Universität Karlsruhe

Fakultät für Informatik
Institut für Theoretische Informatik
Prof. A. Waibel

Diplomarbeit

Translation Model Adaptation for Statistical Machine Translation using Information Retrieval

Almut Silja Hildebrand

1. April - 1. October 2005

Betreuer: Prof. Dr. Alex Waibel
Dipl.-Inform. Matthias Eck

Hiermit erkläre ich, die vorliegende Arbeit selbstständig erstellt und keine anderen als die angegebenen Quellen verwendet zu haben.

Karlsruhe, den 1. Oktober 2005 [illegible]

Contents

1 Introduction

1.1 Motivation

The goal of this research is to improve the translation performance of a Statistical Machine Translation system.

Translation tasks can be very different in nature, because language varies greatly depending on who uses it when and for what purpose. For example the sentence structure and vocabulary in a children's book, a scientific article, a dialog between two persons, a newspaper article and an article on medicine or sports are different. Often the same term has a different meaning in different topics. This project tries to improve translations by adapting the translation system to the translation task at hand.

One challenge in machine translation connected to the domain and topic of a text is word-sense disambiguation. Every language has words that have different meanings when used in varying contexts. In most cases the word has to be translated into different words for each meaning.

For example the word 'operating' has other translations to German in different domains. medicine: 'operating table' or computer related: 'operating system'. 'Operating table' translates to 'Operationstisch', while 'operating system' to 'Betriebssystem'. So, 'operating' translates to 'Operation' or 'Betrieb' depending on whether its used in a medical or a technological domain.

A statistically built translation lexicon usually provides several translation alternatives. The challenge is to select the right translation alternative for the right domain.

Typically, the more data is used in a statistical machine translation system to estimate the parameters of the model, the better it approximates the actual distributions in the language. More training data will usually lead to a higher translation performance.

However if a significant amount of out-of-domain data is added to the training data, translation quality may drop. One reason is that a general translation model $P(s|t)$ trained on in-domain and out-of-domain data does not fit domain, topic or style of individual texts.

The use of a language model and phrase-based transducers enables the translation system to utilize context information to select the right translation alternative. In some cases the immediate context can be enough to determine the domain it is used in and with it the correct translation. In the example 'operating' the next word is enough to do so.

But in other cases the immediate context is not enough. For example the word 'shot' has several connotations depending on whether a policeman fires a shot, a doctor gives a shot, a barkeeper serves a shot, a tourist takes a shot or someone adventurous gives it a shot.

```
The doctor gave me a shot of vaccine.
The waiter gave me a shot of whiskey.
My  friend gave me a shot of the Grand Canyon.
```

In this example the five word phrase 'gave me a shot of' translates into different expressions depending on words, that can not be found in the immediate context of the word 'shot'. German translations of these examples are all different expressions, see table 1.1. The direct translation of 'shot' is 'Schuss' which only occurs in one of the translations.

English	**German**
to fire a shot	einen Schuss abgeben
to give a shot of vaccine	impfen
a shot of alcohol	ein Kurzer
a shot of the Grand Canyon	ein Foto vom Grand Canyon
to give it a shot	es mal probieren

Table 1.1: German Example Translations: 'shot'

Most commonly used language models do not exceed trigrams and it is hard to find long matching phrases for phrase transducers. So long term context information is lost in these models.

1.2 Approach

This project aims to circumvent the limitations of commonly used models by making use of long range context information. It tries to exclude alternative translations from other domains, by training the models only on data that fits the domain and more specific topic and style of the translation task at hand.

To find this in-domain data in all the available training data, Information Retrieval (IR) is a suitable tool because it finds and ranks documents of a document collection according to their relevance to a given topic, and this should limit them to the same domain as well.

Recent approaches in language model adaptation (Eck et al. [2004], Zhao et al. [2004]) successfully used information retrieval techniques to find matching data in a training corpus.

The ideas and results of this project have already been published in the proceedings of EAMT 2005 Hildebrand et al. [2005].

1.3 Outline

Chapter 2, 'Background Information on Statistical Machine Translation and Information Retrieval' briefly explains about systems, models and algorithms used in this project. It includes some general information on statistical machine translation, descriptions of word- and phrase-alignment techniques for building transducers and the two similarity measures used in this project.

Chapter 3 'State of the Art' refers to research work in the fields of language model adaptation and translation model adaptation, on which this project is partially based.

Chapter 4 'Translation Model Adaptation' details the approach to translation model adaptation. It focuses on selection of similar data and automatic determination of selection size.

Chapter 5 is a collection of all experimental results for two experiment settings. The main goal for the first, translating from Spanish to English, is word sense disambiguation by finding in-domain data in a small mixed-domain corpus. In the second, translating from Chinese to English, I mainly try to make use of a very large out-of-domain bilingual corpus without introducing too many undesired translation alternatives.

2 Background Information on Statistical Machine Translation and Information Retrieval

This chapter briefly explains the models and algorithms used in this project. It is not a complete overview of all techniques for statistical machine translation or information retrieval.

2.1 Statistical Machine Translation

Machine translation systems try to translate a text in one language, called the *source language*, into another language, the *target language*. Most Statistical Machine Translation (SMT) systems do the translation sentence-by-sentence. They estimate the probability $P(T|S)$ that a sentence T in the target language is a valid translation of a sentence S in the source language.

According to Bayes' theorem $P(T|S)$ is:

$$P(T|S) = \frac{P(T)P(S|T)}{P(S)} \tag{2.1}$$

The *decoder* is the translation system component that performs the actual translation by searching for the target language sentence $\widehat{T}$, whose probability $P(T|S)$ is maximum. As that maximum is independent of the a priori probability of the source sentence $P(S)$, we search for:

$$\hat{T} = argmax_T(P(T) \cdot P(S|T)) \tag{2.2}$$

$P(T)$ is the a priori probability of the target sentence, which is in SMT typically provided by a language model. $P(S|T)$ is the translation probability of S given T, provided by the translation model.

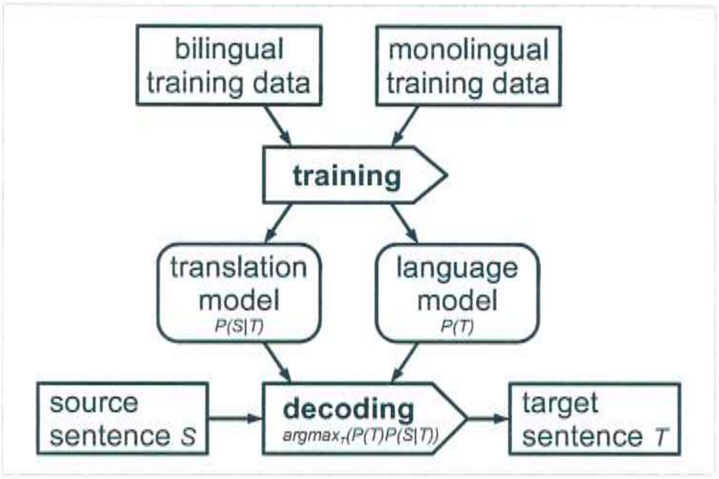


Figure 2.1: SMT System

Statistical approaches to machine translation, pioneered by Brown et al. [1990], estimate parameters for a probabilistic model of word to word correspondences (the translation model) and word order (the language model) directly from large corpora of parallel bilingual text.

2.1.1 Translation Model

Data for training a translation model usually consists of a collection of texts in two languages, where each sentence in the source language is aligned with one sentence that is its translation in the target language. This kind of data is called a *parallel bilingual corpus.*

Given these pairs of sentences, the difficult task is to find which words in the source language translate into which words in the target language.

This mapping is called a *word alignment*. Figure 2.2 shows an example for a manual word alignment between a Spanish and an English sentence.

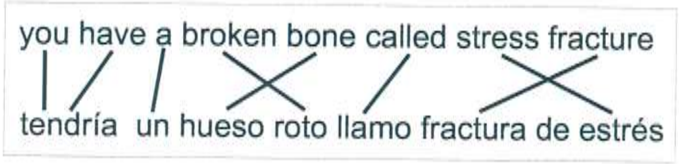

you have a broken bone called stress fracture
tendría un hueso roto llamo fractura de estrés

Figure 2.2: Word Alignment Example for a Spanish – English sentence pair

Next, a translation lexicon is extracted from these word alignments. Because languages have different structure it is beneficial to not only use word to word translations in that lexicon, but to find bigger parts of sentences, consisting of two, three or more words, which are translations of each other. Such parts of sentences are called *phrases*. A phrase can coincide with but is not limited to a grammatical clause of a sentence. It can be any part of a sentence. Phrase-to-phrase translations incorporate context information and local word reordering. From the example sentences in figure 2.2, a phrase could be the main clause: 'you have a broken bone'. Examples for phrase pairs with local word reordering would be: 'broken bone' – 'hueso roto' and 'stress fracture' – 'fractura de estrés'.

Extracted phrase translations from many sentence pairs make up the translation lexicon, called the *transducer*. The transducer aligns each source phrase with several candidate target phrases (translations) and contains an estimated translation probability for each phrase pair and possibly other model scores like phrase fertility.

Table 2.1 shows several transducer entries relevant to the example sentence: 'me gustaría recibir información con respecto a los músculos' (reference translation: 'i would like an information regarding muscles'.

There are several methods to align words and phrases to build transducers. Some of them are explained in the following sections 2.1.3 to 2.1.6. The transducer in table 2.1 was built using the Integrated Phrase Segmentation and Alignment Algorithm (ISA) (section 2.1.5).

source phrase	target phrase	probability
me	i	0.005307
me	me	0.004916
me gustaría	i would like	0.555504
gustaría recibir	like to receive	1.000000
recibir	receive	0.979510
información	information	0.993545
información	information desk	0.079469
con	with	0.973766
con respecto	regarding	0.567744
con respecto a	regarding	0.204682
respecto	own	0.034200
a	to	0.742839
a los	to the	0.039452
los	the	0.002374
los	them	0.001832
músculos	muscles	0.964412
los músculos	muscles	0.930327
los músculos	the muscles	0.230246
los músculos	your muscles	0.764182

Table 2.1: Example Transducer Entries

Hierarchical Transducers

Hierarchical transducers are a set of transducers varying from specific to general. These transducers are applied to the test sentences in order from specific to general. A specific transducer's name can be used in a more general transducer as a label to replace a class of terms. Thus, a phrase is generalized and can be used to translate more sentences.

An example for a specific transducer is a lexicon translating body parts, identified by its label @BP:

```
@BP # head # Kopf
@BP # leg # Bein
@BP # arm # Arm
@BP # hand # Hand
@BP # finger # Finger
```

A general transducer @GTD can use the label to substitute all body parts in its entries:

```
@GTD # my @BP itches # mein @BP juckt
@GTD # does your @BP hurt ? # tut dein @BP weh ?
@GTD # can you move your @BP ? # kannst du deinen @BP bewegen ?
```

These transducers are applied to the source sentence in hierarchical order, the most specific first:

```
=> does your leg hurt ?
=> does your @BP{leg -> Bein} hurt?
=> tut dein @BP{leg -> Bein} weh ?
=> tut dein Bein weh ?
```

In this example the transducer pair can translate: 'does your leg hurt?', 'does your head hurt?' and 'does your arm hurt?' even though probably only one of them was seen in the training data.

2.1.2 Decoding

Translating sentences using previously trained models is called *decoding*. The decoding is divided into two steps: building the translation lattice and best path search.

The translation lattice is built from word to word as well as phrase-to-phrase translations. The source sentence forms the basis of the lattice, each word one edge.

Then all transducers are applied: the decoder searches for matching phrases between the source sentence and the transducer. The target words and phrases are inserted into the translation lattice as new edges over the source sentence. The phrases can overlap. The graph contains words and translation probabilities from several different transducers, as well as other possible scores associated with phrases and words (e.g. co-occurrence statistics). Figure 2.3 shows a translation lattice for the example sentence in Table 2.2. The lattice stores information about how many and which words are already translated - the *coverage information* - as well as backtracking information and cumulative scores.

source	me gustaría recibir información con respecto a los músculos
reference	i would like an information regarding muscles
SMT	i like to receive information regarding your muscles

Table 2.2: Decoder Lattice Example Sentence

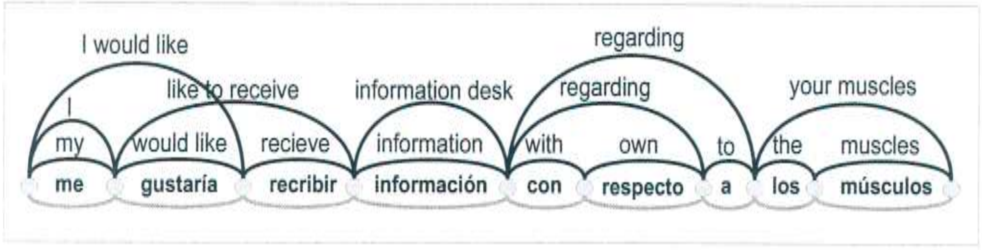


Figure 2.3: Decoder Lattice Example

In the second step, the decoder searching for the best path, it reads the language model scores for the target language from the language model.

A sentence length model consists of fertility scores for words and phrases and a sentence stretch factor for a specific language pair. It contains information abut preferred length of translation or penalizes translations that are too long or too short.

The search through the lattice is first-best search. While decoding, several pruning techniques are applied to reduce decoding time.

Word Reordering

Decoding without word reordering is called *monotone decoding*. Here local word reordering is taken care of by phrase translations only.

Decoding that reorders words is called *non-monotone decoding*. The target language model is the main source of information for word reordering. There are several methods to reorder words while decoding.

The simplest but most computationally expensive is to try all permutations of words in the target sentence and then select the one with the best language model score. To reduce decoding time, it is possible to only try all permutations within a window of e.g. four words, instead of the whole sentence. This reordering window slides over the sentence during the decoding process.

Another word reordering strategy is to jump ahead in the sentence and leave some words untranslated, then come back later to cover those gaps, when their translations fit into the target sentence. Like all word reordering strategies, this one is also computationally expensive and there is need to restrict, for example, the maximum number of jumps per sentence. It is possible to train a jump model with probabilities depending on position or source word to only jump in certain cases.

A third word reordering strategy is to build a tree structure over the target sentence using, for example, a grammar tagger and then swap subtrees. This way whole subclauses can trade places. To speed up this strategy it is possible to only allow certain swaps or only a maximum number of swaps. Also a swap model can be trained depending on part of speech tags or the grammar structure.

2.1.3 Translation Models: IBM Model 1

The IBM1 lexicon, which is the Model 1 introduced by Brown et al. [1993], is a word to word translation lexicon. It estimates word to word translation probabilities using the EM algorithm from a bilingual training corpus. This simple lexicon is basis for many word alignment algorithms. It was used in this project as an additional transducer in the decoder.

For each word in the source sentence we have to decide how to align it to a word in the target sentence. In Model 1 all alignments for each source word are initially assumed to be equally likely. Figure 2.4 shows an example for an initial word alignment between a Spanish and an English sentence.

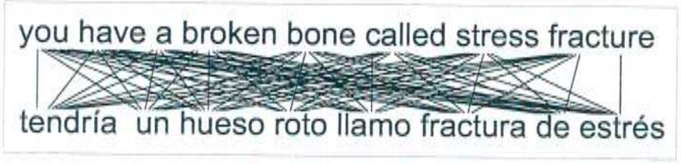

you have a broken bone called stress fracture
tendría un hueso roto llamo fractura de estrés

Figure 2.4: Initial Word Alignment Example: Spanish – English sentence pair

Then the EM algorithm is used to estimate the alignment probabilities.

```
EM algorithm for training an IBM1 Lexicon:

# Accumulation (over corpus)
For each sentence pair
   For each source word s
      Sum = 0.0
      For each target word t
         Sum += p(s|t)
      For each target word t
         Count(s,t) += p(s|t)/Sum
# Re-estimate probabilities (over count table)
For each target word t
   Sum = 0.0
   For each source word s
      Sum += Count(s,t)
   For each source word s
      p(s|t) = Count(s,t)/Sum
```

In IBM Model 1 the alignment probability does not depend on the order of the words in the source sentence *s* and the target sentence *t* but only on the number of times the words appear in their respective sentences.

Figure 2.5 shows how the alignment probabilities could look like after the EM training. The connecting lines between words represent the alignment between the words; the strength of a line represents the alignment probability estimated over all the sentence pairs in the training. Alignments with an alignment probability under a certain level are dropped.

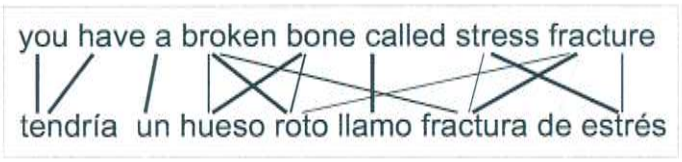

you have a broken bone called stress fracture
tendría un hueso roto llamo fractura de estrés

Figure 2.5: Word Alignment Example for an Spanish – English sentence pair

The alignment probability is given in equation 2.3.

$$P(s_1^J|t_1^I) = p(J|I)\prod_{j=1}^{J}\sum_{i=1}^{I} p(i|j,J,I)p(s_j|t_i) = p(J|I)1/I^J \prod_{j=1}^{J}\sum_{i=1}^{I} p(s_j|t_i) \tag{2.3}$$

with:
t: target word
I: length of target sentence
i: position in target sentence

s: source word
J: length of source sentence
j: position in source sentence

$a_1^J = a_1 \ldots a_j \ldots a_J$: whole alignment

IBM Model 1 supports only alignments, where each source word has exactly one connection, multiple source words may connect to one target word, some words in the target sentence might have no connections. One possibility to cover the case where a source word should have no connection or multiple connections, is to train an additional lexicon with reversed language directions and then invert the resulting lexicon.

In the paper by Brown et al. [1993] they present four more models, that use alignment probabilities from Model 1 as training basis. (Knight [1999] is an easy to understand tutorial workbook on the topic.)

- Model 2: adds absolute position to the alignments
- Model 3: adds fertilities
- Model 4: adds inverted relative position alignment
- Model 5: is an non-deficient version of model 4

2.1.4 Translation Models: HMM alignment model

The HMM alignment model (Vogel et al. [1996]) is a word alignment model based on a first order Hidden Markov Model (HMM) (Jelinek [1976]).

Usually word alignments are not distributed arbitrarily over all word positions in the sentence, but are generally close to the diagonal and form local clusters. Figure 2.6 shows this in a word alignment for a sentence pair from the the Spanish – English experiment setting (see section 5.3).

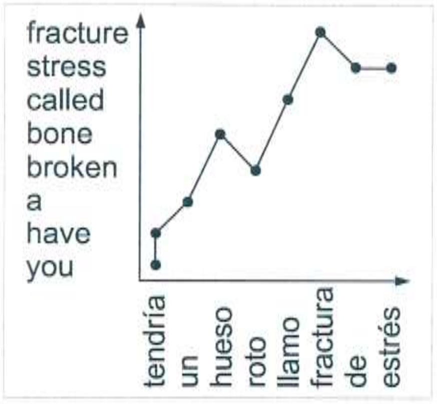


Figure 2.6: HMM Word Alignment for an Spanish – English sentence pair

The HMM alignment model takes this into account by modeling an alignment depending on the previous alignment.

The HMM alignment model is similar to the IBM Model 2. But while the alignments of IBM Model 2 are dependent on the absolute position of the word in the sentence, the alignments of the HMM model are dependent on the relative position to the previous alignment.

2.1.5 Translation Models: ISA

The Integrated Phrase Segmentation and Alignment Algorithm (ISA) (Zhang et al. [2003]) provides phrase-to-phrase translations. It segments the training sentences into phrases and aligns phrase translation pairs in the same step, without using a pretrained word-to-word alignment lexicon.

For each sentence pair from the training corpus, it builds a two-dimensional matrix representing the correlation among all word pairs from the source and the target sentence. Correlation measures such as the Point-wise Mutual Information (MI) or the dependence information from a chi-square test can be used.

1. Point-wise Mutual Information (MI) between word t_i and word s_j:

   $$I(t_i, s_j) = log_2 \frac{P(t_i, s_j)}{P(t_i)P(s_j)} \tag{2.4}$$

   The higher $I(t_i, s_j)$, the more likely t_i is a translation of s_j.

2. The chi-square test (Manning and Schütze [2001]) is a test for statistical dependence of two variables. It compares the observed co-occurrence frequencies with the frequencies expected for independence.

   Chi-square is:

   $$X^2 = \sum_{i,j} \frac{(O_{ij} - E_{ij})^2}{E_{ij}} \tag{2.5}$$

   where O_{ij} are the cells of the table with the observed frequencies (see table 2.3) and E_{ij} are the cells of the table with the expected frequencies for independence (see table 2.4). In this example X^2 is very high: $X^2 = 456400$, which shows that the occurrence of the two words s and t in the sentence pairs is highly correlated.

	s in S	s not in S
t in T	59	6
t not in T	8	570934

Table 2.3: Example values: co-occurrence frequencies for all sentence pairs (S, T): observed values

	s in S	s not in S
t in T	0.00763	66.99
t not in T	66.99	570875

Table 2.4: Example values: co-occurrence frequencies for all sentence pairs (S, T): expected values

Figure 2.7 shows an ISA matrix for an example sentence pair, words of the source sentence $s_1, ...s_7$ and the target sentence $t_1, ..., t_{13}$. The grayscale of the cells represent either the $I(t_i, s_j)$ or X^2 values, depending on which measure is used.

The ISA algorithm first finds the cell with the highest value. Then it expands from that seed cell to the largest possible rectangular region, which meets the following conditions: The value of the cells in the expanded region is within a certain threshold to the seed cell and the expanded region would not block a free cell with higher value. After that expansion is finished, all cells in the same rows and columns as the marked region are marked as blocked.

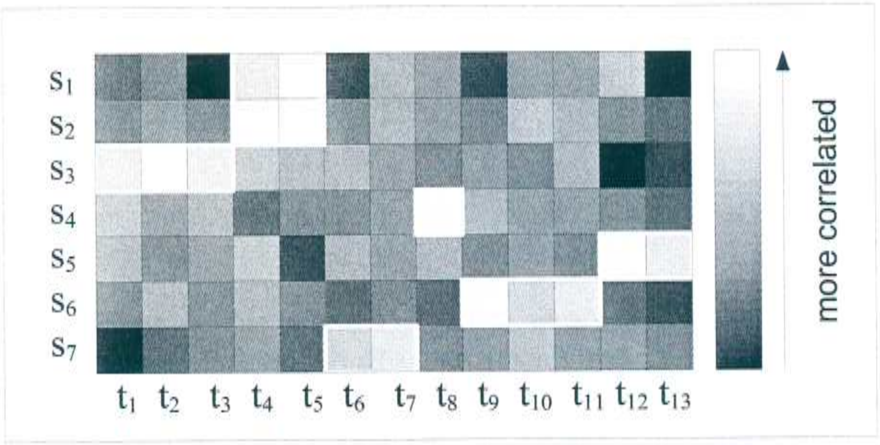


Figure 2.7: ISA: example sentence pair correlation matrix

This procedure is repeated until there are no free cells left.

These marked regions (see Figure 2.7) are the phrase translation pairs, which are then included in the transducer.

2.1.6 Translation Models: PESA

The Phrase Extraction via Sentence Alignment (PESA) (Vogel et al. [2004]) algorithm is another method for aligning a phrase in a source language sentence to a phrase in the corresponding target language sentence.

Given a phrase from the source sentence from left phrase boundary j_1 to right phrase boundary j_2 PESA tries to find the best matching phrase boundaries i_1 and i_2 in the target sentence by calculating a modified IBM1 sentence alignment probability $P_{(i_1,i_2)}(\vec{t}|\vec{s})$.

A regular IBM1 sentence alignment probability $P(\vec{t}|\vec{s})$ gives the probability of sentence t being a translation of sentence s, regardless which words are translations of which words exactly. Using the alignment probabilities from an IBM1 lexicon, it sums over all word to word translation probabilities.

$$P(\vec{t}|\vec{s}) = \prod_j (\sum_i P_{(s_j,t_i)}) \tag{2.6}$$

The modified PESA sentence alignment probability gives the probability that phrase (i_1, i_2) is a translation of phrase (j_1, j_2) and the rest of sentence t being a translation of the rest of sentence s. Using the word translation probabilities from an IBM1 lexicon, it sums over all words but those in 'forbidden' areas see figure 2.8 and equation 2.7.

$$P_{(i_1,i_2)}(\vec{t}|\vec{s}) = \prod_{j=1}^{j_1-1} (\sum_{i \notin (i_1 \dots i_2)} P_{(s_j,t_i)}) \prod_{j=j_1}^{j_2} (\sum_{i \in (i_1 \dots i_2)} P_{(s_j,t_i)})) \prod_{j=j_2+1}^{n} (\sum_{i \notin (i_1 \dots i_2)} P_{(s_j,t_i)}) \tag{2.7}$$

Then it modifies the target phrase boundaries to find the ones, which maximize the sentence alignment probability:

$$(\widehat{i_1}, \widehat{i_2}) = argmax_{(i_1,i_2)} \left\{ P_{(i_1,i_2)}(\vec{t}|\vec{s}) \right\} \tag{2.8}$$

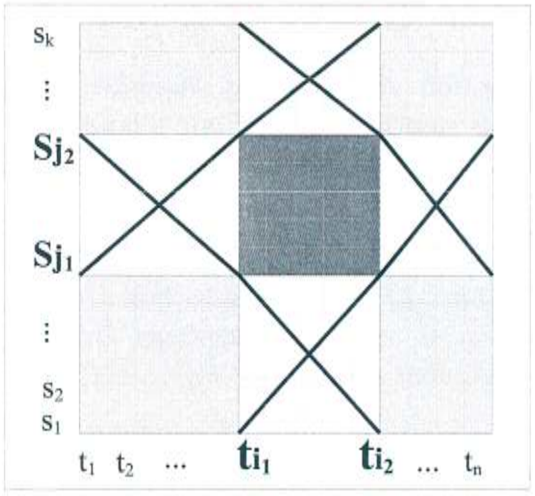


Figure 2.8: PESA: modified sentence alignment

Calculating the alignment in both directions improves translation quality.

$$P_{(i_1,i_2)}(\vec{s}|\vec{t}) = \prod_{i=1}^{i_1-1}(\sum_{j\notin(j_1...j_2)} P_{(t_i,s_j)})\prod_{i=i_1}^{i_2}(\sum_{j\in(j_1...j_2)} P_{(t_i,s_j)}))\prod_{i=i_2+1}^{k}(\sum_{j\notin(j_1...j_2)} P_{(t_i,s_j)}) \tag{2.9}$$

Using the probabilities from both directions the best (i_1, i_2) is:

$$(\widehat{i_1}, \widehat{i_2}) = argmax_{(i_1,i_2)}\left\{(1-c)\,P(i_1,i_2)(\vec{s}|\vec{t}) + cP(i_1,i_2)(\vec{t}|\vec{s})\right\} \tag{2.10}$$

All the phrases whose sentence alignment probability is within a certain range from the best one, are used in the transducer.

2.1.7 Language Model

A Language Model (LM) is used to predict words with associated probabilities given the word sequence up to the present, the i.e. history. If h represents the history, the language model gives a probability $P(w|h)$ for all words w in the vocabulary. Because using the whole history, e.g. the whole document seen up to this point is intractable, the probability is approximated using only the recent history containing a few words. In an n-gram language model, the history h contains $n-1$ words.

$$P(w_i|h) = P(w_i|w_1, w_2, w_3, \dots w_{i-1}) \approx P(w_i|w_{i-(n-1)} \dots w_{i-1}) \quad (2.11)$$

Usually n is between 2 and 4, due to the increase of size of the model, the amount of data required to train it and the time for search with increasing n. To build an n-gram language model and determine the probabilities of different sequences it is necessary to examine a preferably large training corpus.

In this project all language models are trigram language models trained with the SRI Language Model Toolkit (Stolcke [2002]).

The perplexity of a language model with respect to a sentence measures how well the model predicts the sentence; that is, how well it fits the sentence.

The perplexity of a language model LM with respect to sentence s of k words is the reciprocal of the geometric average of the probabilities of the predictions in s:

$$PPL_{LM}(s) = P_{LM}(s)^{-\frac{1}{k}} \quad (2.12)$$

2.2 Similarity Measures for Information Retrieval

Information retrieval tries to solve the problem of finding documents in a database, that are relevant given a query. A query can be a list of keywords or a document. If the query is a document, the goal is to find other documents of the same topic. One approach is to estimate the similarity of the documents to the query and return the documents with the highest similarity score. There are several available similarity measures. Two common measures were used in this project, implemented in the CMU LemurTk.

2.2.1 Okapi model

The Okapi retrieval model is based on a probabilistic model. It ranks documents by the probability of their relevance to a query.

The Okapi BM25 term weighting function (Robertson and Walker [2000]):

$$\sum_{t \in Q,D} w^{(1)} \frac{(k_1+1)tf}{K+tf} \cdot \frac{(k_3+1)qtf}{k_3+qtf} \tag{2.13}$$

where:

Q	is the query
D	is the document
t	is a term (usually a word) from the query
tf	is the term's frequency in a document
qtf	is the term's frequency in the query
N	is the number of documents in the database
df	is the number of documents that contain the term
dl	is the document length
$avdl$	is the average document length
$w^{(1)}$	is the Robertson/Sparck Jones weight of t in Q $w^{(1)} = log\frac{(r+0.5)/(R-r+0.5)}{(df-r+0.5)/(N-df-R+r+0.5}$
R	is the number of documents relevant to a specific topic
r	is the number of relevant documents containing the term
K	is $k_1(1-b)+b\frac{dl}{avdl}$
k_1, b and k_3	are constants

Relevance information is not available for the data used in this project. Without relevance information ($R = r = 0$) the Robertson/Sparck Jones weight (Robertson and Sparck-Jones [1976]) reduces to an inverse document frequency weight (Robertson et al. [1994]), which leads to the document scoring function:

$$\sum_{t \in Q,D} ln \frac{N - df + 0.5}{df + 0.5} \cdot \frac{(k_1 + 1)tf}{(k_1(1 - b) + b\frac{dl}{avdl}) + tf} \cdot \frac{(k_3 + 1)qtf}{k_3 + qtf} \quad (2.14)$$

with k_1 between 1.0-2.0, $b = 0.75$ and k_3 between $1 - 1000$. The Lemur tk default values used in this project are $k_1 = 1.2$, $b = 0.75$ and $k_3 = 7$.

2.2.2 TF-IDF model

The TF-IDF model (Term Frequency - Inverse Document Frequency) is a vector space model for document representation. It uses a cosine distance to estimate document similarity.

Each text, documents in the database or a query, is represented as a vector. The vector space has one dimension for every term in the document collection. A term is usually a word, but it is also possible to use bigrams etc. If a term t_i occurs in the text, the corresponding dimension in the vector representing the text is assigned a non zero value, for example the count of the word in the document x_i. The similarity between a query and a document can then be determined by the distance of the document vector $\vec{d} = (x_1, x_2, \ldots, x_n)$ and the query vector $\vec{q} = (y_1, y_2, \ldots, y_n)$ (n is the number of terms in the document collection) in the vector space.

To emphasize the words that contribute to the topic and weaken the influence of very common words that occur in every document, all terms are weighted by a term weighing function. The weight has two components: the document term frequency (TF) tf_d, which is usually the count of the term in the document, and the inverse document frequency (IDF) $log(N/df_t)$, where N is the number of documents in the database and the document term frequency df_t is the number of documents that contain term t.

The weighted vectors $\vec{d}$ and $\vec{q}$ are:

$$\vec{d} = (tf_d(x_1)idf(t_1), tf_d(x_2)idf(t_2), \ldots, tf_d(x_n)idf(t_n)) \tag{2.15}$$
$$\vec{q} = (tf_q(y_1)idf(t_1), tf_q(y_2)idf(t_2), \ldots, tf_q(y_n)idf(t_n)) \tag{2.16}$$

The score of document d for query q is given by the cosine distance between the vectors:

$$s(\vec{d}, \vec{q}) = \sum\nolimits_{i=1}^{n} tf_d(x_i) \cdot tf_q(y_i) \cdot idf(t_i)^2 \tag{2.17}$$

In this project I used the Lemur TF-IDF model (Zhai [2001]). It uses the Okapi TF formula as document TF function tf_d:

$$tf_d(x) = \frac{k_1 x}{x + k_1(1 - b + b\frac{dl}{avdl})} \tag{2.18}$$

The Lemur tk default values used in this project are $k_1 = 1$ and $b = 0.3$ for the document tf_d and $k_1 = 1000$ and $b = 0$ for the query tf_q.

3 State of the Art

3.1 Language Model Adaptation

The idea to use information retrieval to select data that matches the test data's domain and topic is based on work that was done by Mahajan et al. [1999] for language models for speech recognition. Because standard 3-gram language models cannot model long-term context information, Mahajan et al. use information retrieval to adapt the language model to the topic of discussion. They use the known history of the document as a query against the database of all training data. The known history can be the recognition result from a first pass a multi-pass system, or the already recognized first part of a document in a one-pass system. The top ranking retrieved documents are then used to train a topic-specific language model, which is then interpolated with a general language model built from all data. To answer the question of how many of the retrieved documents to use, they trained several specific language models from the top 50, 100, 200 and 400 documents and interpolated all of them with the general language model.

These concepts were applied to statistical machine translation by Eck et al. [2004]. They used a first pass translation as qurey to retrieve top n documents and then built a new adapted language model from the retrieved data. They made experiments with story- and sentence-based retrieval, and compared different similarity measures and techniques like the use of stemmers or stopword lists for information retrieval. They had the best results training an adapted language model for each sentence from the test data. Similarity measures other than TF-IDF, stemming and stopwords did not give any improvement. To determine the optimal number of sentences to train the adapted language model, they tried using several sets of top sentences between the top 100 and the top 50000 sentences.

Zhao et al. [2004] extend this approach by using several ways to generate the query. For the query they used the best hypothesis, the n-best hypotheses and all possible translations from the translation model. For these three options they generated bag-of-word queries and structured queries, that incorporate information about word order and proximity information.

Kim and Khudanpur [2003] used a similar idea for language model adaptation for speech recognition in a language with sparse data. They translate a first speech recognition result into a language with a lot of available data, then retrieve stories of the same topic using information retrieval. These stories are translated into the language with insufficient training data and then are used to train a better language model for recognition in that language. Kim and Khudanpur use the likelihood of the test data against the adapted language model to find its optimal size. They calculate the likelihood of the test data against the language model built from different sets of retrieved and translated documents. This way they find the optimal number of documents to use.

3.2 Translation Model Adaptation

Little work has been done on adaptating the translation model for Statistical Machine Translation. One method for the adaptation was proposed by Wu and Wang [2003]. They focus on the actual word alignment and improve it by training different alignment models from in-domain and out-of-domain data and interpolate them. It is necessary for this approach to have at least a small separate amount of in-domain training data available. They do not adapt to the test set.

4 Translation Model Adaptation

Generally statistical systems perform better when trained on more data and also on higher quality data. Training data quality improves if the data is less noisy and when it better fits the translation task. For example, in natural language processing domain specific systems usually outperform general domain systems. Another possibility to better fit the system to the task is to adapt it to the specific test data. E.g. off-the-shelf speech recognition systems are often adapted to one specific speaker.

In a statistical machine translation system adaptation to the text to be translated could be done on various levels. Figure 4.1 illustrates possibilities of different adaptation approaches:

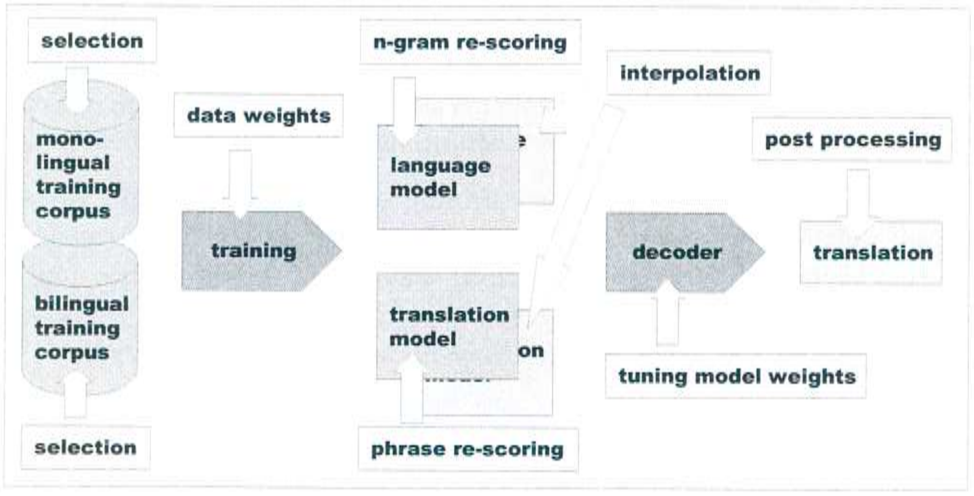


Figure 4.1: Possible ways of Adaptation in the Translation Process

- Selection:
  Choose those parts of the training data, that are especially fitting to the task, for training the language model and the translation model.
- Data Weights:
  Use the more adequate training data with a greater weight in training.
- Phrase and N-gram re-scoring:
  Manipulate the model scores directly, to change the ranking of certain phrase translations or n-grams.
- Interpolation:
  Train several specific and general translation or language models and interpolate them.
- Tuning Model Weights and System Parameters:
  Adjust the weights for different models and other parameters used in the decoder to perform best for a certain development test set.
- Postprocessing:
  Re-write or paraphrase the translation to remove domain-atypical formulations.

In this project the translation model is adapted to the test set by selecting a part of the parallel corpus for training using Information Retrieval.

The idea is to find data in the training corpus that matches the test data's domain or, more specifically, topic and style. The translation system is trained only on this reduced selection of training data. Out-of-domain data might hurt translation quality; by excluding it, we hope to get translations that come from the same domain as the test set.

```
Algorithm:

1. for each test sentence
        use test sentence to select n most similar sentences
        from the training data
2. build translation model using only the training sentences
   found for each test sentence
3. translate with adapted translation model
```

Because information retrieval solves the problem of finding documents in a database that match the topic of a query, it was chosen as a tool to find and rank similar sentences.

4.1 Sentence Selection using Information Retrieval

For all information retrieval tasks in this project, I used tools provided by The Lemur Toolkit for Language Modeling and Information Retrieval (LemurTk).

The document collection consists of the source language part of all available bilingual training data, each sentence representing one document. Using only the source language to build the retrieval index has the advantage, that it is independent from the quality of the translation system, as no first pass translation is necessary.

Each sentence from the test data was used as one separate query.

In Information retrieval there are several methods to score a document against a query. Cosine distance with TF-IDF term weights, Okapi similarity measure or Kulbach-Leibler distance are some popular ones.

For the first experiment setting, Spanish – English (see chapter 5.3), I compared the retrieval results for cosine distance similarity measure with TF-IDF term weights (see section 2.2.2) to those of probabilistic document similarity model with Okapi term weighing function (see section 2.2.1). There was no significant difference.

For individual queries, the portion of TF-IDF retrieved documents that occur in the Okapi retrieval result ranges from 30% to 60%. For the whole test data, this overlap increases to over 75% for the top 300 sentences per query and over 90% for the top 1000 retrieved sentences per query.

It is not surprising that the translation scores for experiments using TF-IDF and Okapi similarity measures show no significant differences (see section 5.3.4). For this reason only TF-IDF retrieval was used for all further experiments.

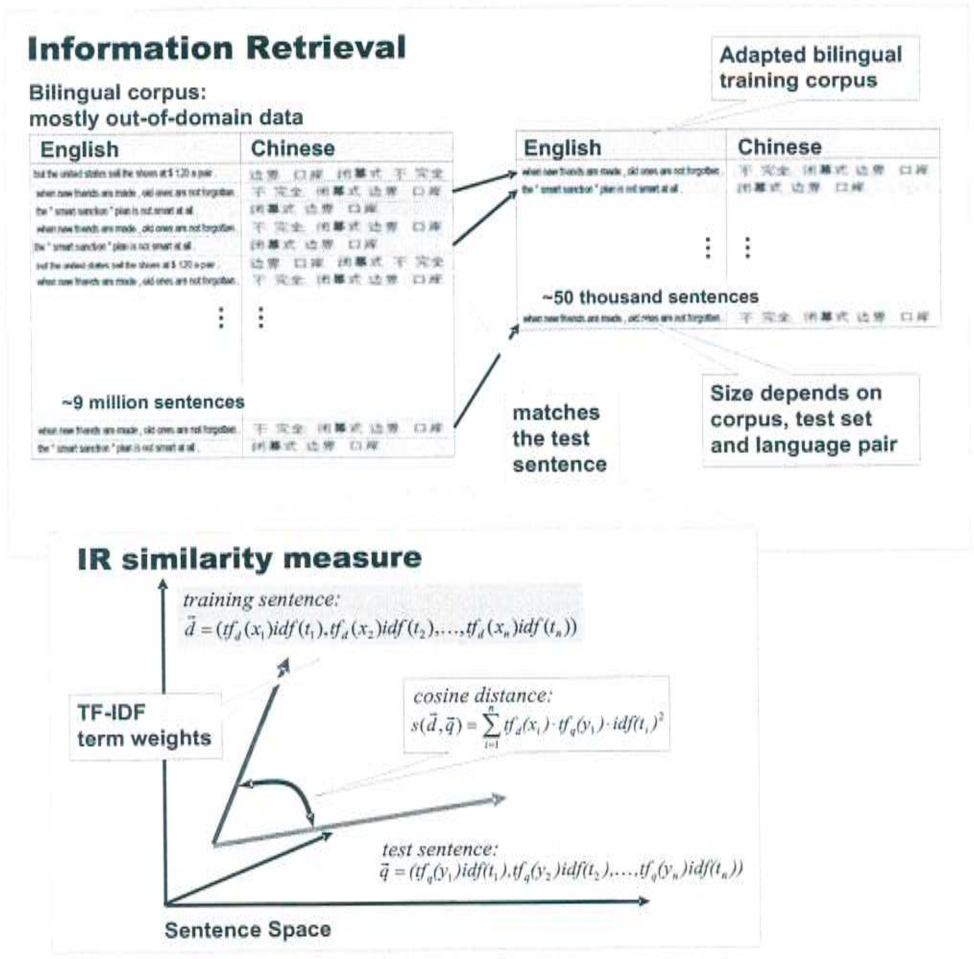

Information Retrieval
Bilingual corpus:
mostly out-of-domain data
English
Chinese
~9 million sentences
Adapted bilingual training corpus
English
Chinese
~50 thousand sentences
matches the test sentence
Size depends on corpus, test set and language pair
IR similarity measure
training sentence:
$\vec{d} = (tf_d(x_1)idf(t_1), tf_d(x_2)idf(t_2), \ldots, tf_d(x_n)idf(t_n))$
TF-IDF term weights
cosine distance:
$s(\vec{d},\vec{q}) = \sum_{i=1}^{n} tf_d(x_i) \cdot tf_q(y_i) \cdot idf(t_i)^2$
test sentence:
$\vec{q} = (tf_q(y_1)idf(t_1), tf_q(y_2)idf(t_2), \ldots, tf_q(y_n)idf(t_n))$
Sentence Space

Figure 4.2: Adaptation based on Information Retrieval

4.2 Adapted Translation System

After selecting the top n similar sentences for each individual sentence from the test data, they were used together to train an adapted translation model.

The main reason for not training individual translation models for each sentence is that a translation model trained on only a few hundred sentences is unlikely to give robust probabilities. If the whole test data is from one domain, it can be expected that a phrase or word translation, that was right for one sentence in the test data, will be correct for other sentences as well. If a particular test set consists of parts from different domains, a solution could be to train separate translation models for these parts of the test data.

The combined selected training data usually contains duplicate sentences, as the same sentences from the document collection might recur in the top n retrieval results for different test sentences, especially for sentences from the same topic or domain. There are more duplicates the smaller the document collection is and with higher values of n. For example in the first experiment setting (see section: 5.3), which uses a small document collection, the combined training set already contained 37.5% duplicates for $n = 50$ and as much as 74% for $n = 1000$.

It is not clear if the duplicates help the translation performance. The duplicates force the translation probabilities towards the more often observed words, which could help. So I compared the translation performance of systems trained on training sets with and without the duplicates. As to be expected the translation scores differ for individual selections, but overall there is no obvious tendency towards one or the other alternative (see section 5.3).

In this project the target language model used in the decoder was always trained only on the target language part of the bilingual training corpus - no further monolingual training data was added. When this language model is not adapted it might contradict a correct translation from the adapted translation model. So it virtually suggested itself to adapt the language model with the same technique. I used the English part of the adapted training set to train a new adapted language model. This gave an improvement in all experiments.

4.3 Language Model Perplexity as Measure to Determine Selection Size and Re-rank

As shown in the experiments (see chapter: 5), the optimal amount of selected data varies for different language pairs, training corpora or test sets. It would be very useful not to be forced to do a time consuming grid search each time we change the setting, to determine the best size of the adapted training corpus.

To do a grid search for the optimal selection size, it is necessary to translate the test set each time, to compare the translation scores for the different selections. It is only possible to compare translation scores for a development test set with reference translations. The estimate for the optimal selection size then has to be transferable to the actual test set.

The optimal number of sentences to select for training might also vary for each individual test sentence. It is easy to imagine, that there are not too many useful sentences to be found for a test sentence like 'Good morning Mr. Smith', but with a sentence like 'I want to get off at the bus stop in front of the Saks Fifth Avenue department store, so could you tell me when we get there' it is a whole different situation.

Looking at the TF-IDF scores for some example queries (Figure 4.3) shows that they are not comparable, as the top sentences for a short query have much lower scores than the ones for a longer query.

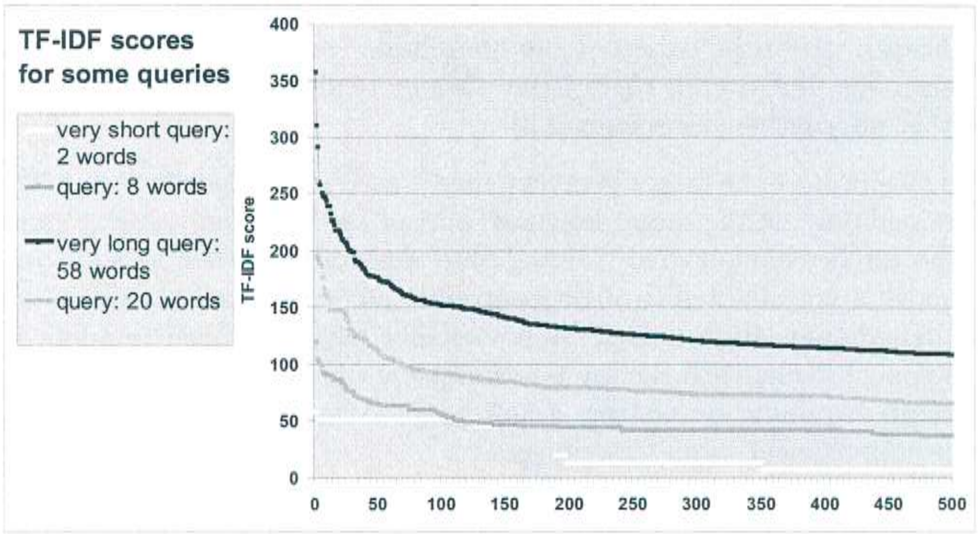


Figure 4.3: TF-IDF scores for some queries

Therefore it is not easy to define a score threshold for including retrieved sentences in the selection. Also, the scores fall smoothly for most sentences, so there is no indication where to stop including the retrieved sentences in the adapted training corpus. A certain range from the top score or a percentage of the score scope is not a lot better than using the top n retrieval results and would have to be determined by a grid search as well.

Kim and Khudanpur [2003] presented an idea to use the likelihood of the test data for measuring the quality of the language model built from a selection.

Idea:

To judge how well a selection of training data fits the test sentence, measure the perplexity of a language model built from this selection against the test sentence. Find the perplexity minimum to determine the optimal selection size for each test sentence.

Figure 4.4 shows the behavior of the perplexity of the language model built from a selection for one test sentence against the respective test sentence. The perplexity was measured for the top $10, 20, \ldots 1000$ sentences, as they were returned by the TF-IDF retrieval. (For the diagram I chose four sentences from the Spanish – English experiment (see chapter 5.3).)

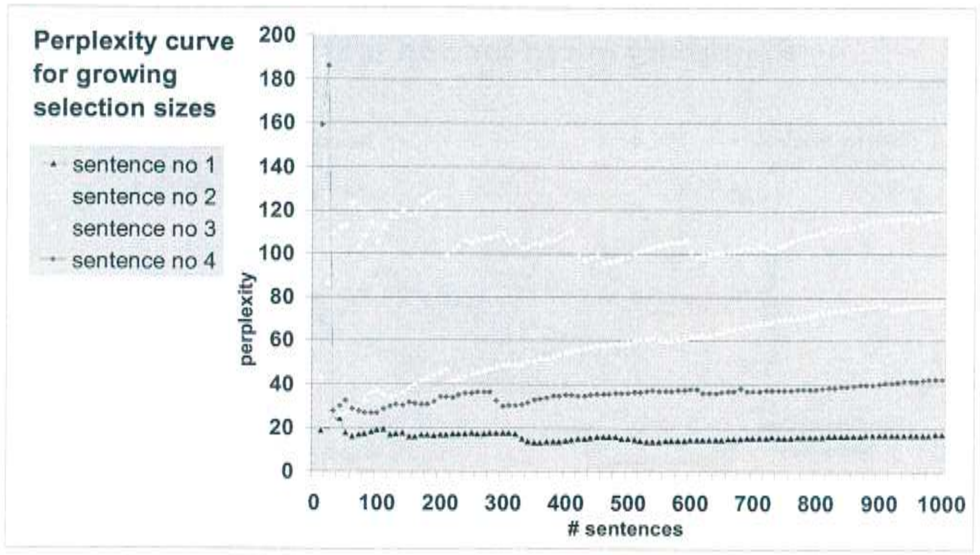
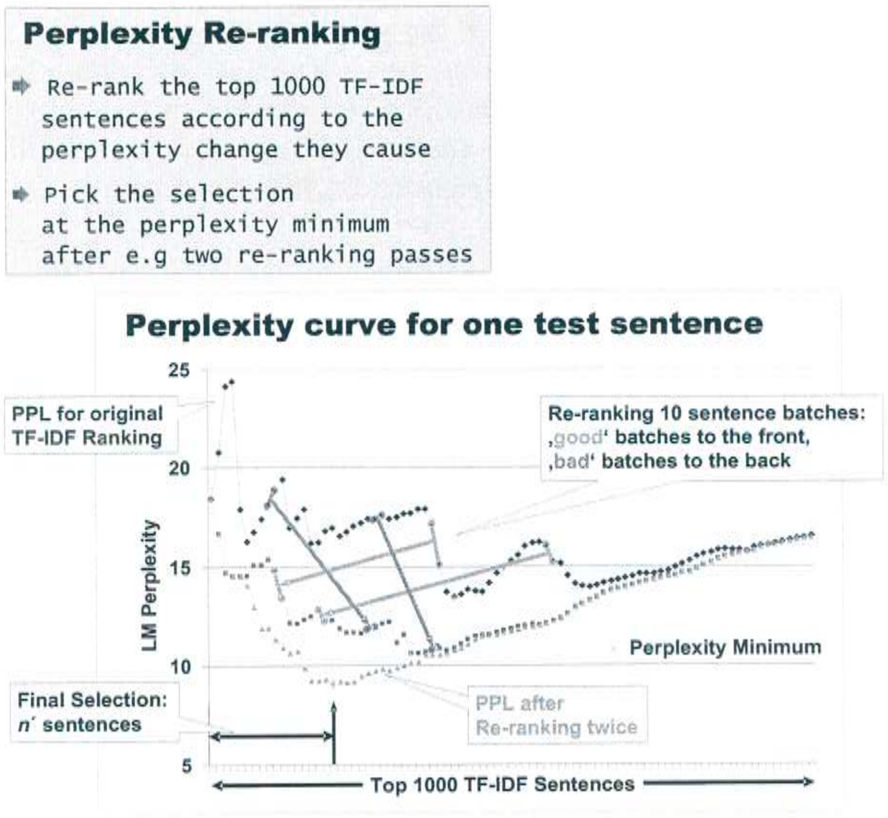


Figure 4.4: perplexity for different selections for some queries

Unfortunately the perplexity curve does not show a nice, convex shape for most sentences. There are even sentences where the perplexity minimum is at the first or second batch. The previous experiments showed that the optimal selection size is definitely bigger than 10 sentences per test sentence. So it seems that picking the selection with the lowest perplexity is not the best method in many cases.

Because information retrieval ranks the sentences according to their term weights, while language model perplexity gives information about matching word order, some of the 10-sentence-batches added early to the selection make the perplexity worse, while some 10-sentence-batches ranked lower in the TF-IDF retrieval improve the perplexity.

To exploit that additional information, I use the perplexity change that each batch of sentences causes as an additional measure for ranking sentences on the top n sentences retrieved by TF-IDF (see Figure 4.5).



Perplexity Re-ranking

- Re-rank the top 1000 TF-IDF sentences according to the perplexity change they cause
- Pick the selection at the perplexity minimum after e.g two re-ranking passes

Figure 4.5: Perplexity Re-ranking

Because the size of the selection increases over the testing run, the changes in perplexity are not comparable - to add e.g. a ten sentence batch might change the perplexity considerably at the beginning, when the selection has only 50 sentences but the same batch will only decrease or increase the perplexity a little when added to an 800 sentence selection. So, the batches cannot be completely re-ranked according to the perplexity change. They are only classified as 'good' or 'bad' during the pass. To get batches that improve the perplexity into the selection, the batches are re-ranked. All the 'bad' batches are taken out of the list and shuffled to the end. Still the main selection criterion is TF-IDF information retrieval, as I look only at the e.g. top 1000 sentences ranked by TF-IDF. Also, the original TF-IDF ranking is kept among the good as well as the bad batches.

After one re-ranking pass the shape of the perplexity curve for 10 sentence batches is already smoother and has a considerably lower perplexity minimum than the original order. After re-ranking a second time, the measured perplexities are even lower (Figure 4.6 and 4.7).

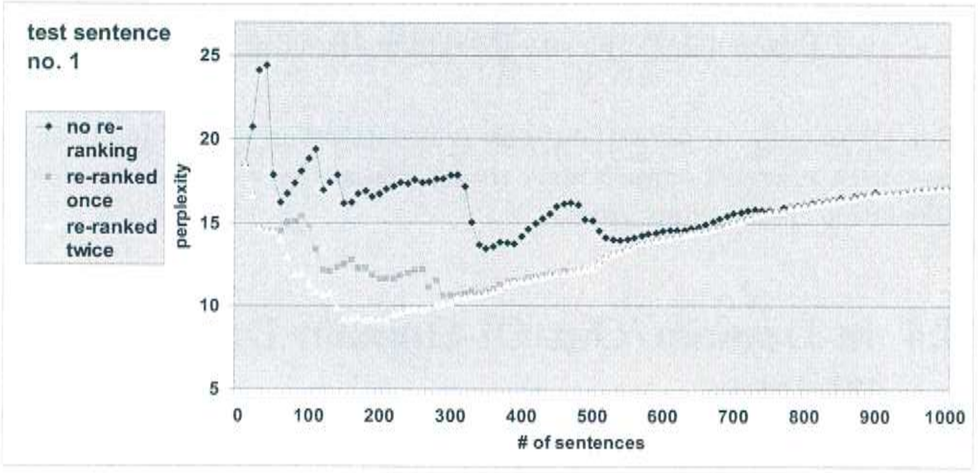


Figure 4.6: Perplexity Re-ranking Results: query 1

On average, like for test sentence number 1 (Figure 4.6), the selection chosen after re-ranking is smaller than without re-ranking. For an extreme example like test sentence number 4 (Figure 4.7), which had the perplexity minimum at the first batch before the re-ranking, more sentences get into the selection after re-ranking. After re-ranking twice there are already nearly no 'bad' batches before the perplexity minimum and 'good' batches after it, so it's not worth the computation time to re-rank a third time.

After re-ranking I determined the selection size for each sentence by picking the selection with the lowest perplexity.

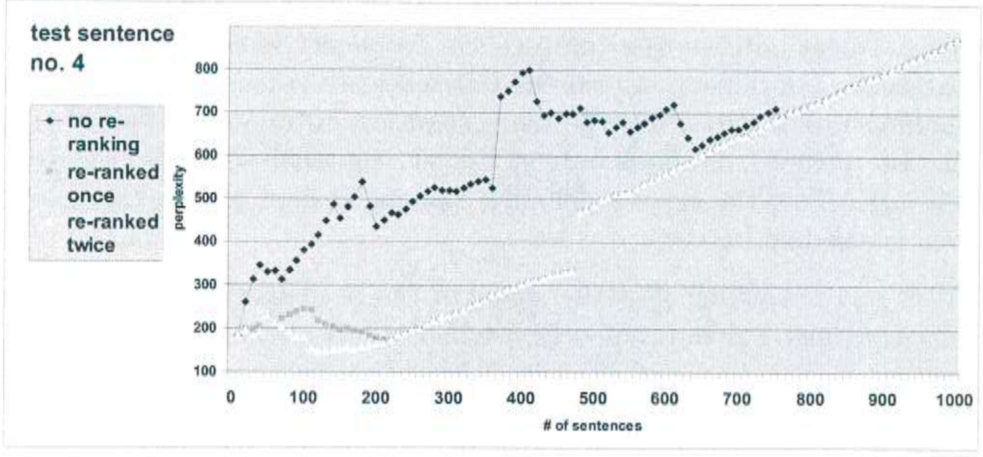


Figure 4.7: Perplexity Re-ranking Results: query 4

This technique can do without any translation run or development test data with reference translations to determine the selection size for the translation model adaptation.

4.4 In-Domain/Out-Of-Domain Data Weights

In the Chinese – English experiment setting (section 5.4), the ratio between the in-domain and the out-of-domain training data is extreme: 20 thousand lines in-domain to 9 million lines out-of-domain data. Adding only a small fraction of this additional data to the in-domain data, e.g. 1%, already reduces translation quality. This is probably because the out-of-domain data is so different that it overrides the correct in-domain translations from the in-domain data. On the other hand it is desirable to make use of as much additional data as possible to cover more words and to estimate commen words more robust.

To address this problem, I assigned a higher weight to the in-domain data in the training for the IBM1 lexica, which are used by the PESA algorithm to score the phrase translations.

Initial experiments with a weight of 3:1 showed, that the translation scores improve and that more out-of-domain data can be added without loosing translation quality.

To find the optimal weight for the in-domain data I did a grid search for three selections picked by the automatic PPL selection size determination. Here it is still an unsolved question how to determine the optimal weight automatically. One possibility would be to train two different lexica and tune weighing factors for them during decoding time. Unfortunately, then some beneficial effects of training the lexica from all data would be lost.

As a rule of thumb the weight w for the in-domain data: $w = \left\lceil \frac{\#lines\ out-of-domain}{\#lines\ in-domain} \right\rceil$ which means balancing the number of lines of in-domain and out-of-domain data, worked considerably well.

5 Experiments

5.1 Overview

The techniques described before were tested on two different corpora and setups.

In the first setup, Spanish to English, the training data is a mixture of data from the medical domain and tourism domain. Both types of data are rather similar in style, as they both consist of dialogs with short sentences. The relatively small retrieval index was built from all available data. In this setup the task was to find the in-domain data in a mixed document collection.

I then applied the ideas to a different scenario. In the experiment translating Chinese to English the amount of out-of-domain data is much larger than the available in-domain data. Also the style is considerably different. The in-domain data consists of tourism dialogs with short sentences, the out-of-domain data is mostly Chinese news wires and speeches. Words that are covered by the available in-domain data can be translated fairly well. Here the major goal of the adaptation is to cover unknown words without loosing translation quality by adding too much out-of-domain data.

In both cases just adding the larger amount of out-of-domain training data to the small amount of in-domain data does either not significantly improve the performance of a baseline system that was trained on the in-domain data only or hurts translation quality considerably.

	Test Data	In-domain	Out-of-domain	Main Goal
Spanish - English	doctor - patient dialogs	25k lines medical	125k lines tourism (BTEC)	find & extract in-domain data for word sense disambiguation
Chinese - English	tourism phrases	20k lines BTEC tourism	9Mio lines news & speeches (TIDES)	cover unknown words without introducing undesired translation alternatives

Table 5.1: Experiment Settings Overview

5.2 Translation System

All experiments were conducted with the CMU statistical machine translation system (Vogel et al. [2003]).

The decoder can integrate several translation lexica and transducers in the translation process. In the Spanish – English experiment setting I used IBM1 lexica and HMM and ISA transducers (see section 2.1.3 – 2.1.5). In the Chinese – English experiments I used the new PESA transducer (see section 2.1.6) to show, that the adaptation technique still gives improvement with the latest translation system. The PESA algorithm uses IBM1 lexica for phrase scoring. In both cases the language model is a trigram language model with Kneser-Ney-discounting built with the SRI Language Modeling Toolkit (Stolcke [2002]) using only the English part of the bilingual corpus.

The best scores for NIST (Doddington [2001]) or BLEU (Papineni et al. [2002]) evaluation metrics are usually achieved using considerably different tuning parameters for the translation system. In the experiments for the Spanish – English translation the system was only tuned towards NIST, in the Chinese – English experiments I tuned the system towards both NIST and BLEU respectively.

5.3 Setting 1: Spanish - English, mixed domain data

5.3.1 Test and Training Data

The test data for the Spanish – English experiments consists of 329 lines of medical dialogs (6 doctor-patient dialogs). It contains 3,065 Spanish words and 3,399 English words (tokens) with one reference translation. I had three different corpora of bilingual training data available. 25,077 lines of medical dialogs can be regarded as in-domain data. Additional out-of-domain data are 2,323 lines of tourism dialogs and 123,416 lines of BTEC (Basic Travel Expression Corpus) data described in Takezawa et al. [2002] (see Table 5.2), which is also from the tourism domain.

Training set	#lines	#words English	#words Spanish
Medical dialogs (in-domain)	25,077	218,788	208,604
Tourism dialogs (out-of-domain)	2,323	26,600	24,375
BTEC data (out-of-domain)	123,416	903,525	852,364
Overall	150,816	1,148,913	1,085,343

Table 5.2: Training Data Sizes for Experiments Spanish – English

5.3.2 Baseline Systems

I compared all adapted systems to two baseline systems. The first one was trained on the medical data only, the second baseline system uses all available training data. The score of the first baseline can be seen as an oracle score, as in this setting the goal of the adaption is to find the in-domain data in an mixed training corpus.

In this experiment the translation system was only tuned towards the NIST score.

The scores in table 5.3 show, that the baseline system that only uses the available in-domain data is not necessarily better than the system that

System	#Lines	NIST
only in-domain data	25k	5.1820
in-domain and out-of-domain data	150k	5.2074

Table 5.3: Baseline System results, Spanish – English

uses all data. The best NIST score is actually a little, but not significantly higher for the second baseline system. There are three possible reasons for the improvement using the additional data:

1. It covers 27% of the previously unknown words (36 of 132).
2. It consists of dialogs like the medical data. Those dialogs cover primarily a different topic, but they still might be helpful for the translation as the sentence structure is fairly similar.
3. Some of the tourism data is actually in-domain, as there are dialogs in this corpus, where tourists have a medical problem and ask for a doctor.

5.3.3 Distinct and Non-distinct Training Corpus

For the Spanish – English setting I built the information retrieval index using the Spanish part of all available in-domain and out-of-domain data. (I used the Lemur Toolkit (LemurTk) for all Information retrieval tasks.)

The top n similar sentences for each Spanish test sentence for $n = 30, 50, 100, 200, 300 \ldots 1000$ were retrieved from the index, using TF-IDF as the similarity measure (see section 2.2.2). For $n = 50$ the selection for the entire test set contained 40% duplicates in 16,196 lines, 75% for n=1000 in 313,201 lines.

The information retrieval sometimes retrieves fewer sentences than asked for. This happens especially for short sentences, when all remaining sentences have a zero TF-IDF score because not even one word matches.

These retrieved sentences form the reduced training set, which was used to train the new adapted translation models. The LM was trained on the entire training data.

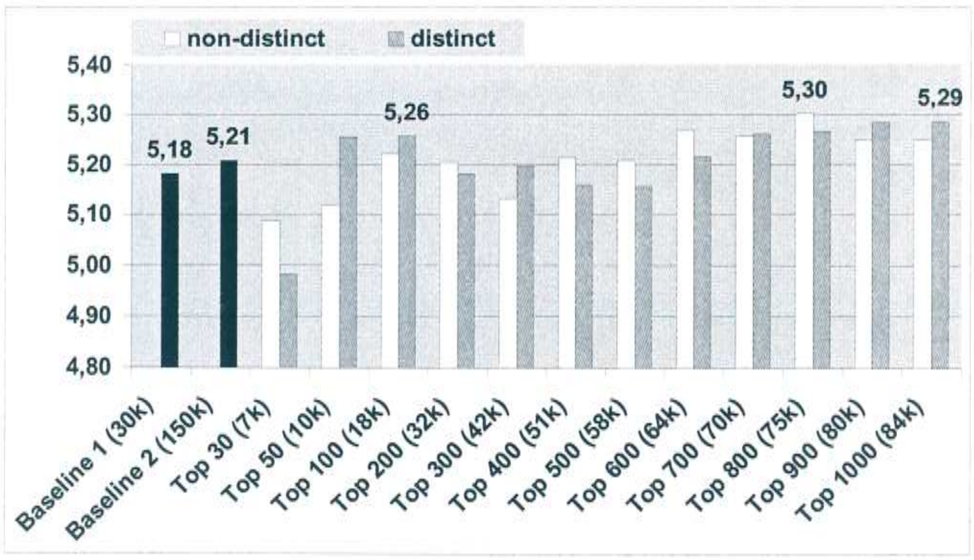


Figure 5.1: Distinct and non-distinct retrieval for Spanish – English (NIST)

Figure 5.1 illustrates the results. The numbers in parentheses on the x-axis denote the number of distinct sentences that were used to train this particular system. The non-distinct training set contained some of those sentences more than once.

The highest NIST score for this experiment in the non-distinct case was 5.3026 at top 800 retrieved sentences. This training set has about 75,000 different sentences which is about half of the original training data and about 250,000 sentences with duplicates. When the duplicate sentences were removed for the actual training the highest NIST score was 5.2878 for top 900 (about 80,000 sentences).

5.3.4 Okapi IR

To see how much impact a different IR similarity measure has on translation results, I compared TF-IDF and Okapi retrieval.

For individual queries the portion of retrieved documents from the TF-IDF retrieval, that can be found in the Okapi retrieval result range from 30% to 60%. Looking at the retrival result for the whole testset the

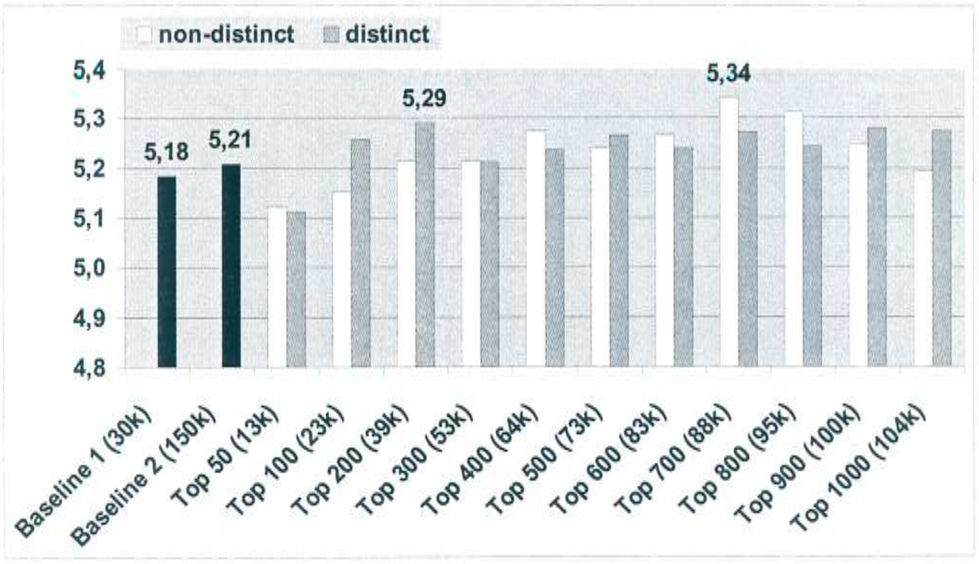


Figure 5.2: Okapi retrieval for Spanish – English (NIST scores)

overlap increases to over 75% for the top 300 sentences per query and over 90% for the top 1000 retrieved sentences per query.

So it is not surprising, that even though the translation scores differ in individual cases, in general none of the alternatives is significantly better than the other.

5.3.5 TM and LM Adaptation

Up to this point I always used the general language model of baseline 2 (all available training data) for the translations.

In this next set of translation runs I used the English part of the adapted training corpus to train a new adapted language model.

This further improved the best NIST score to 5.3264 (Top 200 with about 64,000 sentences of overall training data and just about 32,000 distinct sentences). It had a bigger impact on the smaller systems, as the adapted and the general LM become more similar for larger selection sizes.

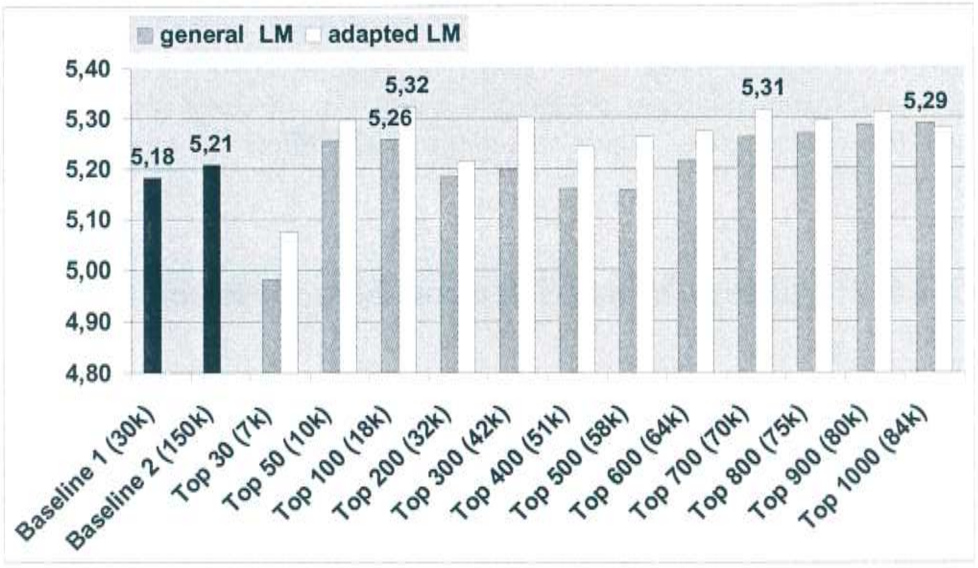


Figure 5.3: TM and LM adaptation for Spanish – English, distinct training corpus (NIST)

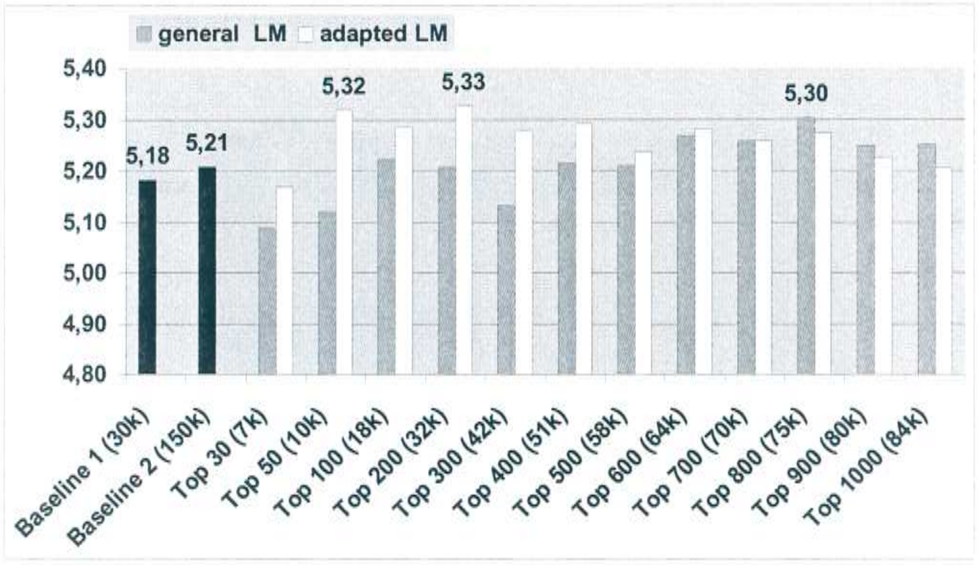


Figure 5.4: TM and LM adaptation for Spanish – English, non-distinct training corpus (NIST)

Diagrams 5.3 and 5.4 compare the NIST scores for translation with and without additional LM adaptation for the distinct as well as the non-distinct training corpus. Both show the same behavior: The matching LM adaptation improves translation quality in all cases, especially for smaller selection sizes, where the adapted LM differs the most from the general LM.

5.3.6 Perplexity based Selection Size Determination

To find the optimal selection size for the adapted training corpus automatically I used the LM perplexity curve calculated for the top 1000 sentences retrieved via TF-IDF retrieval. The perplexity was calculated after adding sentences in batches of 10. Then I re-ranked those 10 sentence batches twice according to the perplexity change they caused.

In this experiment I always built the language model from the adapted training data, as this worked well for the previous experiments.

Diagram 5.5 shows the NIST scores for selection sizes picked at the perplexity minimum before re-ranking and after re-ranking once and twice in comparison to the baselines and the best scores from the previous experiments. The best NIST score of 5.3807 was reached after re-ranking twice.

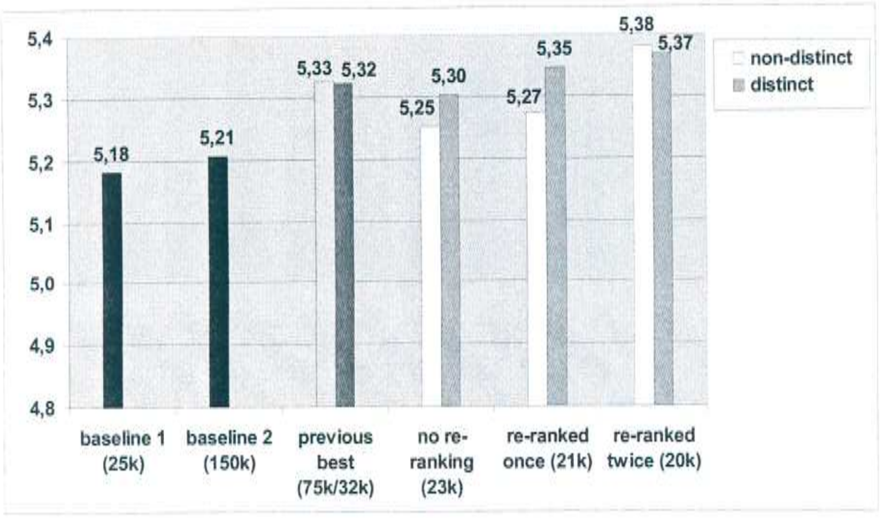


Figure 5.5: PPL based selection size and re-ranking for Spanish-English (NIST)

Figure 5.6 shows histograms for the number of sentences that get selected per test sentence by the PPL selection size determination after the respective re-ranking runs. By re-ranking the most frequent selection size moves from the second batch, which equals 20 sentences, up to 50 sentences and the biggest selections picked ever move from over 800 down to around 400.

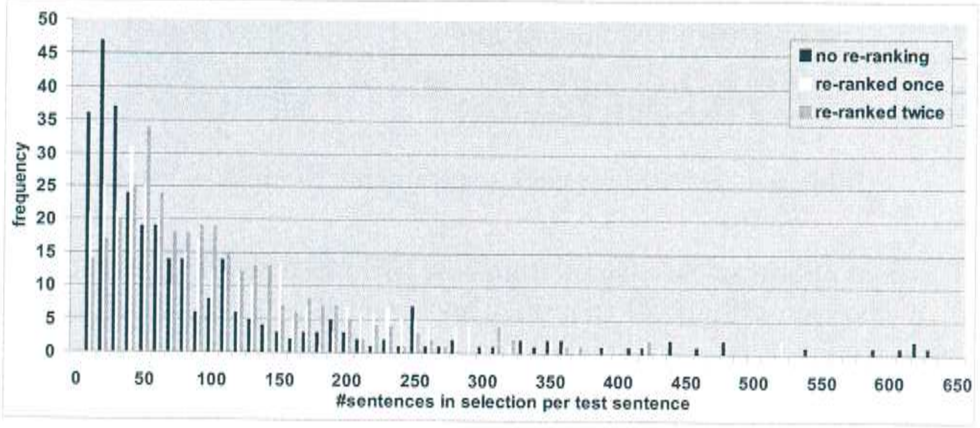


Figure 5.6: Selection sizes per sentence

5.3.7 Summary

The differences between the systems with or without duplicate sentences are not significant. The highest NIST score was reached using a training set that contained duplicates.

Training the language model on a matching adapted data selection clearly improves the performance, especially on smaller selection sizes as the adapted LM differs more from the general LM.

The selection automatically found by perplexity driven selection size determination was able to achieve about the same scores as the best one from the grid search. Perplexity re-ranking improved slightly over them. The selections picked by the PPL based selection size determination are considerably smaller than the ones, that achieved previously the best scores. Probably because the perplexity provides another source of information to further exclude non-matching sentences.

System	#Lines	NIST
baseline 1: in-domain data	25k	5.1820
baseline 2: all data	150k	5.2074
best TF-IDF with duplicates	75k	5.3026
best TF-IDF distinct	84k	5.2878
best Okapi with duplicates	88k	5.3431
best Okapi distinct	39k	5.2902
best with LM adaptation	75k	5.3264
best with PPL selection size	23k	5.3032
best with PPL re-ranking	20k	5.3807

Table 5.4: Results for each experiment: Spanish – English

The final improvement of 3.3% from 5.20 NIST to 5.38 NIST was achieved by a selection of only 20 thousand lines, which is only 13% of the data from baseline 2 and even smaller than the in-domain-only first baseline.

5.4 Setting 2: Chinese - English, in-/out-of-domain data scenario

5.4.1 Test and Training data

The test data for the Chinese – English experiments consists of 506 lines of tourism dialogs. It contains 3,510 Chinese words. There are 16 English reference translations per test sentence available. The in-domain training data consists of exactly 20,000 lines of tourism dialogs from the BTEC corpus (Takezawa et al. [2002]). The index for retrieving additional data was built from about 9.1 million lines of Chinese newswire and speeches.

Training set	#lines	#words English	#words Chinese
in-domain (BTEC)	20,000	188,935	175,284
out-of-domain (newswires & speeches)	9.1 million	144 million	135 million

Table 5.5: Training Data Sizes for Experiments Chinese – English

5.4.2 Baseline System

The baseline system was trained on the available in-domain data only and had a NIST score of 8.1129 achieved a BLEU score of 0.4621.

It was known from earlier results that a system using all available training data does not perform well on this BTEC test set. The vocabulary coverage certainly improves (89 unknown words in the baseline, 4 with the complete TIDES corpus) but the out-of-domain data introduces too many wrong translations as the TIDES corpus is very different from the test set. The ISI/USC machine translation group for example reported at IWSLT 2004 (Ettelaie et al. [2004]) BLEU scores of 0.243 for the unrestricted track and a BLEU score of 0.4405 for the additional data track with in-domain data weights of 5:1. (I used the data from the additional data track at IWSLT 2004 (Akiba et al. [2004]) for this project). I did not explicitly train another baseline from all data for this reason.

In this in-/out-of-domain data scenario one could argue, that adding some data to the small initial system will improve the translation performance, no matter what data is selected. So I selected different numbers of sentences randomly from the complete training corpus as additional training data and compared the translation results to the adaptive selection. From different random selections only small ones could improve over the baseline, where the amount of additional data was not bigger than the in-domain-data. Already at 75 thousand lines additional data, which is less than 1% of the 9 million lines available data, the translation scores drop. Two examples are given in table 5.6.

System	#Added Lines	BLEU	NIST
In-domain data only (20k lines)	–	0.4621	8.1129
best random selection	15k	0.4850	8.2262
largest random selection	75k	0.4501	7.9482
ISI system, additional, weight 5:1	N Million	0.4405	–
ISI system, unrestricted	N Million	0.243	–

Table 5.6: Baseline System Results: Chinese – English

This shows the trade-off between a small domain-specific model that can not cover all words and a larger system that might introduce wrong out-of-domain translations.

5.4.3 In-domain/out-of domain data scenario

With this small amount of in-domain training data at hand the IR index was built for the out-of-domain data only. The top n similar sentences for each Chinese test sentence for $n = 10, 20, 30, \ldots, 300$ were then retrieved from the index, using cosine distance with TF-IDF term weights as the similarity measure (see section 2.2.2).

Then I added the retrieved sentences from the out-of-domain data to the in-domain data to train the translation model.

As I felt that the available in-domain data was too poorly represented especially when adding more and more training data for a larger number of retrieved n I removed the duplicates in all cases. In diagrams 5.7 can be seen and 5.8 that both translation scores start to drop clearly as soon as the out-of-domain data outweighs the in-domain data.

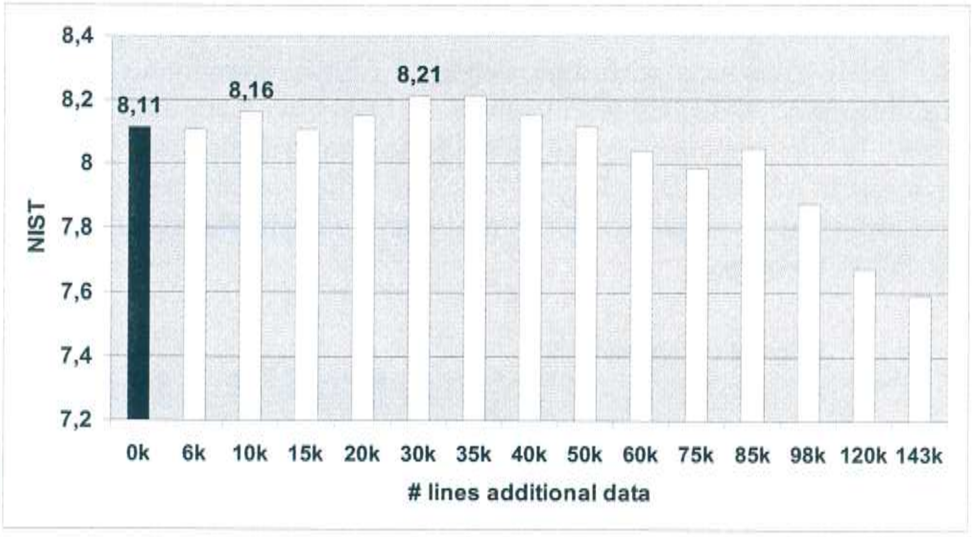


Figure 5.7: Chinese-English: different selection sizes (NIST)

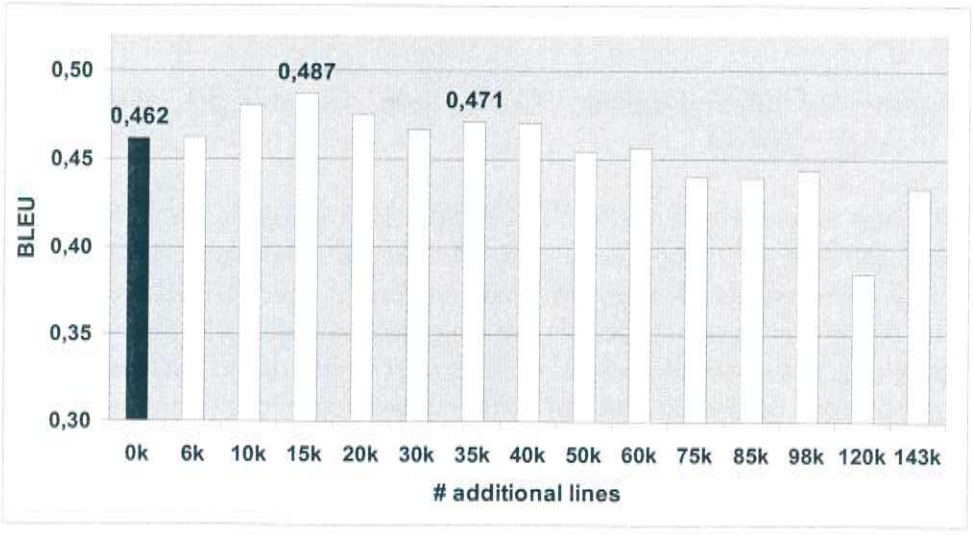


Figure 5.8: Chinese-English: different selection sizes (BLEU)

To get more robust probabilities for the words already known to be in-domain and to be able to add more out-of-domain data without choosing out-of-domain translation alternatives I trained the IBM1 lexica used by the PESA transducer with data weights 3:1 for in-domain:out-of-domain training data. As expected this especially helped with the larger selection sizes. In our system it is not possible to give fractures as weights for training data in the IBM1 lexicon training. Whole numbers as weights are accomplished by duplicating the in-domain data in the training corpus for IBM1 training.

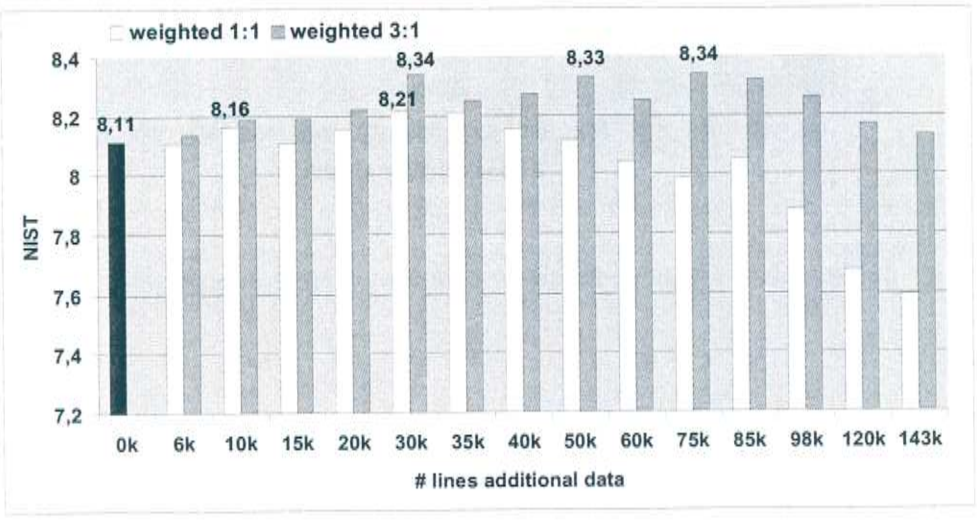


Figure 5.9: Chinese-English: Comparison 1:1 and 3:1 data weights (NIST)

The best scores were 8.3398 (NIST) and 0.4931 (BLEU). Both scores were accomplished with the changed weight for the in-domain data, the best NIST score for the Top 60 retrieved sentences, the best BLEU score for the Top 80 retrieved sentences. In diagrams 5.9 and 5.10 the number on the x-axis denotes the number of lines of training data that was added to the available in-domain data of 20,000 lines to form the training corpus.

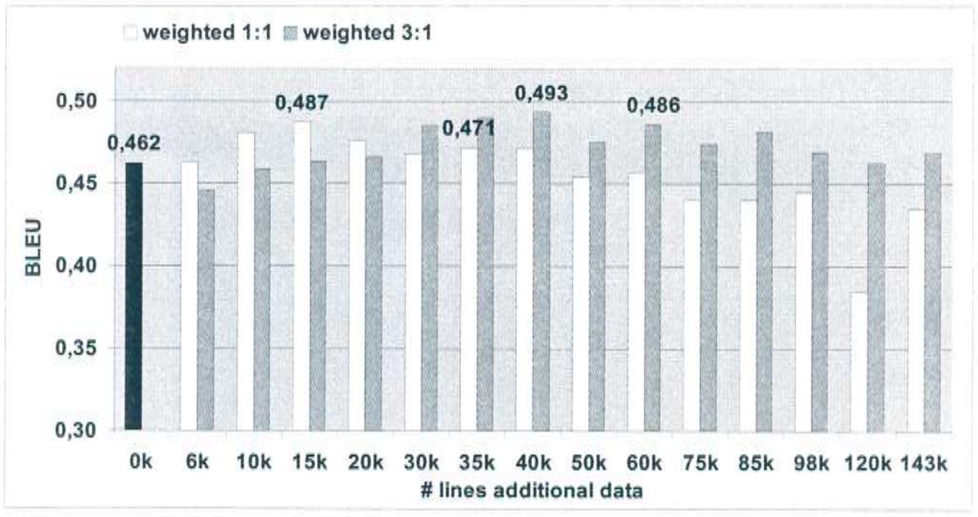


Figure 5.10: Chinese-English: Comparison 1:1 and 3:1 data weights (BLEU)

5.4.4 Perplexity driven Selection Size Determination

In this data setting I used a batch size of 20 instead of 10 for the perplexity driven selection size determination and re-ranked only the top 800 retrieved sentences in the first and the top 600 in the second perplexity re-ranking. These parameter settings seemed to be necessary because of runtime issues due to the big data collection and vocabulary size.

Diagrams 5.11 and 5.12 show the NIST and BLEU scores for selection sizes picked at the perplexity minimum before re-ranking and after re-ranking once and twice in comparison to the baselines and the best scores from the previous experiments.

In this experiment setting the automatic determination of the selection size was able to reach the same results achieved by trying various selection sizes but the re-ranking itself gave no real improvement. The reason might be, that the 3:1 weight for in- and out-of-domain data is far from optimal for these selection sizes. The weighing had a big impact on the scores, drowning out the possibly positive effect of re-ranking. Another reason might be that the double batch size of 20 is not a good choice for Chinese data, especially as here the sentences are generally much longer than in the previous experiment setting.

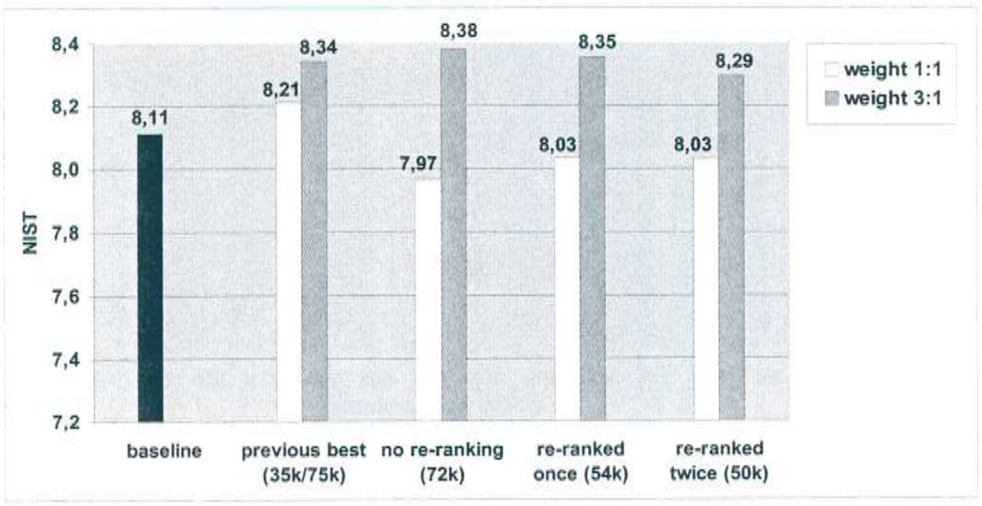


Figure 5.11: Chinese-English: Perplexity determined Selection Size and Re-ranking (NIST)

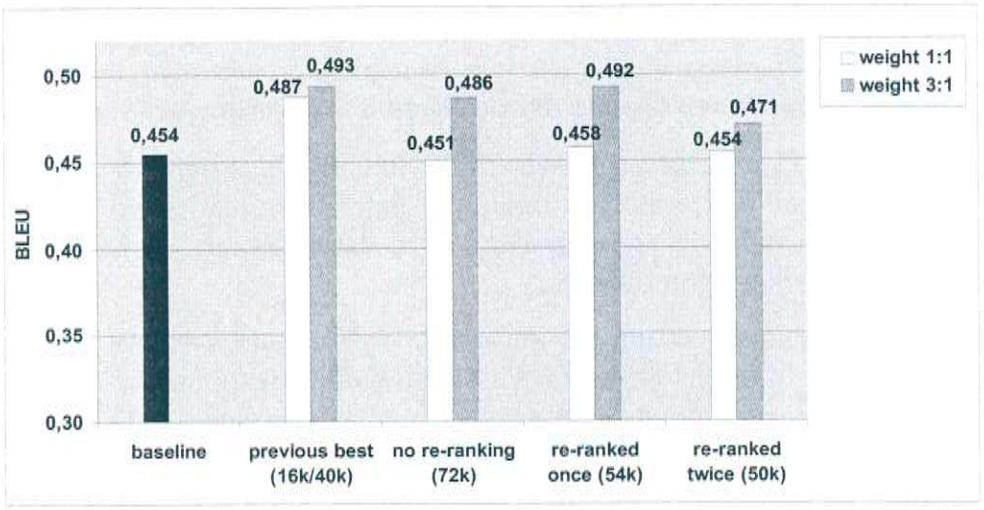


Figure 5.12: Chinese-English: Perplexity determined Selection Size and Re-ranking (BLEU)

Surprisingly the automatically determined selections are in most cases much larger than the ones found to be best in the grid search, especially for the best BLEU scores. In the Spanish – English the automatically found selections were always smaller than the manually determined ones.

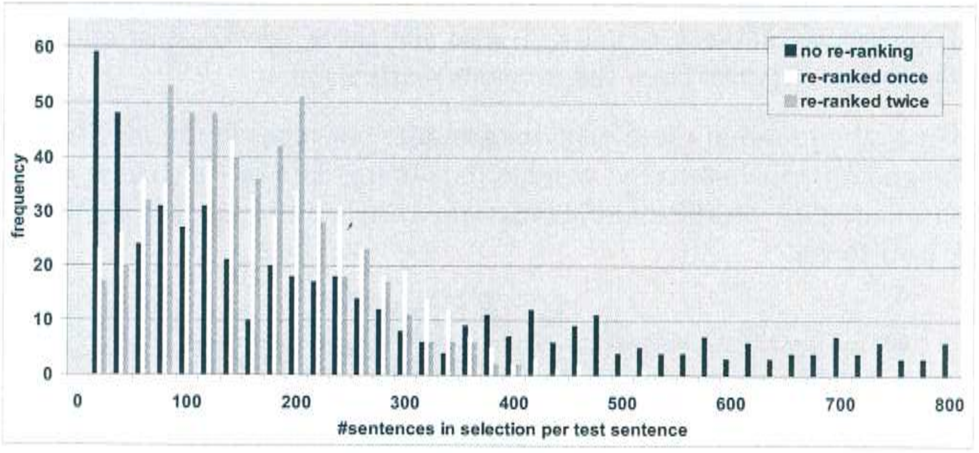


Figure 5.13: Selection sizes per sentence

Figure 5.13 shows histograms for how many sentences get selected per test sentence by the PPL selection size determination after the respective re-ranking runs. Before re-ranking the perplexity minimum is either at the first or second batch for over 100 of the 500 test sentences. Here the original TF-IDF ranking of the sentences seems to totally disagree with the perplexity quality measure. By re-ranking the most frequent selection size moves visibly up to 140 sentences and the biggest selections picked ever move from 800 down to 400.

5.4.5 In-domain/Out-of-domain data weights

As the previous experiments show, it is beneficial to weigh the in-domain data stronger than the additional out-of-domain data. To show how much improvement in translation quality could be gained by finding the optimal weight, I trained the IBM1 lexica used by the PESA algorithm with a whole set of different weights. I used the three selections of additional out-of-domain data from the previous experiment.

The system used in these experiments does not provide the possibility to assign individual fractured weights to training data or tune these weights automatically. So the experiments were restricted to whole weights from 1:1 up to 6:1.

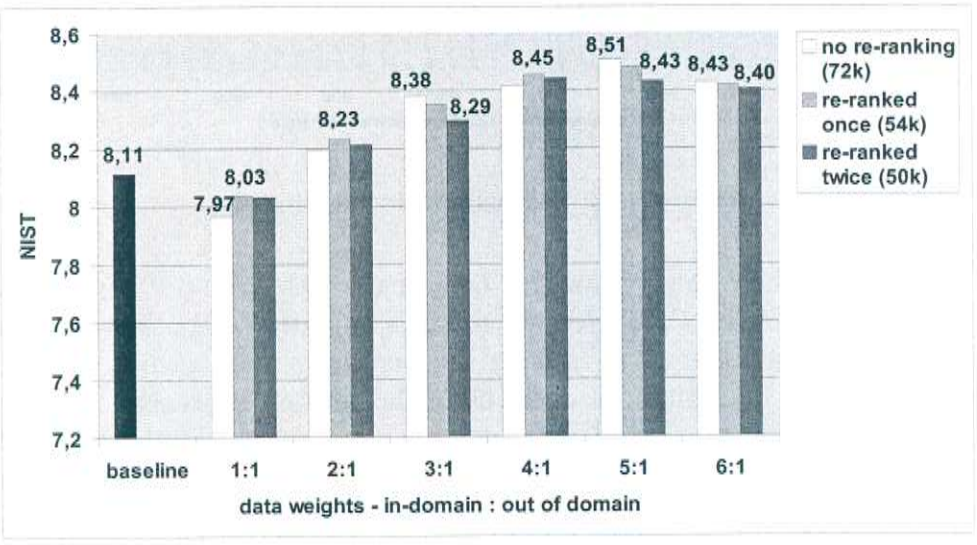


Figure 5.14: Chinese-English: Data Weights for PPL re-ranked selections (NIST)

Diagrams 5.14 and 5.14 show, that by weighing the in-domain data stronger over the out of domain data especially the NIST score can be improved further. Still the smaller selections after perplextiy re-ranking do not significantly outperform the initial selection picked without re-ranking.

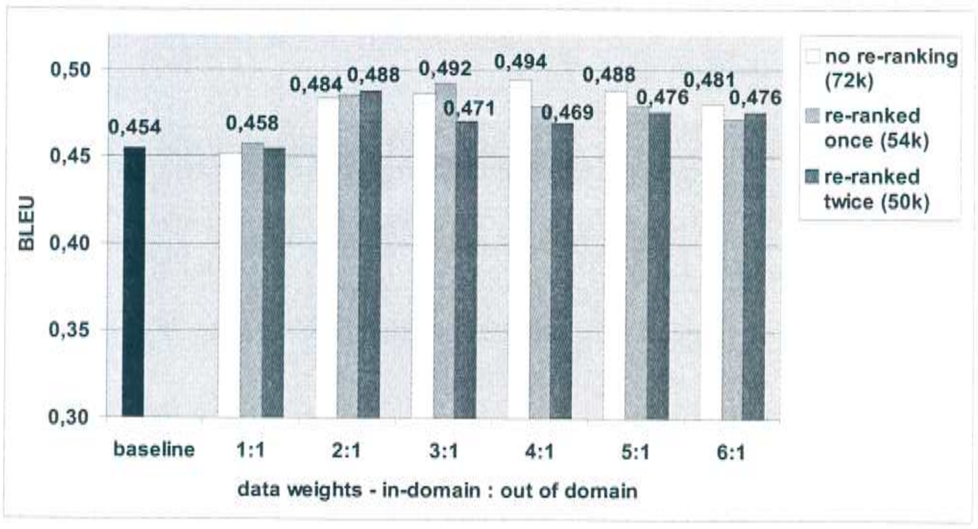


Figure 5.15: Chinese-English: Data Weights for PPL re-ranked selections (BLEU)

5.4.6 Summary

This experiment setup shows, how fast it can be harmful to just add huge amounts of training data, if the additional data comes from a different domain. The more important is it to carefully select the data to use in training. The best results were achieved with rather small selection sizes.

The perplexity driven selection size determination was able to achieve similar scores as the manual grid search. The perplexity re-ranking could not improve over that. The amount of automatically selected data here is rather big, like the bigger ones in the grid search. This is probably the reason why the scores for the 1:1 data weight are not good. Also the too big batch size of 20 is probably responsible for the poor performance of the re-ranking.

Weighing the in-domain data stronger than the out-of-domain data gives opportunity to add more data to cover more words without loosing translation quality.

System	# Added lines	BLEU	NIST
baseline: in-domain data only		0.4621	8.1129
best random	15k	0.4850	8.2262
best with weight 1:1	15k / 30k	0.4871	8.2132
best with weight 3:1	40k / 75k	0.4931	8.3398
best with PPL selection size	54k / 72k	0.4924	8.3812
optimal weights (4:1/5:1)	72k / 72k	0.4942	8.5060

Table 5.7: Results for each experiment: Chinese – English

The overall improvement of 4.8% in BLEU score, from 0.462 to 0.494 and 6.9% in NIST score, from 8.11 to 8.50 shows that translation model adaptation using IR can be succesfully applied in an in-domain / out-of-domain data scenario.

5.4.7 Example translations

As scores like NIST or BLEU provide the possibility to automatically evaluate and compare different translation systems and techniques, they can not give a human reader an idea of how good the performance of a translation system really is. The translation examples in table 5.8 show the improvements in translation quality in a human readable form. Of course not each and every sentence translation improves, the last two examples show sentences, whose translation is worse with the adapted system.

Reference Baseline Best system	no-smoking, please i 'd like a seat please i 'd like a no smoking seat please
Reference Baseline Best system	can i have a medical certificate could you give me a medical open could you give me a medical certificate
Reference Baseline Best system	three glasses of melon juice, please please give me three of those melon juice please please give me three glasses of melon juice please
Reference Baseline Best system	where can i buy accessories where can i get it where can i buy jewellery
Reference Baseline Best system	so , where is the dining room well restaurant where then where 's the dining room
Reference Baseline Best system	which would you like , tea or coffee do you have coffee would you like tea or coffee
Reference Baseline Best system	boarding for delta airlines flight six two three is delayed grand chanel tower is six two three flight be delayed boarding time delta airlines flight two three six will delay departure time
Reference Baseline Best system	excuse me. could you tell me how to get to the getty museum? excuse me could you tell me the way to the art museum yosemite san diego please excuse me could you tell me how to get to the museum
Reference Baseline Best system	please send a bellboy for my baggage . please send a bellboy to my baggage please call me carry my baggage health services
Reference Baseline Best system	i have no time no time a little more time either

Table 5.8: Example Translations: Chinese – English

6 Conclusions and Future Work

This project shows that it is possible to adapt translation models for statistical machine translation by selecting similar sentences from the available training data. There are improvements in translation performance on two different language pairs and overall different test conditions.

The results show that it is helpful to support this adaptation method by analogically adapting the language model as this further improves the translation quality.

Using language model perplexity to determine the selection size automatically renders a development test set with reference translations unnecessary. Re-ranking the retrieval result according to LM perplexity even improved translation quality slightly in one of the cases.

The combined methods of TF-IDF selection and perplexity driven selection size determination can be applied to a training corpus and a test set fully unsupervised. There is no need to run a translation system or for a development test set. Although it might be helpful to use a development test set and the translation system to determine the optimal number of re-ranking passes, a good batch size and the best weighing factor for the in-domain / out-of-domain data.

Advantages

- Uses the source language side only
  No first pass translation (with errors) needed.
- Uses the real test set
  No transfer from a development test set and no danger of overfitting.
- Works for open domain systems too
  We don't necessarily have to know which data is in-domain.

Disadvantages

- Needs the test data before training the system.
- Adaptation can not be applied on the fly to a running system.

Future Work

Different things could be done to further investigate this approach to translation model adaptation.

- Sentence based translation models
  The information retrieval and the perplexity driven selection size determination are both done for each sentence from the test set seperately. So it would be possible to train individual translation models for each sentence. This would be especially interesting, if the test set is not very homogeneous.
- Determine data weights automatically
  It might also be beneficial to use training algorithms that allow sentences to have fractional weights. Section 5.4.3 showed that tuning weights for in- and out-of domain data can give improvements. To be able to determine the best weight in each situation automatically would certainly be helpful.
- Investigate batch size and number of re-ranking passes for PPL driven selection size determination
  Up to this point I did not compare the impact of changing the batch size for the perplexity driven selection size determination. Also in some cases more than 2 re-ranking passes could be helpful.
- Improve IR
  There are some more similarity measures for information retrieval than the two I used – TF-IDF and Okapi, maybe another one would be even more suited to this problem. The ones I used focus only on unigrams. It could be helpful to develop a more sophisticated similarity measure that matches phrases, too. It was demonstrated in Zhao et al. [2004] that language model adaptation could benefit from such an advanced similarity measure and it is certainly possible to apply these ideas here. Other information retrieval techniques like stemmers or the usage of a stop-word list could be applied, too. I used only the default parameters for the IR, maybe some improvement could be achieved by tuning them.

- IR pseudo feedback
  Pseudo feedback could be applied in two different ways: In the regular way, using the first n best retrieval results to expand the query to broaden the result. As the data is available as bilingual corpus, the corresponding target language sentences from the first retrieval step on the source side could be used as queries for a retrieval on the target side.

- Further LM adaptation
  The corresponding target language sentences from the source side retrieval could be used as queries to find more matching data in additional monolingual data for training the target language LM. This has the advantage, that no first pass translation containing translation errors woud be needed.

Bibliography

Yasuhiro Akiba, Marcello Federico, Noriko Kando, Hiromi Nakaiwa, Michael Paul, and Jun'ichi Tsujii. Overview of the IWSLT04 evaluation campaign. In *Proceedings of the International Workshop on Spoken Language Translation*, pages 1–12, Kyoto, Japan, 2004.

Peter F. Brown, J. Cocke, S. Della Pietra, V. Della Pietra, F. Jelinek, J. Lafferty, R. Mercer, and P. Roossin. A statistical approach to machine translation. In *Computational Linguistics*, page 16(2), 1990.

Peter F. Brown, Stephen A. Della Pietra, Vincent J. Della Pietra, and Robert L. Mercer. The mathematics of statistical machine translation: Parameter estimation. In *Computational Linguistics*, pages 19(2):263–311, 1993.

George Doddington. Evaluation of machine translation quality using n-gram cooccurrence statistics. In *NIST Washington, DC, USA.*, 2001.

Matthias Eck, Stephan Vogel, and Alex Waibel. Language model adaptation for statistical machine translation based on information retrieval. In *Proceedings of LREC 2004*, Lisbon, Portugal, May 2004.

Emil Ettelaie, Kevin Knight, Daniel Marcu, Dragos Stefan Munteanu, Franz J. Och, Ignacio Thayer, and Quamrul Tipu. The ISI/USC MT system. In *Proc. of the International Workshop on Spoken Language Translation*, pages 59–60, Kyoto, Japan, 2004.

Almut Silja Hildebrand, Matthias Eck, Stephan Vogel, and Alex Waibel. Adaptation of the translation model for statistical machine translation based on information retrieval. In *Proceedings of EAMT 2005 (10th Annual Conference of the European Association for Machine Translation)*, Budapest, Hungary, May 2005.

Frederik Jelinek. Speech recognition by statistical methods. In *Proceedings of the IEEE*, pages Vol. 64, 532–556, April 1976.

Woosung Kim and Sanjeev Khudanpur. Language model adaptation using cross-lingual information. In *Proceedings of Eurospeech*, Geneva, Switzerland, September 2003.

Kevin Knight. A Statistical MT Tutorial Workbook. In *JHU summer workshop*, Denver, Colorado, April 1999.

LemurTk. The lemur toolkit for language modeling and information retrieval. School of Computer Science, Carnegie Mellon University. URL `http://www.cs.cmu.edu/~lemur/`.

Milind Mahajan, Dough Beeferman, and X.D. Huang. Improved topic-dependent language modeling using information retrieval techniques. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, Phoenix, Arizona, 1999.

Christopher D. Manning and Hinrich Schütze. Foundations of statistical natural language processing. pages 169–172, 2001.

Kishore Papineni, Salim Poukos, Todd Ward, and Wei-Jing Zhu. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the ACL 2002, Philadelphia, USA*, 2002.

Stephen E. Robertson and Karen Sparck-Jones. Relevance weighting of search terms. *Journal of the American Society for Information Science 27*, pages 129–146, May-June 1976.

Stephen E. Robertson and Steve Walker. Okapi/keenbow at TREC-8. In *The Eighth Text REtrieval Conference (TREC-8)*, 2000.

Stephen E. Robertson, Steve Walker, Susan Jones, Micheline Hancock-Beaulieu, and Mike Gatford. Okapi at TREC-3. In *The Third Text REtrieval Conference (TREC-3)*, 1994.

Andreas Stolcke. Srilm - an extensible language modeling toolkit. In *Proceedings International Conference for Spoken Language Processing*, Denver, Colorado, September 2002.

Toshiyuki Takezawa, Eiichiro Sumita, Fumiaki Sugaya, Hirofumi Yamamoto, and Seiichi Yamamoto. Toward a broad-coverage bilingual corpus for speech translation of travel conversation in the real world. In

LREC 2002 (Third International Conference on Language Resources and Evaluation), pages Vol.1, pp.147–152, 2002.

Stephan Vogel, Sanjika Hewavitharana, Muntsin Kolss, and Alex Waibel. The ISL statistical translation system for spoken language translation. In *Proceedings of the International Workshop on Spoken Language Translation*, pages 65–72, Kyoto, Japan, 2004.

Stephan Vogel, Hermann Ney, and Christoph Tillmann. HMM-based word alignment in statistical translation. Lehrstuhl für Informatik VI, RWTH Aachen, 1996.

Stephan Vogel, Ying Zhang, Fei Huang, Alicia Tribble, Ashish Venugopal, Bing Zhao, and Alex Waibel. The CMU statistical translation system. In *Proceedings of MT Summit IX, New Orleans, LA*, September 2003.

Hua Wu and Haifeng Wang. Improving domain-specific word alignment for computer assisted translation. In *Proceedings of ACL 2004*, Barcelona, Spain, July 2003.

Chengxiang Zhai. Notes on the Lemur TF-IDF model. Technical report, School of Computer Science, Carnegie Mellon University, 2001.

Ying Zhang, Stephan Vogel, and Alex Waibel. Integrated phrase segmentation and alignment algorithm for statistical machine translation. School of Computer Science, Carnegie Mellon University, 2003.

Bing Zhao, Matthias Eck, and Stephan Vogel. Language model adaptation for statistical machine translation via structured query models. In *Proceedings of COLING 2004*, Geneva, Switzerland, August 2004.