Statistical Machine Translation Incorporating Morphlogical Knowledge and Using Improved Alignments

Klasinas Giannis
Electronic and Computer Engineering
Technical University of Crete

1st November 2005

Abstract

Machine translation between natural languages is a very interesting problem, to which working solutions are bound to have important social and economical implications. During the last years there has been great improvement in the performance of such systems. Ongoing research in the field of statistical machine translation has resulted in performance comparable to that of other approaches which have been developed for decades. The aim of this project is to improve the performance of such a system using state of the art tools and methods. First, the problem of automatic translation is presented, along with different approaches used. Then, we review the basic theory on which statistical machine translation is based, along with the necessary steps to build a statistical machine translation system from scratch. Afterwards, we describe the baseline system we improved. Then, we describe the two different approaches we used, namely, incorporation of morphological knowledge, and usage of improved alignments. Automated, freely available tools were used to build parts of the system, as well as to evaluate the performance of the resulting system. These tools are explained in detail. We conclude with the evaluation of the results and suggestions for further improvements. In the appendix are listed example tranlsations using the baseline and the improved system.

Contents

1 INTRODUCTION **3**

1.1 Statistical Machine Translation 4

1.1.1 Introduction 4

1.1.2 Problem Presentation 4

2 BASELINE SYSTEM **7**

2.1 Sentence boundary detection 7

2.1.1 Introduction 7

2.1.2 Method for sentence boundary detection 7

2.1.3 Results evaluation 8

2.2 Sentence aligning 8

2.2.1 Introduction 8

2.2.2 Method for aligning sentences 9

2.2.3 Results evaluation 10

2.3 Language modelling 11

2.3.1 Introduction 11

2.3.2 Language model training 11

2.4 Translation modelling 12

2.4.1 Introduction 12

2.4.2 Word to word based translation modeling 13

2.4.3 Parameter estimation 14

2.4.4 Expectation maximization 16

2.4.5 GIZA++ 18

2.4.6 Phrase-Based Translation modeling 18

2.4.7 Bilingual phrases-model building 19

2.5 Decoding 20

3 MORPHOLOGY **22**

3.1 Introduction 22

3.2 Essential background on morphology 23

3.2.1 Morphemes and the kinds of morphologies 23

3.2.2 Learning a morphology 23
3.3 Different approaches 24
3.3.1 Introduction 24
3.3.2 Bootstrapping using a knowledge source 24
3.3.3 Obtaining affix inventories 25
3.4 Performing a complete morphological analysis-Linguistica . . . 25
3.4.1 Introduction 25
3.4.2 Minimum Description Length Model 26
3.4.3 Heuristics for word segmentation 26
3.5 Heuristics proposed 27
3.5.1 Introduction 27
3.5.2 Heuristic proposed 28
3.5.3 Results-evaluation 28
3.6 Using morphology in SMT 29
3.6.1 Combined Lexical-Morphological system implementation 30

4 IMPROVEMENTS ON BASELINE SYSTEM 33
4.1 Introduction 33
4.2 Bilingual phrase extraction 33
4.3 Heuristics proposed 34
4.3.1 diag_and 34
4.3.2 weight 35
4.3.3 exact 39
4.3.4 lenwe 39
4.3.5 cor_icor 40
4.3.6 int 40
4.3.7 uni 42
4.3.8 icor 42

5 RESULTS - EVALUATION 43
5.1 Introduction 43
5.2 BLEU 43
5.3 NIST 45
5.4 Results for system incorporating morphological knowledge . . 48
5.5 Results for system using improved alignments 50

6 CONCLUSION - FUTURE WORK 65
6.1 Using Statistical Machine Translation for Speech Translation . 66

A Tanslation Examples 67

Chapter 1

INTRODUCTION

Machine Translation is the automated process of translating a document from one natural language into another natural language. Machine translation is one of the hardest problems of artificial intelligence because in order to translate a natural language we need to understand the meaning of the source text, the syntax restrictions of every language and other information so as to be able to resolve the ambiguities of natural languages. There have been different approaches to this problem, the most important being the following:

- interlingua
- transfer based
- example based
- statistical

Interlingua This approach, also called knowledge-based, tries to imitate the way a human translates. That is, after reading the sentence it uses syntactic and semantic knowledge about the language as well as the meaning of the sentence so as to transform them into information regarding the syntactic and semantic knowledge about the other language. This approach faces many problems representing all this knowledge.

Transfer-based Transfer-based is similar to knowledge-based, however information is not represented in interlingua form. Instead complex rules that match the syntactic and semantic structures of the two languages are created. As a result this approach requires knowledge of the comparison of the grammars of the languages, that is the differences and how they are solved. The rules are created manually, and are language depended.

Example-based In this approach the system learns to translate from parallel corpora in various languages. In its simplest form the sentence to be translated is compared to a set of sentences whose translation is known and one of these translations is selected for the sentence in hand.

Statistical Statistical machine translation, which is the approach used in this project, can be seen as a somewhat more complicate example-based where the selection criteria are based on probabilistic methods. Its advantage is that it solves the problem of representing concepts of natural language, since knowledge is not represented but is automatically mined from a parallel corpus. The resulting translation quality is good when dealing with texts containing economic, political, technical and business terms.

1.1 Statistical Machine Translation

1.1.1 Introduction

Statistical Machine Translation (SMT) means automated translation from one natural language into another, using statistical methods. The usage of statistical methods for automatic translation was first proposed by Waren Weaver in 1949. The idea was abandoned due to various reasons, one of which was the small computational power available at the time. Today, however, the study, implementation and utilization of a statistical machine translation system is possible using a modern computer.

1.1.2 Problem Presentation

The techniques used in automatic translation are basically the same that have been successfully employed by the field of Speech Recognition. The basic idea of these techniques is the notion of Language Model and Translation Model.

Suppose a sentence f in the foreign language and we are searching for a sentence n in the native language which maximizes the probability $P(n|f)$, that is, we are searching for the most probable translation of sentence f to the native language. This is formulated as

$$\hat{n} = \arg\max_{n} P(n|f) \tag{1.1}$$

Using Bayes rule

$$P(n|f) = \frac{P(n)P(f|n)}{P(f)} \tag{1.2}$$

(1.1) becomes

$$\hat{n} = \arg\max_{n} \frac{P(n)P(f|n)}{P(f)} \tag{1.3}$$

The term $P(f)$ in 1.3 is ignored, since it is the same for all possible sentences n. The above transform enables us to use the Noisy Channel technique which works as follows. We imagine that someone thinks of sentence n, but by the time n gets to the printed page it is corrupted by noise and becomes f. To recover the most likely sentence n, we ask the following:

1. What kind of sentences is used in the native language?
2. How native sentences are transformed into foreign?

As a result now we have to compute two terms $P(n)$ and $P(f|n)$, while initially we only had to compute $P(n|f)$. This might seem like a setback but it is not. If we only depend on $P(n|f)$, we need to have very accurate probabilities, which is not easy. For example, it is possible that high values are given to sentence pairs where words in f are translations of words in n, but the words in n are arbitrarily ordered. Obviously, such a model would not produce quality translations. The problem arises from trying to keep too much information in one model. However, using the noisy channel technique, we solve this problem. We use $P(f|n)$ to be able to tell whether the words in n are good translations of the words in f. It is not important if the ordering of the words in n is correct[1]. $P(n)$ is used to verify this. As a result it is easier to train each model ($P(n)$ is called the language model and $P(f|n)$ is called the translation model).

In order to achieve good performance, it is important to use a big corpus to compute $P(n)$ and $P(f|n)$. In this project, we have used the European parliament records [11]. Below this abstract level there are many details. It is possible to use different approaches so as to improve the resulting translation quality. For example, the way the translation model is built is quite complex and very important. As a result there exists space for different techniques. Also, one might think of using the stems of the words instead of the words themselves to train the model, when the corpus available for training is not enough. In this project, we have used a baseline statistical machine translation system created by Fanouris Moraitis [1] and explored different ways of improving its performance.

[1]This is not always the case. For example the words in *John loves Mary* can be reordered to *Mary loves John*. Both sentences are grammatically and syntactically correct, however the meaning is completely different. So, it might be useful to store some information about word ordering in $P(f|n)$.

The creation of a baseline statistical machine translation system is described in chapter 2. In chapter 3 we discuss morphology, and investigate the incorporation of morphological knowledge into statistical machine translation systems. In chapter 4 we present the modifications used to enhance the performance of the baseline system. The experimental setup and results are presented in chapter 5. Finally, we conclude our work in chapter 6.

Chapter 2

BASELINE SYSTEM

2.1 Sentence boundary detection

2.1.1 Introduction

In order to align bilingual sentences, we first have to detect sentence boundaries. Generally speaking, a sentence ends with a punctuation mark like . ? ! ;. However, a punctuation mark does not always denote the end of a sentence, since it can appear in abbreviations like "i.e.".

2.1.2 Method for sentence boundary detection

In order to solve this problem the text is divided into tokens, token being any sequence of characters between spaces. Any possible sentence boundary, that is, token having at least one punctuation mark, is split into the following parts.

- Prefix
- Candidate (one of ! : ; .)
- Suffix

For example, in the sentence *Do you agree?* Prefix=agree Candidate=? Suffix=NULL. In addition to that, it is checked if the token is an abbreviation or honorific and if the next token starts with an uppercase or lowercase letter. The lists for abbreviations and honorifics were taken from dictionaries. However, because such lists can not be complete, a probabilistic model[1] was

[1]The same model was used to evaluate the algorithm

applied on the training corpus which checks if a token could be abbreviation or honorific. As a result we introduce a feedback that improves the performance of the algorithm.

2.1.3 Results evaluation

The method described above was applied on 111408 English and 112756 Greek sentences, and the results were evaluated by a human judge. The errors are divided into the following two categories.

False negative Failure to identify a sentence boundary

False positive Mistaken identification of sentence boundary

The results are displayed in Table 2.1. As far as false negatives are concerned, it can be noted that:

- Most mistakes are caused by typographic mistakes in the text.
- The rest of the mistakes were abbreviations that were sentence boundaries.

	Greek	English
Sentences	112756	111408
Candidate Punctuation Marks	120225	113207
False Positives	0	0
False Negatives	205	175
total error	0.17%	0.15%

Table 2.1: Results

2.2 Sentence aligning

2.2.1 Introduction

In order to train the machine translation system , we first have to identify the pairs of translated sentences, process known as Bilingual Sentence Alignment. Before aligning the sentences, the paragraphs must be aligned, but that is not difficult as paragraphs have well defined limits. Aligning the sentences, however, is a more difficult problem.

2.2.2 Method for aligning sentences

It is generally true that long sentences[2] tend to be translated into long sentences, while short sentences are translated into short ones. A simple probabilistic model uses the ratio of the two sentences length, and assigns a value to this pair of sentences. This value is then given as input to a dynamic programming algorithm that searches for the maximum likelyhood alignment. The distance between two sentences is defined as $-\log Prob(match|\delta)$ where δ is a random variable that depends on the length l_1, l_2 of the sentences in hand.

Variables l_1, l_2 are supposed to be independent, and normally distributed. Such a model is defined by mean value c and distribution s^2. Mean value actually represents the expected number of characters in one language per character of the other language, and s^2 the dispersion of the mean value per character. Both values are computed empirically[3]. δ is calculated using the following formula,

$$\delta = \frac{l_1 - l_2 c}{\sqrt{l_1 s^2}} \tag{2.1}$$

having mean value zero and standard deviation one[4].

Now we want to compute $Prob(match|\delta)$, that is the possibility that a sentence is translated into another, given δ. Using Bayes rule we have

$$Prob(match|\delta) = \frac{Prob(match,\delta}{)} Prob(\delta) = \frac{Prob(\delta|match)Prob(match)}{Prob(\delta)} \tag{2.2}$$

$Prob(\delta)$ is ignored since it is the same for all possible matching pairs, while $Prob(match)$ is given as prior probability from Table 2.2.

Category	Prob(match)
1-1	0.89
1-0 or 0-1	0.005
1-2 or 2-1	0.0445

Table 2.2: Prior Probabilities

Since δ follows the normal distribution with mean value zero and standard deviation 1 (at least for matching sentences) we can suppose $Prob(\delta|match) \approx$

[2]The legnth is the number of characters

[3]For example to find the mean value, we count the length of language a paragraphs divided by the respective figure for language b

[4]At least for sentences that are translations of each other

$Prob(\delta)$. Supposing there exists a match in $[-\delta_o ... \delta_o]$

$$Prob(\delta|match) = Prob(-\delta_o < \delta < \delta_o) \tag{2.3}$$

Using the cumulative distribution function[3]

$$Prob(-\delta_o < \delta < \delta_o) = 2(1 - Prob(\delta < \delta_o)) \tag{2.4}$$

where

$$Prob(\delta < \delta_o) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\delta_o} e^{\frac{\delta^2}{2}} d\delta \tag{2.5}$$

it is possible to compute

$$Prob(match|\delta) = Prob(match)(2(1 - Prob(\delta < \delta o))) \tag{2.6}$$

The distance computed, is then used by the dynamic programming algorithm as the cost of every change. Allowed changes are:

(1-1) One sentence matches exactly one sentence.

(1-0) One sentence in one language has no corresponding in the other language (deletion).

(0-1) Insertion of a sentence.

(1-2) or (2-1) Two sentences in one language correspond to one sentence in the other language, or vice versa.

The algorithm described above described is an implementation of the William A. Gale & Kenneth W. Church algorithm[2].

2.2.3 Results evaluation

The method described above was tested on a 2051 alignments corpus, and the results were evaluated by a human judge. There were 31 mistakes, so the error was on the order of 1,5%. 1-1 alignments had the smallest error percentage, while in 2-1 and 1-2 this percentage increased. 1-0 o 0-1 alignments were always wrong, but that was caused by a wrong 1-2 or 2-1 preceeding alignment. The results are shown in Table 2.3.

Category	total count	mistake count	Percentage of errors
1-1	1912	19	0.99
1-0 or 0-1	3	3	100
1-2 or 2-1	139	9	6.1

Table 2.3: Alignment evaluation

2.3 Language modelling

2.3.1 Introduction

Using the noisy channel technique, the initial problem is divided into two smaller problems, one of which is the calculation of $P(n)$, namely the language model. Qualitatively the language model represents the probability of using phrase n in a natural languageage. As a result, it assigns small probability[5] to syntactically or grammatically wrong sentences, as such phrases are rarely used. For example it is much more probable to encounter the phrase *Today is Monday* than *Monday today is.* In other words, the language model contains the information of how correct or not is a sentence, making it easier to build the translation model.

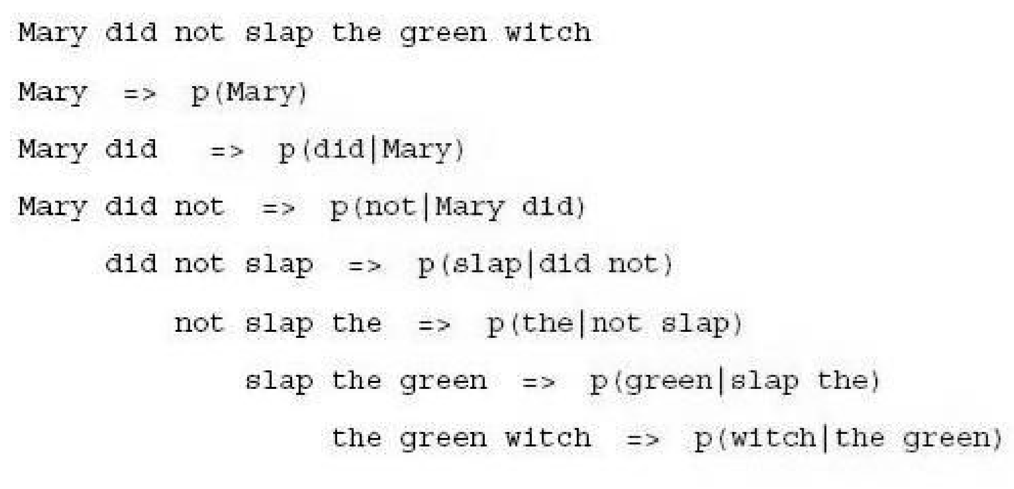
2.3.2 Language model training

Language model training can be defined as the calculation of the prior possibility of a word or phrase. Phrases are represented using ngrams. Ngrams are phrases of n words. For n=1, n=2, n=3 they are called unigram, bigram and trigram respectively. For example consider the bigram *Mary did*, and the probability $P(did|Mary)$. To calculate this possibility we divide the number of observations of *Mary did* by the number of observations of *Mary*, as seen in the following equation.

$$P(did|Mary) = \frac{\#(Mary\ did)}{\#(Mary)} \tag{2.7}$$

Generally we can find *Mary did not slap the green witch* by deduction, as seen in figure 2.1. Several tools exist that train language models, like CMU Toolkit[7], HTK Lattice Toolkit[8] and SRILM Toolkit[9, 10]. The SRILM Toolkit was used to build the language model, which is freely available for noncommercial purposes under the Open Source Community License. The

[5]We assign non-zero probabilities because wrong sentenences are possible in real-life documents.



```
Mary did not slap the green witch

Mary  =>  p(Mary)

Mary did   =>  p(did|Mary)

Mary did not  =>  p(not|Mary did)

     did not slap  =>  p(slap|did not)

         not slap the  =>  p(the|not slap)

             slap the green  =>  p(green|slap the)

                  the green witch  =>  p(witch|the green)
```

Figure 2.1: Calculation of $P(Mary\ did\ not\ slap\ the\ green\ witch)$ using trigrams

SRILM Tookit is a collection of C++ libraries, executable programs and helper scripts that allow both production and experimentation with statistical language model that can be used in statistical machine translation systems, speech recognition and other applications.

Using only base functionality, we create the language model with this command **ngram-count -order 4 -text corpus -lm lm.srilm** . With the order option we specify the rank of ngrams used, in this case fourgrams. The text specifies the corpus used to train the model, which must have one sentence per line. Finally using lm we train the backoff ngram model, and write it to file *lm.srilm*. The output is in ARPA format.

2.4 Translation modelling

2.4.1 Introduction

By translation modeling we mean the calculation of the probability that we have sentence f given sentence n, $P(f|n)$. We will use a Phrase-Based Translation model which uses the results of a Word-Based Translation model. As the names imply Word-Based Translation establishes correspondence between words, while Phrase-Based Translation uses phrases.

2.4.2 Word to word based translation modeling

There exist many techniques to calculate the probabilities for a Word-Based Translation model[3]. One way is to use an interlingua representation of the sentence; sentence n is transformed into predicate logic, or a conjunction of atomic logical assertions. For example sentence *John must not go* is transformed into OBLIGATORY(NOT(GO(JOHN))) and then into the corresponding sentence of another natural language. A different approach tries to build the syntactic tree for sentence n and then using predefined rules the tree is transformed into the corresponding syntactic tree for the foreign language.

The technique we used translates the words of sentence n into the foreign language and then reorders the words. However before discussing the details, first we have to define some parameters. $t(f_i|n_i)$ is the probability of translating word n_i into f_i. $n(k|n_i)$, called fertility, is the probability that word n_i will be translated into k words in the foreign language. $d(p_f|p_n, length_n, length_f)$, distortion, is the probability that word in position p_n in sentence n of length $length_n$ will produce a word in position p_f in sentence f of length $length_f$. In addition, we assume that in position zero of each sentence n is the word NULL which can give words (spurious) in sentence f with probability p_1. After assigning fertilities to all words in n we will have created x words in f.

We are now ready to review the process of transforming sentence n into sentence f. Suppose that the native sentence is
Mary did not slap the green witch
using English as native Language. Then we choose the fertilities for each word.
Mary not slap slap slap the green witch
As we see word slap has fertility three while word did has fertility zero. Then we insert spurious words.
Mary not slap slap slap NULL the green witch
Finally we replace English words with foreign words, in this case Spanish.
Mary no daba una botefada a la verde bruja
Finally we reorder the words, using the distortion probabilities.
Mary no daba una botefada a la bruja verde

The word alignment produced, can be represented as a vector. For this example the vector is [1 3 4 4 4 0 5 7 6], which means that the first Spanish is derived from the first English word, the second Spanish word is derived from the third English and so on. Note that the sixth Spanish word is derived from the word NULL. One should also note that the way we translated the sentence we have introduced an 1-N restriction because one English word can produce many Spanish words, but it is impossible more than one English words to produce one Spanish word.

This method is described by Brown et al.[3], and is named IBM Model 3[6]. This model calculates the parameters $d(p_f|p_n, length_n, length_f)$, $t(f_i|n_i)$, $n(k|n_i)$, p_1 in order to find the best alignment for every phrase pair. Next we will see how these parameters are calculated using word to word alignments, how we calculate the probability of every word alignment and how these associate with the initial problem, the calculation of $P(f|n)$.

2.4.3 Parameter estimation

If we knew beforehand which of the possible alignments is the right we could easily find the values for the parameters. However in the beginning every alignment is possible. For example, suppose we have a two word sentence n which is translated into a two word sentence f. Ignoring NULL, the possible alignments are the following.

$$\begin{tikzcd} n_1 \arrow[d] & n_2 \arrow[d] \\ f_1 & f_2 \end{tikzcd}$$

$$\begin{tikzcd} n_1 \arrow[dr] & n_2 \arrow[dl] \\ f_1 & f_2 \end{tikzcd}$$

$$\begin{tikzcd} n_1 \arrow[d] \arrow[dr] & n_2 \\ f_1 & f_2 \end{tikzcd}$$

$$\begin{tikzcd} n_1 & n_2 \arrow[dl] \arrow[d] \\ f_1 & f_2 \end{tikzcd}$$

Suppose the first alignment has probability 0.3, the second 0.1, the third 0.4 and the fourth 0.2 and that we want to calculate $n(1|n_1)$. We observe that n_1 has fertility 1 in the first and second alignment, so it is observed 0.3+0.1=0.4 times and $n_c(1|n_1) = 0.4$ and in the same manner we find $n_c(0|n_1) = 0.2$ and

[6]In order to produce we actually use IBM Model 4, however we refer to IBM Model 3 for simplicity' s sake.

$n_c(2|n_1) = 0.4$. n_c is called fractional count. Now $n(1|n_1)$ can be calculated from the following equation

$$n(1|n_1) = \frac{n_c(1|n_1)}{n_c(1|n_1) + n_c(2|n_1) + n_c(0|n_1)} \tag{2.8}$$

So knowing the alignment probabilities, which are formulated as $P(a|n, f)$ we can calculate the model parameters. However, the problem is to calculate these probabilities. Let us examine the following transformation.

$$P(a|n, f) = \frac{P(a, n, f)}{P(n, f)} = \frac{P(a, f|n)P(n)}{P(f|n)P(n)} = \frac{P(a, f|n)}{P(f|n)} \tag{2.9}$$

To find the alignment probabilities we need to know two things, $P(a, f|n)$ and $P(f|n)$. Since there are many ways to get to f from n and each way corresponds to one alignment, we can write

$$P(f|n) = \sum_a P(a, f|n) \tag{2.10}$$

so finding the alignment probabilities ends to

$$P(a|n, f) = \frac{P(a, f|n)}{\sum_a P(f|n)} \tag{2.11}$$

The problem now is to compute $P(a, f|n)$. However $P(a, f|n)$ is the result of IBM Model 3. For example to find the probability of producing the following alignment

$$\begin{tikzcd}
Mary \arrow[d] & not \arrow[d] & slap \arrow[d] & NULL \arrow[d] & the \arrow[d] & green \arrow[dr] & witch \arrow[dl] \\
Maria & no & daba\ una\ bote\ fada & a & la & bruja & verde
\end{tikzcd}$$

we have to find the probability of slap having fertility 3, not being translated into no etc. In other words the alignment probability is the product of the parameters calculated by IBM Model 3, as seen in the next formula.

$$P(a, f|n) = \prod_{i=1}^{I} n(\phi_i|n_i) \prod_{j=1}^{K} t(f_j|n_i) \prod_{j=1}^{K} d(p_{fj}|p_{nj}, I, K) \binom{K-\phi_0}{\phi_0} p_0^{K-2\phi_0} p_1^{\phi_0} \prod_{i=0}^{I} \phi_i! \frac{1}{\phi_0!} \tag{2.12}$$

In the above formula I is the length of sentence n and K of sentence f. Term $\binom{K-\phi_0}{\phi_0}$ represents the number of ways it is possible to create spurious words. $p_0^{K-2\phi_0}$ and $p_1^{\phi_0}$ express the possibility of adding or not spurious words. $\frac{1}{\phi_0}!$

expresses the possibility of reordering of spurious words. Also, in case of a word with fertility bigger than one where it produces,for example, f_1, f_2, f_3 we do not know if the words are produced in this order or they appear in this order after reordering, so since this information is lost we multiply the alignment probability by $\prod_{i=0}^{I} \phi_i!$.

2.4.4 Expectation maximization

We have up to now shown that if we know the model parameters we can compute $P(f|n)$ since $P(f|n) = \sum_a P(a, f|n)$. Problem is, to find $P(a, f|n)$ we need to know the parameters. To solve this problem we need to use, arbitrary, initial values. We generally choose uniform values, so if language f had 20000 words, then $t(f_j|n_i) = \frac{1}{20000}$. Using the Expectation Maximization algorithm[4], in every step the values of the parameters are improved, as will be seen in the following example.

Suppose we have two sentence pairs $(n_1 n_2)(f_1 f_2)$ and $(n_1)(f_2)$. If we ignore the NULL word, require every word to have fertility 1, and ignore the distortion probabilities, the only possible alignments are the following.

$$\begin{tikzcd} n_1 \arrow[d] & n_2 \arrow[d] \\ f_1 & f_2 \end{tikzcd}$$

$$\begin{tikzcd} n_1 \arrow[dr] & n_2 \arrow[dl] \\ f_1 & f_2 \end{tikzcd}$$

$$\begin{tikzcd} n_1 \arrow[d] \\ f_2 \end{tikzcd}$$

Given our simplifications the only things that bear on the alignment probabilities are the word translation parameter values. As a result $P(a, f|n) = \prod_{j=1}^{K} t(f_j|n_i)$[7].

Step 1 Set parameter values uniformly.

$t(f_1|n_1) = \frac{1}{2}$
$t(f_2|n_1) = \frac{1}{2}$
$t(f_1|n_2) = \frac{1}{2}$
$t(f_2|n_2) = \frac{1}{2}$

[7]IBM Model 1

Step 2 Compute $P(a, f|n)$ for all alignments.

$$\begin{tikzcd} n_1 \arrow[d] & n_2 \arrow[d] \\ f_1 & f_2 \end{tikzcd} \quad P(a, f|n) = \frac{1}{4}$$

$$\begin{tikzcd} n_1 \arrow[dr] & n_2 \arrow[dl] \\ f_1 & f_2 \end{tikzcd} \quad P(a, f|n) = \frac{1}{4}$$

$$\begin{tikzcd} n_1 \arrow[d] \\ f_2 \end{tikzcd} \quad P(a, f|n) = \frac{1}{2}$$

Step 3 Normalize $P(a, f|n)$ to yield $P(a|e, f)$ values.

$$\begin{tikzcd} n_1 \arrow[d] & n_2 \arrow[d] \\ f_1 & f_2 \end{tikzcd} \quad P(a|f, n) = \frac{\frac{1}{4}}{\frac{1}{4} + \frac{1}{4}} = \frac{1}{2}$$

$$\begin{tikzcd} n_1 \arrow[dr] & n_2 \arrow[dl] \\ f_1 & f_2 \end{tikzcd} \quad P(a|f, n) = \frac{\frac{1}{4}}{\frac{2}{4}} = \frac{1}{2}$$

$$\begin{tikzcd} n_1 \arrow[d] \\ f_2 \end{tikzcd} \quad P(a|f, n) = \frac{\frac{1}{2}}{\frac{1}{2}} = 1$$

Step 4 Collect fractional counts.

$t_c(f_1|n_1) = \frac{1}{2}$
$t_c(f_2|n_1) = \frac{1}{2} + 1 = \frac{3}{2}$
$t_c(f_1|n_2) = \frac{1}{2}$
$t_c(f_2|n_2) = \frac{1}{2}$

Step 5 Normalize fractional counts to get revised parameter values.

$t(f_1|n_1) = \frac{\frac{1}{2}}{\frac{4}{2}} = \frac{1}{4}$

$$t(f_2|n_1) = \frac{\frac{3}{2}}{\frac{4}{2}} = \frac{3}{4}$$
$$t(f_1|n_2) = \frac{\frac{1}{2}}{1} = \frac{1}{2}$$
$$t(f_2|n_2) = \frac{\frac{1}{2}}{1} = \frac{1}{2}$$

Repeating steps 2-5 many times yields:
$t(f_1|n_1) = 0.0001$
$t(f_2|n_1) = 0.9999$
$t(f_1|n_2) = 0.9999$
$t(f_2|n_2) = 0.0001$
The new values of the parameters are different from the initial. For example the probability of n_1 translated into f_2 is boosted because of the second sentence pair, while the translation of n_1 into f_2 is highly unlikely. Generally speaking, EM, which only optimizes numbers and knows nothing of natural language translation, is efficient in computing the parameter values.

2.4.5 GIZA++

In order to produce the alignments we have used GIZA++. GIZA++ is an extension of GIZA which is part of the EGYPT Statistical Machine Translation Toolkit,which was developed during the 1999 John Hopkins Language and Speech Processing summer workshop.

The toolkit is used to train statistical word based translation models [3, 5, 6]. Providing as input senetence level aligned bilingual corpus, it outputs, among other things, a file with the word alignment between sentences. The format of this file is:
Tomorrow morning
NULL ( ) Maniana (1) por (2) la (2) maniana (2)
We used the predefined parameters, five iterations to train IBM Model 3 and IBM Model 4 and $p_0 = 0.02$, that is 2% possibility of creating a spurious word after each regular word. It should be noted that since we have used the noisy channel technique to devide the initial problem into two smaller, when we want to tranlsate from English to Greek we will use Greek as source language and English as target language.

2.4.6 Phrase-Based Translation modeling

Several researchers have recently demonstrated [6] improved performance using phrase to phrase, rather than word to word models. Current state of the art machine translation systems like CMU [7] use phrase-based translation models. The biggest, maybe, problem of word-based translation models is that they ignore context information, since they translate one word at a time, in a context free manner. On the other hand, it is clear that the translation of a word depends heavily on the context. This information is lost, and the only way to reconcile is to use a language model, which stores information regarding the most probable sequence of words. However, as seen in Table 2.4, it does not help much, since it is trained separately for each language. The way to overcome this problem is to train the translation model using phrases rather than words.

FOREIGN	was Halten Sie vom Hotel Gewandhaus?
NATIVE	what do you think about the hotel Gewandhaus?
Word to Word	what do you from the hotel Gewandhaus?
Phrase to Phrase	what do you think of the hotel Gewandhaus

Table 2.4: Comparison of Word-Based and Phrase-Based translation

2.4.7 Bilingual phrases-model building

In order to build the translation model we need to extract the bilingual phrases from the alignments provided by GIZA++. Basically, a bilingual phrase is a pair of m source words and n target words. For extraction from a bilingual word aligned training corpus, we pose two additional constraints:

- The words are consecutive
- They are consistent with the word alignment matrix, but there is an additional rule to add words to phrases.

The consistency means that the m source words are aligned only to the n target words and vice versa. The following criterion defines the set of bilingual phrases BP of the sentence pair $(f_j^J; e_i^I)$ that is consistent with the word alignment matrix A:

$$BP(f_1^J, e_1^I, A) = \left\{ \left(f_j^{j+m}, e_i^{m+n} \right) : \forall (i', j') \in A : j \leq j' \leq j+m \leftrightarrow i \leq i' \leq i+n \right\} \tag{2.13}$$

We have already mentioned that bilingual phrases must be consecutive and consistent with the alignment. In many alignments however, GIZA++ aligns one word with a sequence of words that are not consecutive. To deal with this situation an additional rule has been added; words are separated into groups of consecutive words and then we select the maximum from

every group[8] and compare it with the position of the word in the source sentence with which it is aligned. The maximum which is closest to the source word defines the phrase to be aligned to the word. Phrases or words that are rejected are not ignored, but are regarded as being created from NULL. These words are treated as wildcards, and are put were suitable. For example, the alignment
NULL (5) Mary (1) did ( ) not (2) slap (3 4) the (6 7) green (9) witch (8)
would be transformed into
Mary (1) did ( ) not (2) slap (3 4 5) the (6 7) green (9) witch (8)
As we see the word created from NULL is added to the first phrase which does not become inconsistent.

The resulting file, containing the bilingual phrases can be very big[9]. It is, however, known that bilingual phrases need not be longer than three words.[6] As a result we can ignore longer phrases, reducing the file's size down to one tenth of the original size. After rejecting bilingual phrases that appear only once, the size of the file reduces to a size manageable by the computer on which the experiments are run.

Having created the bilingual phrases we are now ready to calculate the translation probability for every phrase, using the following formula

$$Prob(\bar{f}|\bar{n}) = \frac{count(\bar{f},\bar{n})}{\sum_{\bar{f}} count(\bar{f},\bar{n})} \tag{2.14}$$

2.5 Decoding

Having trained both the language and the translation model, it is now possible to translate a foreign sentence f using decoding to solve find $\hat{n}_{best}$ from the formula

$$\hat{n}_{best} = \arg\max_n P(n)P(f|n) \tag{2.15}$$

Since we have created a phrase based translation model, during decoding the foreign input sentence f is segmented into a sequence of I phrases f^I. We assume a uniform probability distribution over all possible segmentations.

Each foreign phrase f_i in f^I is translated into a native phrase n_i. The native phrases may be reordered. Phrase translation is modeled by a probability distribution $\phi(f_i|n_i)$. Recall that due to the Bayes rule, the translation direction is inverted from a modeling standpoint.

[8]Regarding each word's position in the target sentence

[9]Around 12 GB for 600000 sentences

Reordering of the native output phrases is modeled by a relative distortion probability distribution $d(a_i - b_{i-1})$, where a_i denotes the start position of the foreign phrase that was translated into the ith native phrase, and b_{i-1} denotes the end position of the foreign phrase that was translated into the $(i - 1$th native phrase.

We use a simple distortion model $d(a_i - b_{i-1}) = a^{|a_i - b_{i-1} - 1|}$ with an appropriate value for the parameter a.

In order to calibrate the output length, we introduce a factor ω (called word cost) for each generated native word, in addition to the language model $P(n)$. This is a simple means to optimize performance. Usually this factor is larger than 1, biasing towards larger output.

In summary, the best native output sequence $n\hat{_{best}}$ for a given foreign input sequence f is

$$\hat{n}_{best} = \arg\max_n P(n)P(f|n)\omega^{length(n)} \tag{2.16}$$

where $P(f|n)$ is decomposed into

$$P(f|n) = P(f^I|n^I) = \prod_{i=1}^{I} \phi(f_i|n_i)d(a_i - b_{i-1}) \tag{2.17}$$

The decoder used, which implements the above, is Pharaoh[12]. Pharaoh is a beam search decoder for phrase to phrase statistical machine translation which was developed by Philipp Koehn, as part of his Phd thesis[14].

Chapter 3

MORPHOLOGY

3.1 Introduction

In natural languages, instead of using a totally different word for each and every possible meaning, words that convey similar meaning are similar theselves, usually differing in some parts of them (e.g. their endings). This is the basic concept that constitutes the notion of the morphology of a natural language.

In today's world, morphological analysis is an essential component in language engineering applications ranging from spelling error correction to machine translation. Morphology is the study of the way words are built from smaller meaning-bearing units which are called morphemes. In the information retrieval domain, the similar, but not identical, problem is called stemming, which usually deals with removing endings from words, leaving the stem (root) of the word. However, a full morphological analysis is more than that, and is usually regarded as a segmentation of the word into morphemes combined with an analysis of the interaction of these morphemes that determine the syntactic class of the word form as a whole.

In this chapter we will mention different approaches to the problem of building a morphology for a natural language and describe especially Linguistica, which will be used in conjunction with some heuristics we propose in order to build a morphological analysis used by a statistical machine translation system[20].

3.2 Essential background on morphology

3.2.1 Morphemes and the kinds of morphologies

Morphemes are defined as the minimal meaning-bearing units in a language. Apart from the stem of a word, a morpheme can be an affix, which usually provides additional meaning of some kind to the main concept that is provided by the stem. An affix may be a prefix, suffix, circumfix or infix, whether it precedes the stem, follows it, does both or is being inserted in it, accordingly. Prefixes and suffixes (and circumfixes as well, since they may be viewed as a combination of a prefix and a suffix) are often called concatenative morphology, since a word is composed of a number of morphemes concatenated together. In some languages, morphemes are combined in complex ways, using what is called nonconcatenative morphology. Another kind of this type is the templatic morphology that is very common in languages like Arabic, Hebrew etc. and uses root words and templates that transform them

There are two broad classes of ways to form words from morphemes: inflection and derivation and thus we speak of inflectional or derivational morphology. These two are partially overlapping, since the borders between them are usually not absolutely clear. Inflection mostly deals with the usage of affixes, while derivation is the combination of a word stem with a grammatical morpheme usually resulting in a word of a different class, often with a meaning hard to predict exactly.

Three general classes of linguistic knowledge are needed in order to build a morphological parser:

Lexicon The list of stems and affixes, together with basic information about them.

Morphotactics The model of morpheme ordering that explains which classes of morphemes can follow the other classes of morphemes inside a word.

Orthographic rules Spelling changes that occur due to morpheme attachment.

3.2.2 Learning a morphology

In recent years, there has been much interest in computational models that learn aspects of the morphology of a natural language from raw or structured data. These models are of great practical interest, minimizing the expert resources or need of linguistics ion order to develop stemmers and analyzers.

There are three distinct ways of learning a language' s morphology:

Supervised learning The data consists of a set of pair of words.

Unsupervised learning The data consists of a single set of all the words in the corpus.

Partially supervised learning The data consists of two sets of words, without any indication of the relationship between the individual words.

We will mostly deal with unsupervised learning, since such methods may be used with untagged corpus which is often the case, performing morphological analysis based only on a corpus. This can be a valuable tool that may be used in statistical machine translation, where the system is being trained using such untagged corpora.

3.3 Different approaches

3.3.1 Introduction

In this section, the most important approaches of (mostly) unsupervised morphology learning are presented. One way to categorize the existing approaches on this matter is by evaluating whether human input is provided in the process of deriving the morphology and whether the goal is to only obtain affixes or to perform a complete morphological analysis. According to this categorization, we may therefore cluster the various approaches and techniques as follows:

- Bootstrapping using a knowledge source
- Obtaining affix inventories
- Performing a complete morphological analysis

For the first two categories we will provide short descriptions, while for the third one we will describe in detail an example application.

3.3.2 Bootstrapping using a knowledge source

A first approach in obtaining morphologies is to begin with some initial humanlabeled source from which to induce other morphological components. Although their work may be more suited to information retrieval (IR), Xu and Croft[15] are proposing a technique that is an example to this case. They are basing their work around the hypothesis that the word forms that should be conflated for a given corpus will co-occur in documents from that corpus. They use a co-occurrence measure to modify an initial set of conflation classes generated by a stemmer, refining the output of the well known Porter stemmer. This corpus-based stemming automatically modifies the equivalence classes (conflation sets) to suit the characteristics of a given text corpus. They perform experiments in English and Spanish, but they do agree that generating the initial conflation classes in languages with more complex morphologies may be a problem.

3.3.3 Obtaining affix inventories

A second, knowledge free category of research has focused on obtaining affix inventories. DeJean[16] is inspired by the works of Zellig Harris[17], a distributional approach where the distribution of an element is the set of the environments in which it occurs. His work uses untagged and non artificial corpora without specific knowledge about the studied language. The algorithm is divided into three steps: the first step computes the list of the most frequent morphemes, which is being extended in the second step by segmenting words with the help of the morphemes already generated, while the third step consists in the segmentation of all the words with the morphemes obtained at the second step. A symmetric procedure can be used to identify prefixes; the letters of the words are just reversed. Morpheme boundaries for the most frequent morphemes are discovered when the number of different letters that are found to follow some sequence of letters is higher than a threshold.

3.4 Performing a complete morphological analysis-Linguistica

3.4.1 Introduction

Although the work presented in section 3.3 does induce some information about a language's morphology, finding just the affixes of a language is not a complete morphological analysis of the specific language. Another knowledge-free category of research attempts to induce a complete analysis of the morphology for each word of a corpus. In this section we will describe such an approach, Linguistica, based on Minimum Description Length analysis, which we have used in this project.

3.4.2 Minimum Description Length Model

The central idea of minimum description length (MDL) analysis[19] is composed of four parts:

1. A model of a set of data assigns a probability distribution to the sample space from which the data is assumed to be drawn.
2. The model can then be used to assign a compress length to the data, using familiar information-theoretic notions.
3. The model can itself be assigned a length.
4. The optimal analysis of the data is the one for which the sum of the length of the compressed data and the length of the model is the smallest.

In other words, we seek a minimally compact specification for both the model and the data. Linguistica tries to analyze words into morphemes, using MDL as guideline. In order to provide a morphology to evaluate using MDL, first bootstrapping heuristics are needed that provide an initial morphology.

3.4.3 Heuristics for word segmentation

Two heuristics are used to produce an initial morphology analysis.

- The first one (called take-all-splits), considers for each word of length of length l all the possible cuts into $w_{1,i} + wi + 1, l,\ 1 \leq i < l$. For each cut

  $$H(w_{1,i}, w_{i+1,l}) = -(ilogfreq(stem = w_{1,i}) + (1-i)logfreq(suffix = w_{i+1,l})) \tag{3.1}$$

  is computed; then it is used in the following formula to assign a probability to the cut of w into $w_{1,i} + wi + 1, l$.

  $$prob(w = w_{1,i} + wi + 1, l) = \frac{1}{Z} e^{-H(w_{1,i} + wi + 1, l)} \tag{3.2}$$

  where

  $$Z = \sum_{i=1}^{n-1} H(w_{1,i} + wi + 1, l) \tag{3.3}$$

  For each word the best parse is noted, and then we iterate until no word changes, which typical takes less than five iterations.

- Using the convention that each word ends with an end-of-word symbol we compute the counts of all n-counts between two and six letters (including the end of word). Then for each ngram $[n_1 n_1 \ldots n_k]$ we compute

$$\frac{[n_1 n_1 \ldots n_k]}{total\ count\ of ngrams} log \frac{[n_1 n_2 \ldots n_k]}{[n_1][n_2]\ldots[n_k]'} \tag{3.4}$$

  The top 100 ngrams on the basis of this measure are chosen as candidate suffixes. Then all words are parsed into stem plus suffix, if possible, using a parse from the candidate set. For those words that more than one parsing are possible, we keep the most probable, according to the previous heuristic.

Consequently, for each stem we make a list of all the suffixes which appear with it, called a signature. Stems having the same signatures are merged. Initially all signatures with only one stem (which account for about 90% of the initial signatures) are removed, as well as those with only one stem. The remaining are called regular signatures. The resulting signatures are of the form

$$\left\{ \begin{array}{c} stem_1 \\ stem_2 \\ stem_3 \end{array} \right\} \left\{ \begin{array}{c} suffix_1 \\ suffix_2 \end{array} \right\} \tag{3.5}$$

Variations of the resulting grammar are considered and adopted only if they reduce the description length of the grammar and the corpus. First each suffix is tested to see if it is a concatenation of two independent suffixes. Then suffixes in the same signature are tested to see if they begin with the same letter or sequence of letters, so that these letters can be considered part of the preceding stems. Finally signatures with only a small number of stems are checked to see if they are worth keeping, or discarding them leads to a better model.

3.5 Heuristics proposed

3.5.1 Introduction

Using Linguistica on a corpus we can have a morphological analysis. It is possible, however, especially when the available corpus is small[1] that the produced morphology will not be very good, both in terms of precision and

[1]Even worse when dealing with a language with rich morphology

recall. Linguistica offers the chance of adjusting some parameters, to influence the resulting morphology. However, to use them one must take into account the way the morphology is built. We have tried to offer a simple and cheap (both in terms of time and computational power needed) way of increasing the precision of the resulting morphological analysis, on the expense of recall.

3.5.2 Heuristic proposed

Examination of the resulting morphological analysis provided by Linguistica easily leads to an observation. In most words mistakenly analyzed, the error is assigning stem characters to the suffix. The opposite error, assigning suffix characters to the stem is not so important since it is more easy to identify a stem with extra characters in the end than a chopped stem. The problem of false identification is even more important when dealing with short words, where removal of the suffix usually leaves a very short stem (maybe two or three characters long), which is possibly useless for training a statistical machine translation system. To overcome these problems we use a heuristic rule which uses two parameters

- The length of the words l.
- The ratio r of the length of the suffix divided by the length of the whole word.

We examine every word analyzed by Linguistica. We adopt the analysis only for words that have $l_{word} > l_0$ and $r_{word} < r_0$, or else discard it.

3.5.3 Results-evaluation

In order to be able to choose values for r_0 and l_0 we carried out a simple experiment. We used Lingustica to provide morphological analysis based on a 1M token Greek corpus. Then we randomly picked 1k words (2k tokens) for which Linguistica had produced morphological analysis. To evaluate the performance of the heuristic, a human judge decided for each word if the analysis was correct or mistaken. The results, using different values for r_0 and l_0 are shown in Table 3.1.

r_0	l_0	Precision(%)
1	0	79
0.2	0	89
0.2	4	89
0.2	5	89
0.2	6	93
0.3	0	84
0.3	4	84
0.3	5	90
0.3	6	94

Table 3.1: Results

3.6 Using morphology in SMT

In statistical machine translation, the translation problem is posed as a posterior probability maximization problem.[2] If we consider W_s and W_t to be word sequences for the source and target languages respectively, then the problem can be formulated as:

$$\hat{W}_t = \arg\max_{W_t} P(W_t|W_s) \tag{3.6}$$

where $\hat{W}_t$ is the translated sequence of words in the target language.

Using the algorithms described in Sections 3.4.3, 3.5, we come up with knowledge about the morphology of both the source and target languages. This knowledge can be represented as a (deterministic or statistical) mapping from a sequence of words W to a sequence of stems S. These stems may be extracted in general by the use of a statistical morphological analyzer (stemmer) that computes the probabilities $P(S|W)$. Also, a morphological generator is defined as the model that computes the reverse probabilities $P(W|S)$.

Let us consider S_s and S_t to be sequences of stems for the source and target languages respectively. Then a stem-to-stem machine translation system can be formulated as:

$$\hat{S}_t = \arg\max_{S_t} P(S_t|S_s) \tag{3.7}$$

Using the statistical models for the morphological analyzer $P(S_s|W_s)$ and morphological generator $P(W_t|S_t)$ for the source and target languages respectively, as well as the stem-to-stem translation model $P(S_t|S_s)$ we may

[2]The method described in this section is work of Karageorgakis et.al. presented in [20].

write

$$\begin{aligned}\hat{W}_t &= \arg\max P(W_t|W_s) \\ &= \arg\max_{W_t} \sum_{S_t,S_s} P(W_t, S_t, S_s|W_s) \\ &= \arg\max_{W_t} \sum_{S_t,S_s} P(W_t|S_t, S_s, W_s)P(S_t|S_s, W_s)P(S_s|W_s) \\ &= \arg\max_{W_t} \sum_{S_t,S_s} P(W_t|S_t)P(S_t|S_s)P(S_s|W_s) \qquad (3.8)\end{aligned}$$

provided that W_t, S_s are conditionally independent given S_t; W_t, W_s are conditionally independent given S_t, S_s; and W_s,S_t are conditionally independent given S_s. This equation corresponds to a word-to-word translation model; however, in this system word to word translation is performed via the stem to stem system, i.e., $W_s \rightarrow S_s \rightarrow S_t \rightarrow W_t$.

Eq. (3.8) can be further simplified as follows: the mapping $S \rightarrow W$ is a many to one mapping and $P(S_s|W_s) = 1$, because the mapping $W_s \rightarrow S_s$ is deterministic, so the double summation at Eq. (3.8) becomes a single summation over S_t only, as follows:

$$\hat{W}_t = \arg\max_{W_t} \sum_{S_t} P(W_t|S_t)P(S_t|S_s) \qquad (3.9)$$

We refer to this system as the morphological or stem-based SMT system.

Once we have built the morphological SMT system, we need to combine it with the traditional lexical SMT system. This combination can be done by assuming that each SMT system computes probabilities independently of each other, i.e.,

$$\hat{W}_t = \arg\max_{W_t} [P(W_t|W_s)]^{w_0} [\sum_{S_t} P(W_t|S_t)P(S_t|S_s)]^{w_1} \qquad (3.10)$$

where w_0 and w_1 are weights that model the "confidence" we have in each translation, the lexical and the morphological SMT models. By combining these two SMT systems we hope that we overcome the weakness of the conditional independence assumptions of Eq. (3.8). This combination may be implemented at an early or at a late stage (e.g., word lattice combination).

3.6.1 Combined Lexical-Morphological system implementation

The combined lexical and morphological SMT system is implemented using late integration and lattice re-scoring according to the following steps:

1. The lexical SMT system that computes the probabilities $P(W_t|W_s)$ is built.

2. The training corpus is stemmed using the unsupervised rules derived from the Linguistica system.

3. The stemmed corpus is used to derive the morphological (stem) SMT system that computes the probabilities $P(S_t|S_s)$.

4. Every sentence in the evaluation corpus is decoded using the lexical SMT system producing a lattice of possible word-level translations. This lattice is then represented as a finite state acceptor F_W.

5. Every sentence in the evaluation corpus is stemmed and then decoded using the morphological SMT system. The resulting lattice contains all possible stem-level translations and is represented as a finite state acceptor F_S.

6. The stem to word model $P(W_t|S_t)$ in the target language is constructed by running the Linguistica system on the target language corpus and obtaining the morphological signature. The stem to word model is represented as a unweighted (costless) finite state transducer T_{SW}, i.e., in our case, we assume that all possible words that can be generated from a stem are equiprobable [3].

7. The stem acceptor F_S and the stem to word transducer T_{SW} are composed to obtain a stem to word mapping; the resulting transducer is projected to its output symbols to obtain the finite state acceptor $F_{W'}$.

8. F_W and $F_{W'}$ acceptors are re-weighted (weights multiplied) by the factors w_0 and w_1 as discussed above (in practice, we don't weight F_W and w_0 is always 1.).

9. The weighted acceptors F_W and $F_{W'}$ are intersected and the best path of the intersection is found using Viterbi decoding. The best path T' represents the translated sentence of the combined lexical-morphological SMT system.

The process that has been described above can be formulated as follows:

$$T' = \text{bestpath}\{([F_S \circ T_{SW}]_2 * w_1) \cap F_W\}$$

[3]In order to guarantee non-empty composition in the next step, all words contained in F_W and F_S were added as identity mappings in T_{SW} and then Kleene closure was applied to T_{SW}.

where $\circ$ represents composition, $\cap$ intersection, $*$ weighting and $_2$ projection to the output symbols; T', F_S, T_{SW}, F_W and w_1 are defined above.

Chapter 4

IMPROVEMENTS ON BASELINE SYSTEM

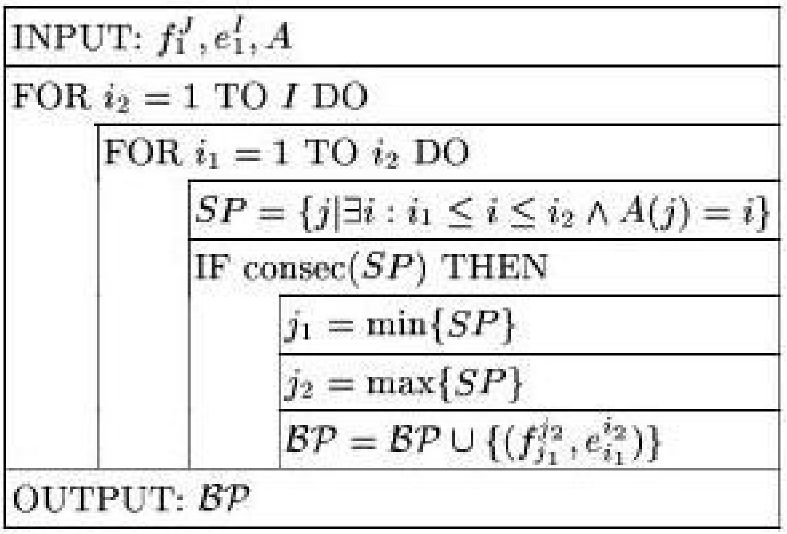
4.1 Introduction

The baseline system uses Giza++ to align a bilingual corpus. In order to translate from a native to a foreign language, the foreign to native alignment is used to extract bilingual phrases and build translation models. However the 1-N restriction between native and foreign words, can seriously degrade the performance of the system, especially when dealing with highly inflective languages. Furthermore, using only one alignment implies information loss, which may be significant when training resources are sparse. The modifications introduced to the baseline system are the following:

- Algorithm for bilingual phrase extraction
- Alignments used for bilingual phrase extraction

4.2 Bilingual phrase extraction

The procedure used to extract bilingual phrases is described in Zens et. al.[5]. It is essentially the same with the one used by the baseline system, described in section 2.4.7 on page 19. The only difference is that there is no additional rule in the second constraint for extracting bilingual phrases. The algorithm in Figure4.1 computes the set of bilingual phrases BP with the assumption that the alignment is a function $A : \{1, \ldots, J\} \rightarrow \{1, \ldots, I\}$



```
INPUT: $f_1^J, e_1^I, A$
FOR $i_2 = 1$ TO $I$ DO
    FOR $i_1 = 1$ TO $i_2$ DO
        $SP = \{j | \exists i : i_1 \leq i \leq i_2 \land A(j) = i\}$
        IF consec($SP$) THEN
            $j_1 = \min\{SP\}$
            $j_2 = \max\{SP\}$
            $\mathcal{BP} = \mathcal{BP} \cup \{(f_{j_1}^{j_2}, e_{i_1}^{i_2})\}$
OUTPUT: $\mathcal{BP}$
```

Figure 4.1: Algorithm for extracting bilingual phrases

4.3 Heuristics proposed

The word alignment produced by Giza++ only allows one target word to be aligned with exactly one source word. In order to overcome this restriction, the corpus is aligned bidirectionally. That is, Giza++ is run two times, using initially the foreign language as source and the native as target (baseline system), and then exchanging the two languages. The result is that now we have two alignments. Intersecting these alignments, results in a high precision/low recall alignment. On the opposite hand the union is a high recall/low precision alignment.

4.3.1 diag_and

Our approach, based on Koehn et.al.[6], is to explore the space between these two alignments, trying to keep precision near intersection levels and recall near union levels. Starting from the intersection alignment, we add points that exist in the union alignment, using the following heuristic (which we will call *diag_and*):

1. First we expand to only directly adjacent points (meaning points that connect to an already adjacent point) under the constraint that the new alignment point connects at least one previously unaligned word. This is done iteratively until no new alignment points can be added.

2. Finally non-adjacent points are added, with otherwise the same requirements.

This heuristic is called **diag_and**. As an example, in Figure 4.2 we can see the two alignments produced by GIZA++ and in Figure 4.3 the resulting alignment, where the points added are marked with arrows.

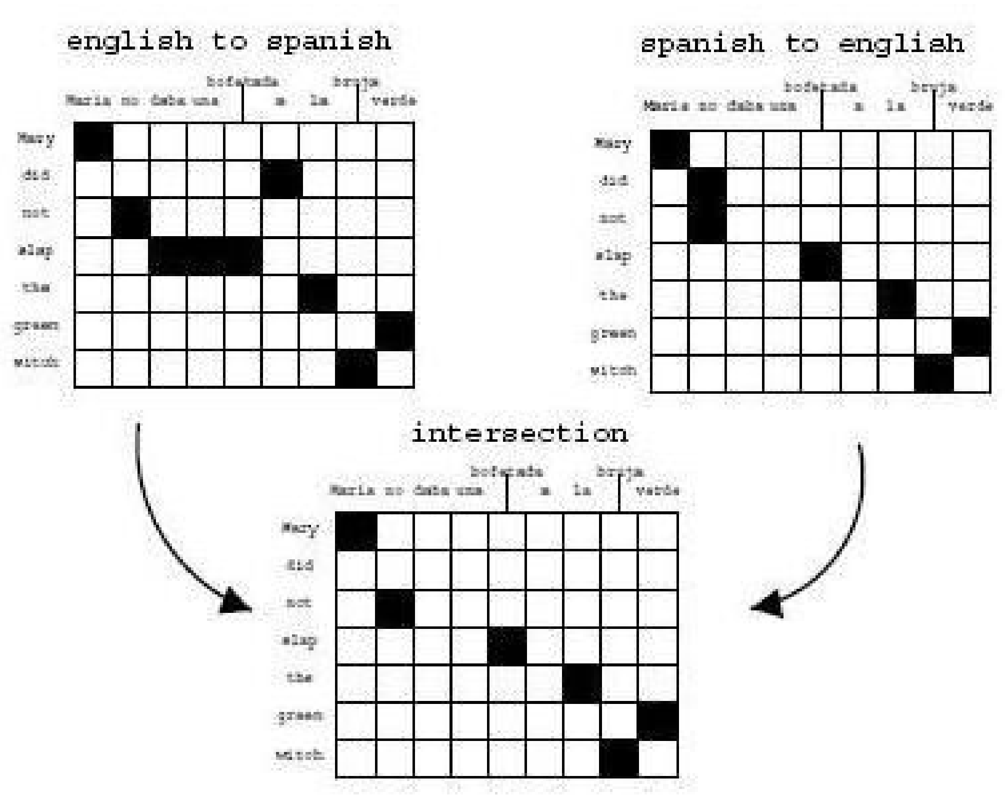


Figure 4.2: Intersection of alignments (from Pharaoh manual[13])

4.3.2 weight

A possible refinement to **diag_and** is **weight**. While in **diag_and** every alignment has equal weight (unary), in **weight** each alignment is assigned a weight according to its similarity to the intersection. If we define the alignment as a function

$$F(i,j) = \begin{cases} 1 & \textit{point } (i,j) \textit{ belongs to alignment} \\ 0 & \textit{else} \end{cases} \quad 1 \leq i \leq I, 1 \leq j \leq J \tag{4.1}$$

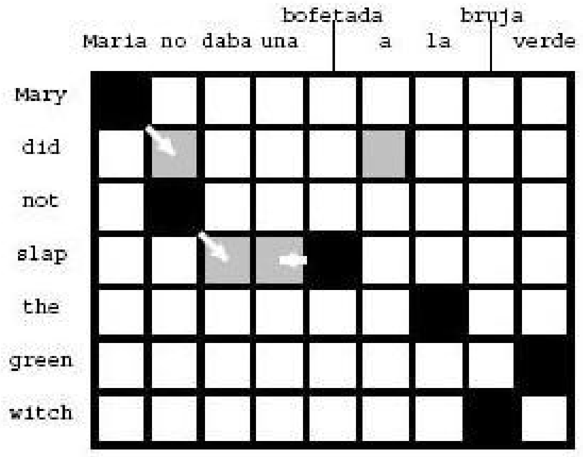


Figure 4.3: Alignment used by *diag_and* (from Pharaoh manual[13])

where I is the length of the source sentence and is J the length of the target sentence, then for each alignment we can compute the metric

$$R(I, W) = \frac{\sum_{i,j} I(i,j)}{\sum_{i,j} W(i,j)} \tag{4.2}$$

where $I(i,j)$ denotes the intersection alignment and $W(i,j)$ the resulting alignment. The distribution of $R(I,W)$ for a set of 358887 sentences is shown in figures 4.4, 4.5. In figure 4.6 is shown the mean value of $R(I,W)$ for every sentence length.

Then, a weight is assigned to every alignment, according to $R(I,W)$. By intuition, alignments with larger values for $R(I,W)$ are better than others with small values, because that means there is bigger similarity between the two alignments. It seems, then, logical to emphasize more on such alignments. Several different weight schemes were implemented.They can be split into two categories:

- Discrete weights
- Continuous weights

In the first category we have implemented four different schemes; the weight is assigned using equations 4.3, 4.4, 4.5, 4.6 respectively. On the other hand, we can use a continuous function of $R(I,W)$ to assign a weight. Functions used are R, $10R$, $100R^2$, e^R, R^2, $\frac{1}{1.0001-R}$, R^3, R^4, R^5, R^6, e^{R^2}, e^{2R}, e^{5R}, e^{10R}, R^7, R^8, R^9, R^{10}.

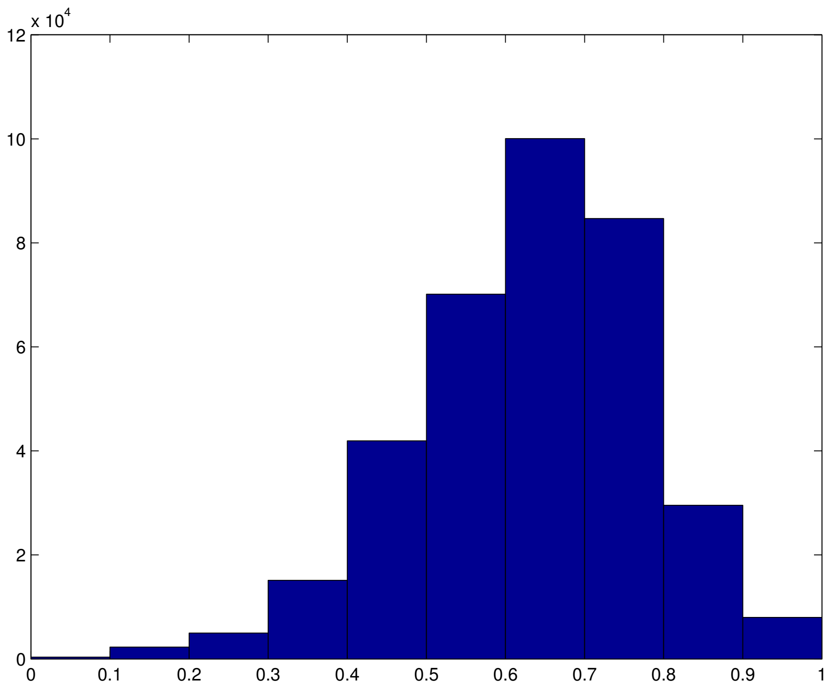


Figure 4.4: $R(I, W)$ distribution

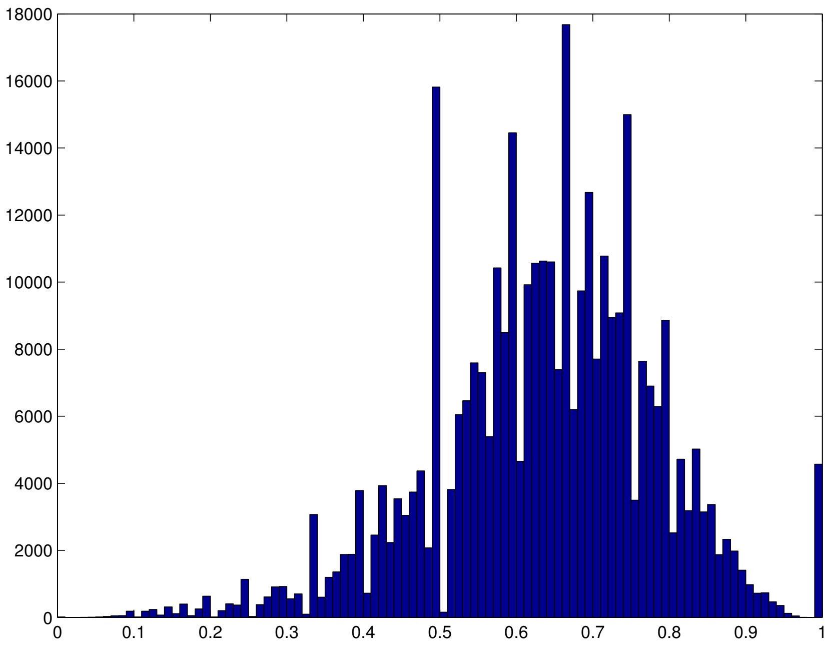


Figure 4.5: $R(I, W)$ distribution

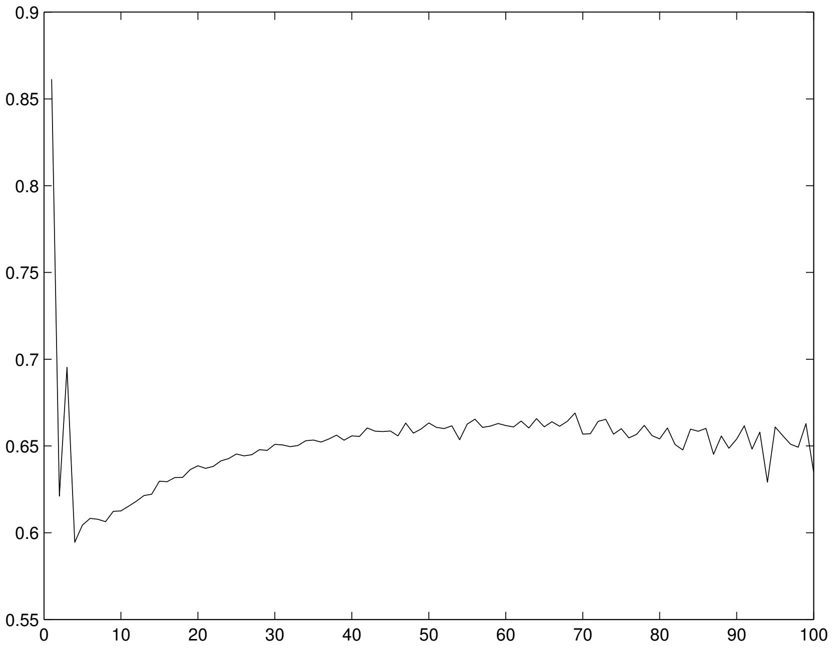


Figure 4.6: $R(length)$ distribution

$$weight(R) = \begin{cases} 1 & R < 0.6 \\ 1.2 & 0.6 \leq R < 0.7 \\ 2 & 0.7 \leq R < 0.8 \\ 9 & 0.8 \ < R < 0.9 \\ 24 & 0.9 \leq R < 1 \\ 30 & R = 1 \end{cases} \tag{4.3}$$

$$weight(R) = \begin{cases} 0.5 & R < 0.6 \\ 1 & 0.6 \leq R < 0.7 \\ 2 & 0.7 \leq R < 0.8 \\ 3 & 0.8 \leq R < 0.9 \\ 4 & 0.9 < \ leqR < 1 \\ 8 & R = 1 \end{cases} \tag{4.4}$$

$$weight(R) = \begin{cases} 0.5 & R < 0.6 \\ 1 & 0.6 \leq R < 0.7 \\ 4 & 0.7 \leq R < 0.8 \\ 8 & 0.8 \leq R < 0.9 \\ 16 & 0.9 \leq R < 1 \\ 32 & R = 1 \end{cases} \tag{4.5}$$

$$weight(R) = \begin{cases} 0.5 & R < 0.6 \\ 2 & 0.6 \leq R < 0.7 \\ 5 & 0.7 \leq R < 0.8 \\ 10 & 0.8 \leq R < 0.9 \\ 15 & 0.9 \leq R \leq 1 \\ 30 & R = 1 \end{cases} \tag{4.6}$$

4.3.3 exact

Exact is very similar to **weight.** Instead of applying the same weight to every alignment point, every alignment point gets a weight depending solely on its existence in the intersection alignment, as shown in equation (4.7). That is, if a point exists in the intersection alignment its weight is 2, else 1. Then for every native word j the weights of its points are summed and divided by their count, as shown in equation (4.8).The result is the weight of the word. The distribution of the weight of the word w for a set of 358887 sentences is shown in figure (4.7). When making the translation rule, we add each word's weight and divide by their count. Using the result in equation (4.9) we get the weight of the translation rule.

$$W(i,j) = I(i,j) + A(i,j) \tag{4.7}$$

where $I(i,j)$is the alignment of the intersection between the outputs of Giza++ and $A(i,j)$is the alignment used

$$w(j) = \frac{\sum_i W(i,j)}{\sum_i A(i,j)} \tag{4.8}$$

$$weight(w) = \begin{cases} 1 & w < 1.75 \\ 2 & 1.75 \leq w < 1.8 \\ 3 & 1.8 \leq w < 1.85 \\ 4 & 1.85 \leq w < 1.9 \\ 5 & 1.9 \leq w < 1.95 \\ 6 & 1.95 \leq w \leq 2 \end{cases} \tag{4.9}$$

4.3.4 lenwe

The bilingual phrase distribution versus their size is shown in figure (4.8). Since bigger bilingual phrases are fewer than the smaller ones but probably offer better translations, each bilingual phrase is given a weight according to its size, as shown in equation (4.10).

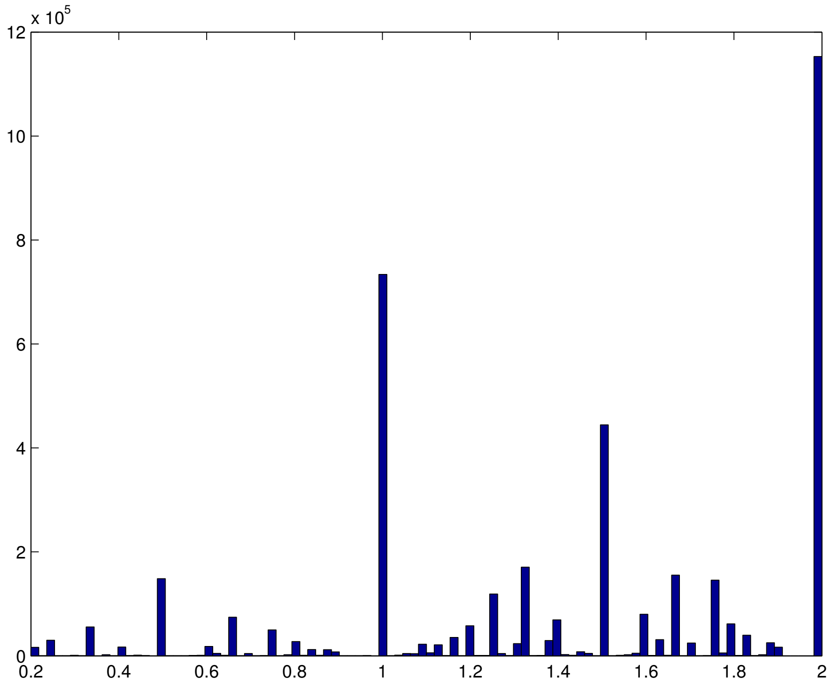


Figure 4.7: w distribution

$$W(BP) = \begin{cases} 3.73154 & w = 1 \\ 4.13777 & w = 2 \\ 5.11651 & w = 3 \\ 6.23713 & w = 4 \\ 7.43149 & w = 5 \end{cases} \tag{4.10}$$

4.3.5 cor_icor

Instead of applying a heuristic rule to create a new alignment using the output of GIZA++, it is possible to use both alignments separately. Since, as the results suggest, the base alignment outperforms the inverse alignment, we assign a weight to each alignment, depending on where it belongs; base or inverse. We have used seven weighting schemes, described in Table4.1.

4.3.6 int

int is the model created using the intersection of the two alignments. If $F2N(i,j)$ is the foreign to native alignment and $N2F(i,j)$ is the native to

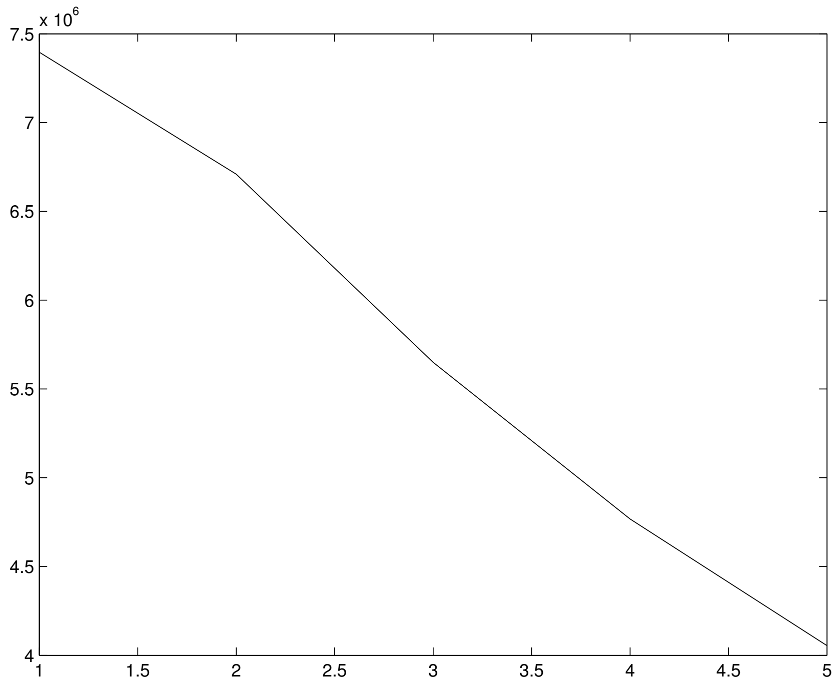


Figure 4.8: Bilingual phrase length distribution

weight scheme	1	2	3	4	5	6	7
base weight	2	3	1	1	4	5	6
inverse weight	1	1	0.2	1	1	1	1

Table 4.1: Weight schemes for **cor_icor**

foreign alignment, then the intersection alignment is given by the formula $INT(i,j) = F2N(i,j) \wedge N2F(i,j)$

4.3.7 uni

uni is the model created using the union of the two alignments.If $F2N(i,j)$ is the foreign to native alignment and $N2F(i,j)$ is the native to foreign alignment, then the intersection alignment is given by the formula $INT(i,j) = F2N(i,j) \vee N2F(i,j)$

4.3.8 icor

icor is the model created using only the inverse (native to foreign) alignment.

Chapter 5

RESULTS - EVALUATION

5.1 Introduction

Human evaluations of machine translation are extensive but expensive. Human evaluations can take months to finish and involve human labor that can not be reused. This fact poses a great problem in the development of machine translation systems, were the impact of different methods must be appreciated quickly and and cost-effectively. As a result quick, cheap and reliable evaluation of the resulting translation quality can improve the performance and reduce the development time and cost. Consequently, many machine translation evaluation techniques have been proposed in the past[21]. The evaluation techniques used in this project, BLEU[22] and NIST[23], which are described in the next sections, are inexpensive, quick, language-independent and correlate highly with human evaluation.

5.2 BLEU

BLEU is based on the idea that the closer a machine translation is to a professional human translation, the better it is. Typically, there are many "perfect" translations of a given source sentence. These translations may vary in word choice or in word order even when they use the same words. And yet humans can clearly distinguish a good translation from a bad one. For example, consider these two candidate translations of a Chinese source sentence:

Candidate 1 It is a guide to action which ensures that the military always obeys the commands of the party.

Candidate 2 It is to insure the troops forever hearing the activity guidebook that party direct.

Although they appear to be on the same subject, they differ markedly in quality. For comparison, we provide three reference human translations of the same sentence below.

Reference 1 It is a guide to action that ensures that the military will forever heed Party commands.

Reference 2 It is the guiding principle which guarantees the military forces always being under the command of the Party.

Reference 3 It is the practical guide for the army always to heed the directions of the party.

It is clear that the good translation, Candidate 1, shares many words and phrases (in other words ngrams) with these three reference translations, while Candidate 2 does not. The cornerstone of BLEU is the modified precision meter. To compute this, one first counts the maximum number of times a ngram appears in any single reference translation. Next, one clips ($Count_{clip} = min(Count, Max\_Ref\_Count)$) the total count of each candidate word by its maximum reference count, adds these clipped counts up, and divides by the total (unclipped) number of candidate words. The above procedure is formulated as

$$p_n = \frac{\sum_{C \in \{Candidates\}} \sum_{ngram \in C} Count_{clip}(ngram)}{\sum_{C' \in \{Candidates\}} \sum_{ngram' \in C'} Count_{clip}(ngram')} \tag{5.1}$$

Modified precision meter gives high scores to sentences that match the reference sentence in word choice and word order. By definition it penalizes sentences longer than the references, however it does not penalize shorter sentences. To overcome this restriction a brevity penalty is introduced; it is defined by the following equation

$$BP = \begin{cases} 1 & if\ c > r \\ e^{(1-r/c)} & if\ c \leq r \end{cases} \tag{5.2}$$

where c is the candidate of the candidate translation and r the effective reference corpus length[1]. Then

$$BLEU = BP \cdot exp\left(\sum_{n=1}^{N} w_n log p_n\right) \tag{5.3}$$

where $N = 4$ and $w_n = 1/N$.

[1]Sum of the best match length(closest length of corresponding reference phrases) for each candidate sentence in the corpus

5.3 NIST

NIST is an alternative statistical machine translation quality automated evaluation technique. Building and using NIST is motivated by two characteristics of BLEU.

- First, the IBM BLEU formulation uses a geometric mean of co-occurrences over N (rank of ngrams). This makes the score equally sensitive to proportional differences in co-occurrence for all N. As a result, there exists the potential of counterproductive variance due to low co-occurrences for the larger values of N. An alternative would be to use an arithmetic average of N-gram counts rather than a geometric average.

- Second, note that it might be better to weight more heavily those N-grams that are more informative • i.e., to weight more heavily those N-grams that occur less frequently, according to their information value. This would, in addition, help to combat possible gaming of the scoring algorithm, since those N-grams that are most likely to (co-)occur would add less to the score than less likely N-grams

Information weights were computed using Ngram counts set of reference translations, according to the following equation:

$$Info(w_1 \dots w_n) = log_2\Big(\frac{the\ \#\ of\ occurrences\ w_1 \dots w_{n-1}}{the\ \#\ of\ occurrences\ w_1 \dots w_n}\Big) \tag{5.4}$$

Using the information weights, a modification of IBM's formulation of the score was chosen as the evaluation measure that NIST uses to provide automatic evaluation to support machine translation research. NIST's formula for calculation the score is

$$Score = \sum_{m=1}^{N} \left\{ \sum_{\substack{all\ w_1 \dots w_n \\ that\ co-occur}} Info(w_1 \dots w_n) \Bigg/ \sum_{\substack{all\ w_1 \dots w_n \\ in\ sys\ output}} (1) \right\} exp\left\{ \beta log^2 \left[min\left(\frac{L_{sys}}{\bar{L}_{ref}}, 1 \right) \right] \right\} \tag{5.5}$$

where β is chosen to make the brevity penalty factor 0.5 when the number of words in the system output is $2/3^{rds}$ of the average number of words in the reference translation, $\bar{L}_{ref}$ is the aveage number of words in a reference translation, averaged over all reference translation andL_{sys} is the number of words in the translation being scored.

Notice that, in addition to the calculation of the co-occurrence score itself, a change was also made to the brevity penalty. This change was made to minimize the impact on the score of small variations in the length of a translation. This preserves the original motivation of including a brevity penalty (which is to help prevent gaming the evaluation measure) while reducing the contributions of length variations to the score for small variations. Figure 5.1 gives a comparison of the two brevity penalty factors.

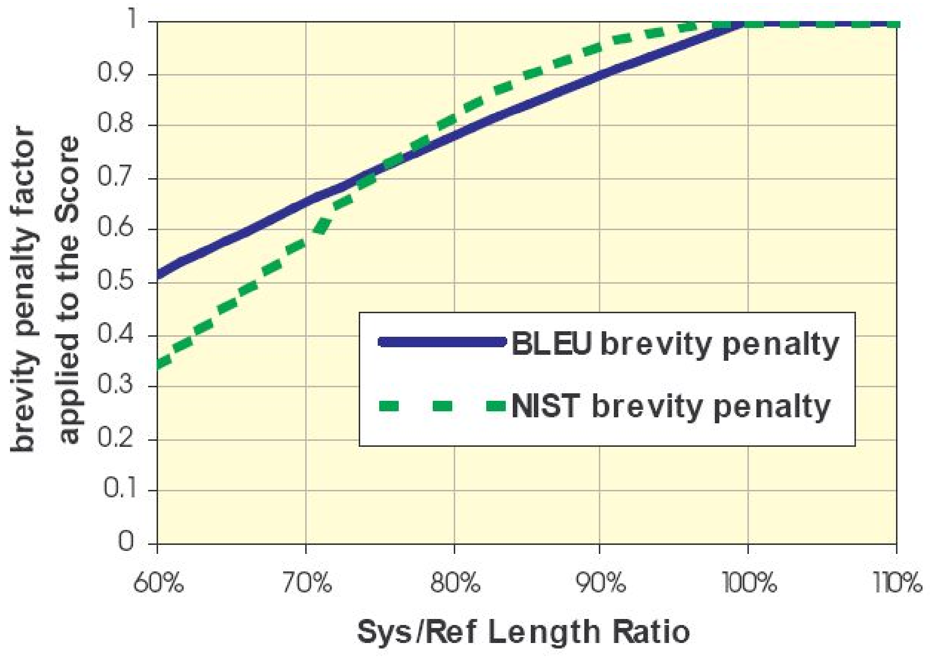


Figure 5.1: Comparison of the BLEU and NIST brevity penalty factor (From NIST[23])

In order to evaluate the difference between NIST and BLEU the F-ratio is introduced.F-ratio is the between system score variance divided by within-system variance. The between system variance is the variance of the average system scores across different systems, and the within-system variance is the variance of document scores for a given system, computed across different documents an different reference translation and then pooled over all systems. The NIST evaluation score is compared with IBM's original BLEU score in Figure 5.2 and Figure 5.3.Is is demonstrated that, for human judgments of Adequacy[2], the NIST score correlates better than the BLEU score on all of the corpora. For Fluency[3] the judgments, however, the NIST score correlates better than the BLEU score only on the Chinese corpus. This may be a mere random statistical difference between corpora. Or alternatively, this may be a consequence of different human judgment criteria or procedures. (The

[2]Each segment is scored according to how well the meaning conveyed by the reference translation is also conveyed by the evaluated segment.

[3]How fluent the evaluated translation is.

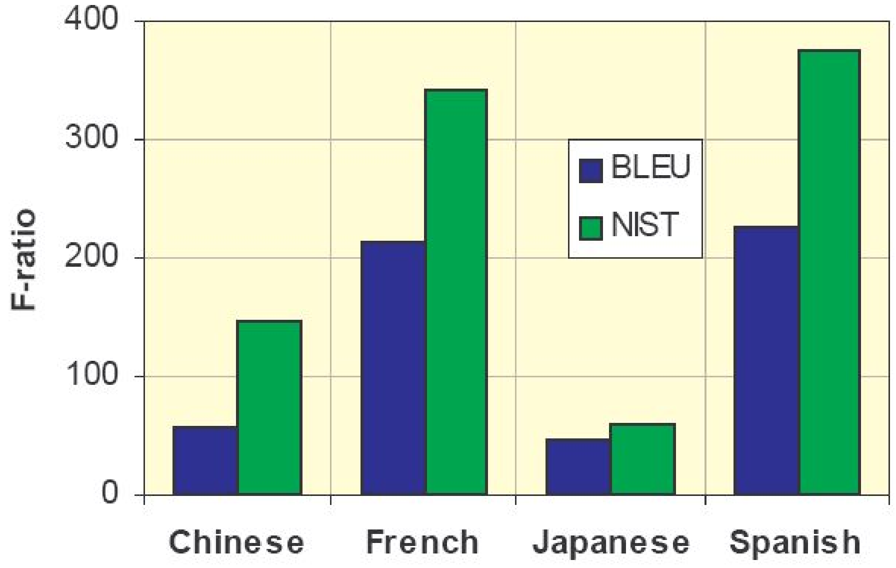


Figure 5.2: F-ratio comparison of the correlation of BLEU and NIST scores for document variance for four corpora (From NIST[23])

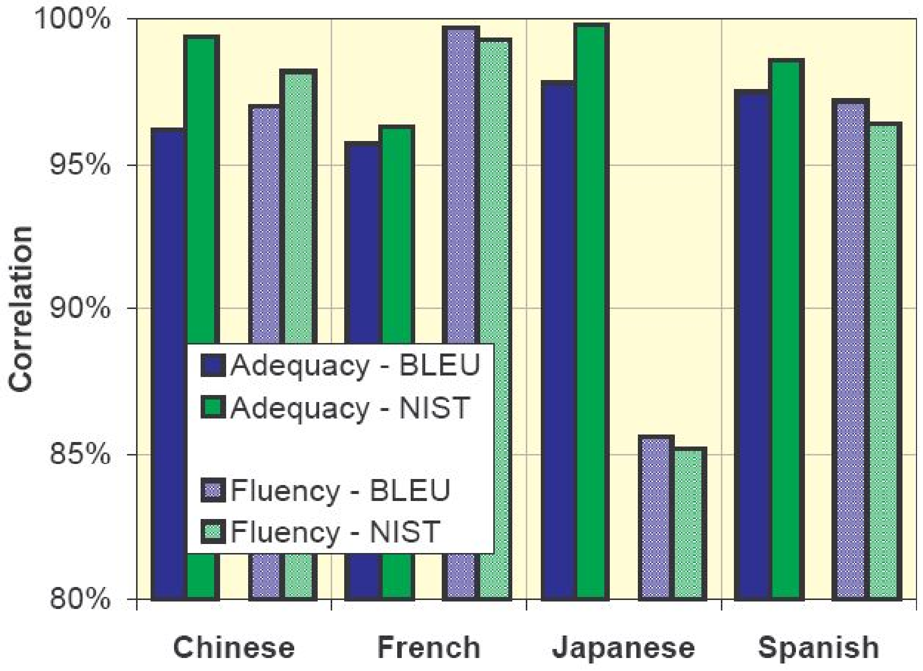


Figure 5.3: Comparison of the correlation of BLEU and NIST scores with human judgments for four corpora (From NIST[23])

Chinese-to-English translations were judged using a different procedure than that used for the other corpora).

5.4 Results for system incorporating morphological knowledge

We provided both systems with an initial test data of approximately 26k sentences. For the 1M corpus, system B yielded approximately 11k sentences that were different compared to those of system A. For the 4M corpus, the different sentences were found to be about 6k. This number of different sentences, is a function of the size of the lattices used by the SMT system, thus enlarging these lattices could produce more different translations.

Table 5.1 summarizes this information; for every training corpus size (TC_s), the number of different sentences that formed the evaluation set are displayed (Ds), as well as the ratio (Ds_r) of the different sentences compared to the total 26k sentences of the test set.

TC_s	Ds	Ds_r
1M	10669	40.57%
4M	6322	24.04%

Table 5.1: Training and evaluation set sizes

For every training corpus size, we performed several experiments by changing the weight w_1 of the FSM containing the morphological information. In order to focus on the real improvement of the new system, our evaluation set does not consist of all the sentences that form the test data, but those that resulted in different translations between the two systems. The results of these experiments are shown in Tables 5.2 and 5.3.

The best score improvement achieved was 0.4326 for the NIST scores and 0.0089 for the BLEU evaluation metrics, which correspond to 14.30% and 14.74% relative score improvement respectively compared to the lexical system. The tables show that smaller weights w_1 provide bigger improvements, i.e., the FSM containing the morphological information should be weighted more. This is probably due to the fact that the statistics of the word stems are better trained than the statistics of the words for training sets of the same size, i.e., the stem model is better trained than the word model. It can also be seen that the incorporation of morphological information provides more improvement for systems that have been trained with smaller data sets, since

TC_s	w_1	N_A	N_B	N_B-N_A	Improvem.
1M	0.05	3.0253	3.4579	0.4326	14.30%
1M	0.1	3.0538	3.4618	0.4080	13.36%
1M	0.2	3.1144	3.4663	0.3519	11.30%
1M	0.3	3.1539	3.4453	0.2914	9.24%
4M	0.1	4.2391	4.4139	0.1748	4.12%

Table 5.2: NIST scores for systems A and B for various combination weights and training corpus sizes

TC_S	w_1	B_A	B_B	B_B-B_A	Improvem.
1M	0.05	0.0604	0.0693	0.0089	14.74%
1M	0.1	0.0611	0.0697	0.0086	14.08%
1M	0.2	0.0629	0.0704	0.0075	11.92%
1M	0.3	0.0635	0.0703	0.0068	10.71%
4M	0.1	0.1006	0.1057	0.0051	5.07%

Table 5.3: BLEU scores for systems A and B for various combination weights and training corpus sizes

the scores for the 1M training corpus are much better than those of the 4M training corpus.

In order to be certain that our test set size is large enough to guarantee true improvement, we perform bootstrap re-sampling [24]. This method has been used in various fields of research, including automatic speech recognition and statistical machine translation [25, 26, 27].

TC_s	w_1	N_d mean	N_d interval	N_B RSD
1M	0.05	0.4325	[0.3996, 0.4654]	0.77%
1M	0.1	0.4082	[0.3751, 0.4405]	0.75%
1M	0.2	0.3520	[0.3199, 0.3836]	0.75%
1M	0.3	0.2913	[0.2600, 0,3217]	0.76%
4M	0.1	0.1748	[0.1298, 0.2204]	0.84%

Table 5.4: 95% confidence intervals for N_d scores (NIST)

Table 5.4 shows the interval mean and the 95% confidence interval for the differences in the NIST scores between systems A and B ($N_d = N_B - N_A$). Assuming the bootstrap hypothesis, we can say that there is 95% confidence that, for example, for the system with $w = 0.01$ the improvement of the NIST score lies between 0.4220 and 0.4887, which corresponds to improvement between 14.06% and 16.28%. The last column of this table also shows the relative standard deviation for the NIST value of the morphological SMT system (*Relative Standard Deviation* or RSD is defined as $(100 * \sigma/\mu)\%$, where μ and σ are the mean and standard deviation respectively). Clearly, the improvements achieved by combining the lexical and the morphological information are statistically significant.

5.5 Results for system using improved alignments

Both NIST and BLEU were used to score the resulting translations. In order to evaluate the performance of each system in different conditions four different corpora were used, with the characteristics shown in Table 5.5. Although training of the translation model can be a time consuming task, we decided to use four training sets, in order to appreciate the performance of the system in different training confitions. Training set 0 provides the opportunity to test the system when training data is not available in large. Combined with the fact that the target language is Greek, a language with rich morphology and much bigger lexicon than English, this training set should give clear signs of how well each systems deals with the problem of data sparseness. One way to appreciate how big this problem is, is to use the ratio of words[4] to tokens[5] in the corpus. Obviously the bigger this figure is, the harder it is to train efficiently the models, since more words will not be seen enough times to estimate realistic probabilities. For test set 0 this figure is 0.049 for Greek and 0.018 for English, while for test set 3 the respective figures are 0.014 and 0.006. As a result the different training sets ensure varying training conditions.

corpus	0		1		2		3	
lang	el	en	el	en	el	en	el	en
tokens	1260076	1261542	2505231	2528720	4473410	4548857	8911269	9030677
words	61725	22832	63448	30835	81983	40137	124121	55373

Table 5.5: Corpora used for training

The same motive leaded to using four different test sets, as shown in Table 5.6. We used different maximum sentence length in each test set, so that we

[4]We use words to denote the number of distinct words in the corpus

[5]The size of the corpus

test_set	words	tokens	max sentence length	min sentence length	sentences
1	4739	41464	12	1	5000
2	5478	50533	15	1	5000
3	5954	64530	20	3	5000
4	6724	86743	30	3	5000

Table 5.6: Test sets used

can see the performance of the systems in different environments. Another factor was that computational time needed for decoding grows very fast with sentence length, so it would not be practical to use very big sentences. Other than that, the nature of the corpus is closer to spoken than written language, so sentences tend to be relatively small. Usually long sentences are mistakenly two consecutive sentences recognized as one by the sentence aligning algorithm described in Section 2.2. Note than in test sets 3 and 4 sentences shorter than three words are discarded. The reason is that sentences longer than twenty words are more rare than smaller ones. In order to ensure that adequate number of longer sentences are present in the test set, we choose to avoid using sentences smaller than three words. If we overlooked this fact, then probably test sets 3 and 4 would not offer more information than test sets 1 and 2.

Both NIST and BLEU need good reference translations to give reliable ratings. Providing more than one reference translations fulfills this need. Since it is quite difficult to hire a professional translator, we decided to use only one reference translation, which is, however, big enough (5000 sentences for each test set) to reconcile. The text used is part of the European Parliament records, not included in the training sets.

For the models that it was possible, we explored the possibility of using different ways to assign weights. In order to do that we trained each variant model using training set 0 and evaluated the resulting translation of test set 1 using both BLEU and NIST, so as to find the best scoring. While it is possible that the differences in the results are not always statistically significant, we believe that they are indicative of the quality of the resulting translation model. Except for that, if we used all the training and test sets available, we would need much more time and computational power to complete the experiments.

Recall from section 4.3 that we used two ways to assign weights in **weight**. In the first case we used equations 4.3, 4.4, 4.5 and 4.6, to find the weight. The results are shown in Table 5.7. The best scoring is the one that uses equation 4.3 to compute the weights, and it was selected for complete testing.

In the following graphs this model is called **weight_d**.

	eq 4.3	eq.4.4	eq.4.5	eq.4.6
BLEU	**0.23**	0.23	0.229	0.228
NIST	**4.66**	4.611	4.612	4.589

Table 5.7: Discrete weight scores

In the second one, different functions of R were used to compute the weight of the alignment and the results are shown in Tables 5.8, 5.9. The best scoring function is R^5, and it was selected for complete testing. In the following graphs this model is called **weight**.

	R	$10R$	$100R^2$	e^R	R^2	$\frac{1}{1.001-R}$	R^3	R^4	R^5
BLEU	0.230	0.231	0.231	0.230	0.231	0.229	0.231	0.232	**0.231**
NIST	4.621	4.626	4.632	4.619	4.632	4.59	4.637	4.641	**4.643**

Table 5.8: R functions

	R^6	e^{R^2}	e^{2R}	e^{5R}	e^{10R}	R^7	R^8	R^9	R^{10}
BLEU	0.223	0.231	0.231	0.231	0.23	0.229	0.23	0.229	0.2315
NIST	4.614	4.625	4.631	4.636	4.616	4.596	4.616	4.614	4.621

Table 5.9: R functions

In the model called **cor_icor**, unlike above, there is no metric like R which can be used to assign a weight to the alignment. Instead, to refine the performance of the model we used another approach. We assign a weight to each alignment depending on whether it is foreign to native (baseline system) or native to foreign (inverse). The results for seven weight schemes are shown in Table 5.10. The best scheme, chosen for complete testing, is 2-1.[6]

weight scheme	1-0.2	1-1	**2-1**	3-1	4-1	5-1	6-1
BLEU	0.236	0.239	**0.239**	0.238	0.237	0.236	0.237
NIST	4.699	4.731	**4.747**	4.723	4.715	4.698	4.7

Table 5.10: **cor_icor** weight schemes evaluation

The baseline system in the graphs is called **base_original** while the baseline using the improved bilingual phrase extraction algorithm **base**. For easy reference we have included the names of all models created along with a short description of each in Table 5.11.

[6]That means weight 2 for baseline alignments and 1 for inverse alignments.

model name	description
base_original	baseline system
base	baseline system using improved bilingual extraction algorithm
diag_and	model created using the diag_and heuristic(Section 4.3.1)
weight	model created using diag_and heuristic and weight assigned according to continuous function of similarity of alignment to the intersection alignment(Section 4.3.2)
weight_d	model created using diag_and heuristic and weight assigned according to quantized function of similarity of alignment to the intersection alignment(Section 4.3.2)
exact	model created using diag_and heuristic and weight assigned according to quantized function of similarity of alignment to the intersection alignment, based only on the bilingual phrase(Section 4.3.3)
lenwe	model created using diag_and heuristic and weight assigned according to bilingual phrase length(Section 4.3.4)
cor_icor	model created using both alignments(Section 4.3.5)
int	model created using the intersection of the two alignments(Section 4.3.6)
uni	model created using the union of the two alignments(Section 4.3.7)
icor	model created using the inverse alignment(Section 4.3.8)

Table 5.11: Short description of models tested

Because **weight**, **weight_d**, **diag_and**, **exact** and **lenwe** all display similar performance, and in order to improve the display of the results we have included for reference the performance of these five models for test set 1 in figures 5.4, 5.5. In the rest figures we have only included **weight**.

The scores achieved for each model and for every corpus are shown in figures 5.6, 5.7, 5.8, 5.9, 5.10, 5.11, 5.12, 5.13.

Looking at the results, one can not fail to notice the remarkable improvement in performance gained using only the improved bilingual phrase extraction algorithm. In all test and training sets the difference between **base** and **base_original** is around 15%, a very important improvement, especially when taking into account that the two models are very similar (they use the same alignment). Other than that, we can divide the models into two sets, which differ around 15% in performance. The first one, low scoring, consists of **int** and **icor**. The second one includes **uni**, **weight**, **base** and **cor_icor**. **icor** performs better than **int** in all test sets, the difference being around 2.5% in all cases. In the the high scoring set the worst performance is achieved by **uni**. At no case does it succeed in providing better performance than any of the other three models. **weight** performs a little better , but it does not succeed in beating **base**, which is always better, except for a few cases. Finally, **cor_icor** displays the best performance, usually a little higher than **base**, with the exception of test set 2 where it performs much better than all competitors, scoring 15% better than **base**. It is interesting to note than in this test set, while the relative order of performance is preserved and that the difference between, for example, **base** and **uni** is around 0.1 NIST like in other test sets, all models achieve higher scores than in other test sets. One should also note that in all test sets, with the exception of test set 2, the relative difference between **cor_icor** and **base** is bigger when using the models created with training set 3, the smaller one, than with other training sets.

All models display reduced scores when translating test sets comprising of longer sentences, but that is normal since longer sentences are more difficult to translate. More detailed examination of the results does not provide any useful information, as deviations from the above remarks are not consistently observed, so they can be attributed to statistical errors.

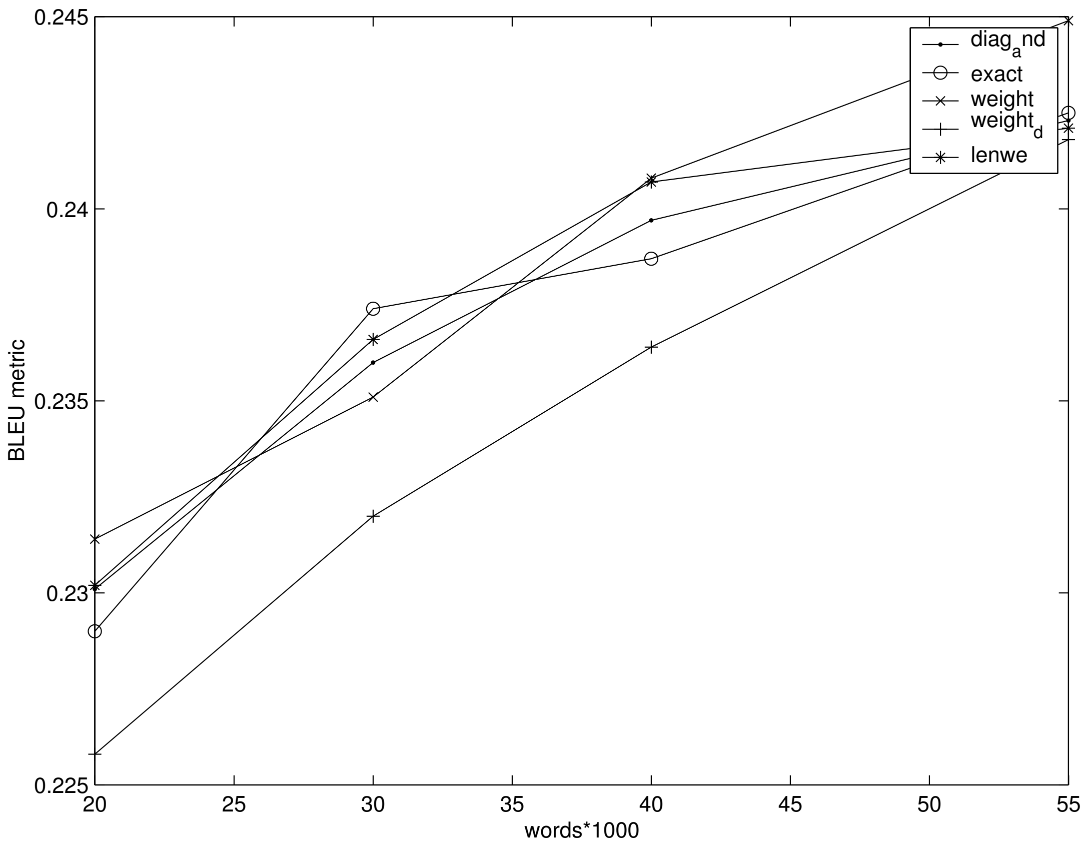


Figure 5.4: test set 1 BLEU for proposed heuristics

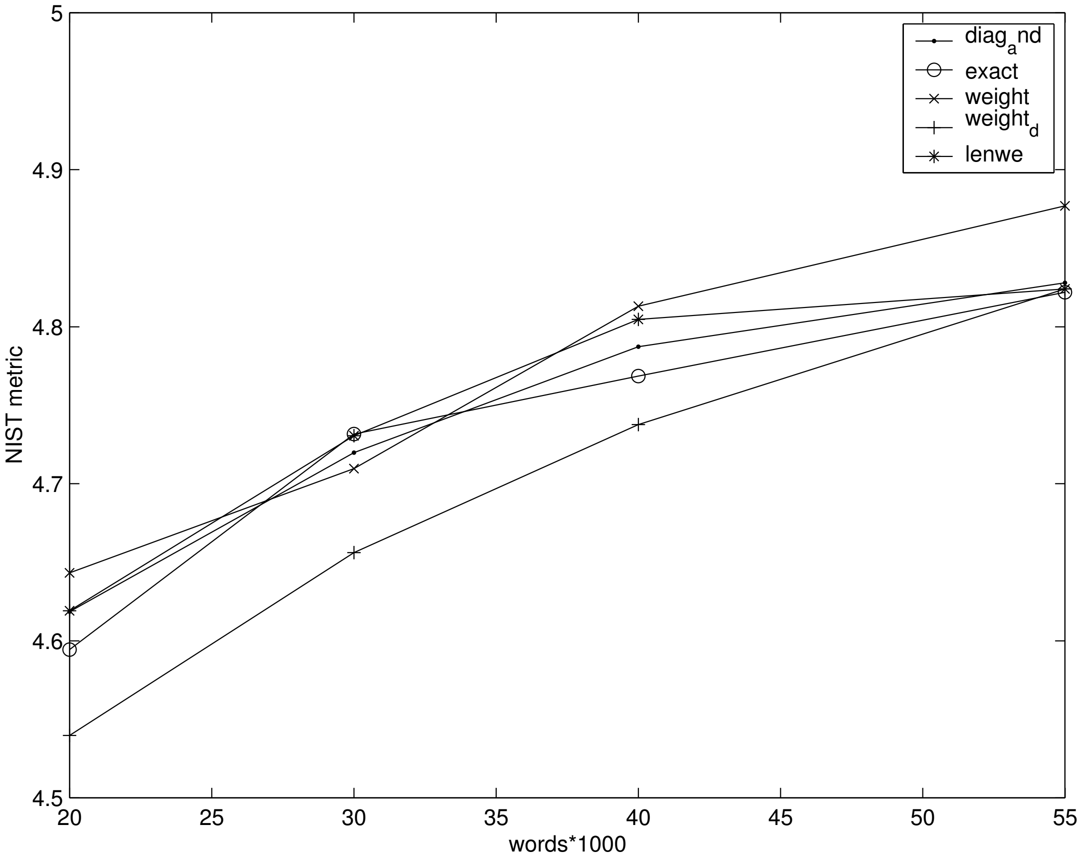


Figure 5.5: test set 1 NIST for proposed heuristics

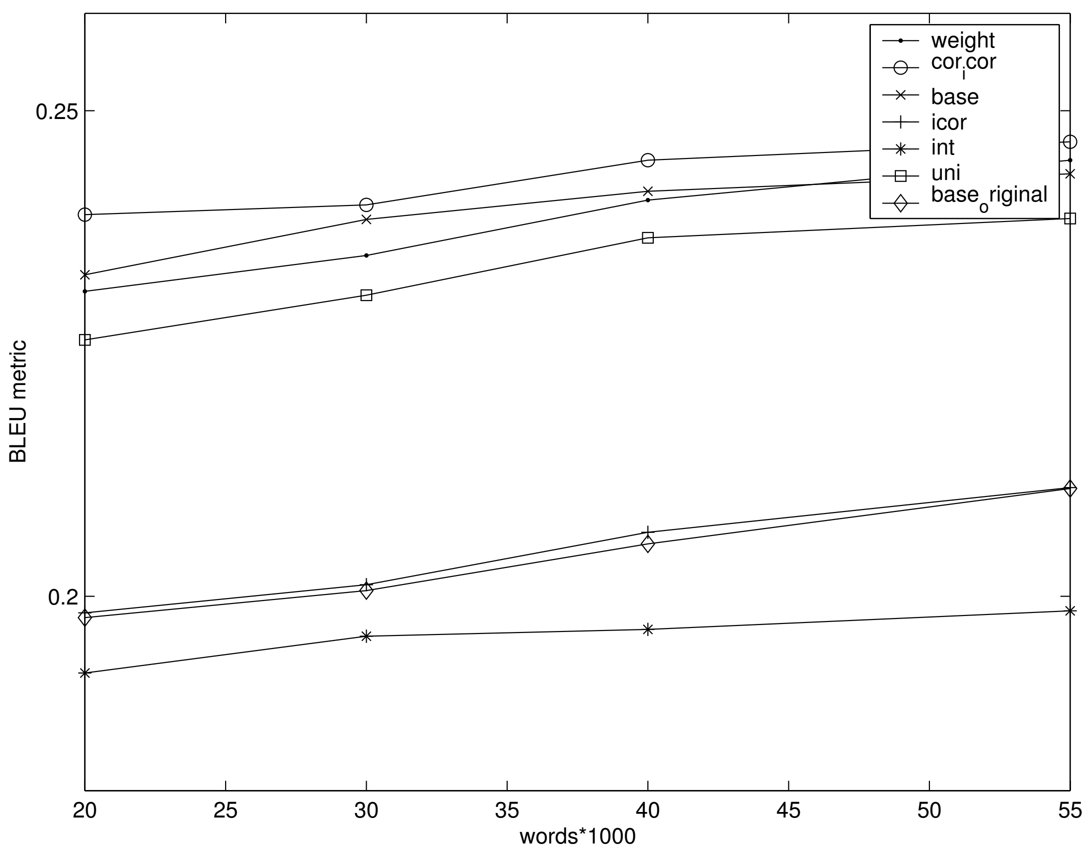


Figure 5.6: test set 1 BLEU

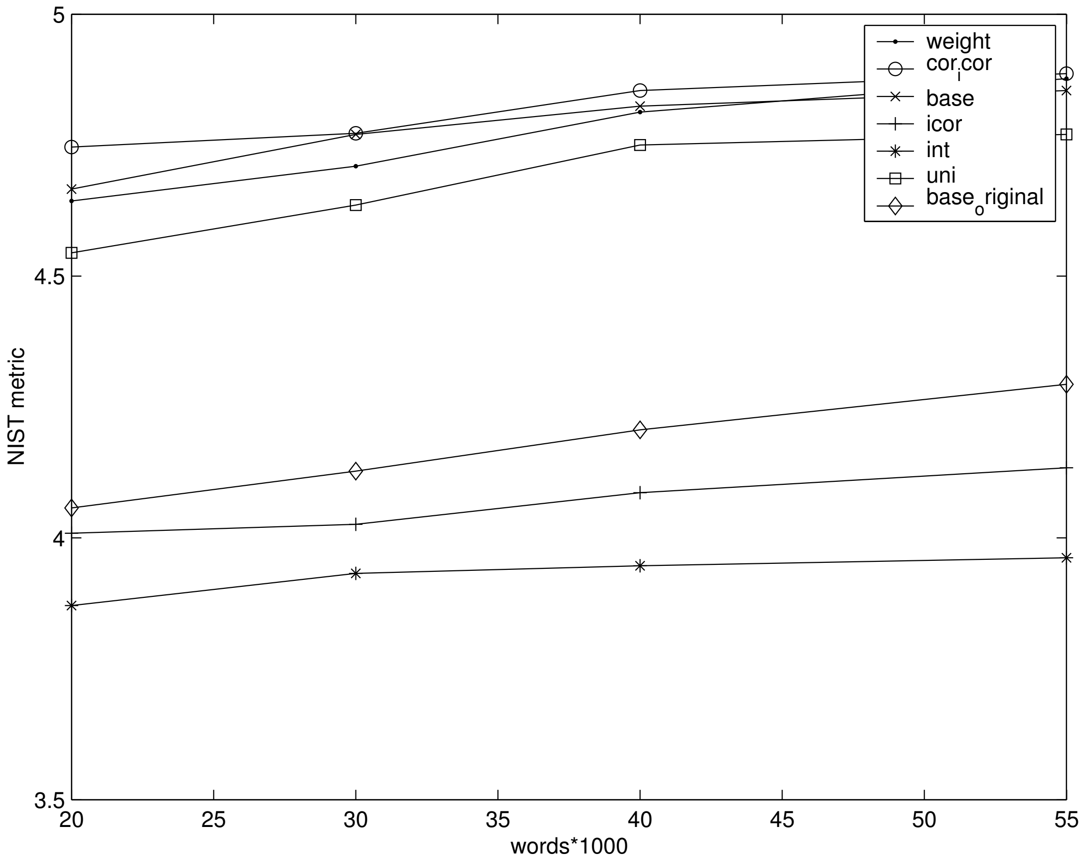


Figure 5.7: test set 1 NIST

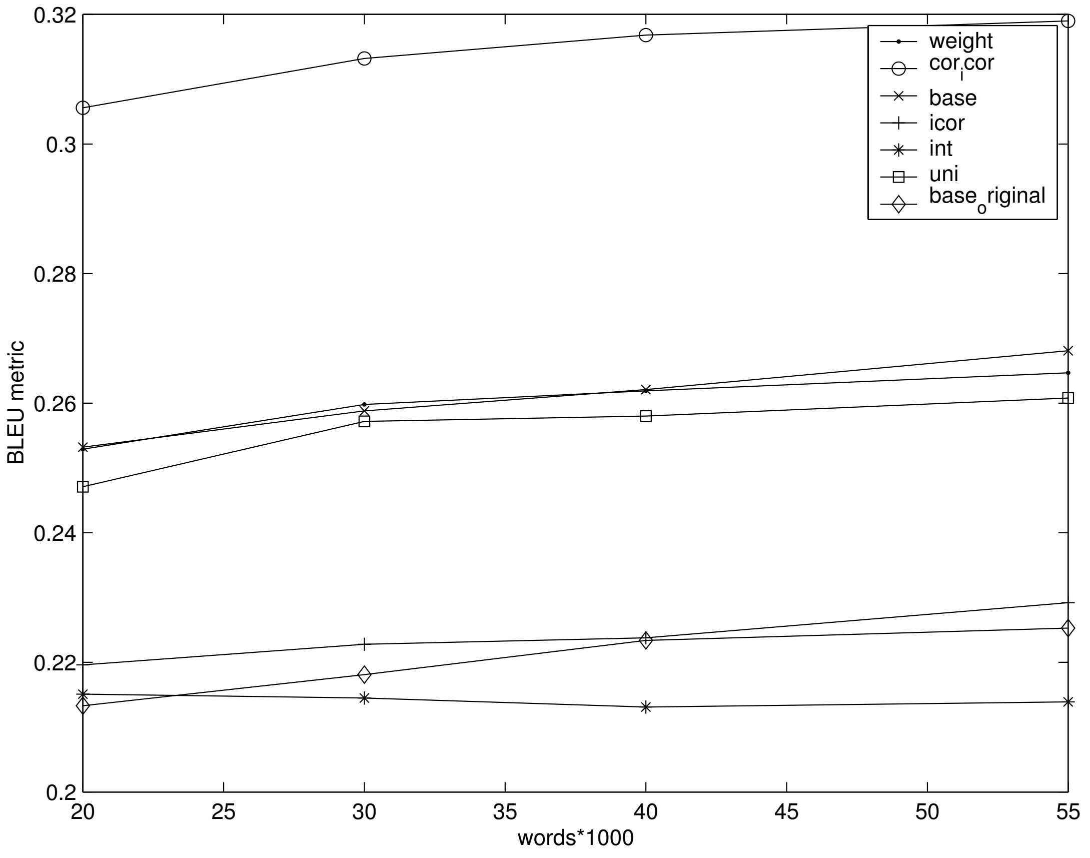


Figure 5.8: test set 2 BLEU

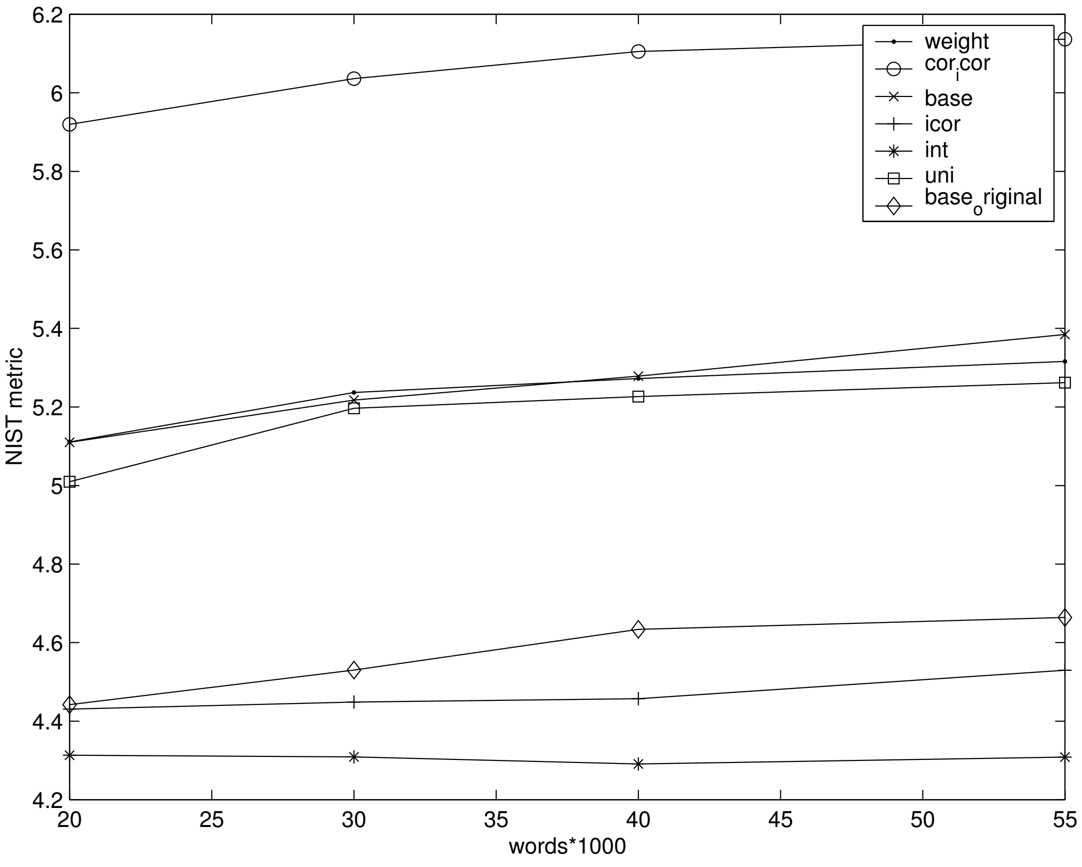


Figure 5.9: test set 2 NIST

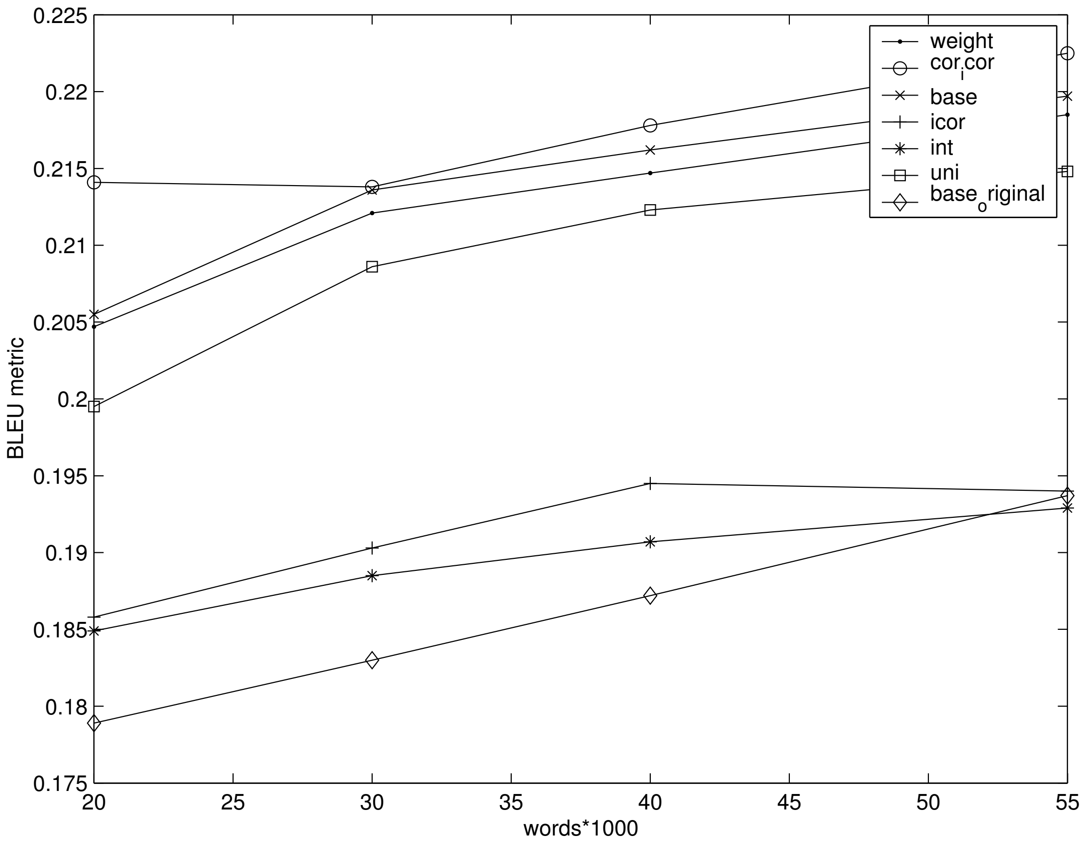


Figure 5.10: test set 3 BLEU

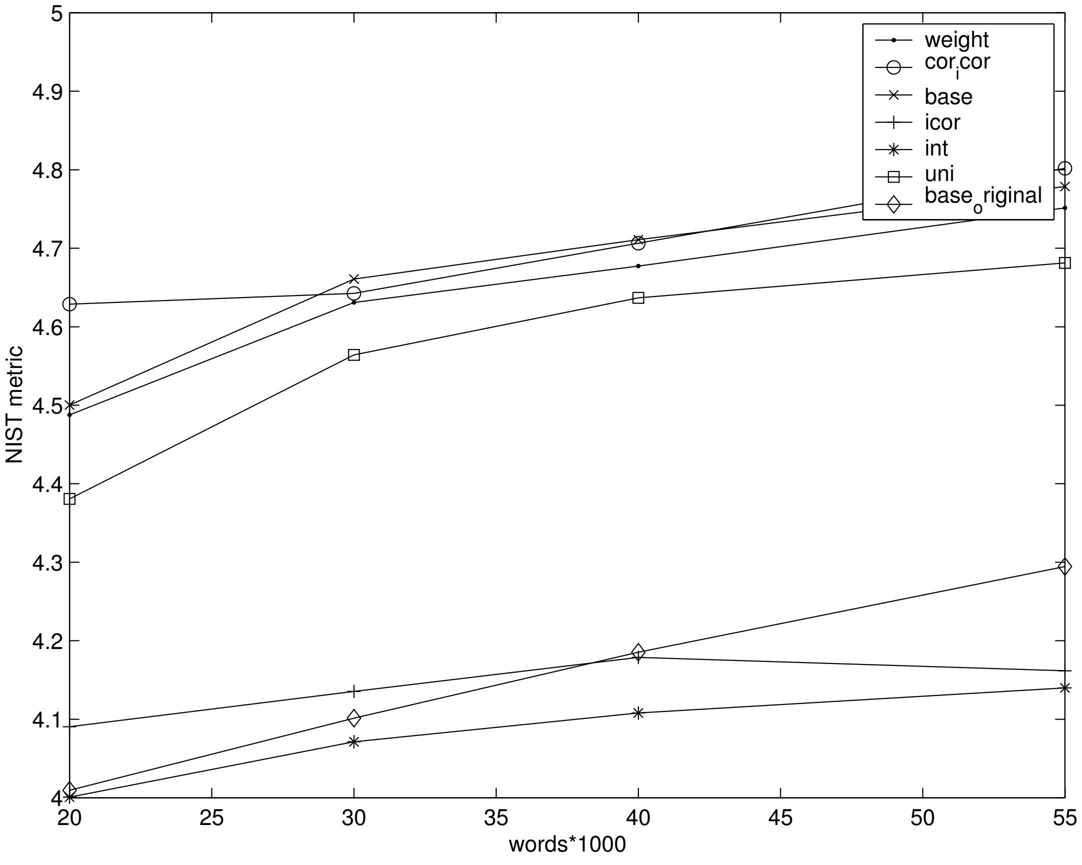


Figure 5.11: test set 3 NIST

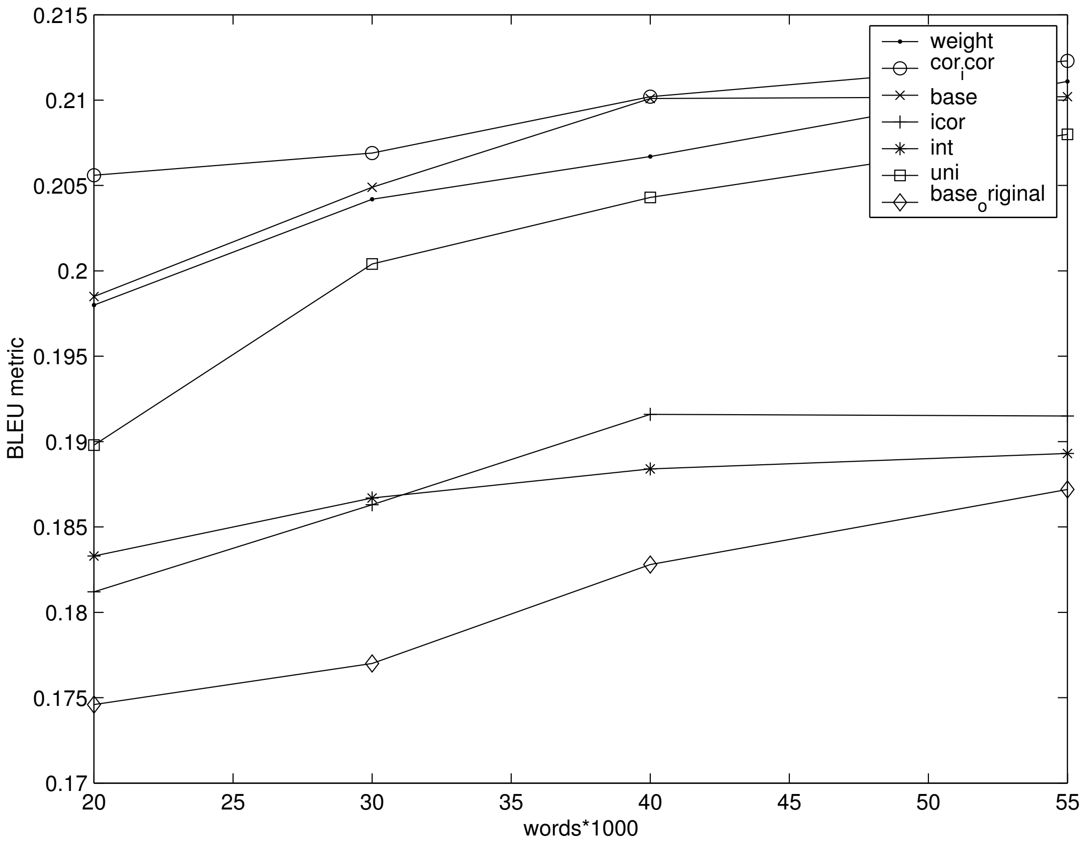


Figure 5.12: test set 4 BLEU

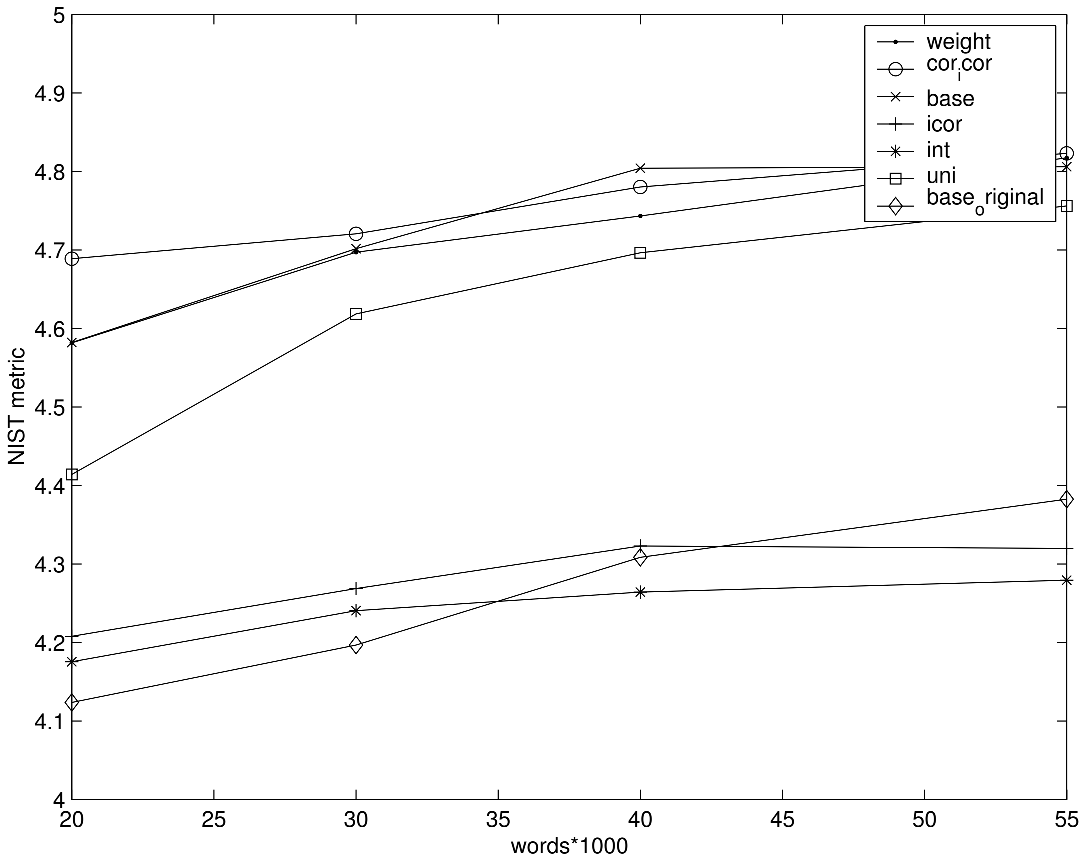


Figure 5.13: test set 4 NIST

Chapter 6

CONCLUSION - FUTURE WORK

The system described in the previous chapters is by no means perfect. There is room for improvements in all levels. However, the experience gained from developing it up to this point can be summarized in the following points.

- The biggest improvement from the baseline system stems from using the improved bilingual extraction algorithm, which gives 15% improvement in performance.
- Since **int** scores 15% worse than models using other alignments, it is better to use rich alignments, that may contain redundant points, than high precision alignments, which obviously leave out important information.
- The alignment created by the **diag_and** heuristic not only does it not offer any improvement over the base alignment, but it actually slightly degrades the system performance. None of the proposed weighting schemes offers significant improvement in performance.
- **cor_icor** offers slightly improved performance over the baseline system, especially when training resources are limited.
- Incorporating morphological knowledge can significantly improve the performance of statistical machine translation systems.

In order to improve the performance of the system, we can follow several ways. The proposed improvements are:

- Improve the preproccesing analysis, especially the sentence aligning process discussed Section 2.2.

- Incorporate knowledge regarding syntax analysis, which has been used in [28, 29].
- Maybe there is little room for improvement using different weighting schemes than those discussed in chapter 4.
- Usage of cognates. By cognates we mean words that are similar in spelling or phonetics and are possible translation of each other. An example is the English word *night* and the German word *nacht*. Various experiments [30] have proved that cognate identification can improve the word alignment, resulting in a more accurate translation system. For example a better sentence.
- Using word classes rather than words when building the billingual phrases.
- Experiments with other language pairs to evaluate the robustness of the system.
- Finally, there is room for improvement in the way morphological knowledge is acquired and used.

aligning algo

6.1 Using Statistical Machine Translation for Speech Translation

During the last years, statistical machine translation is widely used for spoken language translation, were it performs better than other automatic translation approaches.[31] One of the differences between written and spoken language is that in the latter is that the later uses a much more flexible structure, often violating syntactic and grammatical rules. As a result, approaches that depend on this information suffer severely. On the other hand, statistical machine translation does not use stringent rules about syntax or grammar and as a result performs better in such a noisy environment. As a result a statistical machine translation system like the one described here can be used as an integral part of an automatic speech translation system.

Appendix A

Tanslation Examples

Below are listed some translation exapmles from the test sets used. The models used to translate were trained using train set 3. The source sentence is printed in bold; the first translation is the one produced by the **base_original** system and the second one is produced by the **base** system.

parliament adopted the resolution
σώμα εγκρίνει το ψήφισμα
το σώμα εγκρίνει το ψήφισμα

the request is completely unexpected
η πρόταση είναι εντελώς απρόσμενη
η πρόταση είναι εντελώς απρόσμενη

and therefore it rejects amendment no
και γι΄ αυτό απορρίπτει τροπολογία
και γι΄ αυτό απορρίπτει την τροπολογία

this is the reason why i cannot welcome this commission proposal
το λόγο αυτό μπορώ καλωσορίσω την επιτροπή
για το λόγο αυτό μπορώ να χαιρετίσω την πρόταση της επιτροπής

this is not acceptable
αυτό είναι απαράδεκτο
αυτό είναι απαράδεκτο

we must respect their rights
πρέπει σεβόμαστε τα δικαιώματα

οφείλουμε να σεβόμαστε τα δικαιώματά τους

this is not only an economic problem
αυτό δεν είναι μόνο μια οικονομική προβλήματος
δεν πρόκειται μόνο για ένα οικονομικό πρόβλημα

this would mean cutting jobs
αυτό σημαίνει μείωση θέσεων
αυτό σημαίνει μείωση θέσεων εργασίας

let me conclude on one final point
θα ολοκληρώσω με μια τελευταία παρατήρηση
θα ήθελα να τελειώσω με ένα τελευταίο σημείο

but this is not the same thing
αυτό όμως δεν είναι το ίδιο
αλλά αυτό δεν είναι το ίδιο πράγμα

i do not think there are any more problems
δεν νομίζω ότι είναι πια προβλήματα
πιστεύω ότι δεν υπάρχουν πια προβλήματα

the strategy should provide a coherent framework for policy development
η στρατηγική πρέπει παρέχει ένα συνεκτικό πλαίσιο για χάραξη πολιτικής
η στρατηγική πρέπει να παρέχει ένα συνεκτικό πλαίσιο για τη χάραξη πολιτικής

this is a key element of this communication
αυτό είναι ένα βασικό στοιχείο αυτής της ανακοίνωσης
αυτό αποτελεί βασικό στοιχείο αυτής της ανακοίνωσης

mr pronk said previously that we cannot always support social democratic proposals
κύριε pronk ανέφερε προηγουμένως ότι είμαστε πάντα υπέρ σοσιαλδημοκρατικές προτάσεις
ο κ pronk ανέφερε προηγουμένως ότι δεν μπορούμε πάντοτε να υποστηρίξουμε σοσιαλδημοκρατικές προτάσεις

the next point concerns the concept of lifelong learning
το επόμενο σημείο είναι η έννοια της διά βίου
το επόμενο σημείο είναι η έννοια της δια βίου μάθηση

we shall request an inquiry to find the causes of this accident
θα ζητήσω μια έρευνα για τα αίτια του ατυχήματος
θα ζητήσω μια έρευνα για να βρούμε τις αιτίες αυτής της δυστυχήματος

it is just a question of attitudes
δεν είναι μόνο ζήτημα νοοτροπίες
είναι απλά θέμα νοοτροπίες

are we to vote for your amendments tomorrow or the day after
πρόκειται να ψηφίσουμε για τις τροπολογίες αύριο ή την επομένη
πρέπει άραγε να υπερψηφίσουν τις τροπολογίες σας αύριο ή μεθαύριο

i see now the sense in the strategy that she is adopting
βλέπω όμως η έννοια της στρατηγικής που έχει υιοθετήσει
αντιλαμβάνομαι τώρα το νόημα της στρατηγικής που έχει υιοθετήσει

that is of course not necessarily true
αυτό φυσικά δεν ισχύει
αυτό φυσικά δεν είναι αναγκαστικά σωστό

the commission cannot go as fast as it would like in some issues as we are rapidly approaching some of the proposed phaseout dates
η επιτροπή να προχωρήσει όσο γρήγορα θα ήθελα σε ορισμένα θέματα όπως η γοργούς πλησιάζει ορισμένες από τις προτεινόμενες ημερομηνίες απόσυρσης
η επιτροπή δεν μπορεί να προχωρήσει όσο γρήγορα θα ήθελα σε ορισμένα θέματα όπως η ταχεία πλησιάζει ορισμένες από τις προτεινόμενες ημερομηνίες απόσυρσης

since the author is not present question no lapses
εφόσον η συντάκτης απουσιάζει ερώτηση αριθ καταπίπτει
δεδομένου ότι ο συντάκτης της απουσιάζει η ερώτηση αριθ καθίσταται άκυρη

we ask this house under rule to accept this question as an urgent matter
καλούμε το κοινοβούλιο κατά κανόνα να αποδεχθούμε το θέμα ως επείγον ζήτημα
ζητάμε από το σώμα βάσει του άρθρου να αποδεχθεί αυτό το θέμα ως επείγον

can i also ask the commission what action they intend to take against germany

επιτρέψτε μου επίσης την επιτροπή τι μέτρα σκοπεύουν να ληφθούν κατά γερμανίας
θα ήθελα επίσης να ρωτήσω την επιτροπή τι μέτρα σκοπεύουν να ληφθούν κατά της γερμανίας

if national systems are not effective enough the international community ultimately has the responsibility to see that the law is enforced
εάν εθνικά συστήματα δεν αρκεί η διεθνής κοινότητα τελικά έχει την ευθύνη να διαπιστώσουμε ότι η δίκαιο» γίνεται εφαρμοστεί αποτελεσματικά
αν τα εθνικά συστήματα δεν είναι αρκετά αποτελεσματική η διεθνής κοινότητα έχει την ευθύνη για το γεγονός ότι ο νόμος εφαρμοσθεί τελικά

sometimes they are dependent on one large customer
συχνά πρόκειται για ένα μεγάλο customer
μερικές φορές είναι εξαρτώμενοι από ένα μεγάλο πελάτη

even if the judgement had been made it would still have been extremely difficult to complete a codecision procedure on this matter
μολονότι η κρίση είχε σημειωθεί ότι θα ήταν εξαιρετικά δύσκολο να ολοκληρώσει μια συναπόφασης για το θέμα ακόμη
παρόλο που η απόφαση έχει ληφθεί ακόμη θα ήταν εξαιρετικά δύσκολο να ολοκληρώσει μια διαδικασία της συναπόφασης για το θέμα

it would of course have been more reasonable to have financed the reconstruction in kosovo by means of reductions in categories of expenditure other than that of external measures
καλό θα ήταν πιο λογικό να χρηματοδοτήσουμε την ανοικοδόμηση κοσσυφοπεδίου μέσω της μείωσης κατηγοριών δαπανών διάφορο εκείνου των εξωτερικών δράσεων αυτονόητο
θα ήταν πιο λογικό να χρηματοδοτήσουμε την ανοικοδόμηση του κοσσυφοπεδίου μέσω της μείωσης των κατηγοριών δαπανών διάφορο εκείνου των εξωτερικών δράσεων της χρόνω

this is a proposal without debate and we will not have any opportunity therefore to debate the amendments being put forward
η πρόταση χωρίς συζήτηση και δεν θα έχουμε καμία ευκαιρία να συζητήσουμε τις τροπολογίες που προτάθηκαν
πρόκειται για μια πρόταση χωρίς συζήτηση και δεν θα έχουμε καμία δυνατότητα να συζητήσουμε τις τροπολογίες που παρουσιάστηκαν κατά συνέπεια

as such this will not solve any problems within tajikistan
κατά συνέπεια δεν θα επιλύσει κανένα πρόβλημα σε τατζικιστάν
επομένως αυτό δεν θα λύσει τα προβλήματα στο τατζικιστάν

Βιβλιογραφία

[1] Φανούρης Μωραΐτης *Συστήματα Αυτόματης Μετάφρασης Χρησιμοποιώντας Στατιστικά Μοντέλα*, Technical University of Crete, Electronic & Computer Engineering, Chania 2004.

[2] William A. Gale & Kenneth W. Church. *A Program for Aligning Bilingual Corpora*, AT&T Bell Laboratories 600 Mountain Avenue Murray Hill, 07974, (1993).

[3] Peter F. Brown & Stephen A. Della Pietra. *The Mathematics of Machine Translation: Parameter Estimation*, Computational Linguistics, vol. 19, no. 2, pp.263-311, (1993)

[4] Kevin Knight. *A Statistical MT Tutorial Workbook*

[5] Richard Zens, Franz Josef Och, Hermann Ney. *Phrase-Based Statistical Machine Translation.* (2002)

[6] Philipp Koehn, Franz Josef Och, Daniel Marcu. *Statistical Phrase-Based Translation*, In Proceedings of 2003 HLT/NAACL, (2003)

[7] Philip Clarkson, Ronald Rosenfeld. *Statistical Language Modeling using The CMU-Cambridge Toolkit*

[8] J.J. Odell. *Lattice and Language Model Toolkit Reference Manual* Entropic Cambridge Research Laboratories

[9] Andreas Stolcke. *SRILM - An extensible Language Modeling Toolkit*

[10] http://www.speech.sri.com/projects/srilm

[11] http://www.isi.edu/˜koehn/europarl/

[12] http://www.isi.edu/licensed-sw/pharaoh/

[13] Philipp Koehn. *A Beam Search Decoder for Phrase-Based Statistical Machine translation Models. User Manual and Description* (2003)

[14] Philipp Koehn. *Noun Phrase Translation* PHD thesis (2003) University of Southern California, Los Angeles

[15] Jinxi Xu and W. Bruce Croft. *Corpus-based stemming using co-occurrence of word variants.* ACM Transactions on Information Systems, 16(1):61●81, (1998).

[16] H. Dejean. *Morphemes as necessary concepts for structures: Discovery from untagged corpora.* University of Caen-Basse Normandie. http://www.info.unicaen.fr/ DeJean/travail/articles/pg11.htm (1999).

[17] Zellig Harris. *Structural Linguistics.* The University of Chicago Press, (1951).

[18] John Goldsmith. *Unsupervised learning of the morphology of a natural language*, (2000)

[19] Rissanen, Jorma. *Stochastic complexity in Statistical Inquiry.* World Scientific Publishing Co,Singapore.

[20] Panagiotis Karageorgakis, Alexandros Potamianos, Ioannis Klasinas. *Towards corporating Language Morphology into Statistical Machine Translation Systems*, ASRU (2005).

[21] Florence Reeder. Additional mt-eval references. Technical report, International Standards for Language Engineering, Evaluation Working Group. http://isscowww.unige.ch/projects/isle/taxonomy2/ , (2001).

[22] Kishore Papineni, Salim Roukos, Todd Ward, Wei-Jihg Zhu. *BLEU: a Method for Automatic Evaluation of Machine Translation*, In proceedings of ACL, (2002)

[23] National Institute of Standards and Technology. *Automatic Evaluation of Machine Translation Quality Using N-gram Co-occurence Statistics*, http://www.nist.gov/speech/tests/mt/doc/ngram-study.pdf

[24] B. Efron, and R.J. Tibshirani. *An Introduction to the Bootstrap*, CRC Press, (1994).

[25] M. Bisani and H. Ney. *Bootstrap Estimates for Confidence Intervals in ASR Performance Evaluation*, International Conference on Acoustics, Speech, and Signal Processing (ICASSP), (2004).

[26] Y. Zhang, S. Vogel. *Measuring Confidence Intervals for the Machine Translation Evaluation Metrics*, In: Proceedings of The 10th International Conference on Theoretical and Methodological Issues in Machine Translation, Baltimore, MD USA, (TMI 2004).

[27] Y. Zhang, S. Vogel, A. Waibel. *Interpreting BLEU/NIST scores: How much improvement do we need to have a better system?*, Proceedings of the 4th International Conference on Language Resources and Evaluation, Lisbon, Portugal, (2004).

[28] Sonja Niessen and Hermann Ney. *Morpho-Syntactic Analysis for Reordering in Statstical Machine Translation*

[29] Kevin Knight and Philipp Koehn. *ChunkMT: Statistical Machine Translation with Richer Linguistic Knowledge*

[30] Grzegorz Kondrak, Daniel Marcu, Kevin Knight. *Cognates Can Improve Statistical Translation Models*

[31] Hermann Ney. *The Statistical Approach to Machine Translation and a Roadmap for Speech Translation*