Identifying Word Translations in Parallel Corpora

Using Measures of Association

by

Nitin Varma

December 2002

Submitted in partial fulfillment of the

requirements for the degree of

Master of Science

under the instruction of Dr. Ted Pedersen

Department of Computer Science

University of Minnesota

Duluth, Minnesota 55812

U.S.A.

UNIVERSITY OF MINNESOTA

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

Nitin Varma

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.

Dr. Ted Pedersen

Name of Faculty Adviser

Signature of Faculty Advisor

Date

GRADUATE SCHOOL

I am grateful to my advisor Dr. Ted Pedersen for providing me an opportunity to work with him and the valuable guidance that he provided. I would also thank him for the amount of time he has spent in discussing things and for his expertise, comments, feedback and encouragement on many of issues surrounding my work. There were times when he has been thinking about some of the issues more than me. Without his guidance and help this work would not have been possible.

I wish to offer special thanks to Dr. Carolyn Crouch and Dr. Joe Gallian for being on the committee and for their useful comments on the thesis.

I would also like to acknowledge the help of the department of computer science at University of Minnesota Duluth. Specifically I would like to thank Dr. Donald Crouch, Lori Lucia, Linda Meek, and Jim Luttinen for their help. I would also like to thank Jim Luttinen for allowing me to access some of his own machines for running my programs.

I am indebted to my NLP group mates (Satanjeev ”Bano” Banerjee, Siddharth ”Sid” Patwardhan, Saif Mohammad, and Bridget McInnes) who must have been tired of reading my thesis, testing and reviewing my codes. Their comments and suggestions and regarding the same were invaluable.

I also would like to thank my fellow graduate students and friends Krishna, Sree, Amit, Deepa, Kiranmai, Deepa, Kiran, Aditi for their suggestions and comments.

Finally I would like to thank my family for their support and encouragement.

Thank you everyone.

Contents

List of Figures

List of Tables

Abstract

There are increasing amounts of parallel text available online. Such text consists of an original document and its translation into another language. This thesis takes the view that such data is a very rich source of knowledge that can be utilized to learn how languages can be translated from one to the other. In particular, this thesis focuses on developing techniques that can be used to learn which words are translations of each other, simply based on information found in a large sample of parallel text.

The methods employed here are measures of association that have been used in a wide range of statistical applications, and have proven very useful in corpus based natural language processing. In this thesis we explore their use in identifying which words are translations of each other. This thesis starts with an examination of one of the earliest of such approaches, known as K-vec (Fung and Church, 1994). We offer several improvements to this algorithm that lead to demonstrably better results in two very different domains. We also evaluate a number of measures of association and identify the T-score, the Log-likelihood Ratio, and the Odds Ratio as being particularly effective. Finally, we propose two ensemble techniques for combining different measures of associations and also for combining different formulations of the same measure and show that both lead to improved results.

1 Introduction

1.1 Objective and Motivation of the Thesis

There is an increasing amount of information available online in many different languages. Among the most intriguing of these resources are large amounts of parallel text, which is a written text that has been translated into other languages. The objective of this thesis is to use parallel texts as a source of data from which we can learn word translations automatically.

We would particularly like to develop techniques that can help us derive or improve bilingual dictionaries, which contain information about how a word is translated from one language to another. For example, an English-French bilingual dictionary might have an entry that indicates the English word *king* is translated into French as *roi*, and that the English word *people* is translated into French as *peuple*.

There have been a variety of approaches applied to the problem of learning such translations from parallel text. One school of thought, introduced in [10], is that traditional measures of association as used with two dimensional contingency tables could be successfully applied to this problem. This thesis follows in this tradition, and explores a wider range of measures than has previously been considered for this problem. Among the measures we explore are:

1. Pointwise Mutual Information [2],
2. the Dice Coefficient [5],
3. the Log-likelihood Ratio [4],
4. Pearson's Chi-square test [4],
5. the Odds Ratio,
6. T-score [2] and
7. Fisher's Exact Test [15].

We choose to work with the wide variety of measures listed above because each of these tests has different characteristics (as will be described in the Background section). In addition, there have really been no

empirical evaluations of these measures as applied to this problem. Thus, this thesis hopes to establish baselines of performance on this problem using a wide range of fairly well known measures and publicly available data (that will be described later in the thesis).

1.2 Parallel Corpora

This thesis views parallel text as a rich source of knowledge for determining how words can be translated. In effect a parallel corpus represents the distillation of a human translator's expertise. We hope to extract just a small portion of that from the text, that is which words are translations of each other.

In general automatic techniques that identify word translations assume that the text has been *sentence aligned*, that is we know which sentences are translations of each other. This is often a reasonable assumption, as the number of sentences and their ordering does not tend to change radically in a translation, and fairly accurate automatic methods for finding sentence alignment exist. As will be seen later, our technique relaxes this requirement and does not require sentence aligned text.

Determining word translations from parallel text is a difficult problem since the position at which a word and its translation appear in their corresponding sentences might be quite different. In addition, a single word may translate as multiple words, or may not translate at all. Thus, while sentence alignment is fairly easy to determine since sentences do not move around a great deal in a translation, words are more problematic since they do move and may result in more or fewer words in their translated form.

The example in Table 1 shows a small portion of a parallel text that has been sentence aligned. We have inserted D> into the text, where D indicates the sentence number. From this we can see that the English sentence *Canadian Institute of Health Research Act* is translated as *Loi sue les de Instituts Recherche en santé du Canada* in French. It should be noted that this data comes to us from the Canadian Hansard's, which are the bilingual English–French proceedings of the Canadian parliament. We use a very large sample of this data in some of our experiments, which comes to us courtesy of Franz Och [14].

The problem of word movement in translation can be seen in the first English sentence, *Government Orders* of the example parallel text and its corresponding French sentence, *Initiatives Ministèrielles*. In the above case translations of the English words *Government* and *Orders* in French are *Ministèrielles* and *Initiatives* respectively. But if we align the words just by taking the words from corresponding positions we get a

English	French
1> Government Orders 2> Canadian Institutes of Health Research Act 3> Mr. Yvon Charbonneau (Parliamentary Secretary to Minister of Health, Lib.): 4> Mr. Speaker, on behalf of the Minister of Health, I am very pleased to speak today in support of Bill C-13, an act to establish the Canadian institutes of health research, at third reading stage. 5> Last week, on the very day that this House completed debate on the report stage of this bill, members of Canada's health research community gathered together to bid farewell to the Medical Research Council, and to greet the new era of the Canadian institutes of health research.	1> Initiatives Ministèrielles 2> Loi Sur Les Instituts De Recherche En Santé Du Canada 3> M. Yvon Charbonneau (secrétaire parlementaire du ministre de la Santé, Lib.): 4> Monsieur le Président, au nom du ministre de la Santé, je suis trés heureux de prendre la parole aujourd'hui en faveur du projet de loi C-13, Loi portant création des Instituts de recherche en santé du Canada à l'étape de la troisiéme lecture. 5> La semaine derniére, le jour même où les députés de cette Chambre mettaient fin au débat à l'étape du rapport de ce projet de loi, des membres du milieu de la recherche en sant du Canada se réunissaient pour faire leurs adieux au Conseil de recherches médicales et pour souligner la naissance des Instituts de recherche en santé du Canada.

Table 1: Sample English-French Parallel Text

wrong translation with the word *Government* being aligned to the word *Initiatives* and the word *Orders* being aligned to the word *Ministèrielles*. The problem of phrasal translations can also been seen in this example. For instance, the second English sentence has only 7 words, whereas its corresponding French sentence has 10 words.

Despite all of these challenges, our objective is to identify the words that are translations of each other based on information as found in a parallel text. For instance, we would like to know that the word *health* in the English sentence is translated as *santé* in the corresponding French sentence.

The quality of the parallel text used affects the quality of word translations that are discovered. In fact not

all parallel text is translated text. For example, The Bible exists in French, English, Spanish, Mandarin, etc. versions. These are not all translations of each other, yet they contain the same content. These different versions of The Bible may trace their origin back to a common earlier version, but over time they evolve in their own ways that reflect the standards and conventions of the language in which they are being used.

One of the sets of data that we employ is a small amount of French and English text that is from The Bible. This is known as the Blinker data [13], named after the project which put the data into a form suitable for this kind of research. Parallel text that is not a translation of each other is particularly challenging to deal with since there may be a larger number of differences between the two versions than there is in the text that is simply a translation from one to the other. On the other hand, sacred documents such as The Bible are translated with great care, and changes in words are undertaken only after considerable reflection, so even if two versions of The Bible share an origin that is several hundred years in the past (e.g., the King James version of The Bible) they may have evolved in relatively similar fashions in their respective languages.

Parallel text as found on the Internet can be quite noisy. It is often the case that parallel text is not necessarily a complete translation but may simply be a partial translation where some of the material at the end is omitted in the translated version. It is also fairly common that text that is advertised as translation is not really a translation but rather a paraphrase that preserves the original content. Such text is useless for our purposes since we are hoping to identify word translations.

1.3 Word Alignment Using Measures of Association

Gale and Church [10] proposed the use of measures of association in finding word translations. They employed the phi-coefficient and Pointwise Mutual Information as their measures of association. Their technique first aligns the parallel text at the sentence level, which means that it determines which sentences are translations of each other. Then it forms a contingency table for each possible word pair. For example, in Table 2 we see a contingency table that represents the measurement of the English word *health* with respect to whether or not it is translated as the French word *santè*.

In general these tables can be interpreted as follows:

- n_{11} - is the number of times *health* and *santè* occur in the corresponding sentences.

Table 2: Contingency table

		Y: $sante$	Y: $\neg sante$	$total$
X	$health$	$n_{11} = 4$	$n_{12} = 0$	$n_{1+} = 4$
X	$\neg health$	$n_{21} = 0$	$n_{22} = 1$	$n_{2+} = 1$
	$total$	$n_{+1} = 4$	$n_{+2} = 1$	$n_{++} = 5$

- n_{12} - is the number of times *health* occurs in a sentence and *santè* does not occur in the corresponding sentence.
- n_{21} - is the number of times *santè* occurs in a sentence and *health* does not occur in the corresponding sentence
- n_{22} - is the number of times neither *health* nor *santè* occur in the corresponding sentences.

This representation corresponds to a standard two dimensional contingency table that indicates how often two events (the English and French words) occur together. We can therefore apply any one of many measures of association to determine if the two events represented in the table are strongly associated with each other or not. If they are associated then the method of Gale and Church will consider them to be translations.

We use the K-vec algorithm of Fung and Church [7] as a starting point for finding word translations in parallel text. The K-vec algorithm is very closely related to the method of Gale and Church (which is not surprising, given that it is the same Church involved in both techniques!). However, K-vec does not require that the text be sentence aligned. Instead, it divides the parallel text into some number of pieces of fixed size and looks for translations within those pieces. The rationale given is that while sentence alignment is fairly easy for languages that are somewhat closely related (like English and French) it is not so simple for languages that are more distantly related (like English and Mandarin).

The intuition behind the K-vec algorithm is that if two words are translations of each other, then they occur an almost equal number of times and in approximately the same region of the original and translated text. For example, in the parallel text in Table 1 there are 6 occurrences of the English word *health* and 6 occurrences

Table 3: Frequency of occurrence and position of *health* and *santé*

health		*santé*	
Occurrence Number	Position	Occurrence number	Position
1	6	1	10
2	17	2	22
3	27	3	33
4	49	4	58
5	77	5	100
6	107	6	124

of the French word *santé* . For each occurrence of the words *health* and *santé*, Table 3 shows its position in the text.

There is an obvious exception to the above scenario when a word in one language is translated into more than one word in another language. For example if the word *santé* is also translated to some other word *X* and used in that sense only once in the parallel text above, then the frequency of the word *santé* and the word *X* will be different though they are translations of each other.

Our interest in K-vec is motivated by several factors. While many other techniques only work well with related languages, K-vec is asserted to work for any language pair. The only language dependent information that K-vec requires is that of *word segmentation*. It must be able to know where words begin and end in a language. Once that is true, it can proceed in the same way regardless of the languages involved. We are also interested in K-vec since despite being widely known, there have been no comparative evaluations of the approach and possible variations of it. We are also interested in exploring some of the decisions made by Fung and Church with respect to formulating their approach. In particular, they propose the use of the T-score [2] as the measure of association. We were intrigued by this since the T-score was originally proposed to find statistically significant sequences of words in large corpora such as *fine wine* and *major league*. The sample sizes and distribution of data in the contingency tables for such work are quite different than we find with K-vec, so it seemed reasonable to investigate how well the T-score performed under these circumstances. Finally, they also proposed to divide the text into a number of pieces equal in size to the

square root of the number of tokens (words) in the text. This struck us as an unusual recommendation since the size of the pieces will become quite large as the size of the parallel corpus grows. However, in a very large corpus, words and their translations are not going to move greater distances, so it seemed to us that employing a fixed piece size might be more effective.

1.4 The Utility of Bilingual Lexicons

Bilingual lexicons are an important resource for humans and automated natural language processing systems alike. When translating technical or specialized text, it can be very important to have a bilingual dictionary that tells if and how certain terms are translated. However, these are the least likely items to be found in a general bilingual lexicon, and the ability to automatically create such resources from parallel text would make it possible to quickly extend a fairly generic bilingual lexicon to a more specialized domain.

Systems that perform Machine Translation (MT) are big potential users of bilingual lexicons as these are one of the most important components of such systems. The quality of the resulting translation greatly depends on the quality of the lexicons. On a very basic level if an MT system does not have an entry for each word of the text to be translated, then even a word to word translation is impossible. For example consider again that the French sentence *Loi Sur Les Instituts De Recherche En Santé Du Canada*. If this is to be translated to English and if the MT lexicon does not have a English translation for the French word *santé*, then the translation can not be carried out completely.

Information Retrieval (IR) systems search and retrieve relevant documents based on a query. Mono-lingual IR systems find the documents only in the language of the query. For example, a query in English that includes the term *AIDS* in English will not find possibly relevant information in other languages. Thus, there is a need for *cross-language* IR systems which retrieve relevant documents in a language other than the query language. An English-French IR system can take a query in English and find the relevant documents in French and vice-versa. For instance consider the query *AIDS* in English again. The English–French IR system will translate the English word *AIDS* to *SIDA* in French and will retrieve related documents in both the English and French language. To do so requires a bilingual lexicon that includes this translation.

1.5 Evaluation Issues and Observations

To evaluate and compare the performance of the different measures we need to compare the quality of the lexicon produced by each of them. The most accurate method of evaluation would be to have a manually created *gold standard* by which we could compare our automatically derived results. Such a resource has translations of all the words in the parallel text and can be used to compare the lexicon created by each of the measures.

However, manually aligned data of parallel corpora that can serve as a gold standard for evaluation is extremely rare. We have employed the only two generally available sources of such data, which include 250 manually aligned verses from English and French versions of The Bible and 500 sentences of the English and French translations of the Canadian Hansard's. The Bible verses come to us from the Blinker project [12] and the Hansard's data from Franz Och [14]. Both the Blinker and Hansard's data are manually aligned, where human beings have gone through the text word by word and determined which words are translations of each other. This provides a reliable source of data for carrying out this evaluation. Please note that we can not simply use an existing bilingual dictionary for evaluation purposes, since such a dictionary does not include proper nouns, morphological variants (such as different tenses of verbs or number of nouns), and may lack certain specialized terminology as well.

1.6 Summary

In this thesis we evaluate a variety of measures of association when applied to the problem of finding word translations in parallel corpora. To our surprise we found that none of the measures did substantially better than any other others. We varied the way in which the piece size of the K-vec method is determined, and found significant improvements with our new approach. We also developed an ensemble technique that combines the output of different tests and produces decisions about word translations based on taking a vote among the different tests. We find that the ensembles are able to result in highly precise determinations of word translations.

2 Background Concepts

Measures of association are used to determine if two events are dependent on one another or not. In the case of finding word translations in parallel text, the two events under consideration are associated with a word in one language occurring in approximately the same position as another word in a second language translation. From this point forward we will refer to source and target languages to indicate the two different languages. The source language is usually the original language while the target is the language being translated into, although this distinction is not relevant in our case.

2.1 Contingency Table Representation

To measure the dependence of these two events we determine their frequency counts using a simple statistical model. We consider the two events to be represented by binary random variables that simply indicate if a word occurred or not in their corresponding pieces. A word pair can fall into one of the four possible categories, and we can represent the associated count data using a two by two contingency table.

In Table 4, the English word *king* and the French word *roi* are mapped to random variables X and Y respectively. In particular these variables denote the presence or absence of these words in the corresponding pieces of the parallel text.

This is nearly identical to the representation employed by Gale and Church, except that rather than considering if two words occur together in corresponding sentences or not, we consider if they occur in corresponding pieces.

Table 4: Example Contingency Table

		Y		
		roi	$\neg roi$	$total$
X	$king$	$n_{11} = 12$	$n_{12} = 1$	$n_{1+} = 13$
	$\neg king$	$n_{21} = 2$	$n_{22} = 45$	$n_{2+} = 47$
	$total$	$n_{+1} = 14$	$n_{+2} = 46$	$n_{++} = 60$

- n_{11} - the number of times *king* and *roi* occur in the corresponding pieces.
- n_{12} - the number of times *king* occurs in a piece and *roi* does not occur in the corresponding piece.
- n_{21} - the number of times *roi* occurs in a piece and *king* does not occur in the corresponding piece.
- n_{22} - the number of times neither *king* nor *roi* occur in the corresponding pieces.

Above, the four values of n_{ij} represent the joint frequency distribution of the two random variables X and Y. The total frequency counts of the each row and column are called the marginal frequency of X and Y. The row marginal distribution is represented by n_{i+} and the column marginal total distribution is represented by n_{+j}.

Formally speaking, two events X and Y are independent if their joint probability is the same as the product of their unconditional probabilities. This indicates that the two events are just as likely to occur together as they are to occur separately. In particular,

$$prob(X, Y) = prob(X) \times prob(Y) \tag{1}$$

Given the contingency table above we consider the two events to be the occurrence of the two words in the corresponding pieces of the parallel text. The two events will be considered to occur together when they both occur in a corresponding piece of text. Thus, the probability of the two events occurring individually can be calculated as their unconditional probabilities:

$$prob(X = king) = \frac{n_{1+}}{n_{++}} = \frac{13}{60} \tag{2}$$

Similarly,

$$prob(Y = roi) = \frac{n_{+1}}{n_{++}} = \frac{14}{60} \tag{3}$$

The probability of both words occurring in the same corresponding pieces is calculated as follows:

$$prob(X = king,\ Y = roi) = \frac{n_{11}}{n_{++}} = \frac{12}{60} \tag{4}$$

From the above calculations we can determine if the probability of the two events occurring together is equal to the product of the probability of the two events occurring separately. If they are equal then we would know that the events are independent and that the words are not likely to be translations of each other (since if two words are translations of each other it is likely that they would occur in corresponding pieces and demonstrate some dependence).

$$\begin{aligned} \frac{12}{60} &\geq \frac{13}{60} \times \frac{14}{60} \\ prob(X = king,\ Y = roi) &\neq prob(X = king) \times prob(Y = roi) \end{aligned}$$

However, since the joint probability of the *king* and *roi* occurring together is more than the product of their individual unconditional probabilities we can conclude that the words *king* and *roi* are not independent. However, to determine if they are dependent we must resort of more systematic methods and employ a measure of association. In general most of the measures that we will discuss are based on comparisons of the joint frequency of events with the individual occurrence.

The values n_{11}, n_{12}, n_{21} and n_{22} of the contingency Table 4 are called the observed values. These are based on frequency counts taken from a sample of parallel text. Given these observed values, it is possible to estimate what values would be expected if the two events under consideration were independent.

The expected value m_{ij} is calculated based on the assumption that the two events are independent. Therefore,

$$\begin{aligned} prob(X, Y) &= prob(X) \times prob(Y) \\ \frac{m_{ij}}{n_{++}} &= \frac{n_{i+}}{n_{++}} \times \frac{n_{+j}}{n_{++}} \\ m_{ij} &= \frac{n_{i+} \times n_{+j}}{n_{++}} \end{aligned} \tag{5}$$

- n_{i+} - is the marginal total for the i^{th} row of the contingency table.
- n_{+j} - is the marginal total for the j^{th} column of the contingency table.
- m_{ij} - is the expected value for the corresponding cell.

Table 5: Contingency Table with Expected Values

		Y: roi	Y: $\neg roi$	$total$
X	$king$	$m_{11} = \frac{13 \times 14}{60}$	$m_{12} = \frac{13 \times 46}{60}$	$n_{1+} = 13$
	$\neg king$	$m_{21} = \frac{47 \times 14}{60}$	$m_{22} = \frac{13 \times 46}{60}$	$n_{2+} = 47$
	$total$	$n_{+1} = 14$	$n_{+2} = 46$	$n_{++} = 60$

For example, the expected value m_{11} for the Table 4 can be calculated as follows:

$$\begin{aligned} m_{11} &= \frac{n_{1+} \times n_{+1}}{n_{++}} \\ &= \frac{13 \times 14}{60} \end{aligned}$$

Similarly, we can calculate the expected value for the cells n_{12}, n_{21} and n_{22}. The contingency table after calculating the expected values for all the cells of Table 4 is shown in Table 5

2.2 Measures of Association

There are a number of measures that produce a raw score that measures the deviation of the observed data from some point of comparison, most often a model of independence as represented by the product of the probabilities of two individual events. The measures we study include Pointwise Mutual Information (PMI), the Dice Coefficient and the Odds Ratio. Each of these tests produce a score on a different scale, so they can not be compared directly with each other. These scores can only be used to compare and rank events that have been assigned a score from the same measure.

2.2.1 Pointwise Mutual Information

Pointwise Mutual Information [2] is the ratio of the probability of the two events X and Y occurring together to the combined probability of the two events occurring independently. This is a direct comparison of what

is observed to what would be expected if the two events were independent. If these probabilities are the same then the two events are independent (and the PMI score will be 0.0).

$$
\begin{aligned}
PMI &= log_2 \frac{\frac{n_{11}}{n_{++}}}{\frac{n_{1+}}{n_{++}}\frac{n_{+1}}{n_{++}}} \\
&= log_2 \frac{n_{11} \times n_{++}}{n_{1+} \times n_{+1}} \\
&= log_2 \frac{n_{11}}{m_{11}} \\
&= log_2 \frac{prob(X = king,\ Y = roi)}{prob(X = king)\ prob(Y = roi)}
\end{aligned}
\tag{6}
$$

If the observed probability is greater than the product of the two unconditional probabilities then evidence for dependence is higher. Since there is no exact point at which we cross the line from independence to dependence the interpretation of this test and most of the others is something of an art.

A PMI value of 1.9841 indicates that observed frequency of the two words occurring together is almost twice the expected frequency of the two words occurring together. We can thus conclude that the words are related and are translations of each other.

2.2.2 Dice Coefficient

The Dice Coefficient [5] is defined as the ratio of twice the frequency of the two events X and Y occurring together to the sum of the total number of times event X occurs and the number of times event Y occurs.

$$
\begin{aligned}
Dice &= \frac{2 \times n_{11}}{n_{+1} + n_{1+}} \\
&= \frac{2\ freq(X = king,\ Y = roi)}{freq(X = king) + \ freq(Y = roi)}
\end{aligned}
\tag{7}
$$

The Dice Coefficient only depends on the frequencies of the events occurring together and their individual frequencies and does not depend on the sample size, which distinguishes it from Pointwise Mutual Information. As such, the Dice Coefficient does not use the value in cell n_{22}. This is interesting because n_{22} gives the number of times neither of the words occur in the corresponding pieces. This value therefore does not give much information as to whether two words are related and should not play a major role in deciding whether or not the words are translations of each other. We will show how the value of n_{22} affects our experiments using an example later in the comparison of these measures.

Suppose that the sample size is 1 and that we remove the log_2 from the Pointwise Mutual Information formula and that we also remove the constant 2 from the Dice Coefficient formula. Then the only difference is that the Dice Coefficient takes the sum of the marginal totals n_{1+} and n_{+1} in the denominator, while Pointwise Mutual Information uses the product. Of course the product increases more rapidly than the sum, so as these counts grow the resulting score for Pointwise Mutual Information will drop more quickly than does the Dice Coefficient.

The values of the Dice Coefficient fall between 0 and 1. Values close to 1 indicate that the two variables are dependent and values close to 0 indicate that the variables are independent.

The Dice Coefficient value for the above example is as follows :

$$\begin{aligned} Dice &= \frac{(2)(12)}{(12+14)} \\ &= 0.8889 \end{aligned}$$

A Dice Coefficient value of 0.8889 is relatively high and may indicate that the words are related and form a good translation pair.

2.2.3 Odds Ratio

The Odds Ratio [3] is defined as the ratio of the total number of times the event of interest takes place to the total number of times it does not take place. In the case of our running example of *king* and *roi*, the value of the odds ratio is calculated as follows:

$$OR = \frac{n_{11}\ n_{22}}{n_{12}\ n_{21}} \tag{8}$$

$$= \frac{freq(X = king\ Y = roi) \times freq(X = \neg king\ Y = \neg roi)}{freq(X = king\ Y = \neg roi) \times freq(X = \neg king\ Y = roi)}$$

The value of the Odds Ratio is always greater than or equal to 0. Values less than 1 indicates that the two events occur together just by chance. Values greater than 1 indicate that the events occur together more often than just by chance and may be dependent. In general, if the numerator of the Odds Ratio is greater than denominator, then the events may be considered interesting.

The Odds Ratio makes use of values in all the four cells of the contingency table and thus uses the information contained in cell n_{22}. The Odds Ratio is symmetric, meaning that one could exchange the values in n_{11} and n_{22} and not affect the resulting Odds Ratio value.

$$OR = \frac{(12)(45)}{(1)(2)}$$
$$= 270$$

An Odds Ratio value of 270 indicates that the events are taking place together more often than would be expected by chance, and that the word pair is likely to form a good translation pair.

2.2.4 Discussion

To differentiate between these measures, we consider how each will score in several particular scenarios that we illustrate below.

In Case 1, the contingency table has values $n_{11} = 1, n_{1+} = 1, n_{+1} = 1$ and the sample size n_{++} = 60 as shown in the Table 6.

In Case 2, the sample size is also 60, but n_{11} = 5, n_{1+}=5, n_{+1}=5 as shown in the Table 7.

In both the cases, the words *king* and *roi* only occur in corresponding pieces in English and French. In other words, neither the English nor the French words occurs in a piece without the corresponding translation in

Table 6: Contingency Table:Case 1

		Y: roi	Y: $\neg roi$	$total$
X	$king$	$n_{11} = 1$	$n_{12} = 0$	$n_{1+} = 1$
	$\neg king$	$n_{21} = 0$	$n_{22} = 59$	$n_{2+} = 59$
	$total$	$n_{+1} = 1$	$n_{+2} = 59$	$n_{++} = 60$

the other.

For purposes of finding translations we would like a measure to give a higher value for Case 2 because the two words we are considering as possible translations occur together more often and we have greater evidence that they are translations of each other than we do in Case 1.

The Pointwise Mutual Information value for the first case (5.9069) is higher than the Pointwise Mutual Information value for the second case (3.5850). In both these cases $n_{11} = n_{1+} = n_{+1}$. In Case 1 $n_{11} = n_{1+} = n_{+1} = 1$ and in Case 2 $n_{11} = n_{1+} = n_{+1} = 5$. We note that the Pointwise Mutual Information value decreases as n_{11} increases which is just opposite to what we want.

The formulation of the Dice Coefficient, on the other hand, assigns same score, namely 1 for both the cases which is at least better than assigning higher score to Case 1.

The Odds Ratio has higher value for Case 2. The Odds Ratio handles the case of $n_{11} = n_{1+} = n_{+1}$ in a slightly different fashion. For the Odds Ratio the value increases as n_{11} increases until $n_{11} \leq \frac{n_{++}}{2}$. For $n_{11} > \frac{n_{++}}{2}$, the test value decreases until $n_{11} = n_{++}$. As desired, the OR value 275 for the second case ($n_{11} = 5$) is higher than OR value of 59 for first case ($n_{11} = 1$).

Next we check whether these measures are sensitive to the value in cell n_{22}. This cell has the value of the number of times neither of the words occurs in corresponding pieces. For our purpose as the value in cell n_{11} increases there is more evidence for the two words being related, whereas the values in the cells $n_{12}\ and\ n_{21}$ are the ones that increase the evidence of the two events being independent, which means that the two words under consideration are not likely to be translations of each other. If two words are not

Table 7: Contingency Table:Case 2

		Y: roi	Y: $\neg roi$	$total$
X	$king$	$n_{11} = 5$	$n_{12} = 0$	$n_{1+} = 5$
X	$\neg king$	$n_{21} = 0$	$n_{22} = 55$	$n_{2+} = 55$
	$total$	$n_{+1} = 5$	$n_{+2} = 55$	$n_{++} = 60$

translations of each other, we would expect that they would often occur in pieces that did not correspond to each other.

The value in the cell n_{22} is the number of times neither of the two words under consideration as possible translations occurs in a corresponding piece. This actually does not give much information about the two words being related and therefore should not play a major role in deciding whether or not the two words are dependent.

For example, consider a case where a contingency table has values n_{11} = 1, n_{22} = 59 and the sample size n_{++} remains 60. Consider a second case where the sample size is same but n_{11} = 59, n_{22}=1. For our purpose we would like to have the test value to be higher for the second case. Again this is because the words occur together more times in the second case and thus provides more evidence for the two events to be related.

The Odds Ratio is a symmetrical measure, so interchanging the values of n_{11} and n_{22} does not affect the resulting score and thus assigns same score to both the cases.

The Dice Coefficient on the other hand gives a higher value to the second case. This is because the Dice Coefficient does not use the value in cell n_{22} whereas the Odds Ratio does. To see the effect of the value n_{22} on Pointwise Mutual Information consider another simple example where we have n_{11} = 4, $n_{+1} = 5, n_{1+} = 5$ and sample size n_{++} = 10. The value of n_{22} in this case is 4 and PMI value is 0.2360. Now if we increase n_{22} by 1, the marginal totals n_{1+} and n_{+1} decrease and thus increase the PMI value to 0.9069, which is significantly higher than the initial value. Thus increasing the value in the cell n_{22} increases the PMI value.

From the discussion above we conclude that Pointwise Mutual Information has a number of drawbacks that make it unsuitable finding word translations in parallel text. The Odds Ratio and the Dice Coefficient may be more suitable, but still have a few potential problems.

2.3 Tests of Association

There are also statistical tests of association that produce a score that can be assigned a value of statistical significance. These different tests can be compared directly, since the values of statistical significance are probabilities and indicate how likely it would be to draw a sample that adheres to the observed data given that a the hypothesized model of independence is true.

The tests of association that we consider are the Log-likelihood Ratio, Fisher's Exact Test, the T-score, and Pearson's Chi-squared Test. As the value of these tests increases, so does the probability that the two events are dependent. Thus, the higher the score the more likely it is that the two words under consideration are translations of each other.

However, all of these tests presume that the data in the contingency table under consideration will have certain characteristics. Even though we are dealing with large amounts of text, we do not find a large number of occurrences of any combination of word pairs terribly often. This is due to Zipf's Law [16], which seems to hold true for most problems in Natural Language Processing, including this one.

In general Zipf's Law holds that most events occur very rarely and only a few events occur most of the times. This is certainly true of the occurrence of individual words and pairs of words in text. Most words occur fairly rarely in a large corpus of text, and very few words occur with high frequency. As a result our frequency counts will often have a distribution that is skewed towards unobserved events (n_{22} in our tables). Given the relatively small sample sizes present in our experiments this is not an extreme concern, but it is enough to motivate the use of Fisher's Exact Test [15], which does not rely on any underlying assumptions about the data in the contingency table.

2.3.1 Log-likelihood Ratio and Pearson's Chi-square tests

Pearson's Chi-square test and the Log-likelihood Ratio [4] test measure the difference between the observed values and the expected values.

The Log-likelihood Ratio is defined as the sum of the ratio of the observed values and expected values, and is computed as follows:

$$\begin{aligned} G^2 &= 2\sum_{ij} log\frac{n_{ij}}{m_{ij}} \\ &= 2(log\frac{12}{3.03} + log\frac{1}{9.97} + log\frac{2}{10.97} + log\frac{45}{36.03}) \\ &= 41.6001 \end{aligned} \tag{9}$$

Pearson's Chi-square test is defined as the sum of the difference between the observed values and the expected values, and is computed as follows using the data from Table 4:

$$\begin{aligned} \chi &= \sum_{ij} \frac{(n_{ij} - m_{ij})^2}{m_{ij}} \\ &= (\frac{(12 - 3.03)^2}{3.03} + \frac{(1 - 9.97)^2}{9.97} + \frac{(2 - 10.97)^2}{10.97} + \frac{(45 - 36.03)^2}{36.03}) \\ &= 43.1356 \end{aligned} \tag{10}$$

The Log-likelihood Ratio and the Pearson's Chi-square test values calculated for the above example are 41.6001 and 43.1356 respectively. These values can then be assigned statistical significance based on the chi–squared distribution with 1 degree of freedom. However, we do not assign significance to the scores we compute since we are simply ranking possible translation pairs and the rankings remain the same with the raw score of values of statistical significance. If the underlying distributional assumptions are met, then these two tests should produce fairly similar scores.

2.3.2 Fisher's exact test

Fisher's Exact test [15] calculates the exact probability distribution of two random variables and does not appeal to an underlying distribution as does the Log-likelihood ratio or Pearson's chi-square test. Fisher's

Exact Test calculates the probability for every possible table that adheres to fixed marginal totals and a sample size that are based on the observed data. In particular, it generates all of the tables by varying the values of n_{11} when n_{1+}, n_{+1} and n_{++} are fixed to their observed values. This corresponds to the hypergeometric distribution of these random variables. For each possible table that is generated the probability of observing that table is calculated as follows:

$$P = \frac{(n_{1+})!(n_{2+})!(n_{+1})!(n_{+2})!}{(n_{11})!(n12)!(n_{21})!(n_{22})!(n_{++})!} \tag{11}$$

Thus, the larger the value of P, the greater the dependence between the two random variables. The values of statistical significance assigned by any test of association can either one side or two sided. A one sided test may be either right or left sided. In our case this distinction is only relevant for Fisher's Exact Test, since this is the only test for which we assign significance (we use the raw scores computed by the Log-likelihood Ratio and Pearson's chi-square test).

For Fisher's Exact Test, a right sided test is calculated by adding the probabilities of all the possible two by two contingency tables formed by fixing the marginal totals and changing the value of n_{11} to greater than or equal to the given value. A right sided Fisher's Exact Test tells us how likely it is to randomly sample a table where n_{11} is greater than observed. In other words, it tells us how likely it is to sample an observation where the two words are more dependent than currently observed. The right sided Fisher's Exact Test value calculated for our example is 8.1287e-10. The low probability means that the chance of the two words being more dependent than observed is negligible and the words are very likely to be translations of each other.

A left sided test is calculated by adding the probabilities of all the possible two by two contingency tables formed by fixing the marginal totals and changing the value of n_{11} to less than or equal to the given value. A left sided Fisher's Exact Test tells us how likely it is to randomly sample a table where n_{11} is less than or equal to the observed value. In other words, it tells us how likely it is to find an instance where the two words are more independent than observed. The Left Fisher test value calculated for the our example is 1.0000. The high probability value of 1 means that the chance of the two words being more independent than observed is very low. Thus, the pair of words are related and form a translation pair.

A two sided test is calculated by summing the probabilities of the tables whose probabilities are less than equal to probability of the observed table. A two sided test tells us how probable it is that the two words are more independent than observed. The two sided Fisher's Exact Test value for our example is 2.7095e-12.

The low probability means that the chance of the two words being more related or dependent than observed is negligible and the words are translations of each other.

For all the other tests and measures described so far, a high score suggests that the two words under consideration are dependent. Similarly, a left side Fisher's Exact Test assigns a high probability to word pairs which are more related while both the right Fisher's test and two sided Fisher's test give low score to the words which are more related. We, however, take the inverse of the values obtained both for the right sided Fisher's test and two sided Fisher's test so that the higher scores means more dependence between the words. For our experiments we choose right sided Fisher's tests for reasons that will be described below.

For our purpose we want the word pairs which occur in corresponding pieces and have high scores associated with them. Again consider Case 1 from Table 6 where $n_{11} = 1, n_{1+} = 1\ and\ n_{+1} = 1$. Also consider Case 2 from Table 7 where $n_{11} = 5$, $n_{1+} = 5$ and $n_{+1} = 20$. In both the cases assume that the sample size is 60.

The left sided Fisher's Exact Test gives the same score to both the cases, namely 1.0. A two sided test gives almost similar scores for both the cases and does not differentiate clearly between the two cases. However, the right sided Fisher's Exact Test, Pearson's Chi-square test, the Log-likelihood Ratio and the T-score do make this distinction, which leads us to use a right sided Fisher's Exact Test.

2.3.3 T-score

The T-score [2] is defined as a ratio of difference between the observed and the expected mean to the variance of the sample. Note that this is a variant of the standard t-test that was proposed for use in the identification of collocations in large samples of text.

$$t = \frac{\bar{x} - \bar{\mu}}{\sqrt{\frac{s^2}{N}}} \tag{12}$$

Here, $\bar{x}$ is the observed sample mean, $\bar{\mu}$ is the expected sample mean and s^2 is the variance of the sample. In our experiment the data is sampled in such way that we just record the presence or absence of the two words in the corresponding pairs. The observed sample mean in our case thus is the ratio of the number of pieces in which the words occur together to the total number of pieces (sample size). In our case $\bar{x}$ thus is $\frac{n_{11}}{n_{++}}$. The expected sample mean is calculated based on the assumption that the two words are independent.

For our experiments $\bar{\mu}$ thus is $(\frac{n_{1+}}{n_{++}})$ $(\frac{n_{+1}}{n_{++}})$

T-score when used with contingency tables which has counts and proportions the formulation computed above is formulated as:

$$\begin{aligned} T-score &= \frac{\frac{n_{11}}{n_{++}} - (\frac{n_{1+}}{n_{++}})(\frac{n_{+1}}{n_{++}})}{\sqrt{(\frac{1}{n_{++}})(\frac{n_{11}}{n_{++}})}} \\ &= \frac{\frac{1}{n_{++}}(n_{11} - \frac{n_{1+}n_{+1}}{n_{++}})}{n_{++} \times \sqrt{n_{11}}} \\ &= \frac{n_{11} - m_{11}}{\sqrt{n_{11}}} \end{aligned} \tag{13}$$

For the example above

$$\begin{aligned} T-score &= \frac{12 - 3.03}{\sqrt{12}} \\ &= 2.5885 \end{aligned}$$

Like Pearson's Chi-square test and the Log-likelihood Ratio, the t-score can be assigned values of statistical significance. However, instead of basing these on the chi-square distribution, these values are obtained from the t-distribution.

2.3.4 Discussion

There are always at least a few high frequency words in both languages of a parallel text. Each of these words occur in almost all the pieces when the text is divided into different pieces. K-vec forms word pairs using the words in the corresponding pieces. A large number of pairs are thus formed using these high frequency words and only a few of them are correct translations. Therefore we want all the tests to assign low scores all for the cases where n_{11} is approximately equal to the sample size n_{++}.

We take the inverse of the scores for a right sided Fisher's Exact Test for our purposes and we get low scores for all such cases. For such cases right sided Fisher' tests have high scores only for the cases where n_{11} is close to half the sample size. The T-score and the Log-likelihood Ratio also have low score for such cases. The T-score has a slight advantage over the right sided Fisher's Exact test and the Log-likelihood Ratio

because it is not symmetric. It thus can differentiate between the cases where we interchange the values of n_{11} and n_{22} keeping the values of n_{1+}, n_{+1} and n_{++} constant.

An example of such a case where we want these tests to differentiate is n_{11} = 40, n_{22} = 20 and sample size n_{++} is 60. We can interchange the values, n_{11} = 20 and n_{22} = 40, keeping the sample size same. We want the tests to differentiate between these cases as there is difference in the evidence of the two words being related in these cases.

Comparing the Log-likelihood Ratio, the T-score and the right sided Fisher's Exact Test for Case 1 as in Table 6 and for Case 2 as in Table 7, we observe that all the tests, as desired, have a high score for Case 2. The T-score and the Log-likelihood Ratio make a clear distinction between these two cases while with the right sided Fisher's Exact test is not so clear. For the second case the right Fisher's test has a score of 1.000 while for the first case it is 0.9833.

We will employ all the previously mentioned tests and measures in this thesis to determine if there is a test which is more suitable than the T-score as proposed by Fung and Church. We consider this a reasonable issue to pursue since the T-score was designed originally for identifying collocations in large samples of text, where the sample size is very large as compared to the sample size in our experiments.

2.4 Evaluation

Evaluating the performance of a system is an important task in any field. In the case of our experiments we would like know what percent of the word pairs selected by a measure of association are correct and what percent of the correct word pairs are selected by it. This information can be obtained by precision and recall measures respectively which are well established ideas in Information Retrieval and are also frequently used in Natural Language Processing.

Our discussion is based on that of Manning and Schutze [11], which is a standard textbook in Statistical Natural Language Processing. We will illustrate these measures via a simple example.

Consider a case where we have 100 word pairs that we know to be translations of each other. We shall call this the gold standard data since it consists of all the correct word translations. Suppose that an algorithm selects 200 word pairs as translations, and that 40 of them are correct (based on the gold standard data).

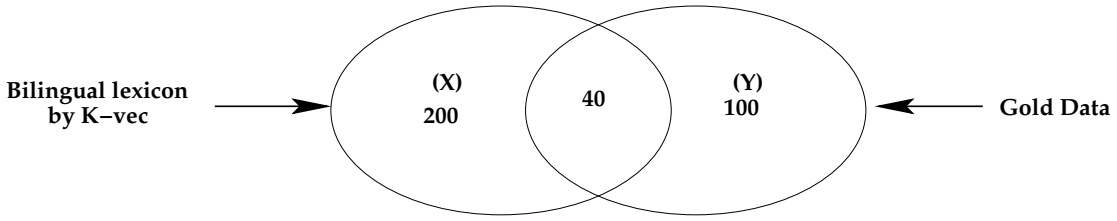

Figure 1: Figure for precision and recall measures. X represents the total word pairs found by a test/measure of association and Y represents the total number of correct word pairs (gold data)

Precision is defined as:

$$precision = \frac{|X \bigcap Y|}{|X|} \tag{14}$$

where,

- X is the set of all the word translations selected by an algorithm and
- Y is the set of all the correct word translations (gold standard data).

Alternatively, precision can be defined as the ratio of the total number of correct word pairs selected by a test/measure to the total number of word pairs found by the test/measure. That is,

$$precision = \frac{C_s}{T_s} \tag{15}$$

where,

- C_s is the total number of correct word pairs found by a test/measure.
- T_s is the total number of word pairs found by the test/measure.

For the example above,

$$precision = \frac{40}{200}$$

Recall is defined as

$$recall = \frac{|X \bigcap Y|}{|Y|} \tag{16}$$

where X and Y are defined as above.

Alternatively, recall can be defined as the ratio of the total number of correct word pairs found by a test/measure to the total number of word pairs present in gold standard data. That is,

$$recall = \frac{C_s}{T_g} \tag{17}$$

where C_s and T_g are defined as above.

Thus, for the example above recall is computed as follows:

$$recall = \frac{40}{100}$$

2.4.1 F-measure

Neither precision nor recall gives a sense of the overall performance of an entire system. One can get a higher recall value at the expense of low precision value and vice-versa. Consider the example above and consider a case where the algorithm finds only one word pair and it is correct. The the precision value is 1 while recall is very low (1/100). Then consider an example where the algorithm finds 300 word pairs and 100 of them are correct. The system thus finds all the correct word pairs and has recall value of 1 but the precision value is low (100/300). We thus achieve a higher recall value at the expense of the precision value. Precision and recall are combined together to give a single measure called as F-measure [11], used to measure the overall performance. It is defined as

$$F = \frac{2PR}{P + R} \tag{18}$$

Here, P and R are precision and recall values respectively.

The information in the Figure 1 can also be presented using a two by two contingency Table as shown in Table 8 where,

- n_{11} is the number of translations found by an algorithm that are also present in gold data (true positives),

Table 8: Contingency Table for precision-recall example

		Y: yes	Y: no	$total$
X	yes	$n_{11} = 40$	$n_{12} = 160$	$n_{1+} = 200$
X	no	$n_{21} = 60$	$n_{22} = 0$	$n_{2+} = 60$
	$total$	$n_{+1} = 100$	$n_{+2}160$	$n_{++} = 260$

- n_{12} is the number of translations found by an algorithm that are not present in the gold data (false positives),
- n_{21} is the number of translations not found by an algorithm but that are present in the gold data. (false negatives), and
- n_{22} is the number of translations neither found by an algorithm nor present in the gold data. This value will be zero for our experiments (true negatives).

It should be noted that the F-measure is in fact equivalent to the Dice Coefficient:

$$\begin{aligned} F &= \frac{2PR}{P+R} \\ &= \frac{2\frac{n_{11}}{n_{1+}}\frac{n_{11}}{n_{+1}}}{\frac{n_{11}}{n_{1+}}+\frac{n_{11}}{n_{+1}}} \\ &= \frac{2n_{11}}{n_{1+}+n_{+1}} \end{aligned} \tag{19}$$

3 The K-vec Algorithm

In this section we describe how the K-vec algorithm [7] determines if two words are translations of each other. This process is applied to every pair of words that occur within some specified number of corresponding pieces. The algorithm proceeds as follows:

1. Divide the two parallel texts into an equal number of pieces. A piece is defined as a part of the text containing a certain number of words.

2. The distinct words in a text are referred to as *word types*. For each word type in both the source and target texts, K-vec creates a k-dimensional binary vector, where k is the number of pieces. Value 1 in the binary vector represents that there is at least one occurrence of the word in the piece. Value 0 indicates that the word does not occur in the piece. Note that K-vec does not consider the frequency of the words in a piece.

3. K-vec forms a two by two contingency table for two word types from the two texts using the binary vectors for the words. It thus categorizes the two words depending on whether or not they occur in the corresponding pieces in the parallel text.

4. K-vec then uses a measure of association to find the degree to which the two words are dependent. If the pair of words are judged to be highly dependent, they may be considered to be translations of each other.

Consider an example where we have an English-French parallel text. Let the number of word tokens in the English text be 1000 and the French text be 900. Suppose that the English text is divided into 10 pieces containing 100 words each, then French text is divided into 10 pieces containing 90 words each.

Consider a word *king* from the English text and a word *roi* from the French text. We are interested in determining if these two words are translations of each other. Suppose the word *king* occurs 5 times in piece 2 and 3 times in piece 7, then the binary vector for the word *king* is:

$$V_{king} =< 0, 1, 0, 0, 0, 0, 1, 0, 0, 0 >$$

Table 9: Contingency Table

		Y: Roi	Y: $\neg Roi$	$total$
X	$king$	$n_{11} = 2$	$n_{12} = 1$	$n_{1+} = 3$
X	$\neg king$	$n_{21} = 0$	$n_{22} = 7$	$n_{2+} = 7$
	$total$	$n_{+1} = 2$	$n_{+2} = 8$	$n_{++} = 10$

Similarly suppose the word *roi* occur 7 times in piece 2, once in piece 5 and 4 times in piece 7, then the binary vector for the word *roi* is:

$$V_{Roi} =< 0, 1, 0, 0, 1, 0, 1, 0, 0, 0 >$$

The two by two contingency table created for the words *king* and *Roi* are created using the binary vector for the two words.

Here,

1. n_{11} is the total number of times 1 occurs in the corresponding positions of the vectors of the two words under consideration. It represents the total number of times a piece contains the word *king* and the corresponding piece contains the word *roi*.

2. n_{1+} is the total number of times 1 occurs in the vector for the word *king*. It represents the total number of pieces containing the word *king*.

3. n_{12} is obtained by subtracting $n11$ from n_{1+}. It represents the total number of times a piece contains the word *king* and the corresponding piece does not contain the word *roi*.

4. n_{+1} is the number of times 1 occurs in the vector for the word *roi*. It represents the total number of pieces containing the word *roi*.

5. n_{21} is obtained by subtracting n_{11} from n_{+1}. It represents the total number of times a piece contains the word *roi* and the corresponding piece does not contain the word *king*.

6. n_{2+} and n_{+2} are the total number of times 0 occurs in the vectors for the word *king* and *roi*, respectively. n_{2+} represents the total number of pieces which do not contain the word *king*. n_{+2} represents the total number of pieces which do not contain the word *roi*.

7. n_{22} is obtained by subtracting n_{21} from n_{+2}. It represents the total number of times a piece does not contain the word *king* and the corresponding piece does not contain the word Y.

8. n_{++} is obtained by adding the values of n_{11}, n_{12}, n_{21} and n_{22}. It represents the total number of pieces into which each of the texts is divided.

This methodology is applied to any two words that occur within the specified number of corresponding pieces, and then ranks all of those pairs based on a measure of association that determines the degree of dependence between the words. Those that score highly are considered to be translations of each other.

3.1 The Number of Pieces

If the number of pieces into which the text is to be divided is very large, then total number of words in each piece is small and a word and its translation may not occur in the corresponding pieces. K-vec may miss such translations as it looks for the word translations in corresponding pieces. If the number of pieces is very small, then the number of words in each piece will be large and the basic advantage of dividing the text into pieces and looking for a word and its translation into corresponding piece is lost.

Fung and Church suggest that K-vec divide the text into a number of pieces equal to the square root of the total number of word tokens in the text. For huge data the number of word tokens in each piece will therefore be large. We thus believe that one does not always get best results by dividing the text in pieces equal to the square root of the total number of words in the text and evaluate this in our experimental work.

3.2 Determining Candidate Translations

Fung and Church do not consider all the word pairs from the two texts as possible or candidate translations because there will be too many such word pairs. They restrict the algorithm to word pairs with frequencies between 3 and 11. They do not consider the low frequency word pairs because the amount of information about these words is not sufficient to find a translation. The words that make high frequency word pairs in

both languages in the parallel text will occur in almost every piece. K-vec looks for word correspondences in corresponding pieces and therefore every high frequency word in one language will be considered as a translation of the high frequency words in another language.

For example, if there are *n* high frequency words in one language and *m* high frequency words in another language, then there will be almost *nm* word pairs formed using these words and only *n* of these can be correct. Measures of association will be unable to differentiate such words and do not consider them for translation. While the idea of a frequency cutoff is sound, we do not believe that an upper limit of 11 (used by Fung and Church) is always the correct choice, but should rather be dependent on the number of pieces.

4 Experimental Data

4.1 Introduction

The parallel text used for the experiments in this thesis are the Blinker data [13] and the Hansard's data [14]. The Blinker data consists of English-French version of The Bible, while the Hansard's data consists of English-French texts from the proceedings of the Canadian parliament.

Both the Blinker data and Hansard's data are *manually aligned*, meaning that human beings have manually determined which words are translations of each other. Manual alignment of this type of data is a very time consuming task and is very rare. These are the only two sources of manually aligned parallel text that we are aware of, and are the only two sources that are commonly cited in the literature.

The main difference between the two sets of data is that the texts in the Hansard's data are actual translations of each other while the texts in the Blinker data are obtained from different versions of The Bible, which are not necessarily translations of each other but are certainly parallel text.

We take the manual alignments from this data and build from that a gold standard bilingual lexicon for both the Blinker and Hansard's data. This gold standard data has translations of all the words and phrases in the context in which they are used in the parallel text. We use this gold standard data to evaluate the results of our different experiments.

We do not use standard bilingual dictionaries for evaluating the translation pairs we identify because the gold standard data includes translations of proper nouns and morphological variants. For example, there may be translations for both *ran* and *running* in lexicons derived from parallel text, since those are types that occur in the parallel text. In a standard bilingual dictionary these forms might not appear but would instead be represented by the infinitive form *run*.

The following is a detailed description of both the Blinker and Hansard's data.

4.2 Blinker data

The main objective of the Blinker project [13] was to create a gold standard lexicon that can be used to evaluate the lexicon created by different algorithms for compiling bilingual lexicons. The first and the

foremost requirement for creating a gold standard lexicon is that it be based on actual parallel text. The Blinker project is based on an English-French version of The Bible because the text is freely available and in the public domain.

According to Melamed [12], out of the 66 books for The Bible, Ecclesiastes, Hosea and Job are not well understood and have inconsistent translations and therefore were not included in the Blinker project data. The remaining 63 books comprise 29614 verses. Out of these verses 250 verses were selected to be manually aligned. Manually aligning the entire Bible is simply too time consuming, so only a portion was carried out. The following is the procedure used to select the verses:

1. Pre-process both the English and the French verses and tokenize the punctuations from the word they are adjacent to and also separate the hyphenated words. The aim was to create multiple words from such words. The resulting parallel text had 814451 *tokens* (total number of words) and 14817 *types* (total number of distinct words) in the English half and 896717 tokens and 21372 types in the French half.

2. Count the frequency of each word type in the verses.

3. Randomly select 25 word types for types with frequencies one, two, three, and four occurrences (for a total of 100 types).

4. A verse is selected if any one of the 25 selected word types occurs in it. If a selected verse has more than one occurrence of the same word type, then it is replaced by another word type of that frequency. Also, if two different word types occur in the same selected verse, then the word type with lower frequency is replaced by another word type of that frequency. The aim was to select only one verse for each occurrence of a word type.

5. Repeat step 4 till the total number verses selected is equal to $(1 + 2 + 3 + 4) \times 25 = 250$. For each word type, the number of verses selected is equal to its frequency.

These 250 verses in their English and French version together form the parallel text for the Blinker data. It has 7510 tokens in the English half, 8191 token in the French half and 1714 and 1912 types respectively.

4.2.1 Description of the Annotation Files

In all seven translators were recruited to manually align the selected verses. Since they annotate the data with alignment information, the translators are usually referred to as annotators.

The 250 verses selected above were divided into two parts. Part 1 contained the verses from 1-100 and was annotated by annotators 1, 2, 3, 4 and 5. Part 2 contained the verses from 101-250 and was annotated by annotators 1, 2, 3, 6, 7. Each verse was therefore annotated by five different annotators. For each verse, the annotator had an output file which contained a set of pair of numbers corresponding to position of the word in the English verse and the position of its translation word in the corresponding French verse. Consider an example where words in an English verse are translated to the words in the corresponding French verse by different annotators as shown in the Figure 2. The annotations shown are for verse 12 of the Blinker data.

The output file by the annotator 1 for the above verse looks like

$0 \rightarrow 1,\quad 1 \rightarrow 2,\quad 2 \rightarrow 3,\quad 6 \rightarrow 4,\quad 3 \rightarrow 5,\quad 4 \rightarrow 5,\quad 5 \rightarrow 6,\quad 7 \rightarrow 7,\ 8 \rightarrow 8,\quad 9 \rightarrow 9\quad 10 \rightarrow 10,\ 11 \rightarrow 10,\quad 12 \rightarrow 11,\ 13 \rightarrow 12,\ 14 \rightarrow 13,\quad 14 \rightarrow 14,\quad 15 \rightarrow 15$

For a word at a particular position the number of entries in the output file is equal to the number of words to which it is translated. In the Figure 2 the word at position 14 is translated to the words at positions 13 and 14 and thus has two entries in the output file as follows

$14 \rightarrow 13,\ 14 \rightarrow 14$

The above translation represents an occurrence of a *phrasal translation*, a translation where one or more words in a language are translated to one or more words in another language.

There is an entry in the output file for each occurrence of a word type. For example for the word type *the* occurring at positions 1 and 8 and translated to the same word *les* at positions 2 and 8 respectively, there are two entries in the output file as follows

$1 \rightarrow 2,\ 8 \rightarrow 8$

A word in the English or French verse which has no translation in the corresponding French or English verse respectively, for example the word *alors* in the French verse in Figure 2, is translated as NULL. It is indicated in the figure by having no link for it and is represented in the output file as follows

$0 \rightarrow 1$

The output file thus has an entry for each word position in the English verse.

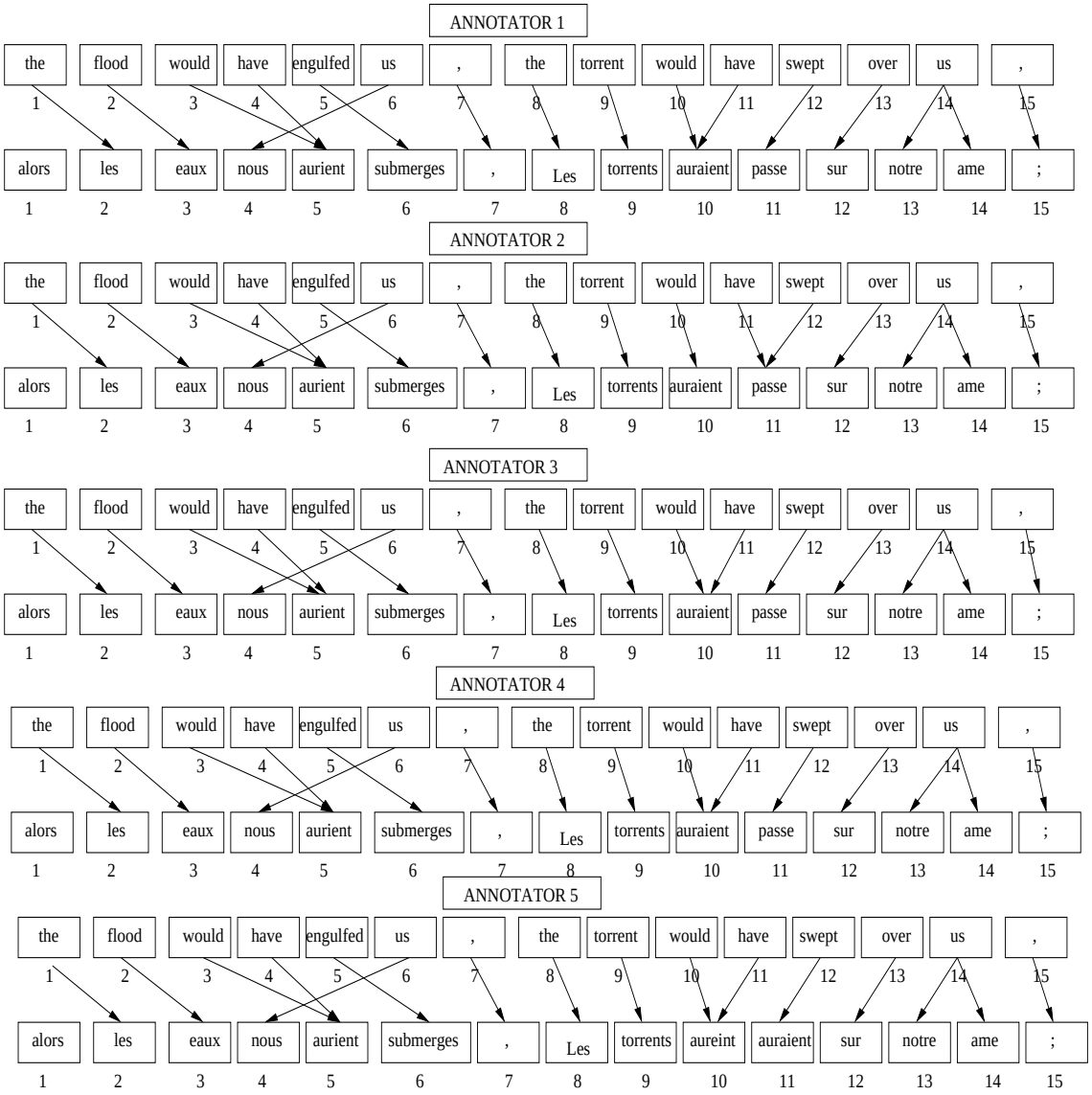

Figure 2: Annotation proposed by 5 different annotators for a verse number 12

Table 10: Annotator Agreement

Number of Annotators	Minimum number that must agree
1	1
2	2
3	2
4	3
5	3

4.2.2 Algorithm for Compiling Gold Standard Lexicon

For this thesis we compiled a gold standard lexicon using only a select subset of the annotators. For verses 1-100 we use annotators 1, 2 and 3 and for verses 101-250 we use annotators 1,3 and 7. This is because for these verses the agreement rate for these annotators was highest. We will explain the steps of the algorithm for one of the verse using 5 annotators (1-5).

Consider the translations proposed by the different annotators for a verse 12 as shown in the Figure 2.

1. For each word position in a verse choose a translation only if some minimum number of annotators agree for it. For different number of annotators the minimum number of annotators that should agree for the translation to be considered valid is as shown in Table 10.

   So, for example, the possible translations suggested for word at position 11 for the verse above by different annotators are as follows

   $11 \rightarrow 10$ (annotator 1) $11 \rightarrow 11$ (annotator 2)

   $11 \rightarrow 10$(annotator 3) $11 \rightarrow 10$ (annotator 4)

   $11 \rightarrow 10$ (annotator 5)

   The translation of the word at position 11 in the English verse is chosen to be the word at position 10 in the corresponding French verse because four out of five annotators proposed it. We ignore all translations where the number of annotators that agree is less than minimum required.

2. After taking only the translations for which minimum number of annotators agree, if a word at a position in the English verse is translated to the words at different positions in the French verse, then they are concatenated to form a single translation pair.

   For example in step 1 the word at position 14 in the English verse is translated to words at positions 13 and 14 because all 5 annotators agree for it. $14 \rightarrow 13$ (all 5 annotators proposed it)
   $14 \rightarrow 14$ (all 5 annotators proposed it)

   In this step such entries are concatenated to form a single entry like $14 \rightarrow 13,\ 14$
   This represents an occurrence of phrasal translations.

3. After step 2 is completed for a verse, if words at different positions are translated to the word/group of words at the same position, then they are concatenated to form a single translation pair. For example if there are translation pairs like
   $10 \rightarrow 10$ (all 5 annotators proposed it)
   $11 \rightarrow 10$ (4 out 5 annotators proposed it)
   are modified to
   $10\ 11 \rightarrow 10$.
   Again this represents an occurrence of a phrasal translation.

   As K-vec does not translate any word to NULL, we do not consider all such entries in the token based gold data. For example, we do not have an entry for the word *alors* which is translated to NULL.
   The final entries for the verse above after step 5 is completed are as follows
   $1 \rightarrow 2,\ 2 \rightarrow 3,\ 6 \rightarrow 4,\ \ 3\ 4 \rightarrow 5,\ 5 \rightarrow 6,\ 7 \rightarrow 7,\ \ 8 \rightarrow 8,\ 9 \rightarrow 9,\ 10\ 11 \rightarrow 10,\ \ 12 \rightarrow 11,\ 13 \rightarrow 12,\ 14 \rightarrow 13\ 14,\ \ 15 \rightarrow 15$

4. The word positions are then replaced by the actual words from the corresponding verses to form the token based gold data. This gold data has 6451 entries. The token based gold data for the sample verse above is follows:
   $the \rightarrow les$
   $flood \rightarrow eaux$
   $us \rightarrow nous$
   $would\ have \rightarrow auraient$

$engulfed \rightarrow submerges$

$, \rightarrow ,$

$the \rightarrow les$

$torrent \rightarrow torrent$

$would\ have \rightarrow auraient$

$swept \rightarrow passe$

$over \rightarrow sur$

$us \rightarrow notre\ ame$

$, \rightarrow ;$

5. Word-type gold data is prepared from the word-token gold data by having a single entry for a word if it is translated to same words/words. For example, if the token based gold lexicon was only made up of the entries above, then for the two entries for the word type *the* are translated to the same word *les*, there is a single entry in the type based gold data. Similarly we have single entry for *would have* translated to *auraient*. But if the two or more entries in token based gold for a word type are translated to different word/words, we keep all of them in the type based gold data. For example, there are 2 entries for **’,’** as it translated to different types **’,’** and **’;’**. The type based gold data is a follows

   $the \rightarrow les$

   $flood \rightarrow eaux$

   $us \rightarrow nous$

   $would\ have \rightarrow auraient$

   $engulfed \rightarrow submerges$

   $, \rightarrow ,$

   $torrent \rightarrow torrent$

   $swept \rightarrow passe$

   $over \rightarrow sur$

   $us \rightarrow notre\ ame$

   $, \rightarrow ;$

   The resulting type based gold data has 2711 entries. The type based gold which we prepare has phrasal

Table 11: Agreement Rates for Verses 1-100 and Annotators 1, 3, 5

Annotator groups	Number of translations
All 3	2139
2 1	652
1 1 1	163

Table 12: Annotator Agreement Rates for Verses 101-250 and Annotators 1, 3, 7

Annotator groups	Number of translations
All 3	3154
2 1	1134
1 1 1	269

translations in it. We make this gold data available but as K-vec does not find phrasal translation we filter those out from the data. The gold data without phrasal translation has 1639 entries. This gold data again has entries for words which are not being translated i.e. translated to NULL. The gold data without NULL entries has 1467 translations.

4.2.3 Details of the data

Table 11 shows the number of annotators who agreed on the translation of a particular word and the number of annotators who did not agree. This data is for verses 1-100 and annotators 1,3, 5.

Table 12 shows the number of annotators who agreed on translation of a particular word the number of annotators who did not agree. This data is for verses 101-250 and annotators 1,3, 7.

Table 14 and 15 gives frequency distribution of word tokens in the English and French text respectively.

Table 13: Number of Entries for each Word Type

Token based gold data	6451
Type based gold data	2711
Without Phrasal translations	1639
Phrasal translations	1072

Table 14: English Frequency Distribution

Range	Count	Range	Count
1-3	1368	4-10	172
11-20	35	21-30	12
31-40	9	41-100	22
101-150	2	273	1
311	1	316	1
418	1	560	1

Table 15: French Frequency Distribution

Range	Count	Range	Count
1-3	1580	4-10	159
11-20	33	21-30	15
31-40	7	41-100	19
101-150	1	150-200	2
250-300	2	308	1
388	1	575	1
619	1	-	-

Table 16: Hansard Alignment Format Example (SENT:401)

Alignment Type	English word position	French word position
S	0	0
S	2	1
P	0	0
P	1	1
P	1	2

4.3 Hansard's data

Like the Blinker data, the main objective in creating a manually aligned portion of the Hansard's data was to create a gold standard lexicon that can be used to evaluate the bilingual lexicon created by different algorithms. This manual alignment was carried out as a part of the research described in [14].

Due to the difficulty of manually aligning parallel text, only 500 sentences from Hansard's data were selected randomly. For the Hansard's data if there are n word tokens in a sentence, then the words are numbered from 0 to n-1. Word 0 therefore represents the first word of the sentence. The format of the aligned data is as shown in Table 16. The S and P alignments distinguish between word by word and phrasal alignments. P alignments are phrasal, and we do not utilize them in this thesis. Thus, our attention is focused on the S alignments.

The above example shows that in the sentence number 401, the entry "S 0 0" tells us that the word at position 0 in English is translated to the word at position 0 in the French sentence. Similarly the entries "P 1 1" and "P 1 2" tells us that the word at position 1 in the English sentence is translated to the words at position 1 and 2 in the French sentence and represents an occurrence of phrasal translations. As K-vec does not deal with phrasal translations, we only use the S alignments for creating the gold standard lexicon. There are 44,351 tokens in the gold standard lexicon created from aligned version of the Hansard's, and from that 1,528 types are found. This is the number of 'entries' in that lexicon.

Tables 17 and 18 give frequency distributions of words (tokens) in English and French text respectively.

Table 17: English Frequency distribution

Frequency range	Number of words	Frequency range	Number of words
1-3	1553	4-10	180
11-20	54	21-30	12
31-40	12	41-100	14
101-150	3	150-200	3
201-300	3	448	1
474	1	-	-

Table 18: French Frequency distribution

Frequency range	Number of words	Frequency range	Number of words
1-3	1820	4-10	196
11-20	32	21-30	12
31-40	8	41-100	13
101-150	6	150-200	2
201-300	3	449	1
521	1	649	1

5 Experimental Results

Here we discuss the results of several different series of experiments. This inquiry was guided by three main motivations. First, we were interested in exploring some of the decisions made by Fung and Church in their original presentation of the K-vec algorithm. As we alluded to earlier, their recommendations regarding the number of pieces to divide corpora into and the frequency cutoffs do not seem suitable for all situations. Second, we were interested in determining if there were measures of association in addition to the T-score and Pointwise Mutual Information (their recommendations) that might perform well or better. Finally, given the very different characteristics of some of these measures of association, it seemed likely that combining multiple measures in some way might be fruitful.

All of our experiments result in a list of word translations. If we observe for top X word translatoins, the number of word pairs considered are more than X. This suggests that there is more one word pair with same rank in some cases. One reason for this may be that we have a precision of only two digits for the scores assigned by different measures for the word pairs. The list of word transalions found are then compared to lexicons derived from the manually aligned versions of the Blinker data (French–English Bible) and the Hansard's (Proceedings of the Canadian Parliament). In order to study the effect of large and small corpora, we use two small data sets, the Blinker data (250 verses) and the manually aligned portion of the Hansard's data (500 sentences). Both of these data sets have been manually aligned, but we ignore that fact during processing and compare the lexicon we derive from that data as if it were not aligned, to a lexicon that is based on the correct alignments. We also find translations in a 10,000 sentence sample from the Hansard's, and compare that with the same gold standard lexicon as found from the 500 sentence manually aligned portion. Tables 19 and 20 give a detailed description of all three of these data sets. We shall refer to these data sets as Blinker, Hansard's (small), and Hansard's (big).

5.1 Results Based on Fung and Church Settings

This experiment was carried out on all three data sets and attempts to duplicate the exact formulation of the K-vec algorithm as proposed by Fung and Church. The purpose of this experiment is to establish baseline results for all three data sets. Then as we vary some of these settings we can clearly see their impact and reach conclusions as to what might be the optimal formulation.

Table 19: Details of the English Experimental Data

Data Set	Sentences	Tokens	Types
Blinker	250	7510	1629
Hansard's (small)	500	7946	1836
Hansard's (big)	10000	124528	7729

Table 20: Details of the French Experimental Data

Data Set	Sentences	Tokens	Types
Blinker	250	8191	1822
Hansard's (small)	500	8749	2095
Hansard's (big)	10000	135738	9707

The number of pieces into which the text is divided is equal to the square root of the total number of word tokens in the English text. The frequency cutoffs are set to 3 and 11, meaning that any words that occur in less than 3 or more than 11 pieces will be excluded as candidate translations. Only word pairs that occur in between 3 and 11 pieces (inclusive) will be considered as valid translation pairs.

We follow the recommendation of Fung and Church and use the T-score and Pointwise Mutual Information as our measures of association. While Fung and Church prefer the T-score, we felt that Pointwise Mutual Information was sufficiently interesting that it merited inclusion all the same.

The baseline results are shown in Tables 21 and 22.

Table 21: Top 50 Translations, T-score, Fung and Church Settings

Data Set	Total Found	Correct	Precision	Recall
Blinker	177	69	0.39	0.05
Hansard's (small)	137	68	0.50	0.04
Hansard's (big)	274	31	0.11	0.02

Table 22: Top 50 Translations, Pointwise Mutual Information, Fung and Church Settings

Data Set	Total Found	Correct	Precision	Recall
Blinker	303	62	0.20	0.04
Hansard's (small)	287	73	0.25	0.05
Hansard's (big)	462	30	0.06	0.02

From Tables 21 and 22 we observe that the results for Hansard's (small) are better than Blinker data results even though the amount of data is almost same. This may be because the Hansard's data is government language that has been translated in a rather functional way, whereas the Blinker data is Biblical text that is not a direct translation.

We also note that the performance on Hansard's (big) is relatively poor, and suggest that the settings as proposed by Fung and Church may not be well suited for this amount of data. In particular, the piece size is much larger for Hansard's (big). This results in many more candidate translations, and it will be difficult to identify true translations given such a large volume of candidates.

Also, Hansard's (big) is divided into 352 pieces (in the Fung and Church formulation) which makes frequency cutoffs of 3 and 11 rather unreasonable. Most words will occur in more than 11 pieces, and these will all be excluded as candidate translations.

5.2 Varying the Formulation of K-vec

In the previous experiment we found that the settings proposed by Fung and Church do not yield good results when the amount of data is large. In this experiment we propose a different set of settings to improve the results for larger amount of data.

We find that the new settings improve the results even when the amount of data is small. The first setting proposed by Fung and Church that we take issue with is dividing the text into a number of pieces equal to the square root of the total number of word tokens. For larger amounts of data this will result in very large piece sizes. However, words that appear in larger corpora are no more likely than those in smaller volumes of text to move great distances during translation. Thus, increasing the piece size as corpora size increases

Table 23: Top 50 Translations, T-score, Hansard's (big)

Tokens in Piece	Cutoff Range	Total Found	Correct	Precision	Recall
352	3-11	274	31	0.11	0.02
100	3-11	42	21	0.50	0.01
352	3-175	116	50	0.43	0.03
100	3-620	74	49	0.60	0.03

seems like an undesirable quality. Also, given the large number of candidate translations that will result, a measure of association will not be able to differentiate between the many possible translations that may be proposed.

Our proposal is that the piece size should not be dependent on the size of the text being processed, but should rather be some constant value that reflects a reasonably sized unit of text in which a word in a source language and its translation into a target might reasonably be expected to appear. In particular, we believe that approximating a paragraph sized amount of text is a reasonable guideline for piece-sizes. This reflects the belief that while a word might tend to move around in a paragraph as it is translated, it is unlikely to move much further than that. We believe that pieces that consist of 100 tokens roughly capture the same amount of information as is found in a paragraph and use that as our piece size in our New Settings.

As discussed previously, we also believe that the upper frequency cutoff should not be set to a constant of 11 as suggested by Fung and Church. If there are a large number of pieces, then 11 is simply too low and will preclude many reasonable candidate translations from being considered. We propose that the an upper frequency cutoff be set to half the number of pieces, in the belief that words that occur more than this number of times are very likely to be function words that are rather difficult to distinguish among and simply introduce noise into the process.

Table 23 shows the results obtained when we use various combinations of the New Settings with the original Fung and Church settings and with each other.

We observe improvements in the results for Hansard's (big) even when just one of these settings is changed. However the results are best with the New Settings, that is when we have 100 word tokens per piece and

Table 24: Top 50 Translations, T-score, New Settings

Data Set	Total Found	Correct	Precision	Recall
Blinker (Old)	177	69	0.39	0.05
Blinker Data (New)	145	77	0.53	0.05
Hansard's (small) (Old)	137	68	0.50	0.04
Hansard's (small) (New)	140	71	0.51	0.05

frequency cutoff range of 3 to (half the number of pieces).

Table 24 shows improvement over the baseline results for the Fung and Church settings (here referred to as "Old") with the New Settings even for all three data sets. Thus, we believe that our New Settings represent a motivated improvement over the settings suggested originally by Fung and Church.

5.3 Comparing Different Measures

Fung and Church suggest the use of the T-score and Pointwise Mutual Information for this task. In particular they favor the T-score. However, there is a wide range of measures of association available, and we undertook to determine if any others might present advantages over either of these two original alternatives.

We include well known measures such as the Odds Ratio, the Log-likelihood Ratio, the Dice Coefficient, and the right sided Fisher's Exact Test in addition to the T-score and Pointwise Mutual Information. Through this experiment we hope to determine which measure performs the best with respect to identify word translations in parallel text.

We conducted experiments to find the top 25 and top 50 translations using each of our proposed measures for all three data sets. We used our New Settings for reasons discussed in the previous section.

For all three data sets we observe that the performance of T-score and Log-likelihood are the best. This is an interesting finding in that the T-score and the Log–likelihood are seemingly different measures. However, it should be noted that the T-score is in fact one of the terms in Pearson's Chi-square test and as such is equivalent to a *Pearson's residual*. Since the Log-likelihood Ratio and Pearson's chi-squared test are

Table 25: Top 25 Translations, Blinker, New Settings

Measure	Total Found	Correct	Precision	Recall
T-score	30	29	0.97	0.02
Log-likelihood	51	36	0.71	0.02
Odds Ratio	111	53	0.48	0.04
Pointwise Mutual Information	184	57	0.31	0.04
Dice Coefficient	370	113	0.31	0.08
Fisher's Exact (right)	4140	185	0.04	0.13

clearly related, it does seem that the T-score and the Log-likelihood Ratio are in fact related somehow. The significance of this relationship with respect to this problem remains an interesting issue for future work.

The Odds Ratio and Pointwise Mutual Information both performed relatively well as well. However, for the Dice Coefficient the total number of candidate translations selected for the top X translations increases drastically as the value of X grows, and the performance declines very rapidly. Pointwise Mutual Information has some of the same characteristics, but is not quite as dramatic as the Dice Coefficient. Fisher's Exact Test (right sided) is the least useful of these measures since it seems to consider nearly every word pair as a possible translation, leading to too many candidates and very low precision and recall.

The following tables show the results for the top 25 and 50 translations for the respective data sets: Blinker (25 and 26), Hansard's (small) (27 and 28) and Hansard's (big) (29 and 30)

From this experiment we conclude that the T-score, the Log-likelihood Ratio and the Odds Ratio represent the most reliable measures for the task of finding translations. Thus, Fung and Church's recommendation in favor of the T-score is validated. We also note that Pointwise Mutual Information also performs well, and suggest that it merit further attention despite Fung and Church's recommendation in favor of the T-score.

5.4 Ensemble Approaches

An ensemble approach combines the output of various techniques in the hopes of getting better collective results than any of the individual methods. We propose two new methods of creating ensembles that will

Table 26: Top 50 Translations, Blinker, New Settings

Measure	Total Found	Correct	Precision	Recall
T-score	145	77	0.53	0.03
Log-likelihood	123	67	0.54	0.05
Odds Ratio	176	70	0.41	0.05
Pointwise Mutual Information	369	75	0.20	0.05
Dice Coefficient	2588	243	0.09	0.17
Fisher's Exact (right)	7068	237	0.03	0.16

Table 27: Top 25 Translations, Hansard's (small), New Settings

Measure	Total Found	Correct	Precision	Recall
T-score	36	24	0.67	0.02
Log-likelihood	33	21	0.64	0.01
Odds ratio	58	32	0.55	0.02
Pointwise Mutual Information	139	48	0.35	0.03
Dice Coefficient	213	53	0.25	0.03
Fisher's Exact (right)	4458	160	0.04	0.10

Table 28: Top 50 Translations, Hansard's (small), New Settings

Measure	Total Found	Correct	Precision	Recall
T-score	140	71	0.51	0.05
Log-likelihood	78	46	0.59	0.03
Odds ratio	133	53	0.40	0.03
Pointwise Mutual Information	358	72	0.20	0.05
Dice Coefficient	1958	156	0.08	0.10
Fisher's Exact (right)	8518	186	0.02	0.12

Table 29: Top 25 Translations, Hansard's (big), New Settings

Measure	Total Found	Correct	Precision	Recall
T-score	30	25	0.83	0.02
Log-likelihood	25	13	0.52	0.01
Odds ratio	37	10	0.27	0.01
Pointwise Mutual Information	44	10	0.23	0.02
Dice Coefficient	367	23	0.06	0.02
Fisher's Exact (right)	107028	457	0.00	0.30

Table 30: Top 50 Translations, Hansard's (big), New Settings

Measure	Total Found	Correct	Precision	Recall
T-score	74	49	0.66	0.03
Log-likelihood	55	24	0.44	0.02
Odds ratio	74	12	0.16	0.01
Pointwise Mutual Information	109	21	0.19	0.01
Dice Coefficient	2896	123	0.04	0.08
Fisher's Exact (right)	184513	502	0.00	0.33

result in a greater number of correct translations being found by these methods.

First, we have noticed that the results of these tests vary quite a bit as the piece size varies. Thus, we use the same measure and take the union of the translations found for two different piece sizes to create a new set of transaltion pairs.

Second, from our discussion in the Background Chapter it seems clear that the different measures of association have very different characteristics, and in fact find very different sets of translations (a point which will be demonstrated shortly). Since our top 3 measures (the T-score, the Log-likelihood Ratio, and the Odds Ratio) are all highly precise measures, we propose an ensemble approach that takes the union of these three measures to create a new set of translation pairs.

5.4.1 One Measures, Varying Number of Pieces

We conducted experiments on all three data sets which utilize the T-score to find translations by first dividing the data into the number of pieces we propose, i.e., the number of pieces having 100 tokens per piece. Then we repeat the process, this time using a piece size based on having 90 tokens per piece. We find that the when using the T-score that the top 25 translations as found by each measure vary considerably from piece size to piece size, as shown in Table 31.

We note that when creating an ensemble in this fashion (using 90 and 100 tokens per piece) that the number of correct translations found by the T-score increases for all data sets and still maintains the same precision as obtained when using a single approach based on 100 tokens per piece.

From the Table 32 we observe that including the additional piece size variation on the T-score increases the number of translations by 12, and of that 10 of those are correct when considering the top 25 translations.

From the Table 33 we observe that just using one more output in the ensemble method we add 25 more word pairs and out of that 16 are correct for the top 50 ranked word pair experiment.

We therefore conclude the the using ensemble approach and combining the word pairs found by the same test but over a range of pieces helps in improving the results.

Table 31: Top 25 Translations, T-score, Hansard's (big)

90 words per piece	100 words per piece
member - dèputé	unemployment - chômage
government - gouvernement	canada - canada
hon. - dèputé	house - chambre
! - !	minister - ministre
price - prix	you - vous
but - mais	! - !
2 - 2	? - ?
you - vous	member - député
unemployment - chômage	income - revenu
two - deux	members - députés
cent - these - ces	–
–	they - ils
1 - 1	2 - 2
like - voudrais	party - parti
income - revenu	speaker - orateur
per - motion - motion	but - mais
speaker - monsieur	programs - programmes
house - chambre	these - ces
our - notre	opposition - opposition
report - rapport	million - millions
: - :	however - toutefois
provinces - provinces	few - quelques

Table 32: Top 25 Translations, Single T-score (100 tokens) versus Ensemble T-score (90 and 100 tokens)

Experiment	Total Found	Correct	Precision	Recall
Single	30	25	0.83	0.02
Ensemble	42	35	0.83	0.02

Table 33: Top 50 Translations, Single T-score (100 tokens) versus Ensemble T-score (90 and 100 tokens)

Experiment	Total Found	Correct	Precision	Recall
Single	74	49	0.66	0.03
Ensemble	99	65	0.66	0.04

5.4.2 Ensemble of Various Measures

From our previous experiments we have concluded that the T-score, the Log-likelihood Ratio and the Odds Ratio are relatively precise measures that perform better than the other measures we considered. We therefore conducted experiments that create an ensemble by taking the union of the translations proposed by each of these three measures.

As an example of these different characteristics, for the Blinker data we observed that for the top X translations, where X is relatively small (less than 50), the Odds ratio has translations formed using words which occur together in the range of 4-10 times. These are relatively low frequency words pairs. The T-score and the Log-likelihood Ratio find relatively similar translations which are usually high frequency word pairs that occur in more than 10 different pieces. However, the difference between the T-score and the Log-likelihood Ratio is in the number of word pairs found by the tests for the same top X ranked word pairs. The T-score finds a small number of translations as compared to both the Log-likelihood ratio and the Odds ratio. The difference in the translations found by these measures can be seen from the Tables 34, 35, and 36.

Considering the translations found by these three measures, we believe than an ensemble that takes the union of these results should improve performance. We go on to compare the results of these ensembles with the single T-score as shown in Table 37.

Table 34: Top 20 Translations, Odds Ratio, Blinker, New Settings

word pair	Frequency
egypt - égypte	5
or - ou	5
christ - christ	5
nations - nations	5
loathe - dgot	4
derbe - derbe	4
sword - épée	4
third - troisime	4
jesus - jésus	4
faith - foi	4
prophets - prophétes	4
hears - entendra	3
jeriah - vaillants	3
carriers - couper	3
woodcutters - puiser	3
jeriah - jerija	3
almighty - armées	3
tingle - oreilles	3
timothy - timothée	3
jehoshaphat - josaphat	3

Table 35: Top 20 Translations, T-score, Blinker, New Settings

word pair	Frequency
israel - isral	14
god - dieu	18
i - je	25
day - jour	9
? - ?	9
lord - Éternel	29
men - hommes	8
people - peuple	8
king - roi	10
all - tous	17
me - moi	10
judah - juda	7
sons - fils	12
like - comme	15
before - devant	11
if - si	8
more - plus	7
egypt - Égypte	5
by - leur	12
we - nous	7

Table 36: Top 20 Translations, Log-Likelihood Ratio, Blinker, New Settings

word pair	Frequency
israel - israìl	14
i - je	25
lord - Èternel	29
egypt - Ègypte	5
god - dieu	18
day - jour	9
people - peuple	8
or - ou	5
? - ?	9
loathe - dégoût	4
derbe - derbe	4
sword - épée	4
third - troisime	4
men - hommes	8
judah - juda	7
all - tous	17
jesus - jésus	23
faith - foi	4
christ - christ	5
nations - nations	5

Table 37: Top 25 Translations, Single T-score versus Ensemble T-score, Log-likelihood Ratio, and Odds Ratio, New Settings

Experiment	Total Found	Correct	Precision	Recall
T-score	30	29	0.97	0.02
Ensemble	128	69	0.54	0.05

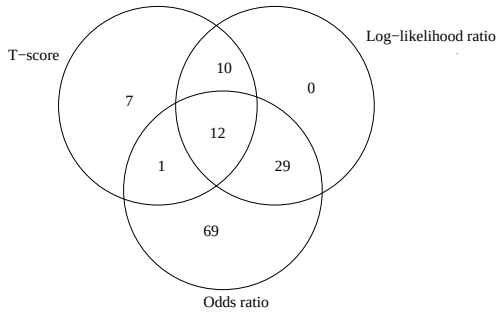

Figure 3: Venn Diagram for Top 25 Translations Per Measure, Blinker, New Settings

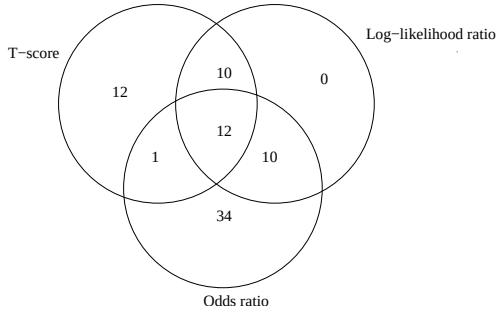

Figure 4: Venn Diagram for Top 25 Translations Per Measure, Hansard's (small), New Settings

As can be seen from the Table 36, when using the ensemble approach the precision goes down considerably but we find 40 more word pairs which are correct. It is interesting to note that most of the translations found by the ensemble involve content words, which are potentially the most interesting from the point of view of translations and bilingual lexicons.

From Figures 3 and 4 we observe that all the word pairs found by Log-likelihood ratio are also found by either T-score or Odds ratio. Thus, it would be possible to simply use the T-score and the Odds Ratio for experiments involving Blinker and Hansard's (small) and achieve comparable results. However, for Hansard's (big) as shown in Figure 5, we observe that there are 10 word pairs found by the Log-likelihood Ratio and that are neither found by the T-score nor by the Odds Ratio.

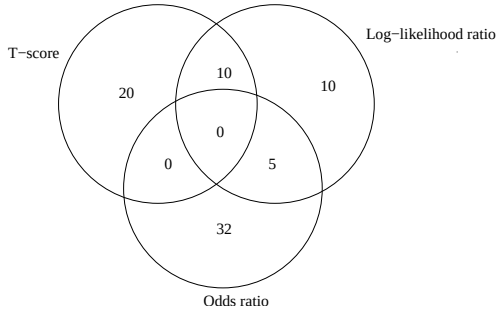

Figure 5: Venn Diagram for Top 25 Translations Per Measure, Hansard's (big), New Settings

6 Related Work

This section gives a description of the various approaches for finding word correspondences and their comparison with the K-vec algorithm of Fung and Church.

In 1991, Gale and Church [10] introduced the idea of using measures of association for finding translations of words based on information in parallel text. They begin by carrying out sentence alignment, which is the problem of determining which sentences are translations of each other. In fact this is a much simpler problem than finding the translations of words, since long sentences in one language tend to translate as long sentences in another language, and the order in which sentences appears doesn't usually change radically in a translation.

In order to deal with very large corpora, they consider a subset of the sentences in a corpus and form the possible translations from within that set. For each translation (X, Y), where X is the word in the source language and Y is the proposed translation in the target, their algorithm forms a contingency table as follows:

-	Y	$!Y$	$total$
X	a	c	$a+c$
$!X$	b	d	$b+d$
$total$	$a+b$	$c+d$	$a+b+c+d$

where:

- *a* is the number of times X and Y occur in the corresponding sentences
- *c* is the number of times X occurs in a sentence but Y does not occur in the corresponding sentence
- *b* is the number of times Y occurs in a sentence but X does not occur in the corresponding sentence

For each proposed translation it calculates a value ϕ, using the formula:

$\phi = \frac{(ad-bc)^2}{(a+b)(a+c)(b+d)(c+d)}$

It selects a candidate translation (X,Y) as a valid translation if its ϕ value is significant as compared to all other pairs (X,Z).

It repeats this process a few times by increasing the number of sentences in the subset. For each iteration it does not consider the word pairs that are selected in any of the previous iterations. Once this is done, for each word in each of the aligned sentence it finds the translation using the selected pairs. In the case where a word has more than one possibility it uses the slope condition [10] to select the best pair.

The idea of forming word pairs between the words of the corresponding sentences and the way the contingency table is created for each word pairs is similar to the K-vec approach. The difference is that K-vec instead of aligning the parallel text at sentence level, divides the two texts into equal regions and forms word pairs in the corresponding regions. The approach we describe here is an iterative method and selects the best translation for each word for each iteration while K-vec is a single iteration method.

Neither K-vec nor the approach of Gale and Church considers the frequency of a word in a region or sentence respectively. Neither of these approaches works for phrasal translations. Aligning parallel text at the sentence level requires knowledge of the languages of the parallel text and hence is an language dependent task. This method need to align the parallel text at sentence level which makes it language dependent as opposed to K-vec. Also it makes the approach computationally more intensive. Finally, neither approach makes use of clues such as cognates, which are words that have essentially the same form in both the source and target languages.

Fung and McKeown [8] present an extension to the K-vec algorithm that is known as DK-vec. It compiles a lexicon from a *noisy parallel corpus*, which is a body of text in which segments from the source or target are completely missing or are not translations of each other. For each word in the source and target text it prepares a position vector and a recency vector. A position vector has the byte positions of all the occurrences of the word in the ascending order. A recency vector is created by finding the difference in the subsequent entries of the position vector.

Candidate translations are formed between the words from the source text and the words from the target text. The word pairs in which the first occurrence of the word is half the text apart are filtered out. Also, the word pairs in which the length of the vector of one of the words is half the length of the vector for another word are filtered out. For all the remaining word pairs it calculates a score using a matching technique known as Dynamic Time Wrapping [8]. For every word X in the source text, the word Y in the target text which gives highest score is taken as its translation.

The original K-vec algorithm of Fung and Church divides the text into segments and forms word pairs using the words in the same segment. This requires the text to be *linear*, without insertion or deletion of sentences or paragraphs. It thus does not work with noisy corpus whereas DK-vec does. To account for the dynamic occurrence of words in the noisy corpus, DK-vec makes use of recency information in addition to position and frequency information used by K-vec. K-vec prepares a binary vector for each word indicating its presence or absence in a particular piece. So it does not make use of the overall frequency of the word in the text as one or more than one occurrence of a word in the same piece is counted as one occurrence. DK-vec makes use of the overall frequency of the word in finding the word correspondences. It uses DTW technique while K-vec uses tests of association to find the best word pairs.

Most approaches to identifying translations and building bilingual lexicons make use of parallel corpora and utilize information like the frequency and position of words to find translations. However, in 1995 Fung [6] proposed a method for finding word correspondences from a *non-parallel corpus*, that is a text in different languages but not translations of each other.

In such a corpus there is no relation between the words and thus this approach cannot make use of the features like word position and its frequency to find word correspondences. Instead, it makes use of the context heterogeneity feature, which is the number of words used in the context of a word and its corresponding word in the second language that are approximately the same.

The context heterogeneity of a word W is defined as an ordered pair (x, y)

x = (left heterogeneity) = $\frac{a}{c}$

y = (right heterogeneity) = $\frac{b}{c}$

where

- a = number of distinct words immediately preceding the word W.
- b = number of distinct words immediately following the word W.
- c = total number of occurrences of the word W in the text.

Once the context heterogeneity vector is calculated for all the words in the source text, the Euclidean distance can be used to find the similarity between two vectors for word X and Y.

$\varepsilon = \sqrt{(x_1^2 + x_2^2) + (y_1^2 - y_2^2)}$

where,

x_1 and x_2 are left heterogeneities of the words X and Y respectively.

y_1 and y_2 are right heterogeneities of the words X and Y respectively.

For each word X in the source text, the word Y is taken to be the translation if the distance between their vectors is less than that of any other word Z in the target text.

The original K-vec algorithm of Fung and Church works only for parallel corpus and makes use of the word position and frequency feature to find word correspondences. This methods works for both parallel as well as non-parallel corpus and makes use of the context heterogeneity feature. K-vec uses tests of association as a similarity measure, while the 1995 approach of Fung uses Euclidean distance. Like K-vec this approach is also language independent and works for different language pairs.

Fung and Ye [9] describe an approach for finding translations from non-parallel yet comparable texts. Texts are called comparable if the content is same but is not exact translation of each other. As it makes use of non-parallel texts, it cannot use word position, and word frequency to find the translation pair. It makes use of the feature that a words that appear in context of the words that are translation of each other are similar.

For each word W it prepares a vector, such that the i^{th} dimension of the vector has value f, if the i^{th} word of the text and the word W appear in the same sentence f times. It makes use of an already existing bilingual lexicon to find the meaning of these context words. For a word X, the word is expected to be its translation if the number of words in common (words that are translations of each other) between X and Y is more than any other pair X and Z. These words in common are called as seed words.

It then arranges the seed words according to the term frequencies. Term Frequency (TF) of the seed word is the the value f in the vector. When arranged in ascending order, the words that are translations of each other rank similar words high up the order. The function words appear in all the sentences and thus will have high term frequency and will be ranked high for each word. To account for this it re-arranges the seed words according to the weighting factor defined as:

word weight factor = $TF \times IDF$

IDF = log $\frac{maxn}{n_i} + n_i$

where

- $maxn$ = the maximum frequency of any word in the corpus.
- n_i = the total number of occurrences of the word i in the corpus.

Once it ranks the seed words it uses similarity measures proposed in /citesalton88termweighting to find the translation pairs.

K-vec only works with parallel text while method of Fung and Ye works for non-parallel texts too. However, this approach requires an existing bilingual lexicon for the language pair under consideration. It is thus a method used to find translation of words whose translation is not known and is present in the document. K-vec does not make use of any previously complied bilingual lexicon for finding the translation for a word. Like K-vec this approach is also language independent as it does not require any prior knowledge of the language to find the translation of the words.

Brown [1] describes a method for extracting a bilingual lexicon from a sentence aligned parallel corpus. It creates a two-dimensional array containing the words from the source text in one dimension and the words from the target text in the other. This structure is known as a co-occurrence table. For each sentence pair in the corpus, it discards the duplicate words and all possible word pairs are entered in the co-occurrence table.

A particular word pair may occur N times in the table, if it occurs together in N corresponding sentences. Once the table is created, all the pairs are passed through two filtering tests. Only the word pairs which pass at least one of the tests are considered as word correspondences, remaining word pairs are discarded. The purpose of both the tests is to make sure that the two words occur together in some minimum number of corresponding sentences to be considered as translations of each other.

The first test sets the threshold value to some unreachably high value for co-occurrence count less than minimum and to some constant for all others. The test is passed only if
$C[S,T] \geq threshold[C] \times count[S] and C[S,T] \geq threshold[C] \times count[T]$

Here,

C[S, T] - is the number of times the word pair occur together in the corresponding sentence.

threshold[C] - selected threshold value

count[S] and count [T] are the overall frequencies of the word in the source and target text respectively.

The second test is passed only if

$C[S,T] \geq thresh1[C] \times count[S] and C[S,T] \geq thresh2[C] \times count[T]$

OR

$C[S,T] \geq thresh2[C] \times count[S] and C[S,T] \geq thresh1[C] \times count[T]$

By changing the values of these threshold, one can vary the precision and recall value. If the threshold values is kept high, the precision value increases but the recall value decreases. If the threshold value is decreased, the recall value increases but the precision value goes down.

Like K-vec this approach only works for parallel corpus but it requires the corpus to be sentence aligned. It is thus language dependent and computationally intensive algorithm as compared to K-vec. K-vec uses tests of association to find the best word correspondences while it filters out the unwanted word pairs using the threshold technique tests discussed above.

7 Conclusions

This thesis investigates the use of measures of association for finding translations in parallel text. We began this inquiry by studying the K-vec algorithm of Fung and Church (1994). We made several findings with respect to this algorithm:

1. The number of pieces in which a text is divided is critical. Fung and Church suggest using the square root of the number of tokens. We point out that this leads to very large piece sizes for larger corpora, and propose instead a constant piece size of 100 tokens. Empirical results show that this improves performance.

2. The frequency cutoffs as proposed by Fung and Church of 3 and 11 (lower and upper) and not appropriate for larger corpora. We suggest that the upper frequency cutoff should depend on the number of pieces, and propose that this should be set to half the number of pieces. Empirical results show that this improves performance.

We also conducted an extensive theoretical and empirical study of measures of association that could be applied to this task. In particular we examined the T-score, the Log-likelihood Ratio, Fisher's Exact Test, the Odds Ratio, the Dice Coefficient, and Pointwise Mutual Information. We reached the following conclusions:

1. The T-score as suggested by Fung and Church is a very suitable measure of association for finding translations in parallel text. It performs as well as any other measure we considered.

2. The Log-likelihood Ratio and the Odds Ratio are equally as effective in many cases, and merit further consideration for this task.

3. The use of Pointwise Mutual Information was discouraged by Fung and Church, and we concur with that finding. While the results it obtains are reasonable for the top few ranks, its performance tends to degrade thereafter.

4. The Dice Coefficient and Fisher's Exact test are too generous and consider too many candidates as valid translations. As such we do not recommend their use.

Since we found that several different measures of association (with very different characteristics) perform relatively well in identifying translations in parallel text, we explored the use of ensemble techniques to try and exploit the strengths of each of these methods. In particular we developed techniques based on taking the union of the results from these various measures and reached the following conclusions:

1. All of these measures are very sensitive to the number of pieces in which a text is divided. As such we developed an ensemble using two different piece sizes (90 and 100 tokens) and found that the union of those two sets of translations represented an improvement over either of the single set of results.

2. The union of our three most effective measures (the T-score, the Log-likelihood Ratio, and the Odds Ratio) results in better performance than any of the individual measures.

We investigated all of the above points using three very different sets of data. Two were taken from the Hansard's, which are the bilingual proceedings of the Canadian Government. One set was fairly small (500 sentences) while the other was quite large (10,000 sentences). The language used is that of government officials and the translations tend to be fairly literal. We also used the Blinker data, which are 250 verses from French and English versions of The Bible. These are not direct translations of each other, and of course religious language is very rich and metaphoric and quite different from that of government officials. Despite these differences, we observed comparable results for all three sets of data for the above points, giving us confidence that our findings do in fact generalize.

8 Future Work

The simplicity of the K-vec algorithm provides us with various possible avenues for future work. In its current form the K-vec algorithm deals only with one to one word translation pairs. One possible extension of this thesis would be extending the K-vec algorithm so that it also works for *phrasal translations*, that is finding the case where a single word is translated to multiple words.

Also, in its current form the K-vec algorithm does not make use of the frequency and the position information of the words in a piece. As it does not account for frequency information, it forms pairs within a piece using low frequency words in one language and the high frequency words in another language, which are very unlikely to be translations of each other. Since it doesn't account for position information it forms word pairs using the words which occur at the beginning of a piece in one language and the words which occur at the end of the corresponding piece in another language. Again, these are very unlikely to be translations of each other. In not accounting for either for these types of information K-vec creates candidate translations that are highly unlikely to really be translations. We plan to extend K-vec to make use of both frequency and position information in order to improve performance.

We also plan to make use of the *cognates*. These are words that are spelled similarly in two languages, or words that simply are not translated, such as proper nouns. For example, if the word *Berlin* appears in an English text, it probably also appears as *Berlin* in a Spanish text. We should be able to pick low hanging fruit such as this to improve the results of K-vec.

We also plan to study the measures of association in more detail and identify additional measures that are significantly different from each other. We would then like to continue exploring the use of ensembles of such measures in order to exploit the different characteristics of these texts.

We also plan to explore additional types of ensembles. One possible formulation would be taking the union of word pairs found using the same measure and the same number of pieces, but varying the frequency cutoffs. Given the number of different measures and the varying settings that each of them provide, we can imagine quite a few different combinations that could lead to interesting results.

References

[1] R. Brown. Automated dictionary extraction for "knowledge-free" example-based translation. In *Proceedings of the Seventh International Conference on Theoretical and Methodological Issues in Machine Translation (TMI-97)*, pages 111–118, Santa Fe, New Mexico, July 1997. `http://-www.cs.cmu.edu/~ralf/papers.html`.

[2] K. Church, W. Gale, P. Hanks, and D. Hindle. Using statistics in lexical analysis. In *Lexical Acquisition: Exploiting On-Line Resources to Build a Lexicon*, pages 115–164. Lawrence Erlbaum, 1991.

[3] J. Davis. Hierarchical models for significance tests in multivariate contingency tables: An exegesis of Goodman's recent papers. *Sociological Methodologies*, 5:189–232, 1973-1974.

[4] T. Dunning. Accurate methods for the statistics of surprise and coincidence. *Computational Linguistics*, 19(1):61–74, 1993.

[5] K. McKeown F. Smadja and V. Hatzivassiloglou. Translating collocations for bilingual lexicons: A statistical approach. *Computational Linguistics*, 22(1):3–38, 1996.

[6] P. Fung. Compiling bilingual lexicon entries from a non-parallel English-Chinese corpus. In *Proceedings of the Third Workshop on Very Large Corpora*, pages 173–183, 1995.

[7] P. Fung and K. Church. Kvec: A new approach for aligning parallel texts. In *Proceedings of 15th International Conference on Computational Linguistics (COLING-94)*, pages 1096–1102, Tokyo, Japan, 1994.

[8] P. Fung and K. McKeown. Aligning noisy parallel corpora across language groups: Word pair feature matching by dynamic wraping. In *In Proceedings of the Biennial Conference of the Association for Machine Translation in the Americas*, pages 81–88, Columbia, Maryland, 1994.

[9] P. Fung and L. Yee. An IR approach for translating new words from nonparallel, comparable texts. In *Proceedings of the Thirty-Sixth Annual Meeting of the Association for Computational Linguistics and Seventeenth International Conference on Computational Linguistics*, pages 414–420, 1998.

[10] W. Gale and K. Church. Identifying word correspondences in parallel texts. In *Proceedings of the Fourth DARPA Speech and Natural Language Workshop*, pages 152–157, Pacific Grove, CA, 1991.

[11] C. Manning and H. Schutze. *Foundations Of Statistical Natural Language Processing pp–267-271*. The MIT Press, 1999.

[12] D. Melamed. *Empirical Methods for Exploiting Parallel Texts*. The MIT Press, Cambridge, MA, 1998.

[13] D. Melamed. *Manual Annotation of Translational Equivalence: The Blinker Project*. Institute for Research in Cognitive Science Technical Report 98-07, University of Pennsylvania, Philadelphia, PA, 1998. `http://www.cs.nyu.edu/~melamed/datasets.html`.

[14] F. Och and H. Ney. Improved statistical alignment models. In *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics*, pages 440–447, Hong Kong, China, October 2000.

[15] T. Pedersen. Fishing for exactness. In *Proceedings of the South-Central SAS User's Group Conference (SCSUG-96)*, Austin, TX, Oct 27-29 1996.

[16] G. Zipf. *The Psycho-Biology of Language: An Intoduction to Dynamic philosphy*. Houghton Mifflin, Boston, MA, 1935.