Ruprecht-Karls-Universität Heidelberg
Bachelor-Arbeit
Betreut durch: Dr. Anke Holler

Statistical Machine Translation Between New Language Pairs Using Multiple Intermediaries

Armin Schmidt
Ringstraße 3
69115 Heidelberg
armin.sch@gmail.com
Matrikel-Nr. 2500590
20.03.2007

Acknowledgements

I want to thank Dr. Andreas Eisele who first introduced me to the topic of statistical machine translation and who supported me throughout with excellent ideas, useful advice, and a very pleasant sense of humour.

Further, I would like to thank Jan Strunk who provided me with his sentence boundary detector *Punkt* and who invested quite some time in order to get the program working with my data.

Last but not least, a huge thank you to Katja Filippova and Torsten Marek for proofreading the script late at night and for being very honest.

Contents

1 Introduction

Enabling machines to automatically translate between natural languages is the vision that first brought to life the field of natural language processing (NLP) and computational linguistics (CL). From its very beginning in the mid-twentieth century onwards, machine translation (MT) as a research area has been under constant investigation and functioned both as objective and trigger that initiated many of the findings in linguistics and NLP. Coinciding with the dawn of the Internet as a mass medium, more and more large text collections have become available in recent years. As a result, statistical approaches to MT have become popular within the research community as well as in industrial environments. Statistical machine translation (SMT) systems proved to be able to not only compare to but even outperform rule-based ones at several evaluation campaigns, e.g. the NAACL/HLT[1] Workshop on Machine Translation (Koehn and Monz (2006)).

Although more and more MT architectures are being developed and despite constant research activity within the field, only few language pairs are currently covered. In order to address new translation directions, Eisele (2006) proposes an approach which aims at the translation between previously unconnected languages[2] by taking the detour over multiple intermediate languages. In this paper, we will give an overview of the current situation and analyse their propositions by setting up an SMT system using existing software and corpora. We will also perform experiments in order to test some of Eisele's hypotheses.

In Section 2, we will give a brief introduction to SMT. The advantages and disadvantages of rule-based and statistical approaches will shortly be discussed in 2.1. We will then outline the underlying mathematical theory in Section 2.2 and introduce most of the software we will use in our experiments.

The motivation for our approach is explained in Section 3, which mainly consists of a review of Eisele (2006). First, some rather general observations will be described about which language pairs are covered by current MT systems. As it will be shown, there is a core set of only ten languages for which MT systems exist that translate between all of them. Most other, though still very few, languages, for which MT engines exists, are covered by only one such system. Secondly, we will apply our conclusions from Section 3.1 to SMT and the availability of parallel corpora in particular and describe some ideas to deal with this situation.

In comparison to the usage of only one language, multiple intermediate languages can improve overall translation quality. There are several incitements that combine linguistic and statistical reasons to back up this statement. Section 3.3 will analyse some of them and propose ideas for corresponding algorithms.

[1] North American Chapter of the Association for Computational Linguistics/ Human Language Technology

[2] Two languages are unconnected if 1. no MT system exists that translates between them, and 2. no parallel corpora exist on which an SMT system could be trained.

In Section 4.1, we will describe the document collections we utilise for our experiments. We will also explain why we did not use our initially planned setup.

Our experiments are discussed in Section 4. The preprocessing of the corpora is described in subsection 4.2 where we will briefly dwell upon the subjects of sentence boundary detection (SBD) and sentence alignment. SBD in particular may have some peculiarities when used with large corpora, especially when these are from specific text domains in different languages, as it is the case in our setting. In particular, common lists of abbreviations will not suffice in order to split sentences correctly. This problem can be addressed by using algorithms that learn abbreviations directly from the data. In Section 4.3, we will introduce our method of performing the translation between French and German via English and Spanish. We evaluate our experiments with the commonly used BLEU metric (Papineni et al. (2002)). Section 4.4 gives an introduction to BLEU, explains how the metric was applied to our setup and interprets the results.

Considering the extremely short amount of time available, it does not come by surprise that we were only able to touch the very surface of what could and should be done regarding the topic. In particular, future work will have to make use of larger training and test sets and must use refined techniques which do not operate on sentence level but on phrase level in order to get as much out of multiple intermediaries as possible, concerning both translation quality and speed. Some such ideas will be discussed in Section 5 where we will also review some errors that have occurred during the experiments and offer suggestions how they could be avoided in the future.

2 Statistical Machine Translation

2.1 Statistical vs. Rule-Based Approaches

Machine translation presupposes that text is processed on almost all linguistic levels. For example, lexical categories (parts of speech, POS) and named entities have to be disambiguated. Syntactic constituents and their functions have to be recognised and annotated. Furthermore, a machine translation system might have to deal with word meaning and anaphora and coreference resolution. With solutions to these tasks implemented, one is still left with the problem of deficient coverage of application-specific terminology and a shortage of existing dictionaries.

Earlier approaches to machine translation were of the rule-based type and involved manually compiled dictionaries and grammars. The disadvantages of such rule-based systems were soon to become clear: they were usually very expensive to build and maintain, and rules tend to cross-influence one another in non-trivial ways which are hard to track and whose repairment may in fact be impossible. In order to implement an only very simple working basic system, already many years of extensive theoretical occupation and manual implementation will be necessary in most cases. Once a rule-based machine

translation system works, it is very difficult to adapt to other domains or languages.

Statistical machine translation (SMT) tries to solve many of these problems by applying statistical learning techniques over large amounts of bilingual data (i.e.parallel corpora - text collections which are available in two or more different languages where one one side is the translation of the other). Using purely quantitative methods, SMT algorithms deal with linguistic knowledge only intrinsically through alignment and reordering probabilities rather than extrinsically formulated rules. Provided with sufficient amounts of training data, statistical approaches to machine translation and other fields of natural language processing are already very successful in dealing with many of the aforementioned problems. One major factor for this development is the growing availability of large monolingual, and increasingly also bilingual, text corpora in recent years. In particular, the advance of the Internet and the globally increasing internationality within social, economic, and political life has produced many new resources for large text collections. The advantages of SMT compared to rule-based approaches lie in their adaptability to different domains and languages: once a functional system exists, all that has to be done in order to make it work with other language pairs or text domains is to train it on new data.

2.2 Theoretical Background

SMT is based on the idea that statistical models for translating between two languages can be learned from large parallel corpora of translated text. In the following, we will introduce some of the most basic concepts and techniques[3]. We will start by giving a brief overview of the general approach. Thereafter, we will outline the design of a full SMT system.

SMT makes a few assumptions that may not seem very intuitive at the beginning. First, we assume that, when translating a German string f to an English string e[4], the German speaker is actually speaking English all along but what they say got somehow distorted on the way, thus resulting in a German string of words. The task we are faced with is to figure out what the original English string was. In other words, we want to find the string e' that maximises the probability $P(e|f)$. This view has the consequence that in theory, any English string may in fact be regarded a translation of f, assigned with a certain probability. Furthermore, we have reversed the translation direction to English being the source, and German being the target language. While this may seem intuitively strange, it does not constitute a problem for our

[3] This chapter is a very elementary repetition of some of the most basic concepts used in SMT. It does not claim to be complete. For an in-depth introduction refer to Koehn (2007)

[4] There is the tradition in SMT of always referring to the source language as *French* or *foreign* and to the target language as *English* to which we will conform.

theory. Bayes' theorem tells us that

$$P(e|f) = \frac{P(e)P(f|e)}{P(f)}$$

We are interested in the English sentence for which $P(e|f)$ is greatest. We therefore write

$$e' = \arg\max_e P(e)P(f|e)$$

The denominator $P(f)$ may safely be omitted since e' is independent of it.

This model proves practical as $P(e)$ can be estimated from monolingual English text, whereas $P(f|e)$ can be estimated from word- or phrase-aligned bilingual data. To make this statement clear we paraphrase the above formula: when translating a German string f to an English string e, we want to know two things:

1. Is our hypothesis e a grammatical sentence of the target language? To answer this, we compare e with a model of the English language, typically an n-gram model which was learned from a large English text collection.

2. Is e really a translation of f? That is to say, we want to make sure that the meaning of f was retained during the translation process. This can by done by looking at how words and phrases of source and target language generally translate into one another. For this, we extract them from bilingual text corpora and align them into a translation model.

The alignment of words or phrases turns out to be the most difficult problem SMT faces. Words and phrases in the source and target languages normally differ in where they are placed in a sentence. Words that appear on one language side may be dropped on the other. Concepts may be expressed by means of different syntactical categories. One English word may have as its counterpart a longer German phrase and vice versa. Figure 1 shows an example.

One of the ground-breaking papers which first described the aforementioned techniques to MT in the early 1990s was Brown et al. (1993). While they used a purely word-based approach, the currently best-performing SMT systems are of the phrase-based type (Koehn et al. (2003)), i.e. they use phrases[5] instead of words as the smallest translation unit. Without going into further detail, we will outline the components of a typical phrase-based SMT system in Figure 2.

The decoder is the component in such a setup that tries to find the best translation hypothesis for an input sentence using a phrase table as well as a language model. The output is not actually a single sentence but an n-best list of sentences, each of them assigned a certain probability.

[5]The term *phrase* is used without any syntactic motivation and refers to any multi-word unit.

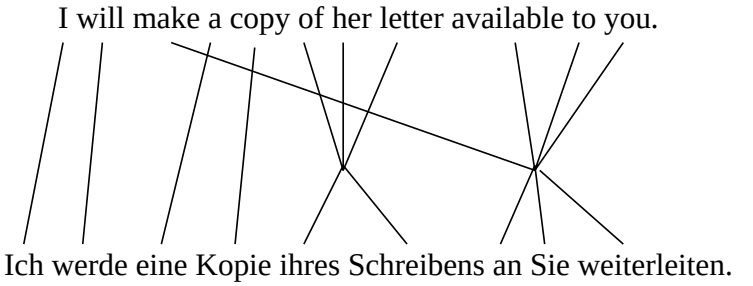

Figure 1: Sample alignment of a German sentence and its English translation

2.3 Software

There are a number of implementations of subtasks and algorithms in SMT and even software that can be used to set up a fully-featured state-of-the-art SMT system. Most of these projects are open source and licenced under the GNU (Lesser) General Public Licence. Although some of them are quite advanced in both functionality and usability, a lot of work always has to be put into the setup of such a system which involves many different kinds of software. Some programs are still under development and have their flaws concerning compatibility with certain computer architectures or compilers. After all components are installed and tested for functionality, the data has to be prepared in order to meet the requirements as input for the training process. The latter typically takes a very long time to run, which increases with the amount of data and the complexity of a system. In this section, we will give a brief overview of some of the most widely utilised ones, which we have also used in our experiments. The scheme in Figure 3 puts each program in relation to the overall translation process.[6]

2.3.1 Moses-Decoder

Moses (Hoang et al. (2006)) is a full-featured, open source SMT system development at the University of Edinburgh. The software includes a phrase-based decoder and supports factored translation models used for integrating linguistic knowledge like syntactic or morphological information into the translation process. The latter is done by accepting input of the form $factor1|factor2|factor3$, where each factor may constitute a different feature of the input, e.g. surface form, lexical category, and word stem.

The Moses project also provides a separate set of scripts which are independent of the decoder itself and which can be used for various pre- and post-processing tasks like tokenization and lowercasing of the training- and test data. These have not been used in our experiments and will mostly not be discussed here. Two exceptions are the script 'train-factored-phrase-model.pl' which is described separately in Section 2.3.4, and the script 'filter-model-given-input.pl'. The latter addresses the large size of the phrase

[6]Note: tools for the preprocessing and evaluation tasks are described separately in Sections 4.2 res. 4.4 below.

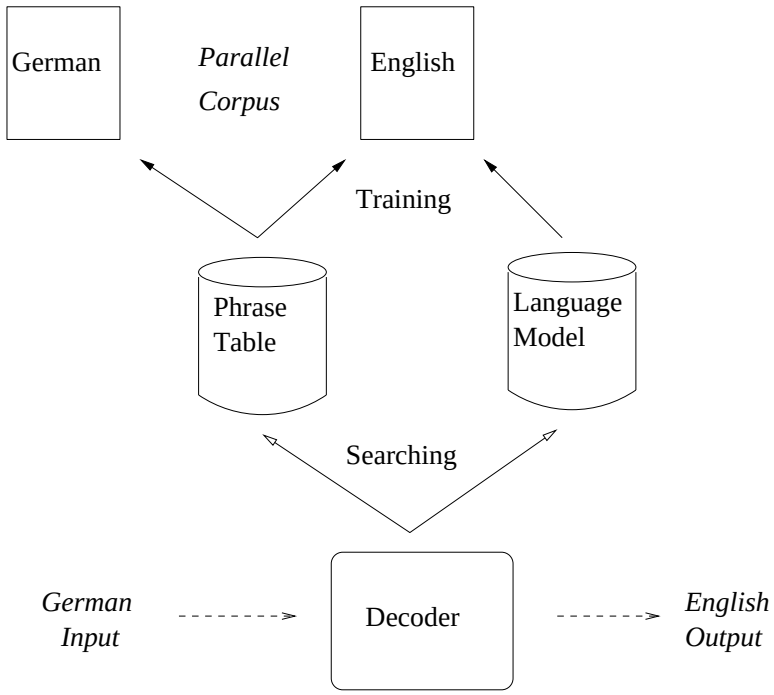

Figure 2: Schematic overview of an SMT system and its components

table as this often exceeds the amount of main memory available. This issue is resolved by extracting only those phrases from the model that actually appear in the input. The filtered phrase tables for our data were usually about 14-15% the size of the original model and fit well into memory.

2.3.2 GIZA++

GIZA++ (Och and Ney (2003)) is a software for learning word-by-word alignments between corresponding bisentences[7] and was developed by Franz Joseph Och and Hermann Ney as an enhancement of the GIZA tool written at the 1999 Summer workshop hosted by the Center for Language and Speech Processing (CLSP) at Johns Hopkins University. GIZA++ implements partly refined versions of all five IBM models (Brown et al. (1993)). GIZA++ is required in order to use the training scripts provided by the Moses project (Section 2.3.1).

2.3.3 SRI Language Modelling Toolkit

The SRI Language Modelling Toolkit (SRILM) has been first developed by Andreas Stolcke for building and applying statistical language models (LMs). It has received some advancements during the CLSP

[7] Two corresponding sentences in different languages.

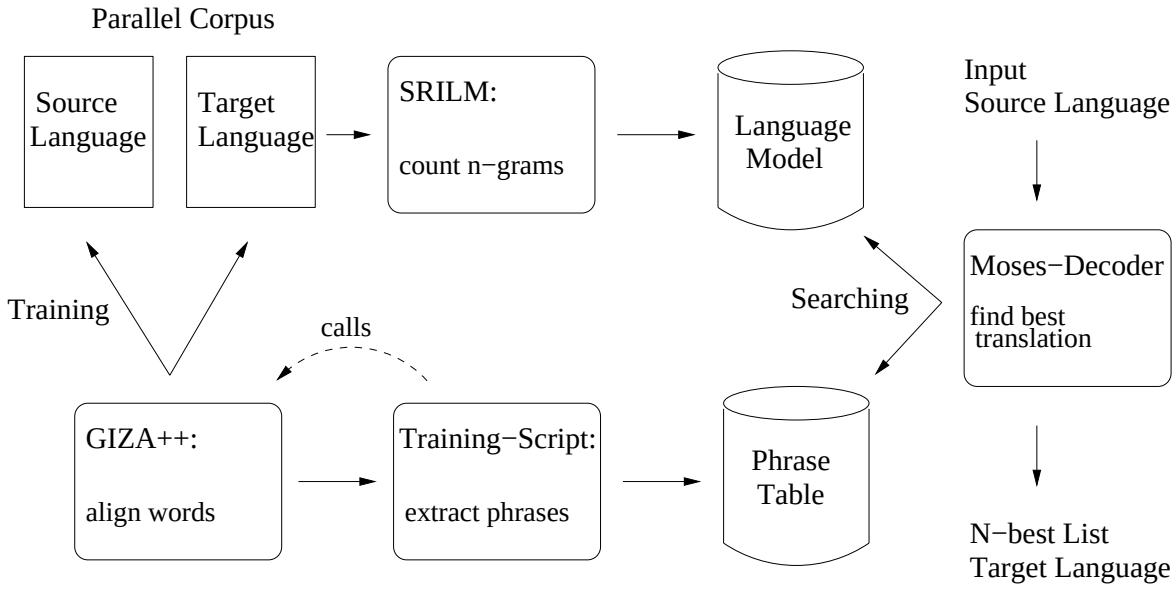

Figure 3: Software in SMT

Summer Workshops between 1995 and 2002 at John Hopkins University. Currently, the SRILM package includes a set of C++ libraries, executable programs as well as miscellaneous scripts, all aiming at tasks related to training LMs and their usage. The capabilities and design of the software are described in Stolcke (2002). SRILM is recommended for use with Moses (section 2.3.1) as the latter depends on some of its class libraries for compilation. Moses provides its own components for language modelling which we have not used so far.

2.3.4 Train-factored-phrase-model.pl

The training process consists of nine steps, all of which are executed by the script 'train-factored-phrase-model.pl'. Each of them will be described briefly below.

1. Prepare Data: The input for word alignment with GIZA++ needs to be in a particular format. A vocabulary file has to be generated containing words, word identification numbers, and frequency information. Aligned sentences have to be written into a file in which the words of each bisentence have been exchanged with their corresponding identification numbers. GIZA++ also requires that all words be clustered into word classes. This is done with the external program 'mkcls' which comes with the GIZA++ package but needs to be compiled separately.

2. Run GIZA++: Although GIZA++ implements all five IBM models, only the word alignment is of interest for training phrase-based models (cf. Section 2.2). This step is very critical in terms of

time and memory requirements and typically takes 16 to 20 hours to run for a corpus consisting of between 300,000 and 400,000 sentence pairs. Corpora should normally be split up into several smaller units in order to reduce memory overhead. Another issue is that GIZA++, following Brown et al. (1993), always aligns only single words from the source language to multiple words on the target side but not the other way around which is, naturally, counterintuitive and unwanted. In order to address this, GIZA++ is run in both directions. Starting from the intersection of the bidirectional runs, the final word alignment is computed by means of various heuristics which may, for example, add alignments that lie in the union of both runs. For the case that a machine with multiple processors is available, the training script offers an option which performs both runs in parallel to speed up the process.

3. **Word Alignment:** The output from step 2 does not yet consist of a word-alignment file which can easily be processed. As described above, we first need to get the intersection of the bidirectional GIZA++ runs and put in a proper format.

4. **Lexical Translation:** From the word alignment the maximum likelihood estimates for each lexical translation in both directions, i.e. $P(f|e)$ and $P(e|f)$ are calculated.

5. **Phrase Extraction:** All phrases are extracted and written into a file together with their alignment points, i.e. information about which two words are aligned.

6. **Score Phrases:** The phrase translation probabilities and several other phrase translation scores such as lexical weighting and phrase penalty, which we will not discuss here, are computed.

7. **Reordering Model:** One or several reordering models may be computed. Available are a purely distance-based model as well as models which take orientation into account and ordering based either one or both of the language sides with respect to the previous and/or next phrase ordering.

8. **Generation Model:** This step is not discussed here.

9. **Configuration:** Lastly, the Moses configuration file is created, defining paths to the respective models as well as standard parameter weights.

3 Motivation

3.1 Insufficient Language Coverage in MT

As described above, automatic translation has been one of the core applications of computational linguistics from its very beginning. Nevertheless, it may not come as a surprise that only very few languages are in fact covered by current MT systems. Hutchins (2005) shows that most existing translation directions

evolve around a small number of core languages, with English being the most frequently utilised one. Figure 4, taken from their paper, gives an overview.

	English	German	French	Japanese	Spanish	Italian	Russian	Portuguese	Ukrainian	Polish	Korean	Czech	Chinese	Dutch	Swedish	Hungarian	Greek	Croatian	Catalan	Arabic		
English	*	47	42	44	45	31	19	31	6	9	16	1	23	8	1	4	3	1	2	15	386	36
German	48	*	25	3	8	9	13	3	2	4	1	1	-	1	1	2	-	1	-	-	127	20
French	41	24	*	3	10	11	5	6	1	2	1	1	1	3	-	-	3	-	-	2	114	15
Japanese	42	3	-	*	3	3	1	3	1	1	16	-	12	-	-	-	-	-	-	-	88	11
Spanish	42	7	10	3	*	8	4	7	1	1	1	1	-	-	-	-	-	-	3	-	88	12
Italian	29	9	11	3	8	*	4	3	1	1	1	1	-	-	-	-	-	-	-	-	71	11
Russian	23	13	5	1	4	2	*	1	2	1	-	1	-	-	-	-	-	-	-	-	53	10
Portuguese	29	4	5	4	7	3	1	*	1	1	2	-	1	-	-	-	-	-	-	-	58	11
Ukrainian	6	2	1	1	1	1	2	1	*	1	-	-	-	-	-	-	-	-	-	-	16	9
Polish	8	3	2	1	1	1	1	2	1	*	-	-	-	-	-	-	-	-	-	-	20	9
Korean	15	-	-	17	-	-	-	-	-	-	*	-	1	-	-	-	-	-	-	-	33	3
Czech	1	1	1	-	-	1	1	-	-	-	-	*	-	-	-	-	-	-	-	-	5	5
Chinese	21	-	-	12	-	-	-	-	-	-	1	-	*	-	-	-	-	-	-	-	34	3
Dutch	8	1	3	-	-	-	-	-	-	-	-	-	-	*	-	-	-	-	-	-	12	3
Swedish	2	1	-	-	-	-	-	-	-	-	-	-	-	-	*	-	-	-	-	-	3	2
Hungarian	2	2	-	-	-	-	-	-	-	-	-	-	-	-	-	*	-	-	-	-	4	2
Greek	2	-	3	-	-	-	-	-	-	-	-	-	-	-	-	-	*	-	-	-	5	2
Croatian	1	1	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	*	-	-	2	2
Catalan	2	-	-	-	3	-	-	-	-	-	-	-	-	-	-	-	-	-	*	-	5	2
Arabic	14	-	2	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	*	16	2
Latin	1	1	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	2	2
Finnish	2	1	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	3	2
Esperanto	1	-	-	-	1	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	2	2
	363	122	113	93	90	70	51	57	16	21	39	6	39	12	2	6	6	2	5	17		
	33	18	13	12	10	10	10	9	9	9	8	6	6	3	2	2	2	2	2	2		

Figure 4: Number of commercial MT systems per language pair, according to Hutchins (2005). Figure taken from Eisele (2006)

As Eisele (2006) points out, it strikes that 10 languages, which we will henceforth refer to as the ”core languages”, are almost completely interconnected while all others are associated with only few other languages. It also stands out that all languages are connected with English in at least one direction. In order to translate between previously unconnected language pairs, it may appear a logical consequence to try to take the detour over English as an intermediate language. But Eisele (2006) already calls attention to the fact that current MT systems still obtain rather poor overall results, and hence, it can be expected that after such indirections the final translation will most probably be of very bad quality. Therefore, he suggests to make use of several instead of only one intermediate language. In the most simple case, the one hypothesis with the best score from all output sentences could be used.

3.2 Shortage of Parallel Text

Concerning statistical machine translation, the observations described above appear in a different light specific to the availability of parallel corpora in that shortage of parallel text is the problem SMT most frequently faces. Several corpora have been collected and made available by the research community. Among them are mostly document collections from the domain of political discourse as this field very often takes place in a multilingual environment and aims at an international audience. Brown et al. (1993) used the Hansard Corpus containing the proceedings of the Canadian Parliament in both French and English. The Europarl corpus (Koehn (2005)) consists of parallel text in altogether 11 languages, thus providing 55 possible language pairs and 110 different translation directions. The Europarl corpus contains the proceedings of the European Parliament, which are transcribed for Danish, Dutch, English, French, Finnish, German, Greek, Italian, Portuguese, Spanish, and Swedish. Furthermore, Eisele (2006) describes the document collection of the United Nations Organisation (UNO) which is distributed in 6 languages throughout: Arabic, Chinese, English, French, Russian, and Spanish. A smaller subset of this document collection is also available in other languages like German, which are not official languages of the UNO. It is interesting to note that 4 of the languages of the UNO corpus belong to the core set of languages as can be seen in Table 4. 3 of them are also part of Europarl. We would like to think that this observation could be a pivotal starting point in order to bridge between previously unconnected languages.

3.3 Translating via Multiple Intermediate Languages

There are several reasons why using multiple intermediate languages can improve the quality of translations between new, not directly connected languages. In this section, we will motivate some of them on a theoretical basis and illustrate them with a few examples. In our considerations, we will refer to statistical machine translation although most of our hypothesis may apply to rule-based machine translation as well.

First, see the schematic overview of the translation process with multiple intermediaries in Figure 5. Henceforth, we will refer to this scheme as well as its abstract language denominations in order to make our explanations more conceivable.

Multiple intermediaries can improve coverage of a machine translation system by combining the resources of the individual translation directions. In its most straight-forward fashion, this is especially true when the translation models to and from the individual intermediate languages are trained on different corpora, ideally covering more or less different domains. In case a word or phrase cannot be found while translating between SL and IL1, we can try to find it in translation directions SL-IL2 or SL-IL3. The same is true for translating from IL[123] to TL. But while this is a reason for using multiple corpora for whichever language direction, this is not a reason specific to translation via multiple intermediaries.

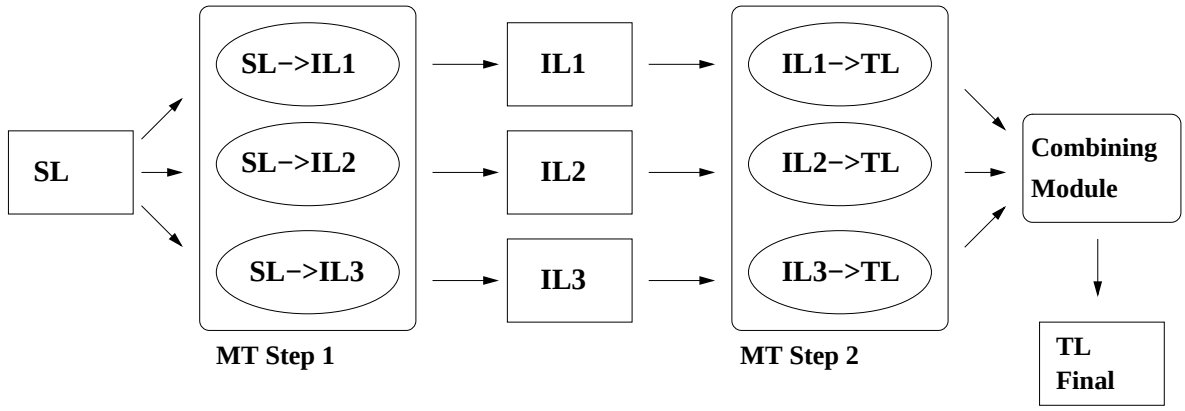

Figure 5: Translating with multiple intermediaries - schematic overview (SL = source language, IL = intermediate language, TL = target language)

Let us walk through a slightly more subtle example which has nothing to do with the domains of the training corpora used. Assume that we have trained our translation models on four different text sets: SL-IL1, IL1-TL, SL-IL2, and IL2-TL. This makes four translation models which may or may not belong to the same text collection or domain. Now assume that there is a word w_{SL} in SL for which several synonymous translations exist in IL1. We call these w'_{IL1}, w''_{IL1}, and w'''_{IL1}. Now, translating w_{SL}, step 1 may produce, say, w''_{IL1} as shown in Figure 6.

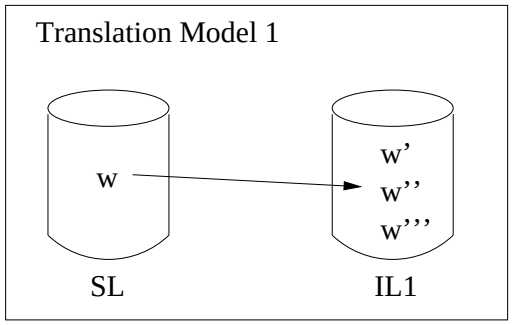

Figure 6: Step 1: translating w_{SL} to w''_{IL1}.

But w''_{IL1} may not occur in the model used in step 2, i.e. IL1-TL, because there, one or both of the other synonyms, i.e. w'_{IL1} and w''_{IL1}, were used throughout. Thus, the translation of w''_{IL1} will fail in step 2 as shown in Figure 7.

If multiple intermediaries are available, an alternative path via IL2 could be taken in such a case. This can be advantageous for two reasons: IL2 may have only one translation w_{IL2} for w_{SL}, or at least fewer

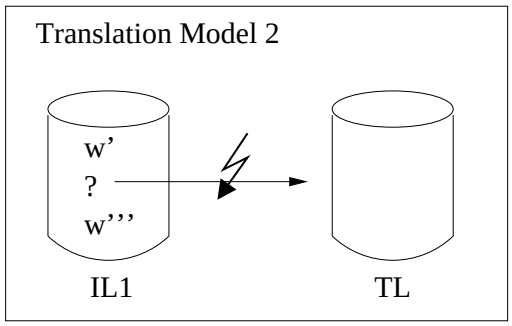

Figure 7: Step 2: translating w''_{IL1} fails because it is not contained in model 2.

synonyms than does IL1. But even if there are just as many synonyms for w_{SL} in IL2 as there are in IL1, there still is the chance that step 1 produces a word, say, w''_{IL2} which is contained in the translation model for step 2 as well.

There is yet another reason that speaks for using multiple intermediaries. No two languages use the same set of concepts[8] relating to one particular word or phrase. Every human translator will be able to tell a story about how certain terms of one language simply do not exist in another. Typical examples include the German *Schadenfreude* or the French *savoir-vivre*, both of which do not have equivalents in English (or, for that matter, French res. German). An interpretor has two choices in such cases: they will either have to paraphrase an expression or use a less specific term and accept a certain loss of information. Also, it happens quite often that such terms are adopted in other languages, just as you find *savoir-vivre* in German dictionaries and *Schadenfreude* in English ones.

An SMT system has to face the exact same problem and for translating through intermediate languages this turns out to be somewhat specific. Assume that a particular word w_{SL} in SL corresponds to a concept c which exists in TL and has a precisely equivalent word there. If IL1 has a term w_{IL1} which fits c well, it will be used frequently and receive a high score. But IL1 may not know of c or at least does not have a particular lexical unit able to express c properly. The translation step from SL to IL1 will therefore have to use a less specific circumscription of c which will yield a low score. In the second translation step from IL1 to TL, the system again has several choices, all of which will result in a loss of specificity:

1. A less specific term is used in step 1. It can be expected that a term just as unspecific will be chosen in step 2. This is because less specific terms usually occur more often than more specific ones, therefore yielding higher overall probabilities.

[8] We define *concept* as a semantic unit that has a singular core meaning and is associated with one particular word or lexicalised phrase.

2. A paraphrase is used in step 1 which will result in a rather literate translation in step 2, especially since a paraphrasal circumscription of a term will be different, depending on the situation in which they are used and also on external factors like the interpretor who translated a particular document in a text collection. For these reasons, the score of the current "correct" translation option will be low.

3. A translator switches to a term whose concept is similar but not quite the same and whose specificity is about similar to the one of the original term. In different situations, different ones of such translations will be used, lowering the scores of all the individual possibilities and making way for rather unspecific options.

In all the aforementioned cases, the correct counterpart in TL for w_{SL} res. c will not be found. Using multiple intermediate languages, such cases can be handled in the following way: When a term of SL cannot be translated into IL1 with the appropriate specificity, IL2 is used instead. If a more appropriate translation option exists in IL2, this alternative path is taken, otherwise IL3 will be used. If none of the alternative paths provides better options, the one with the highest score is used.

3.3.1 Algorithmic Alternatives

One question that remains is how the path to be taken should actually be chosen, i.e. how do we know when to translate via IL1, IL2, or IL3? There obviously are several possibilities. The most simple one is to operate on sentence level only. We could, for example, first translate via all paths separately, i.e. SL-IL1-TL, SL-IL2-TL, and SL-IL3-TL, and then pick only those sentences from all the generated hypotheses in TL that are best according to some score. Because of its simplicity, this is the approach we have chosen for our experiments. This suggests that the adequacy of a translation option for a particular word or phrase is directly reflected in the score of the respective sentence. The truthfulness of this assumption lies within the way such a sentence score is calculated, which is utterly up to the implementation of the decoder.

Much better it would be to operate on smaller units, i.e. words or phrases. It would be most advantageous if such units would even be linguistically motivated - after all, we were talking about concepts above. Such concepts are semantic units which are assigned syntactic constituents. But, as we have already mentioned (cf. Section 2.2), current SMT systems have a different notion of the term *phrase*, and statistically motivated phrases do not necessarily correspond to syntactic or semantic constituents. If this really is a problem remains to be subject for future research. The good news is that SMT always assigns scores to what it considers a phrase and these scores we can work with in order to decide on which phrase to take when translating. For example, we could define a threshold so that, if a phrase in TL, while translating IL1 to TL, gets a scores below it, we would discard it and try to find a better translation via IL2.

4 Experiments

4.1 Data

4.1.1 Bilingual Corpora

At our disposal we had the Europarl corpus as well as the UNO document collection (for both cf. Section 3.2). Our initial plan was to extend and build upon the ideas and observations described in Section 3.2, aimed at the translation between the so far largely unconnected languages Russian and German using English and Spanish as intermediates. For this, we wanted to prepare the UNO document collection and use it in order to train translation models for the language directions RU-EN and RU-ES. The cleaned and well-prepared Europarl corpus was used in order to train models for the translation directions EN-DE and ES-DE. Figure 8 gives an overview of the initially intended setup.

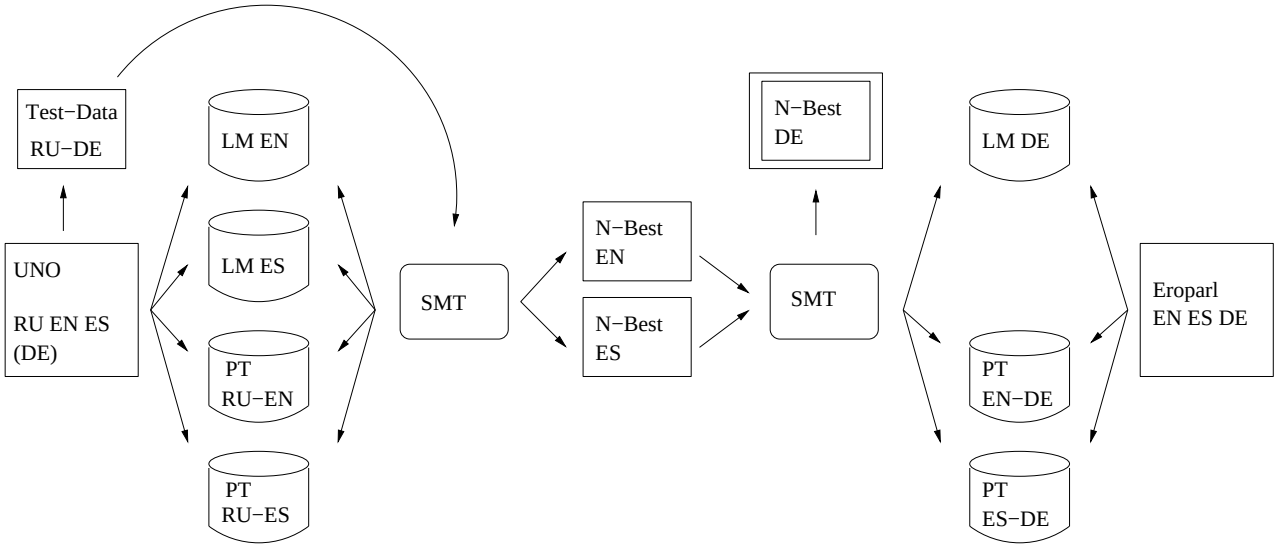

Figure 8: Intended setup. PT = 'Phrase Table', LM = 'Language Model'

The UNO corpus also contains a small set of German documents which can in principle be matched with its Russian counterpart, thus providing a test set for the eventual translation direction RU-DE. Unfortunately, this setting had to be given up after it turned out that the UNO corpus was yet very noisy and contained too many passages which could not be properly aligned. Due to the lack of time it would have taken to deal with the issue of cleaning the data, we had to decide in favour of a different setup. As an alternative we chose French as the source language. As French is covered by the Europarl corpus, this has the advantage that the preprocessing is equally simple for all language blocks. Further, the setup (cf. Figure 9) provides a large test set as FR-DE can be directly aligned. The disadvantage of the setting lies

merely in the fact that it is somewhat artificial with regard to our goal, which is translating between new, i.e. unconnected, language pairs.

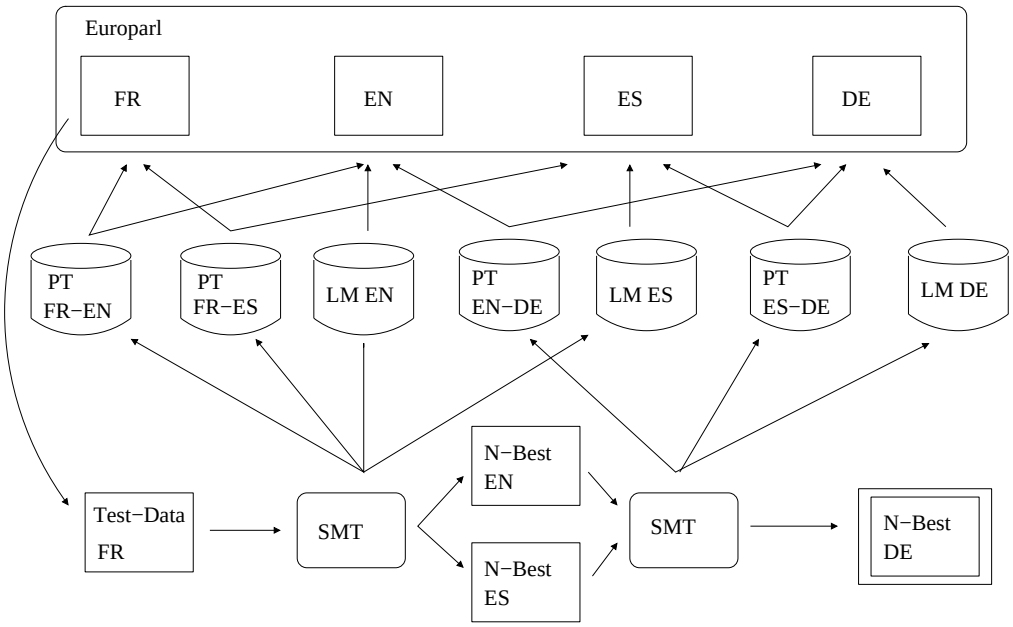

Figure 9: Final setup.

4.1.2 Monolingual Language Models

As for the monolingual language models, we took ready-to-use LMs that were provided by Philipp Koehn for the Shared Task ”Exploiting Parallel Texts for Statistical Machine Translation” at the NAACL 2006 Workshop on Statistical Machine Translation[9]. These are 3-gram models that are trained on parts of the Europarl corpus.

4.2 Preprocessing

It seems obvious that the quality of a statistical machine translation system largely depends on the phrase tables that are used. For the latter to achieve good quality, the preprocessing of the parallel corpora needs to be conducted with great care.

As we deal with very large amounts of data, we split up each of the monolingual data sets into three smaller blocks each in order to reduce the risk of memory-related problems and because certain training steps, in particular word alignment with GIZA++, may take a very long time, up to several days, to run.

[9] http://www.statmt.org/europarl/

Since we were not primarily interested in setting up a translation system for commercial purposes, but rather in observing whether our system improves translation quality over a baseline, we used only part of the available parallel corpus. This might in fact decrease the overall translation quality but should still be absolutely sufficient for our experiments. Table 1 gives an overview of the data blocks and their average size in megabytes and amount of word tokens. For our experiments, we used only the first of the three smaller data blocks in each language set. The test data was extracted from each of the second ones, which had not been used for training. Henceforth, we will refer only to the data that has actually been used when discussing block size or amount of tokens and sentences.

Data Sets	*MB*	*Tokens*
Unsplit	165.5	26,855,241
Split	55.25	8,934,839.3

Table 1: Size of (uncompressed) data sets before and after they were split into three parts each. All values are averages of the respective FR, EN, ES, and DE blocks.

The input for the training script consists of bitext in the source and target languages as two files with lowercased and tokenized sentences per line. Each line in one file has to correspond to its translation on the same line in the other file. In most cases, there should be exactly one sentence in the source language corresponding to exactly one sentence in the target language but this does not always have to be the case.

Prior to the training task the following steps have to be performed on the data as explicated below:

1. sentence boundary detection;
2. tokenization, normalisation, and conversion to lower case;
3. sentence alignment;
4. filtering of improbable alignments, removal of long sentences.

4.2.1 Sentence Boundary Detection

Sentence boundary detection is an area of active research. The main difficulty of the task lies in disambiguating if a dot ('.') occurs at the end of a sentence or rather marking an abbreviation, or both. Most algorithms use a manually compiled set of abbreviations and try to determine if one of them is sentence-final due to heuristics like the capitalisation of the following word. Palmer and Hearst (1994) introduce an approach capable of adopting to new languages and domains via learning techniques. The disadvantage of their algorithm is that it needs to be trained on manually annotated data.

A new and quite successful approach was taken by Kiss and Strunk (2005) who propose a method for unsupervised, language-independent extraction of abbreviations and sentence boundaries. They view

abbreviations and their corresponding dots as being collocations and use statistical tests to detect such relations. For our experiments we used their program 'Punkt' which is an implementation of the algorithm mentioned above. There are two major benefits in the application of their system we hoped to be able to take advantage of. First of all, we didn't have any lists of abbreviations for any of the languages in question. Such lists can be found elsewhere but it can be expected that none of them will in fact cover all the abbreviations that actually occur in our corpora. Furthermore, in many cases, some abbreviations for one language will not have correspondences in sets for the other languages which would cause a certain undesirable instability for the task of sentence splitting. This is particularly true for special text domains, in our case political discourse, that bring along own abbreviations and terminology. Another, related, requirement was for a sentence boundary detecting tool to be able to handle all of the languages in question at the same time. This was considered important since presumably all such programs will make systematic errors. However, for the alignment task, this does not constitute such a great problem as long as the same systematic errors occur for all languages equally.

4.2.2 Tokenization

'Punkt' inserts SGML-like tags into the output to mark abbreviations ('$< A >$'), sentence boundaries ('$< S >$'), ellipses ('$< ... >$'), and sentence-final abbreviations ('$< A >< S >$'). According to these, the text was split up into exactly one sentence per line. It was then lowercased and tokenized. During the latter, multiple whitespace was globally reduced to one single space character and punctuation characters where separated from the preceding word so to be viewed as an autonomous token. As a further step, all XML-tags where altered so that only their name without opening or closing brackets remained. We did not simply remove the full tags for the reason that our corpora contain paragraph annotation tags ('$< p >$'), each on separate lines and the sentence alignment tool we used is able to use such hints for improving alignment quality.

4.2.3 Sentence Alignment

We decided on the software 'Hunalign'[10] (Varga et al. (2005)) for aligning sentences between the respective language blocks. Hunalign uses an approach similar to the widely used one described in (Gale and Church, 1993) where the correspondence of two sentences is measured by the similarity of their length in characters. In addition to the basic algorithm, Hunalign takes a lexicon into consideration. If, like in our case, no lexicon is available, Hunalign first aligns in a purely length-based manner and tries to automatically generate a lexicon from this first alignment. Then, in a second step, a realignment is performed, this time also based on the previously generated dictionary. The program provides several

[10] http://mokk.bme.hu/resources/hunalign

post-filtering options, thus restricting the output to sentence pairs with a score higher than a particular threshold.

Finally, we removed all sentence pairs with more than 40 tokens on either side in order to adhere to sentence length restrictions as required by GIZA++ and filtered out still rather improbable alignments by deleting sentence pairs with a length ratio of less than 0.2 as well as with more than three sentences aligned altogether, i.e. anything that is not a one-to-one or one-to-two alignment. This last requirement was necessary because we observed that one-to-two alignments mostly turn out to be correct whereas two-to-two alignments or many-to-many alignments do not. Hunalign provides options to only output one-to-one alignments, but this would have thrown away too many correct pairs.

After the application of all filtering steps, the data amount is reduced by roughly 30% (calculated by means of byte counts).

4.3 Using Multiple Intermediaries

Our method which uses multiple intermediate languages operates on sentence level only. This approach is of course very basic and only touches the very surface of all possibilities described in Section 3.3. Our setup may in fact even lead to certain problems which will be discussed below in Section 5. Considering the lack of time and because it did not seem necessary, we desisted from writing the functionality directly into the source code of the decoder. We therefore operated on the decoder output which, by configuration, contains not only the translation hypotheses but also information about individual scores.

We extracted 2000 sentences from each test set in all 4 languages. The ones from the French (FR) set were used as input for the decoder, the others were kept as reference for evaluation (section 4.4). As Figure 10 shows, the FR sentences were translated in two separate decoder runs into the intermediate languages IL1 and IL2 (English res. Spanish; Henceforth, we will use the generic names IL[12] instead of 'EN' and 'ES' in order to distinguish between EN/ES being generated by the decoder and EN/ES as reference input for evaluation purposes). The 2000 IL1 and IL2 output sentences were in return used as input for two further decoder runs, both of them translating into the target language DE. For each two DE hypothesis (one from the IL1-DE run, the other from IL2-DE), we extracted the one with the highest score.

4.4 Evaluation

4.4.1 The BLEU Evaluation Metric

There are many different ways machine translation results may be evaluated, each of which has certain flaws. The by far best one of them in terms of accuracy and impartiality toward different approaches in machine translation is manual evaluation. Human validators need to be native or near-native speakers

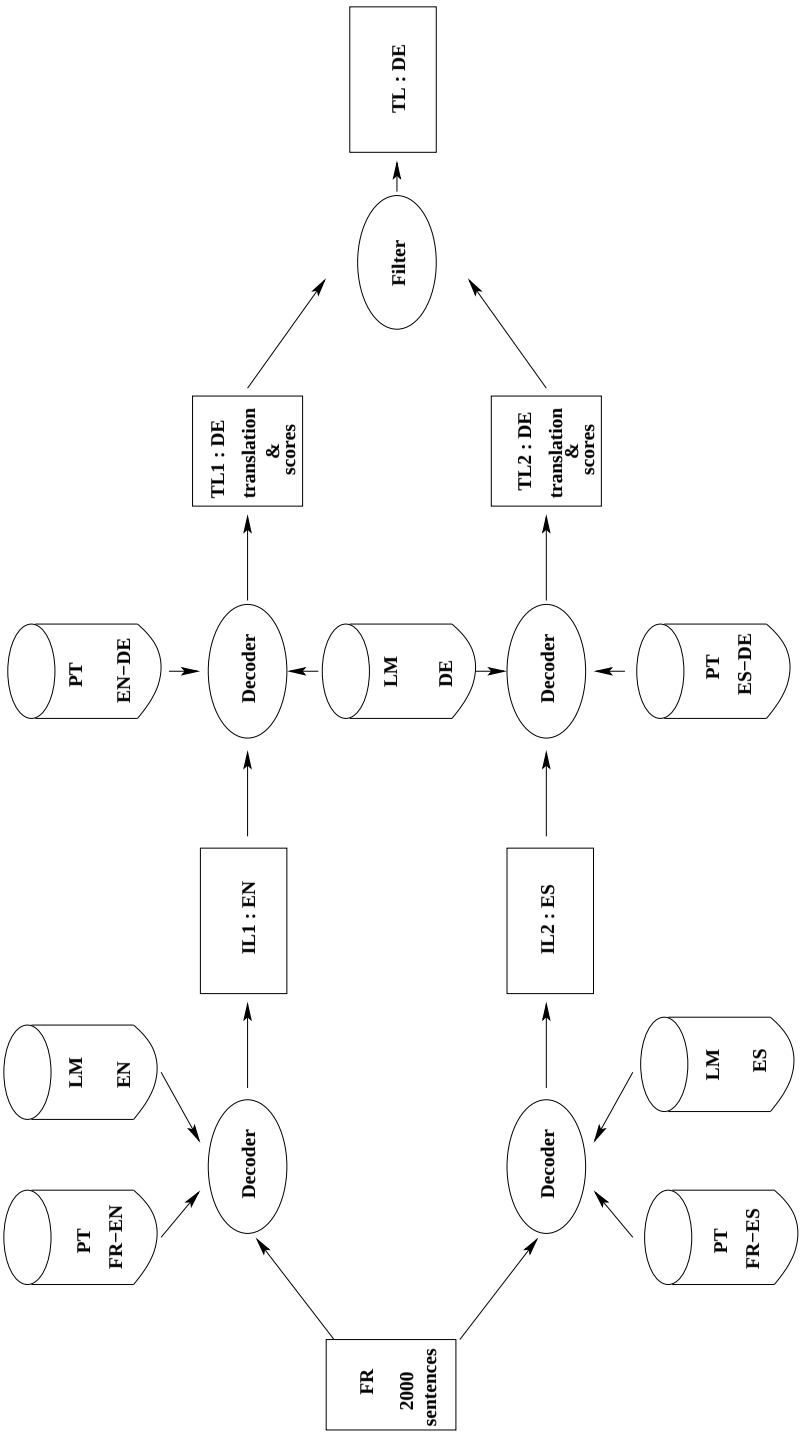

Figure 10: Schematic overview of the experiment's setup.

of the target language in order to judge grammaticality of the MT output. For being able to assure that the output is not only grammatical but actually a correct translation of the source sentence in respect of content, they also need to know the source language to a certain degree. Human validators need to look at every single translated sentence, analyse it thoroughly, and give it several scores, each of which is to capture a certain quality aspect. This, by definition, is an enormously time-consuming task which depends on competent staff and therefore is not well-suited for judging larger amounts of MT output on a regular basis. In addition, it turns out that human validators tend to show large differences in their judgements. To address this issue, much effort has been put into developing aiding software and well-defined scaling schemes.

In order to avoid the cost of manual evaluation, the machine translation community has put much effort into developing automatic evaluation algorithms. These typically compare the MT output against a human-generated reference translation by means of n-gram matches. MT evaluation metrics must also allow for certain variation regarding both word choice and phrase ordering. In recent years, the most frequently used one is the BLEU metric proposed in Papineni et al. (2002). BLEU tries to capture variation in word choice by comparing MT output not against one but multiple references. In order to address the problem of variation in phrase order, Papineni et al. (2002) use modified n-gram precision which is calculated by summing over the matches for every hypothesis sentence S in the entire corpus C.

$$p_n = \frac{\sum_{S \in C} \sum_{n-grams \in S} Count_{matched}(n-gram)}{\sum_{S \in C} \sum_{n-grams \in S} Count(n-gram)}$$

As typical for precision-based metrics, there is no penalty for dropping words. This is addressed by means of a brevity penalty, calculated as:

$$BP = min(1, \frac{output - length}{reference - length})$$

A 4-gram-based BLEU score, then, is computed as:

$$BLEU = BP \cdot \prod_{n=1}^{4} p_n$$

BLEU is currently the de-facto standard evaluation metric in machine translation as it has been shown to very often correlate with human judgement (Coughlin (2003)). Nevertheless, there are also situations in which BLEU may not be a good choice as it tends to allow an overly huge amount of variation for different hypothesis with the same score. It may also give scores very different from human judges when comparing systems that explore different areas of the translation space (Callison-Burch et al. (2006)).

4.4.2 Results

In our setup, we performed the evaluation by means of the BLEU metric. The goal was to show that a translation via multiple intermediaries may achieve better results than a baseline setup which translates via only one intermediate language.

We first extracted 2000 corresponding sentences from the test sets in all four languages, an thus had reference translations to which to compare the output of each translation step. Table 2 lists the corresponding BLEU scores for the output of translating via intermediaries and from references of intermediate languages and target language.

	Language direction	*BLEU score*
Multiple intermediaries	EN&ES → DE	10.12
Separate intermediaries	IL1:EN → DE	9.94
	IL2:ES → DE	11.06
Reference translations	EN → DE	11.60
	ES → DE	13.53

Table 2: Results

A few things may catch one's attention: first of all, the overall BLEU score is rather low - this will be discussed in Section 5. Secondly, the scores of translation results using only one intermediate language (IL1-DE res. IL2-DE) are, as was predicted, lower than those of the reference translations, i.e. EN-DE and ES-DE respectively. Furthermore, taking only the better hypotheses from both IL1-DE and IL2-DE achieved a score better than IL1-DE but worse than IL2-DE. It seems that these results are rather accidental and due to the very basic setup that was viable in the short amount of time. Nevertheless, it has been shown that using multiple intermediaries can improve translation quality over a baseline that uses only one such intermediate language. In fact, we believe that a real-world setting could profit even from these very simple findings. Such an application would not make use of all the available intermediaries since it would have to consider the extra cost that is caused by every additional translation. We suggest, that it would normally be best to translate via a default intermediary and make use of further ones only in case that no good hypothesis can be found. Such a system would probably not know in advance which intermediary may give the best results. With our setup and assuming that English would be decided on as the default, the system could have improved its performance even when operating on sentence level only, as it was done in our experiments.

5 Conclusions & Future Work

We have described and interpreted the fact that current MT systems cover only few language pairs. We have explained that, for SMT, this is due to missing parallel corpora and have proposed methods,

based on Eisele (2006), to overcome this shortcoming. We have outlined the core concepts in SMT and its mathematical foundations and included an overview of software components used in current SMT systems. We have described the setup and implementation of experiments that we had conducted in order to test our hypotheses. We have explained how the evaluation was performed and drew conclusion from our findings.

A lot of what could and should be done on multiple intermediate languages in (statistical) machine translation still lies before us. Although our experiments have not provided a clear evidence for our hypothesis, we think that they have given valuable hints toward which directions are to be taken in the future.

The hypotheses we have layed open in Section 3.3 will need some more substantiation. In particular, it will have to be analysed how often the described phenomena actually occur in typical parallel corpora used for SMT. This would allow conclusions about the range of improvement to be expected from the approach. Also, it would most probably give information on which text domains may be best-suited and how large data sets would have to be at a minimum in order to anticipate significantly better results in comparison to some baseline.

Future experiments will have to analyse the test data to a much greater extent. The test sentences we have used were extracted from a larger test set without us having investigated the question if they were actually suitable for the task. Clearly, much more conclusive results could be expected from test sentences that we would also in theory, or, in fact, on the basis of linguistic intuition, expect to translate more adequately into, res. from, one intermediate language compared to another.

Generally, we expect that training on larger data sets would improve the results. After all, in order to save time, we had to train our models on only part of the data available. It is not clear if these amounts of training data are capable of exploiting all linguistic phenomena that have been described in Section 3.3 and which we have suggested to be the theoretical foundation of our approach.

Furthermore, more attention has to be given to the basic decoding process. We haven't applied any tuning in order to find the best hypotheses in each language direction. Instead, we used unit weights. Looking at sample output, errors stood out that are typical for untuned decoding results, e.g. when translating from English to German, the final verb would often be lost in the output, which might indicate that the reordering model weights are not set correctly.

Lastly, more refined algorithms, some of them have been introduced in Section 3.3.1, will have to be implemented in order to fully exploit the merits of multiple intermediate languages.

6 References

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

Callison-Burch, C., Osborne, M., and Koehn, P. (2006). *Re-evaluating the role of BLEU in machine translation research*. In *Proceedings of EACL 06*.

Coughlin, Deborah (2003). *Correlating automated and human assessments of machine translation quality*. In *Proceedings of MT Summit IX*.

Eisele, Andreas (2006). *Corpora and phrase-based statistical machine translation for new language pairs via multiple intermediaries*. In *Proceedings of LREC-2006*.

Hoang, Hieu, Koehn, Philipp, and Federico, Marcello (2006).
URL `http://www.statmt.org/moses/`

Hutchins, John (2005). *Directory of commercial machine translation systems and computer-aided translation support tools*. Tech. rep.

Kiss, Tibor and Strunk, Jan (2005). *Multilingual unsupervised sentence boundary detection*. In *Computational Linguistics*.

Koehn, Philipp (2005). *Europarl: a parallel corpus for statistical machine translation*. In *MT Summit 05*.

Koehn, Philipp (2007). *Statistical machine translation*. Cambridge University Press.

Koehn, Philipp, Och, Franz Josef, and Marcu, Daniel (2003). *Statistical phrase-based translation*. In *Proceedings of HLT/NAACL 2003*.

Koehn, Phillip and Monz, Christof (2006). *Manual and automatic evaluation of machine translation between European languages*. In *Proceedings of the Workshop on Statistical Machine Translation*.

Och, Franz Josef and Ney, Hermann (2003). *A systematic comparison of various statistical alignment models*. In *Computational Linguistics*, vol. 29(1).

Palmer, David D. and Hearst, Marti A. (1994). *Adaptive sentence boundary disambiguation*. In *Proceedings of ANLPC 94*.

Papineni, K., Roukos, S., Ward, T., and Zhu, W. (2002). *BLEU: a method for automatic evaluation of machine translation*. In *Proceedings of ACL 02*.

Stolcke, Andreas (2002). *SRILM - An extensible language modeling toolkit*. In *Proceedings of ICSLP 02*.

Varga, Dániel, Halácsy, Péter, Kornai, András, Nagy, Viktor, Németh, László, and Trón, Viktor (2005). *Parallel corpora for medium density languages*. In *Proceedings of RANLP 2005*.