Thomson-Urrutia_Treaty between the United_States and Colombia .
1597	4	September_17_,_2005 marked the 60th anniversary of the Wanfried_agreement , also referred to as the Whisky-Vodka -Line agreement , signed between the United ...
1618	15	ANZUS ANZUS joined the nations of Australia , New_Zealand and the United_States in a defence security pact for the Pacific region .

Table C.6: *Top ranked ten sentences for rule entity type. Part II of II*

Entity Type	Question No.	Answer No.	Sentence
person	9	16	Fred Rogers was a producer , writer , puppeteer , composer , lyricist , ordained minister and devoted student of child development .
person	118	7	In 2006 , actor Tom Cruise was named Forbes_magazine 's most powerful celebrity , with three Golden_Globe_Awards , three Academy_Award nominations , ...
person	177	14	Jared Drake Bell was born on June_27_,_1986 to Robin Dodson , a professional billiards player , and Joe Bell in Orange_County , California .
person	188	19	Corbin Bleu Reivers , better known as Corbin Bleu , is an American actor , model , rapper , and singer .
company	501	19	Oct_25_,_2004 ... PetroChina is a spin-off of the China_National_Petroleum_Company , which is involved in a more than $_1_billion joint venture with the
company	517	7	Total Petrochemicals USA , Inc. , part of the chemical branch of Total S.A. , is a worldwide producer of polypropylene , polyethylene , styrenics (including ...
company	549	16	American_International_Group (AIG) is a leading provider of property and ... After the terrorism attacks of September_11 , AIG asked Congress to pass ...

Table C.7: *Sentences placed correctly in top-1 in the rows of the result across all entity types. Table I of II.*

Entity Type	Question No.	Answer No.	Sentence
disease	1098	12	The complaint of excessive daytime sleepiness , commonly encountered in neurological practice , may arise from a variety of disorders .
disease	1131	7	Guinea worm disease is a debilitating and painful infection caused by a large nematode ... At the beginning of the 20th century , guinea-worm disease , ...
disease	1157	2	Infantile neuroaxonal dystrophy is a disorder that primarily affects the nervous system .
disease	1236	6	Premenstrual syndrome (known as PMS) involves a variety of physical , mental , and behavioral symptoms tied to a woman 's menstrual cycle .
rule	1530	4	The_Lansing-Ishii_agreement is the crown of the high achievements of the Imperial Mission .
rule	1554	6	On July_6_,_1914 , he signed the Thomson-Urrutia_Treaty between the United_States and Colombia .
rule	1597	13	September_17_,_2005 marked the 60th anniversary of the Wanfried_agreement , also referred to as the Whisky-Vodka-Line_agreement , signed between the United ...

Table C.8: *Sentences placed correctly in top-1 in the rows of the result across all entity types. Table II of II.*

Question No.	Answer No.	Sentence
3	8	Jan 4_,_2010 ... Mother_Teresa , whose original name was Agnes Gonxha Bojaxhiu , was born on August_26_,_1910 in what is now Skopje , Macedonia .
9	16	Fred Rogers was a producer , writer , puppeteer , composer , lyricist , ordained minister and devoted student of child development .
27	16	Shakespeare - Shakespeare is the world 's most influential playwright and poet .
31	5	Terry Bradshaw led the Steelers to 4 SuperBowl victories , in just 6_years .
43	19	Britannica online encyclopedia article on Carol Burnett (American comedian and actress) , April_26 , 1933San Antonio , Texas , USAmerican comedian and actress ...
46	10	Neil Leslie Diamond was born January 24th , 1941 to Rose and Akeeba Diamond in Brooklyn , New_York .
52	14	Writer Marilyn vos Savant (born 1946) has an I.Q. of 228 , the highest ever recorded .
60	11	One of the highest-paid and best-known leading men in American movies , Harrison Ford has put his mark on two great franchises : the first Star_Wars trilogy ...

Table C.9: *Sentences placed correctly in top-5 rows of the result for person entity type. Table I of II.*

Question No.	Answer No.	Sentence
65	15	Arthur Ashe was a tennis star of the 1960s and '70s and an African-American pioneer : the first black man to win at the US Open and Wimbledon .
78	3	Oprah Gail Winfrey (born January_29_,_1954) is an American television host , producer , and philanthropist , best known for her self-titled , multi-award winning ...
92	3	Shirley MacLaine was born Shirley MacLean Beaty on April_24_,_1934 , to ... Visit IMDb for Photos , Filmography , Discussions , Bio , News , Awards , Agent , ...
118	7	In 2006 , actor Tom Cruise was named Forbes_magazine 's most powerful celebrity , with three Golden_Globe_Awards , three Academy_Award nominations , ...
154	10	Samuel Langhorne Clemens (November_30_,_1835 â April_21_,_1910) , better known by his pen name Mark_Twain , was an American humorist , novelist , writer , ...
177	14	Jared Drake Bell was born on June_27_,_1986 to Robin Dodson , a professional billiards player , and Joe Bell in Orange_County , California .
188	19	Corbin Bleu Reivers , better known as Corbin Bleu , is an American actor , model , rapper , and singer .

Table C.10: *Sentences placed correctly in top-5 rows of the result for person entity type. Table II of II.*

Question No.	Answer No.	Sentence
501	19	Oct_25_,_2004 ... PetroChina is a spin-off of the China_National_Petroleum_Company , which is involved in a more than $_1_billion joint venture with the
511	18	Petroleo Brasileiro SA-Petrobras is an integrated oil and gas company .
517	7	Total Petrochemicals USA , Inc. , part of the chemical branch of Total S.A. , is a worldwide producer of polypropylene , polyethylene , styrenics (including ...
531	3	Cisco_Systems , Inc. (NASDAQ : CSCO , SEHK : 4333) is an American multinational corporation that designs and sells consumer electronics , networking and ...
541	1	Intel , the world leader in silicon innovation , develops processor technologies and supports global initiatives to continually advance how people work and ...
549	16	American_International_Group (AIG) is a leading provider of property and ... After the terrorism attacks of September_11 , AIG asked Congress to pass ...
555	2	Schlumberger is the leading oilfield services provider , trusted to deliver superior results and improved E&P performance for oil and gas companies around ...

Table C.11: *Sentences placed correctly in top-5 for company entity type. Table I of II.*

Question No.	Answer No.	Sentence
615	10	Occidental_Petroleum , which is ranked on the 2009 Fortune 1000 , a list of America 's largest companies .
632	5	Fortis (Euronext : FORA , Euronext : FORB , LuxSE : FOR) is a company that was active in banking , insurance , and investment management .
646	10	Honda_Motor_Co. , Ltd. was founded in 1946 and is based in Tokyo , Japan .
652	4	Lloyds TSB_Bank_Plc is a retail bank in the United_Kingdom .
654	1	American_Express offers access to world-class Credit Cards , Charge Cards , rewards , travel , financial and business services including Corporate Cards and ...
661	1	Caterpillar is the world 's leading manufacturer of construction and mining equipment , diesel and natural gas engines , industrial gas turbines and a wide and ...
815	19	DirecTV_Group is the largest satellite television provider in the US , with 18 million customers - DirecTV has held its own in the pay television market , ...

Table C.12: *Sentences placed correctly in top-5 for company entity type. Table II of II.*

Question No.	Answer No.	Sentence
1005	6	Jan 8_,_2005 ... Secondary hypoadrenalism , or ACTH deficiency hypoadrenalism , is caused by diseases of the pituitary gland , which lead to adrenal failure as ...
1008	19	Attention Deficit Hyperactivity Disorder , ADHD , is one of the most common mental disorders that develop in children .
1015	2	AIDS is caused by a virus called HIV , the Human Immunodeficiency Virus .
1043	7	CADASIL is a microangiopathy mainly affecting the brain .
1053	15	Chronic fatigue immune dysfunction syndrome (CFIDS) is a chronic autoimmune inflammatory disease .
1069	4	Chronic prostatitis_chronic (CP_CPPS) is a debilitating condition diagnosed in the presence of chronic pelvic pain and lower urinary
1072	6	Chronic prostatitis_chronic (CP_CPPS) is a debilitating condition diagnosed in the presence of chronic pelvic pain and lower urinary

Table C.13: *Sentences placed correctly in top-5 rows of the result for disease entity type. Table I of II.*

Question No.	Answer No.	Sentence
1157	2	Infantile neuroaxonal dystrophy is a disorder that primarily affects the nervous system .
1170	1	May_26_,_2005 ... Lymphocytic choriomeningitis , or LCM , is a rodent-borne viral infectious disease that presents as aseptic meningitis (inflammation of the ...
1186	3	Mucolipidosis (ML) is a group of inherited metabolic disorders that affect the body 's ability to carry out the normal turnover of various materials within ...
1212	10	Neuromyelitis optica (NMO) is an idiopathic , severe , inflammatory demyelinating disease of the central nervous system , that causes severe optic neuritis and
1236	6	Premenstrual syndrome (known as PMS) involves a variety of physical , mental , and behavioral symptoms tied to a woman 's menstrual cycle .
1254	6	Mar_2_,_2009 ... Severe acute respiratory syndrome (SARS) is a serious form of pneumonia , caused by a virus isolated in 2003 .
1256	7	Shaken baby syndrome is the term that is used to describe a form of child abuse caused by vigorously shaking an infant , often in anger , to get a child to ...

Table C.14: *Sentences placed correctly in top-5 rows of the result for disease entity type. Table II of II.*

Question No.	Answer No.	Sentence
1501	1	In 1901 the United_States and the United_Kingdom signed the Hay-Pauncefote_Treaty .
1517	3	Treaty_10 , signed in 1906â07 , covered the northern portions of the province which were not included by Treaties 6 or 8. The area of Treaty_10 also ...
1530	4	The_Lansing-Ishii_agreement is the crown of the high achievements of the Imperial Mission .
1553	1	The Peace of Riga , also known as the Treaty of Riga ; was signed in Riga on 18_March_1921 , between Poland , Soviet Russia and Soviet Ukraine .
1554	6	On July_6_,_1914 , he signed the Thomson-Urrutia_Treaty between the United_States and Colombia .
1566	20	Feb 13 , 2010 ... The Locarno treaties of 1924 to 1930 to a large extent led to the revision of the Treaty_of_Versailles ; however they were not the sole ...

Table C.15: *Sentences placed correctly in top-5 rows of the result for rule entity type. Table I of II.*

Question No.	Answer No.	Sentence
1575	7	The German-Polish Non-Aggression Pact was an international treaty between Nazi Germany and the Second Polish ... The Nazi-Soviet Non-Aggression Pact ...
1597	13	September_17_,_2005 marked the 60th anniversary of the Wanfried_agreement , also referred to as the Whisky-Vodka-Line_agreement , signed between the United ...
1604	5	General_Agreement_on_Tariffs_and_Trade_Set of multilateral trade agreements aimed at the abolition of quotas and the reduction of tariff duties among .
1631	1	The 1958 USâUK Mutual_Defence_Agreement is a bilateral treaty between the United_States and the United_Kingdom on nuclear weapons cooperation .
1647	6	Limited success was achieved with the signing of the Partial_Test_Ban_Treaty in 1963 , which banned nuclear tests ...
1683	15	Mar_6_,_2009 ... Treaty of Amity and Cooperation in Southeast_Asia , which was signed on 24_February_1976 in Bali , Indonesia , was amended by the

Table C.16: *Sentences placed correctly in top-5 rows of the result for rule entity type. Table II of II.*

References

Abney, S. P., Collins, M., and Singhal, A. (2000). Answer extraction. In *ANLP*, pages 296–301.

Aizerman, M. A., Braverman, E. A., and Rozonoer, L. (1964). Theoretical foundations of the potential function method in pattern recognition learning. In *Automation and Remote Control,*, number 25, pages 821–837.

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

Barzilay, R., Elhadad, N., and Mckeown, K. (2002). Inferring strategies for sentence ordering in multidocument news summarization. *Journal of Artificial Intelligence Research*, 17:2002.

Barzilay, R. and Lee, L. (2004). Catching the drift: Probabilistic content models, with applications to generation and summarization. In *HLT-NAACL*, pages 113–120.

Biadsy, F., Hirschberg, J., and Filatova, E. (2008). An unsupervised approach to biography production using wikipedia. In McKeown, K., Moore, J. D., Teufel, S., Allan, J., and Furui, S., editors, *ACL*, pages 807–815. The Association for Computer Linguistics.

Bilotti, M., Katz, B., and Lin, J. (2004). What works better for question answering: Stemming or morphological query expansion? In *SIGIR 2004 Workshop IR4QA: Information Retrieval for Question Answering*.

Blok, S., Medin, D., and Osherson, D. (2003). Probability from similarity. In *American Association for Artificial Intelligence Spring 2003 Symposium, Palo Alto, CA*.

Bouma, G., Mur, J., and van Noord, G. (2005). Reasoning over dependency relations for qa. In *Proc. IJCAI-05 Workshop on Knowledge and Reasoning for Answering Questions*, pages 15–20.

Brill, E., Lin, J., Banko, M., Dumais, S., and Ng, A. (2001). Data-intensive question answering. In *TREC*.

Brown, P. F., Pietra, S. D., Pietra, V. J. D., and Mercer, R. L. (1993). The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*, 19(2):263–311.

Burger, J., Cardie, C., Chaudhri, V., Gaizauskas, R., Harabagiu, S., Israel, D., Jacquemin, C., Lin, C.-Y., Maiorano, S., Miller, G., Moldovan, D., Ogden, B., Prager, J., Riloff, E., Singhal, A., Shrihari, R., Strzalkowski, T., Voorhees, E., and Weischedel, R. (2001). Issues, tasks and program structures to roadmap research in question & answering (Q&A). Technical report, NIST.

Carlson, L., Marcu, D., and Okurowski, M. E. (2003). Building a Discourse-Tagged Corpus in the Framework of Rhetorical Structure Theory. In van Kuppevelt, J. and Smith, R., editors, *Current Directions in Discourse and Dialogue*, pages 85–112. Kluwer Academic Publishers.

Chen, G., Tang, Y., Pan, Y., and Deng, Q. (2011). Question classification using multiple kernel learning and semantic information. *JCP*, 6(11):2325–2334.

Chomsky, N. (1985). Syntactic structures. Mouton.

Clarke, C. L. A., Cormack, G. V., Kisman, D. I. E., and Lynam, T. R. (2000). Question answering by passage selection (multitext experiments for trec-9). In *TREC*.

Collins, M. and Duffy, N. (2002). New ranking algorithms for parsing and tagging: Kernels over discrete structures, and the voted perceptron. In *Proceedings ACL'02*.

Cortes, C. and Vapnik, V. (1995). Support-vector networks. In *Machine Learning*, pages 273–297.

Cui, H., Li, K., Sun, R., Chua, T.-S., and Kan, M.-Y. (2004). National university of singapore at the trec 13 question answering main task. In Voorhees, E. M. and Buckland, L. P., editors, *TREC*.

Cui, H., Sun, R., Li, K., Kan, M.-Y., and Chua, T.-S. (2005). Question answering passage retrieval using dependency relations. In Baeza-Yates, R. A., Ziviani, N., Marchionini, G., Moffat, A., and Tait, J., editors, *SIGIR*, pages 400–407. ACM.

Dang, H. T., Lin, J., and Kelly, D. (2006). Overview of the TREC 2006 question answering track. In *Proceedings of the Fifteenth Text REtrieval Conference (TREC 2006)*.

de Marneffe, M.-C. and Manning, C. D. (2008). Stanford typed dependencies manual. Available at `http://nlp.stanford.edu/pubs/dependencies-coling08.pdf`.

Deerwester, S. C., Dumais, S. T., Landauer, T. K., Furnas, G. W., and Harshman, R. A. (1990). Indexing by latent semantic analysis. *Journal of the American Society of Information Science*, 41:391–407.

Dijkstra, E. W. (1959). A note on two problems in connexion with graphs. *Numerische Mathematik*, 1:269–271.

Fellbaum, C., editor (1998). *WordNet: An Electronic Lexical Database*. MIT Press.

Figueroa, A. (2010). *Finding Answers to Definition Questions on the Web*. PhD thesis, Universitaet des Saarlandes.

Firth, J. R. (1957). A synopsis of linguistic theory 1930-55. 1952-59:1–32.

Green, B. F., Wolf, A. K., Chomsky, C., and Laughery, K. (1961). BASEBALL: an automatic question answerer. In Grosz, K. S.-J. B. J. and Webber, B. L., editors, *Readings in Natural Language Processing*, pages 545–549. Morgan Kaufmann.

Greenwood, M. (2004). Using pertainyms to improve passage retrieval for questions requesting information about a location. In *Proceedings of the Workshop on Information Retrieval for Question Answering (SIGIR 2004)*.

Han, K.-S., Song, Y.-I., and Rim, H.-C. (2006). Probabilistic model for definitional question answering. In Efthimiadis, E. N., Dumais, S. T., Hawking, D., and JÃ¤rvelin, K., editors, *SIGIR*, pages 212–219. ACM.

Harris, Z. (1954). Distributional structure. *Word*, 10(23):146–162.

Hecht-Nielsen, R. (1994). Context vectors: general purpose approximate meaning representations self-organized from raw data. *Computational Intelligence: Imitating Life, IEEE Press*, pages 43–56.

Heie, M. H., Whittaker, E. W. D., and Furui, S. (2010). Optimizing question answering accuracy by maximizing log-likelihood. In *ACL (Short Papers)*, pages 236–240.

Hildebrandt, W., Katz, B., and Lin, J. (2004). Answering definition questions using multiple knowledge sources. In *Proc. NAACL 2004*.

Hovy, E., Gerber, L., Hermjakob, U., Junk, M., and Lin, C. (2000). Question answering in webclopedia. In *Proceedings of the 9th Text Retrieval Conference*.

Jijkoun, V. and de Rijke, M. (2005). Retrieving answers from frequently asked questions pages on the web. In Herzog, O., Schek, H.-J., Fuhr, N., Chowdhury, A., and Teiken, W., editors, *CIKM*, pages 76–83. ACM.

Joachims, T. (1999). Making Large-Scale SVM Learning Practical. In Schölkopf, B., Burges, C. J., and Smola, A., editors, *Advances in Kernel Methods - Support Vector Learning*. MIT Press.

Johnson, W. B. and Lindenstrauss, J. (1984). Extensions of lipshitz mapping into hilbert space. In *Conference in modern analysis and probability*, volume 26, pages 189–206.

Kaisser, M., Scheible, S., and Webber, B. L. (2006). Experiments at the university of edinburgh for the trec 2006 qa track. In Voorhees, E. M. and Buckland, L. P., editors, *TREC*. National Institute of Standards and Technology (NIST).

Karamanis, N. (2003). Entity coherence for descriptive text structuring. Technical report, Edinburgh University.

Karamanis, N. and Mellish, C. (2005). Using a corpus of sentence orderings defined by many experts to evaluate metrics of coherence for text structuring. In *Proceedings of ENLG*, pages 174–179.

Ko, J., Si, L., Nyberg, E., and Mitamura, T. (2010). Probabilistic models for answer-ranking in multilingual question-answering. *ACM Trans. Inf. Syst.*, 28(3).

Landauer, T., Foltz, P., and Laham, D. (1998). *Discourse Processes*, 25(1):259–284.

Landauer, T. K. and Dumais, S. T. (1997). A solution to plato's problem: The latent semanctic analysis theory of the acquisition, induction, and representation of knowledge. *Psychological Review*, 104:211–140.

Lapata, M. (2003). Probabilistic text structuring: Experiments with sentence ordering. In Hinrichs, E. W. and Roth, D., editors, *ACL*, pages 545–552. ACL.

Lapata, M. (2006). Automatic evaluation of information ordering: Kendall's tau. *Computational Linguistics*, 32(4):471–484.

Lee, L. (1999). Measures of distributional similarity. In *Proceedings of the 37th annual meeting of the Association for Computational Linguistics on Computational Linguistics*, pages 25–32. Association for Computational Linguistics.

Levenshtein, V. (1966). Binary codes capable of correcting deletions, insertions, and reversals. *Soviet Physics Doklady*, 10(8):707–710.

Li, P., Jiang, J., and Wang, Y. (2010). Generating templates of entity summaries with an entity-aspect model and pattern mining. In

Hajic, J., Carberry, S., and Clark, S., editors, *ACL*, pages 640–649. The Association for Computer Linguistics.

Li, X. and Roth, D. (2002). Learning question classifiers. In *Proceedings of the 19th international conference on Computational linguistics*, pages 1–7. Association for Computational Linguistics.

Light, M., Mann, G. S., Riloff, E., and Breck, E. (2001). Analyses for elucidating current question answering technology. *Natural Language Engineering*, 7(4):325–342.

Lin, C.-Y. (2002). The effectiveness of dictionary and web-based answer reranking. In *Proceedings of the nineteenth International Conference on Computational Linguistics*.

Lin, D. (1998). Dependency-based evaluation of minipar. In *Proceedings of the 1st International Conference on Language Resources and Evaluation*.

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

Lin, H.-T., Lin, C.-J., and Weng, R. (2003). A note on Platt's probabilistic outputs for support vector machines. (technical report), National Taiwan University.

Lin, J. and Demner-Fushman, D. (2006). Will pyramids built of nuggets topple over?. In Moore, R. C., Bilmes, J. A., Chu-Carroll, J., and Sanderson, M., editors, *HLT-NAACL*. The Association for Computational Linguistics.

Llopis, F. and GonzÃ¡lez, J. L. V. (2001). Ir-n: A passage retrieval system at clef-2001. In Peters, C., Braschler, M., Gonzalo, J., and Kluck, M., editors, *CLEF*, volume 2406 of *Lecture Notes in Computer Science*, pages 244–252. Springer.

Loni, B., van Tulder, G., Wiggers, P., Tax, D. M. J., and Loog, M. (2011). Question classification by weighted combination of lexical, syntactic and semantic features. volume 6836 of *Lecture Notes in Computer Science*, pages 243–250. Springer.

MacKay, D. J. C. and Peto, L. (1994). A hierarchical Dirichlet language model. *Natural Language Engineering*, 1:1–19.

Magnini, B., Negri, M., Prevete, R., and Tanev, H. (2002). Is is the right answer? exploiting web redundancy for answer validation. In *Proceedings ACL 2002*, pages 425–432.

Mann, W. C. and Thompson, S. A. (1988). Rhetorical structure theory: Toward a functional theory of text organization. *Text*, 8(3):243–281.

Moldovan, D., Harabagiu, S., Girju, R., Morarescu, P., Lacatusu, F., Novischi, A., Badulescu, A., and Bolohan, O. (2002). Lcc tools for question answering. In Voorhees and Buckland, editors, *Proceedings of the 11th Text REtrieval Conference (TREC-2002)*.

Moldovan, D., Harabagiu, S., Marius, P., Mihalcea, R., Goodrum, R., Girju, R., and Vasile, R. (1999). Lasso: A tool for surfing the answer net. In *Proceedings of TREC-8*.

Moschitti, A. (2006a). Efficient convolution kernels for dependency and constituent syntactic trees. In Fürnkranz, J., Scheffer, T., and Spiliopoulou, M., editors, *Proceedings of the 17th European Conference on Machine Learning (ECML 2006)*, volume 4212, pages 318–329.

Moschitti, A. (2006b). Making Tree Kernels Practical for Natural Language Learning. In *EACL*. The Association for Computer Linguistics.

Moschitti, A., Quarteroni, S., Basili, R., and Manandhar, S. (2007). Exploiting syntactic and shallow semantic kernels for question answer classification. In *ACL*. The Association for Computer Linguistics.

Ng, V. (2008). Unsupervised models for coreference resolution. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 640–649. Association for Computational Linguistics.

Nivre, J. (2005). Dependency grammar and dependency parsing. Technical report, Växjö University: School of Mathematics and Systems Engineering.

Nivre, J., Hall, J., and Nilsson, J. (2006). Maltparser: a data-driven parser-generator for dependency parsing. In *Proceedings of LREC-2006*.

Padó, S. (2006). *User's guide to* `sigf`*: Significance testing by approximate randomisation*.

Platt, J. (1999). Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. *Advances in Large Margin Classifiers*, 10(3).

Porter, M. (1980). An algorithm for suffix stripping. *Program*, 14(3):130–137.

Prager, J. M., Brown, E. W., Coden, A., and Radev, D. R. (2000). Question-answering by predictive annotation. In *SIGIR*, pages 184–191.

Punyakanok, V., Roth, D., and Yih, V. (2004). Mapping dependencies trees: An application to question answering. In *Proceedings of AI&Math*.

Qiu, X., Li, B., amd Lide Wu, C. S., Huang, X., and Zhou, Y. (2007). Fduqa on trec2007 qa track. In *TREC*. National Institute of Standards and Technology (NIST).

Quarteroni, S., Moschitti, A., Manandhar, S., and Basili, R. (2007). Advanced structural representations for question classification and answer re-ranking. In Amati, G., Carpineto, C., and Romano, G., editors, *Advances in Information Retrieval — Proceedings of the 29th European Conference on Information Retrieval (ECIR 2007), 2-5 April 2007, Rome, Italy*, pages 234–245.

Ravichandran, D. and Hovy, E. (2002). Learning surface text patterns for a question answering system. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, pages 41–47. Association for Computational Linguistics.

Robertson, S. E., Walker, S., Hancock-Beaulieu, M., Gatford, M., and Payne, A. (1995). Okapi at trec-4. In *TREC-4*.

Robertson, S. E., Walker, S., Jones, S., Hancock-Beaulieu, M., and Gatford, M. (1994). Okapi at trec-3. In *TREC-3*.

Rose, D. E. and Levinson, D. (2004). Understanding user goals in web search. In *WWW 04: Proceedings of the 13th international conference on World Wide Web*, pages 13–19.

Schone, P., Ciany, G. M., Cutts, R., McNamee, P., Mayfield, J., and Smith, T. (2005). Qactis-based question answering at trec 2005. In Voorhees, E. M. and Buckland, L. P., editors, *TREC*. National Institute of Standards and Technology (NIST).

Sekine, S. (2008). Oak: English analyzer software.

Sekine, S., Sudo, K., and Nobata, C. (2002). Extended named entity hierarchy. In *Proceedings of the 3rd International Conference on Language Resources and Evaluation*.

Surdeanu, M., Ciaramita, M., and Zaragoza, H. (2008). Learning to rank answers on large online qa collections. In *Proceedings of the 46th Annual Meeting for the Association for Computational Linguistics: Human Language Technologies (ACL-08: HLT*, pages 719–727.

Tai, K.-C. (1979). The tree-to-tree correction problem. *J. ACM*, 26:422–433.

Tellex, S., Katz, B., Lin, J., Fern, A., and Marton, G. (2003). Quantitative evaluation of passage retrieval algorithms for question answering. In *Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, pages 41–47.

Vapnik, V. (1982). *Estimation of Dependences Based on Empirical Data*. Springer-Verlag.

Verberne, S., Boves, L., Oostdijk, N., and Coppen, P. (2007). Evaluating discourse-based answer extraction for why-question

answering. In *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, Amsterdam.

Voorhees, E. (1999). The trec-8 question answering track report. In *Proceedings of TREC-8*.

Voorhees, E. M. (2003). Overview of the trec 2003 question answering track. In *TREC*, pages 54–68. National Institute of Standards and Technology (NIST).

Voorhees, E. M. (2004). Overview of the trec 2004 question answering track. In *TREC*. National Institute of Standards and Technology (NIST).

Voorhees, E. M. and Dang, H. T. (2005). Overview of the trec 2005 question answering track. In Voorhees, E. M. and Buckland, L. P., editors, *TREC*, volume Special Publication 500-266. National Institute of Standards and Technology (NIST).

Whittaker, E. W. D., Furui, S., and Klakow, D. (2005). A statistical classification approach to question answering using web data. In *CW*, pages 421–428.

Widdows, D. and Ferraro, K. (2008). Semantic vectors: a scalable open source package and online technology management application. In *LREC*. European Language Resources Association.

Woods, W. A. (1977). Lunar rocks in natural english: Explorations in natural language question answering. In Zampolli, A., editor, *Linguistic Structures Processing*, volume 5 of *Fundamental Studies in Computer Science*, pages 521–569. North Holland.

Xu, J., Licuanan, A., and Weischedel, R. M. (2003). Trec 2003 qa at bbn: Answering definitional questions. In *TREC*, pages 98–106.

Yeh, A. S. (2000). More accurate tests for the statistical significance of result differences. In *COLING*, pages 947–953. Morgan Kaufmann.

Zhai, C. and Lafferty, J. (2004). A study of smoothing methods for language models applied to information retrieval. *ACM Trans. Inf. Syst.*, 22(2):179–214.

Zhang, D. and Lee, S. S. (2003). Question classification using support vector machines. In *Proc. SIGIR 03*. ACM.

Zhang, K. and Shasha, D. (1989). Simple fast algorithms for the editing distance between trees and related problems. *SIAM J. Comput.*, 18(6):1245–1262.

Zhang, L., Zhang, D., Simoff, S. J., and Debenham, J. K. (2006). Weighted kernel model for text categorization. In Christen, P., Kennedy, P. J., Li, J., Simoff, S. J., and Williams, G. J., editors, *AusDM*, volume 61, pages 111–114. Australian Computer Society.

Zhou, Y., Yuan, X., Cao, J., Huang, X., and Wu, L. (2006). Fduqa on trec 2006 qa track. In Voorhees, E. M. and Buckland, L. P., editors, *TREC*, volume Special Publication 500-272. National Institute of Standards and Technology (NIST).