PARAPHRASE EXTRACTION FROM PARALLEL NEWS CORPORA

by

Bengi Mizrahi

A Thesis Submitted to the
Graduate School of Engineering
in Partial Fulfillment of the Requirements for
the Degree of

Master of Science

in

Electrical & Computer Engineering

Koç University

September, 2006

Koç University

Graduate School of Sciences and Engineering

This is to certify that I have examined this copy of a master's thesis by

Bengi Mizrahi

and have found that it is complete and satisfactory in all respects,
and that any and all revisions required by the final
examining committee have been made.

Committee Members:

Assist. Prof. Deniz Yuret (Advisor)

Assoc. Prof. Aylin C. Küntay

Assist. Prof. Engin Erzin

Date: _____

To my family...

ABSTRACT

Different expressions of the same statement is said to be paraphrases of each other. An example is the phrases 'solved' and 'found a solution to' in 'Alice solved the problem' and 'Alice found a solution to the problem'. Paraphrase Extraction is the method of finding and grouping such paraphrases from free text. Finding equivalent paraphrases and structures can be very beneficial in a number of NLP applications, such as Question Answering, Machine Translation, and Multi-text Summarization, e.g. in Question Answering, alternative questions can be created using alternative paraphrases. We attack the problem by first grouping news articles that describe the same event and then collecting sentence pairs from these articles that are semantically close to each other, and then finally extracting paraphrases out of these sentence pairs to learn paraphrase structures. The precision of finding two equivalent documents turned out to be 0.56 and 0.70 on average, when matching criterion was strict and flexible, respectively. We tried 9 different evaluation techniques for sentence-level matching. Although, exact word match count approach had a better precision value than the n-gram precision count approaches, paraphrase extraction phase shows that the latter approaches catch sentence pairs with higher quality pairs for paraphrase extraction. Our system can extract paraphrases with 0.66 precision when only equivalent document pairs are used as a test set.

ACKNOWLEDGMENTS

I would like to express my gratitude to the members of my thesis committee for critical reading of this thesis and for their valuable comments.

I would like to thank to my supervisor Deniz Yüret for his help throughout my academic research. Thank you for your genuine support! Hope to work with you again in future!

I thank to Deniz Yüret, Burak Görkemli, Tayfun Elmas, Tuğba Özbilgin, Mehmet Ali Yatbaz, and Ergün Biçici for their help in the judgment phases of my studies and thanks to Başak Mutlum, Zülküf Genç, Tayfun Elmas, Ozan Sönmez, Utkan Öğmen, and Gökçe Görbil for their support.

Very special thanks to ARGELA Technologies and my work friends for their support!

Last but not least, I must give immense gratitude to my parents Silvet Sara and Albert, to my cousins Sandy, Sheila, Sandra and İno, to my uncles Robert and Beno, to my aunt-in-law Jane, to my friends Korhan, Yakup, Can, Ceki, Lisya, Esen, Etel, Renin, Yaşar, İzel, Betsy, Yosi, Şila, Sabi, Viki, Jeffy, Michel, Deno, Fiona and Sibel for their extreme support and love.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

INTRODUCTION

Parsing, processing and understanding Natural Languages (e.g. English, German, etc.) have always been challenging in Computational Linguistics. The main reason is that natural languages have excessive amount of irregularities in not only their grammars but also in how words are used in combinations to yield a meaning. One can express a situation in so many different ways, using different grammar structures, using different words or word groups (or collocations, you could say) that creating sound models of such languages in computational perspective is still something not achieved. However, even if we do not have sound models of natural languages, we can at least approximate them by training systems using the vast amount of text available.

In this thesis, parallel with this approach, we try to collect paraphrases from news articles regarding the same events. The definition of a paraphrase is 'a restatement of a thought, passage, or text that significantly alters both the words and the grammatical structure of the original.'

Detecting paraphrases, that is recognizing multiple expressions having the same meaning, is crucial for a natural language system, since understanding natural language requires semantic acknowledgement, and such systems should do computations on semantic level rather than lexical level.

An example natural language application can be Question Answering Systems (QA). Question Answering Systems take a factual question as an input and are supposed to give the exact answer to it. A example question can be 'Which company produces NSeries mobile phones?' and the answer of the question can reside in such a sentence: 'The manufacturer of NSeries mobile phones is Nokia.' For a Question Answering System to answer this question, it should semantically recognize that if a company produces something, it is also the manufacturer of that thing. So, a QA system augmented with a database of paraphrases

can approach the problem in the following manner:

Given an input of question query: (e.g. *Who is the author of the book Ramses?*)

- Parse the question and search for phrases (e.g. *X is the author of Y*)
- Search for paraphrases in the database (e.g. *X authored Y, X is the writer of Y, Y is written by X, the author of Y is X, X wrote Y*, etc.)
- For each paraphrase detected:

  Rewrite the question with the alternative paraphrases

  Search each question independently

- Merge the results

The answer will be detected if the text being searched contains the sentence: *Christian Jacq wrote the book Ramses.*

Paraphrase detection is also important for other applications such as Machine Translation or Text Summarization. For instance, in Text Summarization, we can use alternative paraphrases to change the length of a summary. So a generic paraphrase database is a valuable resource and this study focuses on creating one such database using parallel news corpora.

From the Computational Linguistics perspective, there are three different categories of paraphrases:

1. Sentence-Level Paraphrase

   ‘‘A suicide bomber blew himself up in a southern city killing 13 people and injuring 27.’’

   ‘‘13 people were killed and 27 wounded when a suicide bomber blew himself up in a southern city.’’

2. Phrase-Level Paraphrase

   mammal’s length

   length of mammal

3. Entailment

   ‘‘A force majeure is an act of God,’’ said attorney Phil Wittmann, who represents the New Orleans Saints and owner Tom Benson’s local interests.

   Phil Wittmann works for Tom Benson.

Although sentence-level paraphrases can be very beneficial in Machine Translation and Text Summarization, they are unsuitable for generic use so our focus will be on collecting Phrase-Level Paraphrases. Entailment can be defined as follows: A text A entails another text B, if the meaning of B can be inferred from A. Although, some of the studies on detecting entailments is parallel with paraphrase detection, it is not suitable for our purposes because they cannot be used as templates.

Our approach to the problem is motivated by the following hypothesis: There is a strong chance that different news articles written by different journalists but about the exact same event are a valuable source for harvesting paraphrase pairs.

Our approach borrows ideas from studies of [Ibrahim et al., 2003], [Lin and Pantel, 2001], [Finch et al., 2005]. It can be summarized as follows: Given a large amount of news articles from different sources, we pair articles that are considering the exact same events but have been written by different journalists, and we call this process ‘Document-Level Equivalence Detection’. After this, we search for ‘Sentence-Level Equivalence’ (or ‘Sentence-Level Paraphrase’) between the sentences of the matching documents by examining their lexical features. Having harvested a set of equivalent sentence pairs, we finally parse the sentences and search for similarities in their parse trees.

[Ibrahim et al., 2003], instead of using parallel news corpora as input source, used multiple translations of foreign novels. The advantage of this approach is that the events are occurring in the same order in each text so alignment of these translations becomes an easier task. However, the disadvantage of the approach is that the data set being used is rather small compared to using parallel news corpora.

[Finch et al., 2005] tried some Machine Translation evaluation techniques on Paraphrase Detection. They did training using Support Vector Machines (SVM) using a set of sentence pairs, each of them judged by annotators, stating whether there is a sentence-level paraphrase or not. Although over 0.8 precision values reported, this work does not deal with

extracting paraphrases but detecting them.

Another interesting study was conducted by [Dolan et al., 2004]. They extracted sentence-level paraphrases from parallel news corpora using two simple heuristics: 1) Pair the first two sentences of documents describing the same events. 2) Pair sentences if their Levenstein distance is smaller than k from documents describing the same events. However, the results are not comparable because they used Machine Translation evaluation metrics. They claim that if a Machine Translation system is trained with better paraphrase pairs, it will generate paraphrases with smaller Alignment Error Rate. Our study differs from their approach in that we take into account all sentences in an article.

The most sophisticated studies were conducted by [Lin and Pantel, 2001] and [Barzilay and Lee, 2003]. Both approaches apply pattern comparison by analysis of the patterns' respective arguments. Although, the results of [Barzilay and Lee, 2003] is outstanding compared to [Lin and Pantel, 2001], their paraphrases are not suitable for generating new sentences. [Lin and Pantel, 2001] try to discover inference rules from text using their assumption called "Extended Harris' Distributional Hypothesis".

To summarize, [Lin and Pantel, 2001] and [Ibrahim et al., 2003] are the closest approaches to ours in that they both try to detect Phrase-Level Paraphrases for generic use. However, approaches in [Dolan et al., 2004] and [Barzilay and Lee, 2003] yield Sentence-Level Paraphrases which more suitable for Text Summarization[1].

In the subsequent chapters, we describe our approach and show the results we obtained. After this, we continue with the detailed explanation of the related work summarized above and finish with a conclusion.

[1]For instance, the machine can switch between paraphrases to shorten a summary.

Chapter 2

OUR APPROACH

Our aim is to extract paraphrase pairs from a large corpus obtained from a number of new wires. We approach to the problem in three main steps:

1. Finding equivalent document pairs originating from different news wires, but describing the same world event.

2. Finding equivalent sentences pairs between matching documents that are similar in meaning

3. Finding equivalent paraphrases between matching sentences.

The expression 'finding equivalent document pairs' is somewhat ambiguous and needs to be clarified. For instance, finding a document pair, in which both documents are giving information about the peace negotiations between Palestine and Israel is not sufficient for them to be labeled as equivalent for our purposes. Clearly, there is a chance that there will be equivalent paraphrases in the pair but the system will not confidently be able to extract them and the precision will be very low. What we mean by equivalent document pairs is that we want to find documents that are describing the same world event.

By 'Finding equivalent sentence pairs', what we mean is the following: Suppose D_1 and D_2 are two documents and s_1 and s_2 are two sentences from D_1 and D_2, respectively. Moreover, suppose D_1' and D_2' are documents obtained by swapping the place of s_1 and s_2 in D_1 and D_2. If both D_1 and D_1', and D_2 and D_2' are equivalent document pairs, then we can conclude that s_1 and s_2 are equivalent sentence pairs. Although this is the kind of sentence pairs we want to extract, sometimes the number of sentence pairs of this kind are less than expected even though the documents are perfectly equivalent. So we are not only going to extract equivalent sentence pairs but also sentence pairs that have partial equivalence.

Similar to the definition of equivalent sentence pairs, what we mean by 'Finding equivalent paraphrases' is as follows: Suppose s_1 and s_2 are sentences and p_1 and p_2 phrases in

s_1 and s_2, respectively. Suppose s_1' and s_2' are obtained, when the place of p_1 and p_2 are swapped in s_1 and s_2. If both s_1 and s_1' are equivalent and s_2 and s_2' are equivalent sentence pairs, we conclude that p_1 and p_2 are equivalent paraphrases. For instance, consider the following two sentences: 'Alice loves Adam'; 'Alice adores Adam'. Here, loves and adores are equivalent paraphrases.

In the following sections we describe how we find each kind of semantic equivalence for each level in detail.

2.1 Document-Level Equivalence Detection

In this section we describe how we harvest document pairs that are describing the same world event. What we are doing in theory is as follows: We assume that every world event is processed by each news service group. So, according to our assumption, if there are N news services, then we have at least N articles authored by distinct journalists about the same world event. We have a Gigaword Corpus of four newswires available, which are text files in XML format:

- APW: Associated Press Worldstream English Service
- AFE: Agence France Press English Service
- NYT: The New York Times Newswire Service
- XIE: The Xinhua News Agency English Service

Thus, we want to extract 4 news articles about a specific world event from these corpora.

2.1.1 The Approach

The following steps were applied to extract documents about the same world event:

1. Parse all XML files and extract documents with the following attributes
   (a) Document ID (e.g. `APW19960101.0013`)
   (b) Type (e.g. `story`)

(c) Headline (e.g. `Sri Lankan President Says Military Will Leave Jaffna Soon`)

(d) Date, which is extracted from Documents ID (e.g. `1996-01-01`)

(e) Text (the document itself)

2. Index all the documents, whose type are `story`.

3. For each document in one specific news wire, such as APW, query all the documents in the other news wires, in a date window of [-3 days, +3 days].

4. Rank the matches according to $TF * IDF$ scores.

5. Pick the highest ranked documents from each news service, except the one whose documents are being searched.

In the first step, we parse all the documents. Figure 2.1 shows a sample document from APW news service.

In the second step, we index all the documents by creating an inverted index. We create a hashtable, where the keys are all words encountered and the values are the list of Document IDs, whose text contains the key word. (Figure 2.2)

Figure 2.3 shows the inverted index created from the two documents in Figure 2.2.

We also store a hashtable that holds the attributes of the documents, as illustrated in Figure 2.5.

During the indexing, we ignore the words in the stop list since almost all documents contain these words. (For more information about stop lists, refer to Appendix B.3). Another technique that could have been used is stemming. We chose not to use it in the document-level matching, because as we see in the results subsection, the named entities or the proper nouns are sufficient as keywords to distinguish relevant matches from the irrelevant ones.

In the third step, we create queries out of each document in a specific news service corpus and search it in the other news service corpora. The structure of the queries is simply OR's of the words in the whole document being searched. Suppose the document being searched is as in Figure 2.6.

Then, the query would be as shown in Figure 2.7

```
<DOC id="APW19960101.0013" type="story">
<HEADLINE> Sri Lankan President Says Military Will Leave Jaffna Soon</HEADLINE>
<DATELINE> COLOMBO, Sri Lanka (AP) </DATELINE>
<TEXT>
<P>
The Sri Lankan president Monday indicated that the military
could soon leave Jaffna city, the rebel stronghold in the
north that was captured after 50 days of fierce fighting.
</P>
<P>
‘‘It is our aim to establish a civil administration in the area
as soon as possible and vest the administration in the Tamil
people,’’ Mrs. Chandrika Kumaratunga said in her New Year’s message
to the nation.
</P>
...
<P>
The rebels allege widespread discrimination by majority Sinhalese
who control the government and military. More than 39,000 people
have been killed in the fighting since 1983.
</P>
</TEXT>
</DOC>
```

Figure 2.1: A news article represented in XML format

DOCID: docid1 (with date1)

TEXT: a b c

DOCID: docid2 (withdate1)

TEXT: a d e

Figure 2.2: Raw documents

a: docid1 docid2

b: docid1

c: docid1

d: docid2

e: docid2

Figure 2.3: Inverted index from the words in documents

date1: docid1 docid2

Figure 2.4: Inverted index from the dates of the documents

docid1: date1, headline1, text1

docid2: date2, headline2, text2

Figure 2.5: A hashtable holding the attributes of the documents

DOCID: APW19960101.0013

TEXT: a d r f a b e d

Figure 2.6: Document being searched

(a d r f e) AND [dates ranging from 1995/12/29 to 1996/01/03]

Figure 2.7: Query created from the document in Figure 2.6

In the fourth step, we score each of the documents' similarity using the $TF * IDF$ score (For detailed explanation of $TF * IDF$ score refer to Appendix B.5). Let's consider the similarity of the documents in Figure 2.2 with the search document in Figure 2.6 using the inverted index in Figure 2.3. For example, the Term Frequency (TF) of the word 'a' is 2/8, and the Inverted Document Frequency (IDF) of the word 'a' is log (2/2) = 0. So $TF * IDF$ of 'a' equals 0. In other words, word 'a' has no effect on the final score. When all the $TF * IDF$ scores are added, we obtain the final score for similarity. The score for docid1 is 1/8 and for docid2 it is 2/8.

In the final step, we pick the documents that get the highest score from each newswire corpus.

2.1.2 Evaluation and Results

Although, the inter-annotator agreement in the judgments was 89.23%, evaluating the performance of the system was rather difficult. In most of the cases, it was even hard for humans to conclude whether two documents are exact match or not. Let's consider the following document pair:

```
HEADLINE: Israeli Foreign Minister Arrives in Jordan AP Photos
Planned
PART OF THE TEXT: Israeli Foreign Minister Ehud Barak
arrived here tuesday for talks with Jordanian officials on progress
in mideast peacemaking and bilateral relations.

HEADLINE: Israeli Foreign Minister Arrives in Jordan
PART OF THE TEXT: Israeli Foreign Minister Ehud Barak arrived
here today on his first official visit to Jordan as a Foreign
Minister since last November, Radio Jordan reported.
```

We can definitely conclude that the world event is *the arrival of Israeli Foreign Minister to Jordan* in both of the documents. However, let's compare the following document with the documents above:

```
HEADLINE: Jordanian PM Stresses Jordan's Peace Initiative
```

```
PART OF THE TEXT: Jordanian Prime Minister Sharif Zeid Ben Shaker
stressed here today that Jordan is commited to realizing a just,
comprehensive and lasting peace on all tracks of the Middle East
peace process.
```

In this document, the world event is *the talks between Israeli Foreign Minister and the Jordanian Prime Minister*. We conclude that Israeli Foreign Minister has already arrived to Jordan and talked with the Jordanian Prime Minister and the news article summarizes the contents of the talk. So, are the documents exact matches?

To solve this kind of uncertainty, we came up with 4 categories:

- 0, if they are not relevant at all
- 1, if they are same type of news but still irrelevant
- 2, if they are about the same event or news but different parts
- 3, if they match exactly

So the first example should be judged as 3 and the second example should be judged as 2.

To test the performance of the system with these news corpora, we conducted the following survey:

We picked 30 random documents from APW news group. We searched for the documents in our search engine and picked the best match from each of the other news groups. So, we obtained 90 document pairs:

```
APW1: (NYT1, XIE1, AFE1)
APW2: (AFE2, XIE2, NYT2)
APW3: (AFE3, NYT3, XIE3)
...
APW30: (AFE30, XIE30, NYT30)
```

We divided these document pairs and distributed them to 6 annotators so that each document pair is judged by 4 annotators:

- Annotator 1 and 4 evaluated pairs originated from the first 20 searches.
- Annotator 2 and 5 evaluated pairs originated from the last 20 searches.
- Annotator 3 and 6 evaluated pairs originated from the first 10 and last 10 searches.

The final scores were decided by the majority decision. When the judgments were evenly distributed, we picked the higher one. Then, we mapped the judgments to either to 0 or 1, using the following two functions:

$f1 : \{0, 1, 2, 3\} \rightarrow \{0, 0, 1, 1\}$

$f2 : \{0, 1, 2, 3\} \rightarrow \{0, 0, 0, 1\}$

We calculated the average precision of finding exact matches for one document search and the average confidence weighted score for one document search (For a description of confidence weighted score, refer to Appendix C.5. The recall is difficult to obtain since we do not know how many exact matches there are in the corpora.

Suppose the document being searched is APW1 and the retrieved documents are NYT1, XIE1, and AFE1 and suppose annotator 1 gives the following judgment for the comparison.

```
APW1:     (NYT1, XIE1, AFE1)
Judgment: (3,2,0)
```

Then, the scores, when mapped with $f1$, are:

```
APW1:     (NYT1, XIE1, AFE1)
Judgment: (1,1,0)
```

The precision in this search would be $(1 + 1 + 0)/3 = 0.66$ and the confidence weighted score would be $(1/1 + (1 + 1)/2 + (1 + 1 + 0)/3)/3 = 0.88$.

When the mapping function is $f1$, the average precision of finding exact matches equals 0.52 and the average confidence weighted score is 0.64. When the mapping function is $f2$, the results are 0.38 and 0.51, respectively. The inter-annotator agreement in the judgments was 89.23%.

	between APW and AFE	between APW and NYT	between APW and XIE
exact matches	56.67	20.00	40.00
about the same event or news but different parts	13.33	6.67	20.00
same type of news but still irrelevant	13.33	16.67	6.67
not relevant at all	16.67	56.67	33.33

Table 2.1: The percentage of matches from different newswires corpora

The interpretation of these values is that for each of the searches only half of the retrieved documents are exact matches.

The distribution of the matches for different newswires corpora is listed in Table 2.1.

Moreover, when the mapping function $f1$ is used, the average precision of retrieving exact matches from only AFE, NYT and XIE are 0.56, 0.20 and 0.40, respectively. When the mapping function $f2$ is used, the average precision becomes 0.70, 0.26 and 0.60, respectively.

Although the performance of the system is satisfactory, we could not find exact matches for all document queries because of the limited resources we have. Most of the world events are not processed by all the news corpora we have.

Additionally, the assumption we made, that if there are N news groups, there are N news articles about the same world event, is actually an optimistic one[1].

We conclude that this simple technique is sufficient to find some exact document pairs but not sufficient to eliminate all the non-exact matches.

2.2 Sentence-Level Equivalence Detection

In this section, we describe how we obtain equivalent sentence pairs from equivalent document pairs. For this purpose, we use a number of statistical evaluation algorithms, which

[1]This information is given by only examining the subset of the Gigaword Corpus that we use.

are described in [Finch et al., 2005]

The algorithms are: WER, PER, sNIST2, sNIST3, sNIST4, BLEU1, BLEU2, BLEU3, and BLEU4.

2.2.1 The Approach

[Finch et al., 2005] proposed to use Automatic Machine Translation[2] evaluation techniques in paraphrase detection and they compared the performances of these techniques. They trained their system with human-annotated set of sentence pairs using Support Vector Machines (SVM). According to their results BLEU1 and sNIST1 had the highest precision. We also use these techniques in our study. However, we do not do training on our system, we score each sentence pair with each technique and pick the best k sentence pairs and accept them as equivalent sentence pairs.

The following are the techniques applied. s denotes the MT system output and r denotes reference translation:

Word Error Rate (WER)

This is the measure of the number of modification operations required to transform one sentence to the other in terms of number of insertions, deletions and substitutions. The formula is:

$$WER(s,r) = \frac{I(s,r) + D(s,r) + S(s,r)}{|r|}$$

$I(s,r)$ is the number of insertions, D(s,r) is the number of deletion and $S(s,r)$ is the number of substitution required in terms of words.

This measure is also known as the Levenstein Edit Distance Algorithm.

Position-Independent Word Error Rate (PER)

This is a similar measure with WER, but we ignore the positions of the words in the sentences. This yields the formula:

[2]Automatic machine translation evaluation is the task of scoring the output of a machine translation system with respect to a set of reference translations. This task is very close the evaluating the degree of similarity between two sentences.

$$PER(s,r) = \frac{\text{diff}(s,r) + \text{diff}(r,s)}{|r|}$$

Bilingual Evaluation Understudy Score (BLEU)

In this formula the quality of translation is between 0 and 1 measured by the statistical closeness to a set of reference translations. The formula counts the n-gram co-occurrences between the given system output and the set of reference translations and then taking the weighted geometric mean. I is the system output, $s_i \in I$ is the i$^{\text{th}}$ sentence in the output and r_i is the corresponding reference translation. The formula is:

$$BLEU = BP * exp(\sum_{n=1}^{N} \frac{1}{N} * log(p_n))$$

N is the maximum n-gram[3] size considered.

The n-gram precision is calculated as:

$$p_n = \frac{\sum_{i=1}^{I} \sum_{\text{n-gram} \in s_i} count(\text{n-gram})}{\sum_{i=1}^{I} \sum_{\text{n-gram} \in s_i} count_{sys}(\text{n-gram})}$$

$count$(n-gram) is the count of n-grams found both in s_i and r_i , $count_{sys}$(n-gram) is the count of n-grams found only in s_i.

The brevity penalty is used to penalize the output sentences that are shorter than their reference sentences. Its formula is:

$$BP = exp(min(1, 1 - \frac{L_{ref}}{L_{sys}}))$$

BLEU-N is the score where the length of the maximum n-gram measured is N. We apply BLEU-N to two sentences since we are comparing sentences.

NIST Score

The NIST score is similar to BLEU score in that it also uses n-gram co-occurrence precision. However, it takes the arithmetic mean of the n-gram counts. The formula is:

$$NIST = \sum_{n=1}^{N} BP * \frac{\sum_{\text{all n-grams that co-occur}} \text{info}(\text{n-gram})}{\sum_{\text{n-gram} \in s_i} 1}$$

[3]An n-gram is a sequence of n words, for instance "I love" is a 2-gram of "I love ice-cream."

info(n-gram) is defined to be:

$$\text{info(n-gram)} = \log_2 \frac{\text{count((n-1)gram)}}{\text{count(n-gram)}}$$

count(n-gram) is the count of occurrences of (w_1 w_2 ... w_n) and count((n-1)gram) is the count of occurrences of (w_1 w_2 ... w_{n-1}) in all reference translations.

The brevity penalty BP again penalizes shorter system outputs compared to reference translations. The formula is:

$$BP = \exp(beta * \log_2 \min(\frac{L_{sys}}{\text{avg}(L_{ref})}, 1))$$

L_{sys} and L_{ref} are the length of the whole system output and the reference translation.

We ignore BP because we are comparing two sentences, not two documents. Moreover, we do not use info(n-gram) because it is not applicable for sentence comparison; instead, we add 1 if the n-gram co-occurs and 0 if not. We call this simple version of NIST as sNIST. sNIST-N is the score where the length of the maximum n-gram measured is N.

Exact Match Count

We also add the exact match count to serve as a baseline:

$$EM = \#\text{ of word matches}$$

These steps are applied to obtain equivalent sentence pairs:

1. for each matching document D_1 and D_2:
   (a) Split D_1 and D_2 into sentences, using MontyLingua NLP Toolkit
   (b) For each sentence pair (s_1, s_2), where $s_1 \in D_1$ and $s_2 \in D_2$:
   Apply each of the statistical evaluation algorithms above.
   (c) Sort the sentence pairs with respect to their scores
   (d) Pick the best k sentence pairs

We try each of these evaluation techniques, have the results judged by 6 annotators and compare the performance of the system.

Moreover, we also want to find out the correlation between the source of documents used and the performance of the sentence-level matching. We used 3 sources:

- All document pairs extracted in the document-level matching.
- All document pairs that were judged as 2 or 3 in the document-level matching.

2.2.2 Evaluation and Results

For the survey, we picked $k = 2$. We collected 1260 sentence pairs from these three sources and using all the evaluation techniques. We distributed the sentence pairs to 6 annotators so that every sentence pair is evaluated by 2 annotators and every annotator evaluates the same amount of sentence pairs from each source.

Although common sense is used for judgments, the main judgment criterion was:

Look for similarities in the *(verb phrase, subject, object)* tuples of the two sentences.

A clear example would be:

```
Alice is the author of the book.
```

```
The book is written by Alice.
```

For the first sentence *(verb phrase, subject, object) = (is the author of, Alice, the book)* and for the second sentence *(verb phrase, subject, object) = (is written by, the book, Alice)*. The verb phrases semantically match, and the object of one sentence matches with subject of the other one.

Here are some examples of sentence pairs and judgments from the corpora used. In the following pair, even though the second sentence does not have an object, the common sense says they are semantically overlapping, so it is judged as 1.

```
UEFA denies this.
```

```
However, UEFA Disciplinary Chief Rene Eberle disagreed.
```

The following two sentence pairs are giving the same information; they are judged as 1.

```
In a 1,652 page opinion released on Thursday, Judge Gladys Kessler
ruled the largest US cigarette companies violated anti-racketeering
```

```
laws and ordered them to make corrective statements about the health
effects and addictiveness of smoking.

In a 1,652-page opinion, Kessler ruled that defendants including
Altria Group's Philip Morris USA and Reynolds American had lied for
decades about the risks of smoking and ordered them to make
"corrective statements" about the addictiveness of smoking and its
adverse health effects.

---o---

The Justice Department said in a statement it was disappointed the
court didn't impose all suggested penalties.

The Justice Department said in a brief statement that it was
``pleased with the court's finding of liability  but disappointed
that the court did not impose all of the remedies sought by the
government.''
```

A difficult sample for the annotators would be the following:

```
Papandreou has been on life support for most of the time since Nov.
20, when he was hospitalized with Pneumonia, but he has not shown
any readiness to resign.

Since last November, Papandreou has been incapacitated, his vital
organs faltering, his kidneys damaged beyond repair.
```

Both of the sentences are describing the health situation of Papandreou, but there are no paraphrases between the sentences and the *(verb phrase, subject, object)* similarity is not found. This pair must be judged as 0.

The Table 2.2 shows the precision results for sentence pairs from different sources of document pairs.

	All document pairs	Judgment 2 and 3	Judgment 3
EM	0.40	0.67	0.81
WER	0.13	0.13	0.13
sNIST2	0.28	0.61	0.64
sNIST3	0.30	0.59	0.63
sNIST4	0.35	0.60	0.67
BLEU1	0.34	0.62	0.64
BLEU2	0.38	0.37	0.41
BLEU3	0.28	0.31	0.33
BLEU4	0.23	0.27	0.28

Table 2.2: Precision values of each MT evaluation technique, when given all the document pairs, document pairs with judgment 2 and 3, and document pairs with only judgment 3

METHOD	EM	sNIST-N	BLEU-N	ALL
N=2	8	3	3	4
N=3	8	4	4	2
N=4	10	5	2	0

Table 2.3: Number of samples detected by only one specific evaluation technique

These results show that using exact match counts reveals equivalent sentence pairs with the highest precision. We observe that sNIST precision improves as the length of the ngrams increase but this is not the case for BLEU. The most important difference between sNIST and BLEU is that the former one takes arithmetic average precision of ngrams, whereas the latter takes geometric average precision of ngrams.

Furthermore, we wanted to see how each technique differs in detecting equivalent sentence pairs. The following tables shows the number of samples detected only by a specific evaluation technique and not by others.

We deduce from the table that as the size of the ngrams increase, the number of common equivalent sentence pairs detected decrease. However, even with the longer ngrams, the equivalent sentence pairs detected by BLEU can also be detected by other methods.

Here is an example of two sentence pairs, labeled as equivalent by sNIST and EM:

sNIST:

`Chinese authorities plan to prosecute a 17-year-old worker who fell asleep while reading by candlelight, sparking a dormitory fire which killed 20 of his colleagues, a Hong Kong newspaper said Friday.`

`The report said Liu fell asleep while reading by candlelight.`

EM:

`It recovered two diaries that it says show that between April 1988 and March 1991 the jains paid more than 600 million (dlrs 17 million) to 115 politicians and bureaucrats to promote their business in the power sector.`

`The explosive jain diary includes the names and initials of 115 politicians and bureaucrats who allegedly received payments totalling more than 21 million dollars between 1988 to 1991.`

We see how sensitive sNIST is in ngram precision. Although, this does not necessarily mean these kinds of sentence pairs yield better paraphrases, they actually do as described in detail in Section 2.3

2.3 Paraphrase Extraction from Equivalent Sentence Pairs

In this section, we describe how we approach to the problem of collecting paraphrase pairs out of dependency trees of equivalent sentence pairs. However, before going into details, let's define what a dependency tree is.

A dependency relationship is an asymmetric binary relationship between a word called head, and another word called modifier. The structure of a sentence can be represented by a set of dependency relationships that form a tree which we call a *dependency tree*. Figure 2.8 illustrates a dependency tree. We used MINIPAR to create dependency trees of out of sentences. According to [Lin, 1998], MINIPAR is able to cover about 79% of the dependency relationships in the SUSANNE corpus with about 89% precision.

Having collected equivalent sentence pairs, in this level, we apply the following procedures:

For each sentence pair (s_1, s_2):

- Create dependency trees of the sentences: (T_1, T_2) using MINIPAR.

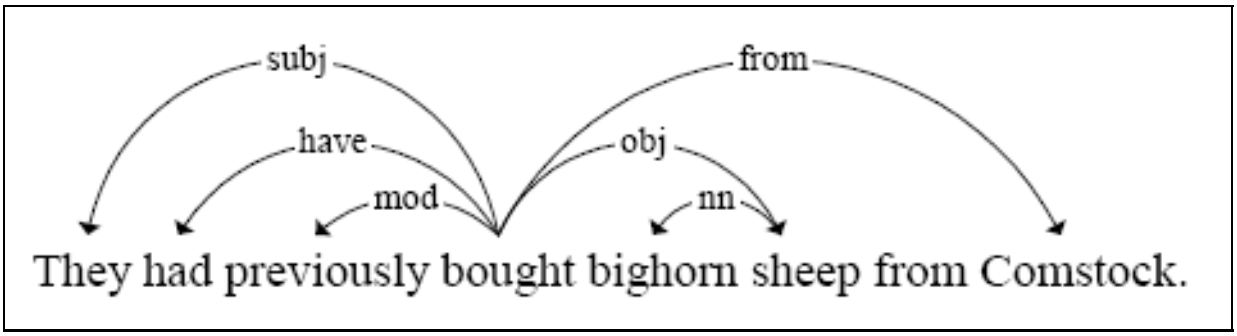

Figure 2.8: An example of a Dependency Tree

- Search for common nouns between T_1 and T_2 and pair them as anchors: $A = \{(n_{1,1}, n_{2,1}), (n_{1,2}, n_{2,2}), ..., (n_{1,k}, n_{2,k})\}$.

- For each anchor pair $a_1 = (n_{1,i}, n_{2,i})$ and $a_2 = (n_{1,j}, n_{2,j}) \in A$:

  - Find and collect the shortest path pairs $p_1 = n_{1,i} \longrightarrow n_{1,j}$ and $p_2 = n_{2,i} \longrightarrow n_{2,j}$.

- return the pair of paths whose frequency of internal relations has the highest value.

After parsing the sentences with MINIPAR, we do post-processing on the dependency trees. If there is a *poss* relation between two nodes, we swap them as illustrated in Figure 2.9 and Figure 2.10:

For instance, such a swap allows us to detect $'s \longrightarrow of$ paraphrases as illustrated in Figure 2.11.

To compare the performance of our system with [Ibrahim et al., 2003] (For more detail about [Ibrahim et al., 2003], refer to Section 3.1, we ran all the paraphrase pairs that are correctly detected in [Ibrahim et al., 2003] with our system. We were able to detect 75% of the paraphrases exactly. 12.5% of the paraphrases were detected as a subset in our paths:
beams of X dazzled her Y (X: sun / Y: eyes)
rays of X dazzled her Y (X: sun / Y: eyes)

The other 12.5% of the paraphrases were detected as a subset in our path with some noise:
averted X head risen to Y mouth (X: poison / Y: her)
turned away X head as poison rose to Y mouth (X: poison / Y: her)

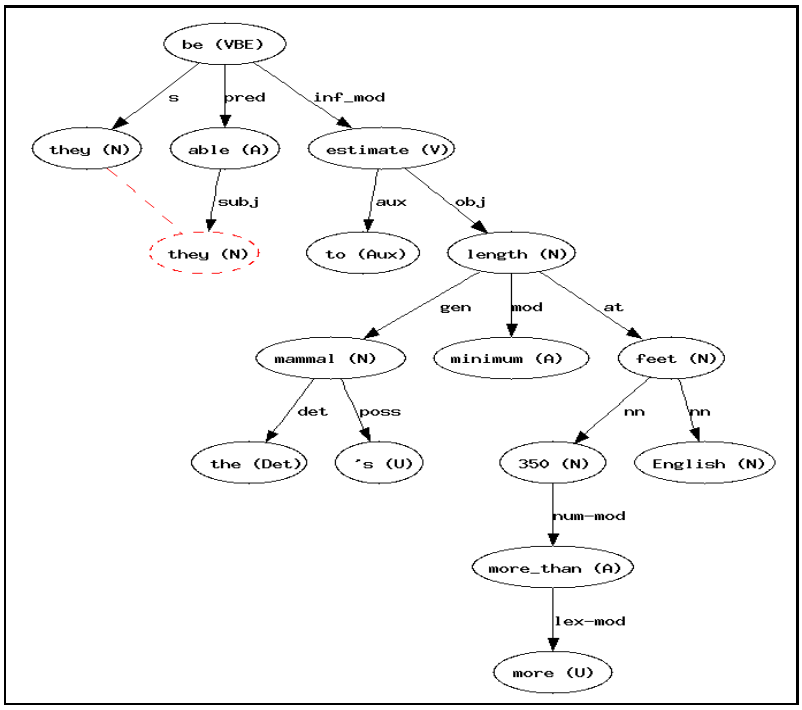

Figure 2.9: Raw output of MINIPAR for the sentence "They were able to estimate the mammal's minimum length at more than 350 English feet."

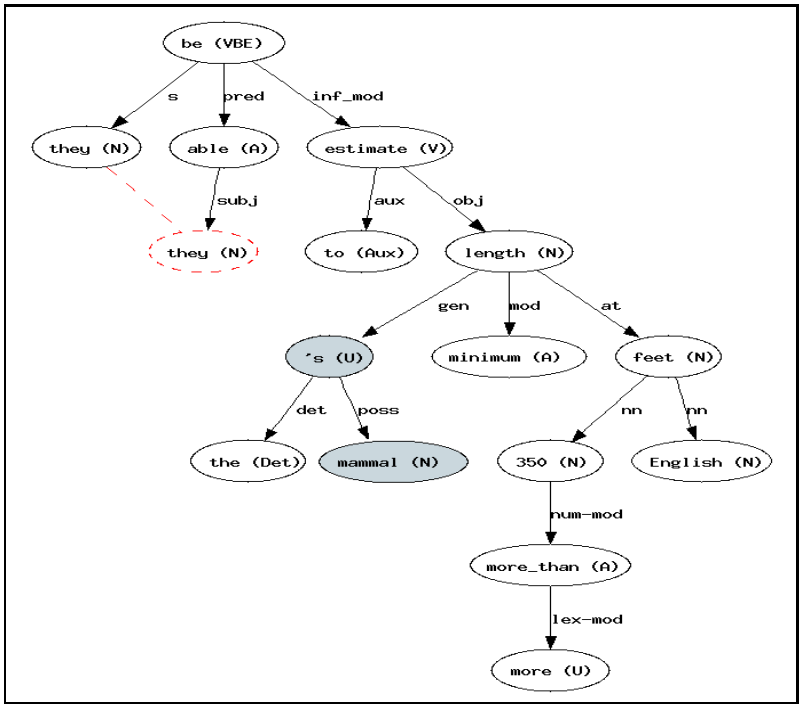

Figure 2.10: After the post-processing on the 'poss' relation for the dependency tree in the previous figure

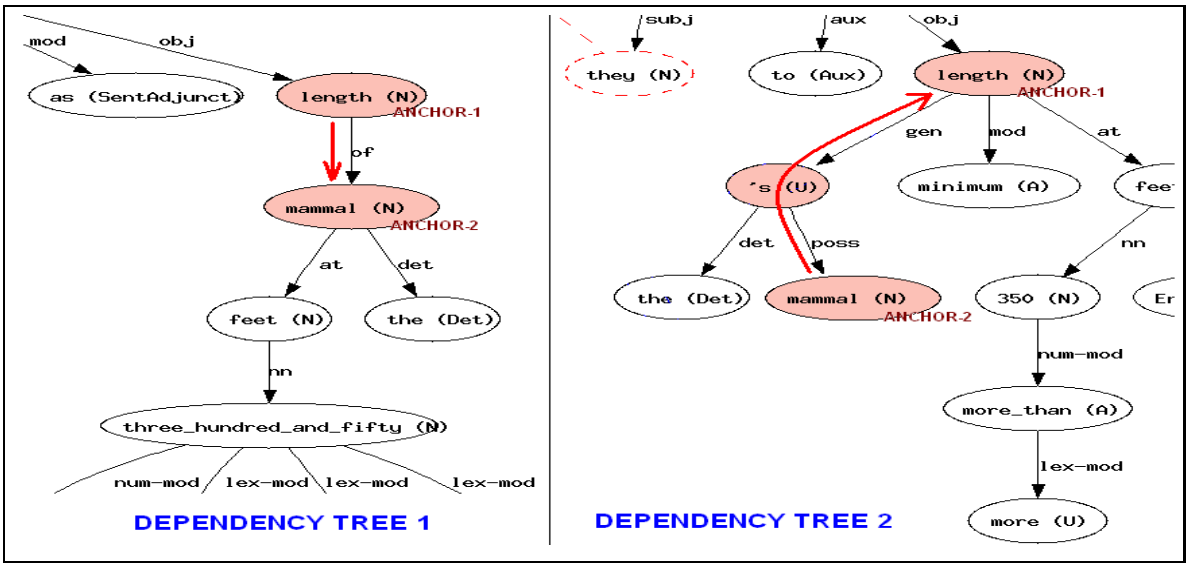

Figure 2.11: A sample match of paths from different dependency trees

The results show that our system performance is comparable with [Ibrahim et al., 2003].

We used 102 sentence pairs. This set includes sentence pairs that were evaluated by all of the techniques in [Finch et al., 2005]. However, note that we just used the sentence pairs originating from document pairs judged as 3. 15% of them are annotated as *equivalent* and the rest are annotated as *not equivalent*. When we investigated the sentence pairs, we found that 12.08% of them contains paraphrases that we are interested in. The remaining 2.92% were either not suitable for paraphrase extraction or they were just identical.

We used three metrics: precision, recall and f-measure$_{0.5}$ (For detailed explanation of these terms, refer to Appendix C.4). The precision value reveals the ratio of the correct paraphrases detected to all the phrases detected:

$$\text{precision} = \frac{\text{\# of correct paraphrases found}}{\text{\# of all phrases pairs found}}$$

Fortunately, we can calculate the recall of our system, since we did hand-examination for our test set. The recall value reveals the ratio of the correct paraphrases detected to all the correct paraphrases in the data set:

$$\text{recall} = \frac{\text{\# of correct paraphrases found}}{\text{\# of all correct paraphrases}}$$

Precision value being higher is more important than the recall value being higher because eliminating false positives is more critical than missing correct paraphrases. However,

	precision	recall	f-measure$_{0.5}$
EM	0.44(±23)	0.70(±28)	0.50
WER	0.33(±54)	0.50(±70)	0.48
sNIST2	0.77(±27)	0.77(±27)	0.77
sNIST3	0.63(±29)	0.77(±27)	0.67
sNIST4	0.63(±29)	0.77(±27)	0.67
BLEU1	0.41(±28)	0.71(±34)	0.48
BLEU2	0.54(±30)	0.75(±30)	0.60
BLEU3	0.55(±33)	0.71(±34)	0.60
BLEU4	0.54(±30)	0.75(±30)	0.60

Figure 2.12: Precision, recall, and F-measures obtained for parphrase extraction for each data set originating from a specific MT evaluation technique (The values are within the intervals with 95% confidence)

precision value being low means we are also learning phrases that are not actually paraphrases.

So, we use f-measure$_\alpha$, which is the weighted harmonic mean of precision and recall. We use $\alpha = 0.5$ so that the precision is weighted twice the recall:

$$\text{f-measure} = \frac{(1+\alpha) * \text{precision} * \text{recall}}{(\alpha * \text{precision}) + \text{recall}}$$

These results show using sNIST evaluation technique yields the highest quality sentence pairs for paraphrase extraction. However, for a more reliable result, the test should be done on much larger corpus of sentence pairs. Even though, the precision value for sNIST was smaller than that of EM, the false positives were eliminated in the paraphrase extraction phase.

Here are some examples that are false positives, originating from EM sentence pairs:

X tell Y (X: PROPER NOUN / Y: PROPER NOUN)

X arrested by Y (X: PROPER NOUN / Y: PROPER NOUN)

X system with Y (X: traffic / Y: one)

X of Y controllers (X: one / Y: traffic)

X was satisfied Y said (X: he/ Y: PROPER NOUN)
X head Y will meet with (X: he / Y: PROPER NOUN)

The following are some of the correctly extracted paraphrases, originating from EM sentence pairs:
X will **inaugurate** Y embassy (X: he / Y: Dutch)
X will **open** Y embassy (X: he / Y: Dutch)

X has been **made** on Y (X: decision / Y: considerations)
X has been **taken** on Y (X: decision / Y: considerations)

X **were part of** force replaced by Y (X: troops/ Y: mission)
X **take part in** Y (X: troops/ Y: mission)

Here is an example that is a false positive, originating from the sNIST sentence pairs:
start marathon at Olympics in X Y said (X: morning / Y: he)
X made decision on changing marathon to Y event (X: morning / Y: he)

And here are some of the correctly extracted paraphrases, originating from sNIST sentence pairs:
X officer **appointed to lead** the Y (X: PROPER NOUN / Y: force)
X national **named to head** the Y (X: PROPER NOUN / Y: force)

X **served on** Y (X: officer / Y: mission)
X **worked on** Y (X: officer / Y: mission)

X **informed** Y (X: PROPER NOUN / Y: PROPER NOUN)
X **told** Y (X: PROPER NOUN / Y: PROPER NOUN)

purpose of X **is to reinforce** Y (X: conference / Y: peace)

X **is a step in** Y **process** (X: conference / Y: peace)

To conclude, we obtained comparable results with [Ibrahim et al., 2003]. However, to obtain more accurate results, we need to test the system with more sentence pairs.

Although the precision values we obtained are comparable with other studies[4] we need higher precision values to create a generic database of paraphrases. The test set we used yielded paraphrases that are in general synonymous words of each other, such as:

ray of X ... Y $\longleftrightarrow$ **beam** of X ...Y

X **informed** Y $\longleftrightarrow$ X **told** Y

However, our system was also able to extract valuable paraphrases, such as:

purpose of X **is to reinforce** Y $\longleftrightarrow$ X **is a step in** Y **process**

[4]We obtained 77% precision when only the exactly matching document pairs are used. However, for a more reliable measure, we need to test the system with a much larger test set.

Chapter 3

RELATED WORK

The related works that are very similar to our work are [Ibrahim et al., 2003] and [Finch et al., 2005] which are described in the previous chapters. In this chapter, we describe [Lin and Pantel, 2001], [Barzilay and Lee, 2003] and [Dolan et al., 2004] in more detail.

3.1 DIRT: Discovery of Inference Rules from Text

DIRT [Lin and Pantel, 2001] is an unsupervised method for discovering inference rules from text. An example to an inference rule is "X found a solution to Y" $\longrightarrow$ "X solved Y". It is an idea related to the Harris' Distributional Hypothesis, which states that words occurring in the same context tends to be similar.

To clarify with an example, let's consider these two words 'responsibility' and 'duty'. Both of the words can be modified by *additional, administrative, assigned, assumed, collective, congressional, constitutional* and both of the words can be objects of verbs such as: *accept, articulate, assert, assign, assume, attend to, avoid, become, breach.* So, according to this hypothesis, these two words are similar in meaning.

[Lin and Pantel, 2001] extends this hypothesis, called 'Extended Harris' Distributional Hypothesis', which states that if two paths tend to link the same set of words, their meanings tend to be similar. Here is an example of two paths as illustrated in Figure 3.1:

Based on this hypothesis, they parse every sentence in a corpus and obtain a dependency tree (Figure 2.8) and extract paths out of them. The ends of the paths are called *slots* and their instances are called *slot fillers.* The paths are picked so that:

- Their slot fillers are *nouns.*
- Any dependency relation that does not connect two content words (i.e. nouns, verbs, adjectives or adverbs) is excluded from a path.
- The frequency count of an internal relation must exceed a threshold.

"X finds a solution to Y"		*"X solves Y"*	
SLOTX	SLOTY	SLOTX	SLOTY
commission	strike	committee	problem
committee	civil war	clout	crisis
committee	crisis	government	problem
government	crisis	he	mystery
government	problem	she	problem
he	problem	petition	woe
legislator	budget deficit	researcher	mystery
sheriff	dispute	sheriff	murder

Figure 3.1: Two paths that tend to link the same set of words

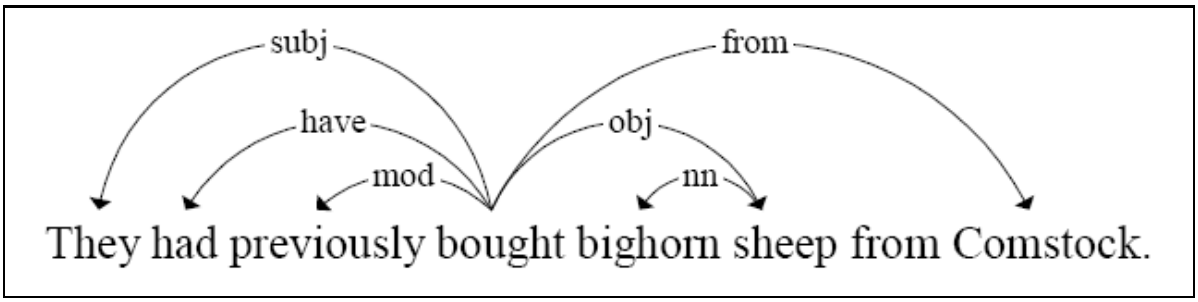

Figure 3.2: Dependency Tree of "They had previously bought bighorn sheep from Comstock."

For instance, the following paths can be extracted from the dependency tree illustrated in Figure 3.2:

p_1 : $\overset{N:subj:V}{\longleftarrow}$ **buy** $\overset{V:from:N}{\longrightarrow}$ (X buys something from Y)

p_2 : $\overset{N:subj:V}{\longleftarrow}$ **buy** $\overset{V:obj:N}{\longrightarrow}$ (X buys from Y)

p_3 : $\overset{N:subj:V}{\longleftarrow}$ **buy** $\overset{V:obj:N}{\longrightarrow}$ *sheep* $\overset{N:nn:N}{\longrightarrow}$ (X buys Y sheep)

p_4 : $\overset{N:nn:N}{\longleftarrow}$ *sheep* $\overset{N:obj:V}{\longleftarrow}$ **buy** $\overset{V:from:N}{\longrightarrow}$ (X sheep is bought from Y)

p_5 : $\overset{N:obj:V}{\longleftarrow}$ **buy** $\overset{V:from:N}{\longrightarrow}$ (X is bought from Y)

They collect the frequency counts of all paths in a corpus and the slot fillers for the paths. For each instance of a path p that connects two words w_1 and w_2, they increase the frequency counts of the two triples $(p, SLOTX, w_1)$ and $(p, SLOTY, w_2)$. They call

$(SLOTX, w_1)$ and $(SLOTY, w_2)$ features of path p. The more features two paths share, the more similar they are.

Similarity of two paths p_1 and p_2 are measured by the following formula:

$$S(p_1, p_2) = \sqrt{\text{sim}(SLOTX_1, SLOTX_2) * \text{sim}(SLOTY_1, SLOTY_2)}$$

Similarity of two slots are measured by:

$$\text{sim}(slot_1, slot_2) = \frac{\sum_{w \in T(p_1,s) \cap T(p_2,s)} \text{mi}(p_1, s, w) + \text{mi}(p_2, s, w)}{\sum_{w \in T(p_1,s)} \text{mi}(p_1, s, w) + \sum_{w \in T(p_2,s)} \text{mi}(p_2, s, w)}$$

$T(p, s)$ is the set of words that fill slot s. $\text{mi}(p, slot, w)$ is the mutual information between a path slot $(p, slot)$ and its filler w:

$$\text{mi}(p, slot, w) = \frac{\Pr(p, slot, w)}{\Pr(p, slot) * \Pr(slot, w)}$$

$\Pr(p, slot, w)$ is the frequency of occurrence of w in path slot $(p, slot)$, $\Pr(p, slot)$ is the frequency of occurrence of $(p, slot)$, and $\Pr(slot, w)$ is the frequency of occurrence of w in slot $slot$.

They compared the inference rules their algorithm generated with a set of human generated paraphrases of the first six questions of the TREC8 Question Answering Contest. The results show that the overlap between the human generated paraphrases and system output is very little but the percentage of the correct paraphrases is quite high. This suggests that finding potentially useful inference rules is very difficult for humans as well as machines.

3.2 Extracting Structural Paraphrases from Aligned Monolingual Corpora

[Ibrahim et al., 2003] used multiple translations of foreign novels. The advantage of this approach is that the events are occurring in the same order in each text so alignment of these translations becomes an easier task. However, the disadvantage of the approach is that the data set being used is rather small compared to using parallel news corpora.

The sentence pairs produced in the alignment process are then parsed by the Link Parser [Sleator and Temperley, 1993], which is a dependency-based parser developed at CMU. The resulting parse structures are, then, post-processed to obtain more consistent links, because Link Parser does not directly identify the subject of a passive sentence. The auxiliary verbs are also discarded so that linkages remain consistent with subject and object linkages.

Moreover, common nouns denoting places and people are marked by consulting WordNet dictionary.

After these steps, to extract paraphrases, they started with finding common anchors between aligned sentence pairs, which is an idea borrowed from [Lin and Pantel, 2001]. They only used nouns and pronouns as anchors. The anchors are then scored using the following heuristics:

- If the anchors are exact word matches, this denotes correspondence.
- The same genders or numbers in the nouns or pronouns denote correspondence. This kind of matches penalize the score by 1/2.
- If the anchors are from unique semantic class, such as places or people, such matches penalize the score by 1/2.
- If the anchors are the only noun or pronoun pair in the sentences, such matches penalizes the score by 1/2.
- If the anchors occur more than once in the aligned sentences, all possible combinations are considered but each such match penalizes the score by 1/2.

For each anchor pair that is initially penalized by the heuristics above, a breadth-first search is applied between the anchor words. If there is a conjunction or punctuation within the path, that path is directly rejected. If valid paths are found between anchor pairs in both of the sentences, these paths become candidate paraphrases with their initial scores, set by the heuristics above.

The initial default score of a paraphrase is one, assuming that it is a perfect anchor match. Every additional occurrence of the paraphrase contributes to the score by $(1/2)^n$, where n is the number of times this set of anchors have been seen. So seeing a new set of anchors for a specific paraphrase has a big impact on the score, but re-occurrences of the anchor set has decreasing impact on the score. If the final score a paraphrase is above a threshold, the paraphrases are labeled as correct.

The average precision is reported as 41% and the average length of the paraphrases learned is 3.26 words long. All of the evaluators agreed on the judgments, either positive or

negative, only 75.4% of the time. The highest score obtained by the system is the equivalence of possessive morpheme *s* with the preposition *of*. Some other interesting equivalence examples are:

Paraphrases:

$$A_1 \overset{*}{\longleftrightarrow} liked \overset{O}{\longleftrightarrow} A_2$$

$$\Longleftrightarrow$$

$$A_1 \overset{*}{\longleftrightarrow} fond \overset{OF}{\longleftrightarrow} of \overset{J}{\longleftrightarrow} A_2$$

Sentences:

The clerk liked Monsieur Bovary.

$\Longleftrightarrow$

The clerk was fond of Monsieur Bovary.

Paraphrases:

$$A_1 \overset{s}{\longleftrightarrow} rush \overset{K}{\longleftrightarrow} over \overset{MV}{\longleftrightarrow} to \overset{J}{\longleftrightarrow} A_2$$

$$\Longleftrightarrow$$

$$A_1 \overset{s}{\longleftrightarrow} run \overset{MV}{\longleftrightarrow} to \overset{J}{\longleftrightarrow} A_2$$

Sentences:

And he rushed over to his son, who had just jumped into a heap of lime to whiten his shoes.

$\Longleftrightarrow$

And he ran to his son, who had just precipitated himself into a heap of lime in order to whiten his boots.

Paraphrases:

$$A_1 \overset{*}{\longleftrightarrow} fit \overset{MV}{\longleftrightarrow} to \overset{I}{\longleftrightarrow} give \overset{O}{\longleftrightarrow} A_2$$

$$\Longleftrightarrow$$

$$A_1 \overset{*}{\longleftrightarrow} appropriate \overset{MV}{\longleftrightarrow} to \overset{I}{\longleftrightarrow} supply \overset{O}{\longleftrightarrow} A_2$$

Sentences:

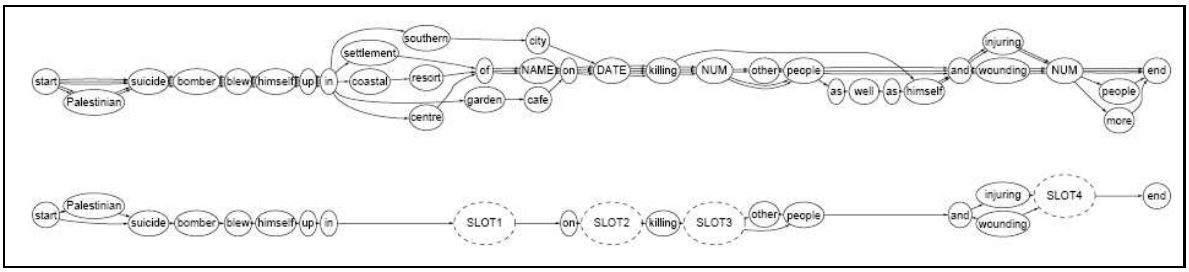

Figure 3.3: A sample of lattice and its slotted lattice

He thought fit, after the first few mouthfuls, to give some details as to the catastrophe

$\Longleftrightarrow$

After the fist few mouthfuls he considered it appropriate to supply a few details concerning the catastrophe.

It is reported that increasing the threshold for generating paraphrases increases the precision up to a certain point. Moreover, the highest ranking structural paraphrases consists of simple word paraphrases of prepositions, such as $at \Longleftrightarrow in$.

3.3 Learning to Paraphrase: An Unsupervised Approach Using Multiple-Sequence Alignment

[Barzilay and Lee, 2003] focus on sentence-level paraphrase generation. They apply multiple-sequence alignment[1] to sentences on un-annotated comparable news corpora. The system learns a set of paraphrasing patterns represented by word lattice pairs and automatically determines how to apply these patterns to create new sentences.

They first divide the corpus into partitions and create clusters of similar sentences. They accomplish this by applying hierarchical complete-link clustering to the sentences using n-gram overlap. After obtaining clusters, they do multiple-sequence alignment based on minimal number of insertion deletion and substitutions and obtain lattice as illustrated in 3.3.

The next step is determining the slots in these multiple-sequence alignments. Areas of large variability should be selected as slots. They first search for commonalities. They

[1] A Multiple-Sequence Alignment is a structure borrowed from Bioinformatics, which is a sequence alignment of three or more biological sequences, generally protein, DNA, or RNA.

identify the backbone nodes, which constitute those nodes shared by more than 50% of the cluster's sentences. Then, they identify slots in the lattices by looking at the degree of variability of the regions. Figure 3.3 shows a slotted lattice.

After generating lattices from each corpus, they take two lattices from different corpora and search for similarities in their slot values by looking back at the sentences of the lattice's clusters. If the similarity is above a threshold, the lattice pair is accepted as paraphrases.

Paraphrase generation is performed by, given an input sentence to paraphrase, first finding the lattice that has the best alignment with the sentence. If a lattice is found, one of the paths in the lattice is chosen to rewrite the sentence. As many paraphrases can be generated as they are different paths in the lattice. An example of an input sentence and its generated paraphrase is as follows:

Original: `A spokesman for the group claimed responsibility for the attack in a phone call to AFP in this northern West Bank town.`

Paraphrase: `The attack in a phone call to AFP in this modern West Bank town was claimed by a spokesman of the group.`

They ran the exact corpus on DIRT and their own system. According to the judgments [Barzilay and Lee, 2003] performed much better then [Lin and Pantel, 2001] with a 38% performance gap.

3.4 Unsupervised Construction of Large Paraphrase Corpora: Exploiting Massively Parallel News Sources

[Dolan et al., 2004] investigated how a good corpus of paraphrase pairs can be obtained by just considering the first two sentences of the news articles. They claim that the first two sentences of the news articles tend to be suitable to serve as a summary of the article. So, if matching articles can be found, their two sentences can be used as paraphrase sentences of each other, which is called the F2 corpus. To compare their system with a baseline, they also collected sentence pairs from matching documents using Levenstein distance metric, which is called L12 corpus.

Moreover, they state that Bilingual Machine Translation is very similar to Monolingual Paraphrase Generation, since they both deal with aligning one sequence of words into another. So, they trained the well known Giza++ alignment application with a subset of both

of the corpora and used the rest of the corpora to test the system using Alignment Error Rate metric. To clarify, for each sentence pair in the test set, they input one sentence of the pair as input and use the other as a gold reference for the output of the system.

They report more paraphrase alternations from F2 corpus, compared to L12 corpus. Here are the clasifications:

- Elaboration: the NASDAQ — the tech-heavy NASDAQ
- Phrasal: a million people — a massive crowd
- Spelling: color — colour, email — e-mail
- Anaphora: Mr. Smith — he
- Reordering: He said "..." — "..." he said.

Chapter 4

CONCLUSION

In this thesis, we explore the recent studies conducted to extract paraphrases from large amount of raw text and we come up with our own approach.

We approached to the problem in three steps. Given a set of news articles, we first identified news articles that describe the same events to maximize the chances of seeing equivalent sentence pairs. Second, we find the two best matching sentence pairs from these document pairs. Finally, we parse the sentence pair, obtain their respective dependency trees and search for similar paths in the dependency trees.

We obtained 77% precision with sNIST2, 55% precision with BLEU3 and 44% precision with EM, when only equivalent document pairs are used. However, the data set we used is around 80KB and the experiment should be performed on much bigger data set to come up with more reliable results[1].

We showed that our results are comparable with [Ibrahim et al., 2003] by running every paraphrase pair extracted by their system. We obtained 85% exact match and the rest of the pairs were part of the matches as a subset[2].

To improve the performance of our system, we need a better document-level equivalence match. If we use publicly available parallel news corpora, instead of using a limited offline corpora such as Gigaword, we can obtain much better resource for sentence-level matching and thus for the phrase-level matching. Another improvement would be on eliminating the false positive pairs. We expect the correct paraphrases to re-occur more frequently then the false positive pairs. Using a threshold value, we can eliminate the infrequent pairs.

[1] The reason we worked on a small data set is the limitation of human annotators

[2] Example:
beams of X dazzled her Y (X: sun / Y: eyes)
rays of X dazzled her Y (X: sun / Y: eyes)

Appendix A

WORDNET

WordNet is a semantic lexicon for the English language. It groups English words into sets of synonyms called synsets and provides short, general definitions, and records semantic relations between these synonym sets. Most synsets are connected to other synsets through a number of semantic relations. These relations vary based on the type of the words. Some of them are:

- For Nouns:
  - Hypernym(X, Y): X is a kind of Y
  - Hyponym(X,Y): Y is a kind of X
  - Holonym(X, Y): X is part of Y
  - Meronym(X, Y): Y is part of X
  - Coordinate terms(X, Y): X and Y share a hypernym
- For Verbs:
  - Hypernym(X, Y): The action X is a kind of action Y (travel and move)
  - Hyponym (X, Y): The action Y is a kind of action X.
  - Entailment(X, Y): If action X is performed, action Y is also performed automatically. (snore and sleep)
  - Coordinate terms (X, Y): X and Y share a hypernym
- For Adjectives:
  - Related nouns
  - Participle of verb

- For Adverbs:
  - Root adjectives

Both nouns and verbs are organized into hierarchies, defined by hypernym relationships. For instance, the fourth sense of the word chair has the following hypernym hierarchy. The words on the same level are synonyms of each other, that is, some sense of electric chair is synonymous with some other senses of chair and death chair and so on. Each set of synonyms, also known as a synset, has a unique index and share their properties, such as gloss definition.

electric chair, chair, death chair, hot seat – (an instrument of execution by electrocution; resembles an ordinary seat for one person; "the murderer was sentenced to die in the chair")

⇒ instrument of execution
⇒ instrument
⇒ device
⇒ instrumentality, instrumentation
⇒ artifact, artefact
⇒ whole, unit
⇒ object, physical object
⇒ physical entity
⇒ entity

Appendix B

COMMON DESIGN FEATURES OF SEARCH ENGINES

Some common design features of IR Systems are inverted indexing, position information, phrases, list of stop words and stemming.

B.1 Inverted Index

An inverted index is simply a hashtable, where the keys are the whole words occurring in a corpus of documents and the values are the lists of documents these words occur with their frequency of occurrences.

B.2 Position Information

An inverted index with position information is a hashtable, which additionally stores a list the positions of each word in the corpus. This enables phrasal searches. For instance, if 'travel agency' is being searched, first the positions of the word 'travel' is retrieved, then the positions of the word 'agency' is retrieved, and then the positions from the two results are merged, and finally consecutive position values are returned. This is much faster than searching for 'travel agency' inside the whole corpus.

B.3 Stop List

Some words are very frequently used in natural languages. They are so frequent that they have no distinguishable effect on the results. For instance, independent of the subject or the event being described in the document, the following list of words appear frequently: *a, also, an, and, as, at, be, but, by, can, could, do, for, from, go, have, he, her, here, his, how, i, if, in, into, it, its, my, of, on, or, our, say, she, that, the, their, there, therefore, they, this, these, those, through, to, until, we what, when, where, which, while, who, with, would, you, your*. So, these words can safely be ignored while indexing and searching.

B.4 Stemming

Stemming is the process of obtaining the roots of the words. This is a very useful method to increase the number of hits since a word and its root has very high semantic similarities. We can appreciate this by considering examples like 'cars' and 'car', or 'laughing', 'laughter' and 'laugh'.

B.5 Scoring and Ranking with TF-IDF Weight

tf-idf weight (term frequency inverse document frequency) is a weight to evaluate how important a word is to a document. In this weighting function the importance of a word increases as the frequency of the word increases in a document, but the degree of importance of diminishes depending on how common the word is in all documents. Term frequency of a word t_i is calculated with the following formula:

$$tf = \frac{n_i}{\sum_k n_k}$$

n_i is the number of occurrences of the word t_i, and

n_k is the number of occurrences of all words in the document.

Inverse document frequency is calculated by the following formula:

$$idf = log(\frac{|D|}{|t_i \in d_i|})$$

$|D|$ is the number of all documents in the system

d_i is the document being considered

Then,

$$\text{tf-idf} = tf * idf$$

A high weight in tf-idf is reached by a high term frequency in the given document and a low document frequency of the term in the whole collection of documents; the weights tend to filter out common terms. For instance, if the word occurs in all documents, idf becomes 0 and t_i will have no contribution to the end score.

Lucene is one such search engine written in Java that provides fast indexing and searching and this tool is used as our document retrieval system.

Appendix C

SCORING METRICS

There are a number of commonly used scoring metrics that quantify performance of a system. These scoring metrics measure how well the retrieved information matches the desired information. Here is a brief explanation of some of them.

C.1 Precision

Precision of a system is the proportion of retrieved information that is relevant to all the retrieved information. The formula of precision is as follows:

$$\text{Precision} = \frac{|\{\text{relevant information}\} \cap \{\text{retrieved information}\}|}{|\{\text{retrieved information}\}|}$$

Although precision score being high is very good for information retrieval systems, this score does not give any clue about how much of all relevant information in a corpus is retrieved.

C.2 Recall

Recall of a system is the proportion of retrieved information that is relevant to all the relevant information. The formula of recall is as follows:

$$\text{Recall} = \frac{|\{\text{relevant information}\} \cap \{\text{retrieved information}\}|}{|\{\text{relevant information}\}|}$$

The percentage of relevant information retrieved among all relevant information in a corpus does not give any clue about the precision in the retrieved documents.

Precision and recall are inversely proportional. We can visualize it with the following graph:

A 100% recall can be achieved by retrieving all the documents in a corpus. However, the precision will be dramatically low. Conversely, a 100% precision can achieved by just

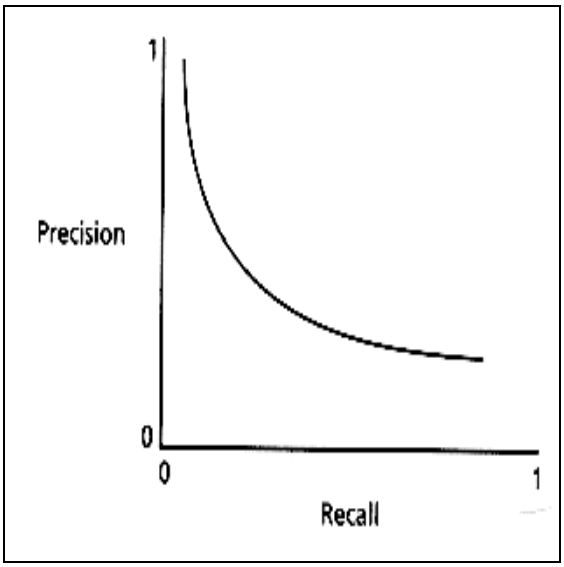

Figure C.1: Precision - Recall Graph

retrieving very little but relevant information but the rest of the relevant information will be missed.

So, the performance of the system increases as the average distance of the curve to origin increases. The importance of precision and recall depends on the goal of the search. Knowing the goal of the search determines what strategies the searcher will use (to find everything on a topic, just a few relevant papers, or something in between.

C.3 Fall-Out

Fall-out of a system is the proportion of the retrieved information that is irrelevant to all the retrieved information. Precision + Fall-out equals 1.

C.4 F-measure

F-measure is the weighted harmonic mean of precision and recall. The formula of F-measure is as follows:

$$F_{\text{alpha}} = \frac{(1 + \text{alpha}) * \text{precision} * \text{recall}}{((\text{alpha} * \text{precision}) + \text{recall})}$$

As the alpha value increases, the weight of recall increases in the measure.

C.5 Confidence Weighted Score

Confident Weighted Score is similar to the precision score but the order of the retrieved information also gets into the calculations. The formula of Confidence Weighted Score (CWS) is as follows:

$$CWS = \frac{1}{N} * \sum_{i=1}^{N} \frac{\text{number of relevant documents in the first i documents retrieved}}{i}$$

The power of this measure is that it also scores the confidence of the system for retrieved documents.

BIBLIOGRAPHY

[Bannard and Callison-Burch, 2005] Bannard, C. and Callison-Burch, C. (2005). Paraphrasing with bilingual parallel corpora. In *Proceedings of the 43rd annual meeting of the Association for Computational Linguistics (ACL2005)*.

[Barzilay and Lee, 2002] Barzilay, R. and Lee, L. (2002). Bootstrapping lexical choice via multiple-sequence alignment. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

[Barzilay and Lee, 2003] Barzilay, R. and Lee, L. (2003). Learning to paraphrase: An unsupervised approach using multiple-sequence alignment. In *HLT-NAACL*, pages 16–23.

[Barzilay and McKeown, 2001] Barzilay, R. and McKeown, K. R. (2001). Extracting paraphrases from a parallel corpus. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL/EACL 2001)*.

[Brockett and Dolan, 2005] Brockett, C. and Dolan, W. B. (2005). Support vector machines for paraphrase identification and corpus construction. In *IWP2005*.

[Doddington, 2002] Doddington, G. (2002). Authomatic evaluation of machine translation quality using n-gram co-occurrence statistics.

[Dolan and Brockett, 2005] Dolan, W. B. and Brockett, C. (2005). Automatically constructing a corpus of sentential paraphrases. In *IWP2005*.

[Dolan et al., 2004] Dolan, W. B., Quirk, C., and Brockett, C. (2004). Unsupervised construction of large paraphrase corpora: Exploiting massively parallel news sources. In *COLING 2004*, Geneva, Switzerland.

[Dras and Yamamoto, 2005] Dras, M. and Yamamoto, K., editors (2005). *3rd International Workshop on Paraphrasing*.

[Finch et al., 2005] Finch, A., Hwang, Y. S., and Sumita, E. (2005). Using machine translation evaluation techniques to determine sentence-level semantic equivalence. In *IWP2005*.

[Gale and Church, 1991] Gale, W. A. and Church, K. W. (1991). A program for aligning sentences in bilingual corpora. In *ACL*.

[Gale and Church, 1993] Gale, W. A. and Church, K. W. (1993). A program for aligning sentences in bilingual corpora. *Computational Linguistics*, 19(1):75–102.

[Geffet and Dagan, 2005] Geffet, M. and Dagan, I. (2005). The distributional inclusion hypothesis. In *Proceedings of the 43rd annual meeting of the Association for Computational Linguistics (ACL2005)*.

[Hirst, 2003] Hirst, G. (2003). Paraphrasing paraphrased. IWP2003 Invited Talk.

[Ibrahim, 2002] Ibrahim, A. (2002). Extracting paraphrases from aligned corpora. Master's thesis, Massachusetts Institute of Technology.

[Ibrahim et al., 2003] Ibrahim, A., Katz, B., and Lin, J. (2003). Extracting structural paraphrases from aligned monolingual corpora. In *IWP2003*.

[Inui and Hermjakob, 2003] Inui, K. and Hermjakob, U., editors (2003). *2nd International Workshop on Paraphrasing*.

[Katz, 1997] Katz, B. (1997). Annotating the world wide web using natural language. In *RIAO*.

[Katz and Levin, 1988] Katz, B. and Levin, B. (1988). Exploiting lexical regularities in designing natural language systems. In *COLING*.

[Lin, 1998] Lin, D. (1998). Dependency based evaluation of minipar. In Proceedings of the Workshop on the Evaluation of Parsing Systems, First International Conference on Language Resources and Evaluation, Granada, Spain.

[Lin and Pantel, 2001] Lin, D. and Pantel, P. (2001). Dirt – discovery of inference rules from text. In *Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining*.

[Marton, 2006] Marton, G. A. (2006). Relation acquisition over compositional phrases. PhD proposal.

[Pang et al., 2003] Pang, B., Knight, K., and Marcu, D. (2003). Syntax-based alignment of multiple translations: Extracting paraphrases and generating new sentences. In *Proceedings of the North American Association for Computational Linguistics and the Human Language Technologies Conferences (NAACL/HLT2003)*.

[Papineni, 2001] Papineni, K. (2001). Bleu: a method for automatic evaluation of machine translation.

[Quirk et al., 2004] Quirk, C., Brockett, C., and Dolan, W. B. (2004). Monolingual machine translation for paraphrase generation. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pages 142–149, Barcelona Spain.

[Sato and Nakagawa, 2001] Sato, S. and Nakagawa, H., editors (2001). *1st International Workshop on Paraphrasing*.

[Sekine, 2005] Sekine, S. (2005). Automatic paraphrase discovery based on context and keywords between ne pairs. In *IWP2005*.

[Shinyama and Sekine, 2003] Shinyama, Y. and Sekine, S. (2003). Paraphrase acquisition for information extraction. In *IWP2003*.

[Sleator and Temperley, 1993] Sleator, D. and Temperley, D. (1993). Parsing english with a link grammar. In *Third international workshop on parsing technologies*.

VITA

Bengi Mizrahi was born in Istanbul in January 6, 1980. He graduated from Ayazaga Isik High School in 1998. He got his Bachelor's Degree in Computational Science from Bilkent University in 2003, and his Master's Degree in Electrical and Computational Engineering from Koc University in 2006. He did internship in Turkcell Iletisim Hizmetleri for 3 months in 2002 and as of 2005, he is working at ARGELA Technologies in department of Next Generation Networks.