Lizenziatsarbeit der Philosophischen Fakultät der Universität Zürich
im Fach Computerlinguistik

Generalized Templates in Example-Based Machine Translation

Sarah Ebling

November 30, 2010

Betreut durch:
Prof. Martin Volk, Universität Zürich
Prof. Andy Way, Dublin City University

I would like to express my gratitude to my supervisors Prof. Martin Volk and Prof. Andy Way, who have given me the opportunity to write my thesis at Dublin City University, have supported my work and have encouraged me to explore new ideas. My thanks also go to Sudip Kumar Naskar for sharing his MT knowledge and resources and providing me with input for my work. Furthermore, I would like to thank Sandipan Dandapat for many fruitful discussions about EBMT. I am also grateful to Mikel Forcada and Sergio Penkale for sharing their experiences with *OpenMaTrEx*.

I am greatly indebted to *SDI Media* for providing the subtitle data for my experiments; this allowed me to explore a special type of data for machine translation and investigate its suitability for EBMT. My thanks also go to Ralf Brown for making available his English–German equivalence classes.

I further wish to thank the people at the DCU School of Computing for making my stay in Dublin such a pleasant one. Last but not least, my gratitude is to my family for their continuous support.

Contents

Figures and Tables

List of Figures

List of Tables

List of Abbreviations

MT Machine Translation

CBMT Corpus-Based Machine Translation

EBMT Example-Based Machine Translation

SMT Statistical Machine Translation

SL source language

TL target language

DCU Dublin City University

CMU Carnegie-Mellon University

POS part of speech

MERT Minimum Error Rate Training

1 Introduction

Example-Based Machine Translation (EBMT), Statistical Machine Translation (SMT) and Rule-Based Machine Translation (RBMT) form the three major paradigms of Machine Translation (MT). The state-of-the-art approach in MT is phrase-based SMT (Koehn et al., 2003).[1] Both SMT and EBMT are instances of Corpus-Based Machine Translation (CBMT). Hence, they rely on a sententially aligned bilingual corpus, called a *parallel corpus* or *bitext*. At the very least, the corpus is divided into a *training set* and a smaller *test set*.[2] Depending on the approach, the training set is used in different ways. The test set is always used to determine an MT system's performance. It consists of a set of *input sentences*. Accordingly, an MT system produces a set of *output sentences*. Output sentences are also referred to as *(translation) hypotheses*.

Several approaches have combined the two CBMT paradigms SMT and EBMT with good results, relying on "the best of both worlds" (Groves and Way, 2005b, 183). The present work focuses on the EBMT paradigm. Our aim is to show what the "best" of the EBMT "world" is but also to tackle the question of why EBMT approaches are not among the state-of-the-art approaches in MT. We discuss the constitutive features of EBMT and present various approaches that exist under its umbrella. One group of approaches makes use of *generalized templates*. Generalized templates are pairs of source language (SL)–target language (TL) units in which some parts have been replaced by variables. Our emphasis for this work is on approaches that apply such templates, as these approaches generally perform better than purely lexical EBMT approaches.[3]

Our own contribution to the field of EBMT research consisted of combining two existing systems that rely on generalized templates. Our goal was to see whether a statistically significant improvement over the individual performances of these two systems could be achieved. We will show that this was not the case. We also revised and extended an existing scheme of generalized templates. In addition, we developed an algorithm that context-sensitively instantiates generalized templates. We embedded it into an SMT system, as only few SMT approaches have made use of generalized templates so far. The results of evaluating this algorithm were promising. The algorithm can be incorporated into an EBMT system.

In most cases, the two factors that decide on the MT paradigm(s) to be applied for a given

[1] In this work, we will refer to phrase-based SMT as SMT.
[2] Usually there is a third set called the *development set*. It is used to tune a system's parameters.
[3] By *(purely) lexical* we refer to non-generalized approaches.

task are (i) the type of data to be translated and (ii) the language pairs involved. In what follows, we discuss these two aspects. As the present work is concerned with EBMT and (albeit to a lesser extent) SMT, we first introduce a number of existing parallel corpora and discuss the process of obtaining a new parallel corpus. We then deal with linguistic features that make translating from one language into another a difficult task. Based on these preliminary considerations, we will discuss which types of data and which language pairs are suitable for use in EBMT systems in the main part of this work.

1.1 Parallel Corpora

The parallel corpora most frequently used for the purposes of MT are the European Parliament Proceedings Parallel Corpus (Europarl) (Koehn, 2005),[4] the Corpus de Bitextes Anglais-Français (BAF),[5] the Aligned Hansards of the 36th Parliament of Canada,[6] the JRC-Acquis Multilingual Parallel Corpus (Steinberger et al., 2006),[7] the OPUS corpus (Tiedemann and Nygard, 2004)[8] and the Corpora of the United Nations (Rafalovitch and Dale, 2009).[9] These examples show that the sources of parallel corpora are often the publications of multinational institutions, such as the UN or the EU, or of governments of multilingual countries, such as Canada (Koehn, 2005).

The corpora listed above contain between 400,000 (BAF) and 40 million (Europarl) words. With regard to the size of the data used to build a CBMT system, there exists a consensus that generally, more data is better data (Koehn et al., 2003; Groves and Way, 2005a,b). Way (2010a, 594) pointed out that "[l]arge parallel corpora exist only for a limited number of language pairs". Hence, there is an ongoing effort to obtain new parallel data from the web. It is important to note that different language versions of a web document are not necessarily parallel. Two documents are parallel if "a noticeable number of sentences can be recognized as mutual translations" (Tomás et al., 2008, 124). If this is not the case, the documents are considered to be merely comparable. An algorithm that collects parallel data from the web will always include a document alignment step that determines whether two documents are, in fact, parallel. The entire process of acquiring a parallel corpus from one or multiple web pages comprises the following steps:

1 extracting raw documents from the web

2 aligning the documents

4 http://www.statmt.org/europarl/
5 http://rali.iro.umontreal.ca/Ressources/BAF/
6 http://www.isi.edu/natural-language/download/hansard/
7 http://langtech.jrc.it/JRC-Acquis.html
8 http://urd.let.rug.nl/tiedeman/OPUS/
9 http://www.uncorpora.org

3 splitting the documents into sentences

4 aligning the sentences

Resnik (1999) was the first to discuss the possibility of collecting parallel corpora from the web. His system *Strand* extracts and aligns web documents.[10] For this, it first identifies web pages that potentially contain parallel text. In practice, this amounts to locating parent pages and sibling pages. Parent pages are pages that point to multiple language versions of a text without containing the text itself, while sibling pages contain the relevant text and point to versions of the same text in other languages.[11] The system uses a search engine to retrieve parent and sibling pages. To locate parent pages, it issues, for example, a query that demands for the string *English* to appear within a predetermined distance from the string *Spanish*. The system then forms document pairs. For this, it takes into account, among other features, the structure of the documents' URLs. This feature exploits the fact that pages that are translations of each other are likely to exhibit parallel URL naming conventions.[12] The system also measures the lengths of the two documents. It then creates a linearized HTML structure for each document. The motivation behind analyzing structural markup is that "authors exhibit a very strong tendency to use the same document structure … [w]hen presenting the same content in two different languages" (Resnik and Smith, 2003, 350).

The system aligns the two HTML structures with the help of a dynamic programming technique. The algorithm takes into account features like the number of aligned non-markup text chunks of unequal length, the length correlation of the aligned non-markup chunks and the significance level of the correlation. All of these features are language-independent. Resnik (1999) pointed out that the quality of the alignment could be improved by applying additional filtering algorithms, such as language recognition or cognate matching. Resnik and Smith (2003) combined *Strand* with an approach that takes into account the content of web pages and requires a language-specific resource, i. e., a bilingual dictionary. The system computes a cross-language similarity score for pairs of SL and TL words. Resnik and Smith (2003) found that their combined approach outperformed *Strand*.

Tomás et al. (2008) extracted parallel corpora from *Wikipedia*[13] and focused on the distinction between parallel and comparable texts. They used web mining and machine learning techniques

[10] We are not aware that it is freely available. A similar tool is *Bitextor* (http://bitextor.sourceforge.net).

[11] Resnik (1999, 532) pointed out that there exists a third kind with which he was not concerned, namely pages that contain a "completely separate monolingual sub-tree for each language".

[12] Note that this does not always hold true. For example, while the structure of the news center on the *Fifa* website (http://www.fifa.com) is parallel across all six languages available, the conventions for naming the news items are not: the news IDs that form part of the news items' URLs are not identical across multiple languages. Valid criteria for forming document pairs in this case are document-internal features like the date of release of the news items and the URL of images contained in the items.

[13] http://en.wikipedia.org/wiki/Main_Page

to extract parallel sentences from parallel documents. To extract parallel sentences from documents that were merely comparable, they used a log-linear feature function classifier. Among other features, they took into account the sentence length ratio, the percentage of cognates and word alignment.

1.2 Languages and Translation Directions

The experiments we performed for the present work are centered around machine translation from English to German. Although these two languages derive from the same family, the West Germanic languages, their combination forms one of the most difficult translation directions. In fact, Koehn (2005) found English to German to be the second most difficult translation direction among the 110 directions he explored, the only more difficult pair being Dutch to Finnish. However, he used the Europarl corpus (cf. Section 1.1) as data, which means that he considered only European languages. Translating between European and non-European languages is usually more difficult. For example, Wu (2009, 1) posited that "language pairs like Chinese–English have presented MT with so much more difficulty than others". Du and Way (2010) pointed out that in the case of Chinese and English, the difficulty arises mainly from the difference in word order between the two languages.

In general, translating from an information-poor into an information-rich language is more difficult than the other way around. German is an example of a morphologically rich language: its noun phrases are marked with case, leading to different inflectional forms for articles, pronouns, adjectives and nouns. English is poorer in morphology. Apart from the morphological differences between German and English, there also exists a difference in syntactic structure. In what follows, four important characteristics of German syntax are presented and the differences to English reflected (Collins et al., 2005):

1 In German subordinate clauses, finite verbs are found in final position, e. g., *damit Sie das übernehmen **können***. In English, the finite verbs have the same position as in coordinate clauses: they follow the subject, e. g., *so that you **can** adopt this*.

2 In German infinitive constructions, the infinitive is found in final position, e. g., *der Sache **nachgehen***. In English, it is found in initial position, e. g., ***look** into the matter*.

3 If a finite verb in a German declarative or *wh*-interrogative clause contains a separable prefix, the prefix is placed at the end of that clause, e. g., *wir fordern das Präsidium **auf*** (English: *we ask the Bureau*). English does not feature separable verb prefixes; in English phrasal verbs, the preposition or adverb is detached by default.

4 If a German sentence contains an infinitive and a negation particle, the negation particle follows the direct object, e. g., *wir konnten es **nicht** mehr rechtzeitig einreichen*. In English, the negation particle follows the finite modal or auxiliary verb, e. g., *we could **not** hand it in in time*. If there is no such verb, it has to be inserted into the sentence (e. g., *we handed it in in time → we **did not** hand it in in time*).

As a further difference, German compound nouns are always combined into one word, while in English they can be distributed across two or more words. For example, the German compound noun *Lehrer-Ausbildung* corresponds to two words in English: *teacher education*. When translating between English and German, a common preprocessing step is to split German compounds to avoid instances of one-to-many word alignments.[14] Furthermore, in German, (letter) case is used to distinguish between parts of speech, for example, between a verb and a noun. It is common to lowercase data during preprocessing, as this reduces the vocabulary size of a corpus. When dealing with German, lowercasing implies the loss of an information source for word sense disambiguation.

1.3 Structure

The remainder of this work is structured as follows: in Chapter 2, we discuss the phases commonly found in an EBMT system (Section 2.1) and give an overview of the approaches found within the EBMT paradigm (Section 2.2). We then focus on EBMT using generalized templates and introduce two systems that follow this approach (Sections 2.3 and 2.4). These are the two systems which we used for our experiments. In Chapter 3, we compare EBMT to SMT and discuss the strengths as well as the weaknesses inherent in the EBMT paradigm (Sections 3.1 to 3.3). We also describe the developments which EBMT and SMT have undergone since their introduction. Providing a comprehensive overview of the SMT framework is beyond the scope of this work, but we will point out references where necessary. We then give an introduction to hybrid and multi-engine MT systems and present a sample of hybrid SMT/EBMT systems (Section 3.4). In Chapter 4, we introduce the most widely used MT evaluation metrics. In Chapter 5, we introduce our experimental data set (Section 5.1) and our approach (Section 5.2). We then present the results of our experiments (Section 5.3) and a discussion thereof (Section 5.4). In Chapter 6, we give an overview of the issues which we tackled in this work and offer an outlook on future research questions.

[14] Such instances can complicate phrase extraction and phrase alignment, which are discussed in more detail in Section 2.3.

2 Example-Based Machine Translation

In the last chapter we outlined our research questions. We stated that the emphasis of this work is on Example-Based Machine Translation (EBMT). EBMT was introduced as "MT by analogy principle" (Nagao, 1984). According to Nagao (1984), it relies on the intuition of humans to make use of translation examples which they have previously encountered in order to translate new input sentences. Apart from the term "analogy-based", EBMT has gone by the names of "case-based", "memory-based" and "experience-guided" machine translation (Somers, 2003, 4). In contrast to SMT, EBMT lacks a well-defined modeling framework. The consequence of this is that a great variety of approaches exist under its umbrella. The approaches have in common that they rely on an example base that contains at least a set of parallel sentences. Much of what an EBMT system requires depends on what particular approach it pursues. There are two main groups of approaches: those that include a training stage and those that do not. The latter are often referred to as *pure EBMT* approaches or *runtime* approaches, as they derive all of the information necessary to translate a sentence from the example base at runtime, i. e., during the actual translation step. These approaches have the advantage that they do not depend on any time-consuming preprocessing steps. On the other hand, their runtime complexity can be considerable. Pure runtime approaches are rare in EBMT nowadays. Section 2.2.1 discusses an approach that started out as a runtime approach and has since incorporated additional knowledge in the form of morphological information.

Approaches that incorporate a training stage are called *compiled approaches*, as training consists of compiling units below the sentence level. These approaches may be further distinguished according to the nature of the precompiled units they rely on: a first group of sub-approaches makes use of derivation trees,[1] and a second relies on generalized templates. Generalized templates are SL–TL pairs in which some parts have been replaced by variables. They provide an additional layer of abstraction and can thus prevent a system from having to revert to word-by-word translation.[2] Naturally, there is a risk of replacing too many parts of an SL–TL pair with variables. To avoid this risk of overgeneralization, generalized templates are usually restricted to certain categories of words. Common candidates for generalization are content words, as re-

[1] Derivation trees are not part of this work.

[2] It is generally acknowledged that translating a sentence word by word leads to poorer translation quality than translating it in larger segments; cf., e. g., Groves and Way (2005a).

placing them with other content words does not affect the grammar of the sentence. Semantic generalization was explored by Kitamura and Matsumoto (1995) and Sumita (2001). Kaji et al. (1992) applied semantic constraints to their approach to syntactic generalization. Pure syntactic generalization was performed by Güvenir and Tunc (1996). Cicekli and Güvenir (2001) generalized over sequences of words. A more recent generalization approach, pursued by Phillips et al. (2003), includes generalization over morphological features. In the following section, we give an overview of the phases that are common to all EBMT systems. Section 2.2.1 then provides an example of an EBMT system that follows the runtime approach. Section 2.2.2 describes some of the systems mentioned above which make use of generalization. The emphasis of the descriptions is on the nature of the generalized templates and on the generalization step.

2.1 Phases of an EBMT System

We stated earlier that EBMT systems following the runtime approach do not incorporate a training stage. The stage that is common to all EBMT systems is the *translation* stage. It is divided into three phases: matching, alignment and recombination. They correspond to the phases commonly attributed to MT, which are analysis, transfer and generation (Somers, 1999). During the *matching* step, those sentence pairs from the example base whose SL halves are sufficiently similar to the input sentence are retrieved. Similarity can be established on the basis of characters, words or syntactic structures. The simplest way to measure similarity on the character or word level is to apply the Levenshtein distance (Levenshtein, 1966). Here, matches are permitted as long as the number of edit operations does not exceed a predetermined threshold. An alternative way of establishing similarity on the word level is to look at semantic similarity. Somers (2003, 24) asserted that semantic matching draws on the strength of EBMT, which is to use "examples with a similar meaning, rather than a similar structure". Nagao (1984) performed semantic matching in his seminal EBMT paper by looking for words and their near-synonyms. Nirenburg et al. (1993) introduced a more refined algorithm for semantic matching. Their distance measure makes use of knowledge obtained from *WordNet* (Miller, 1993) and is computed as shown in equation 2.1.

$$d = 20W + 10w + 5H + 4Y + 3M + 0C\,; \tag{2.1}$$

where W is the number of words of an input sentence that are not in the SL half of the example under consideration, w the number of words of the SL half of the example that are not in the input sentence, H the number of matching words between the input sentence and the SL half of the example when permitting hyperonym matches, Y the number of matching words when permitting synonym matches, M the number of words when permitting morphological variants of a word and C the number of exact word matches. Note that the measure computed is a distance measure, which means that the factors attached to the individual scores are penalty factors (hence a factor of 0 for C).

Once a similar SL sentence is found in the example base, the relevant portions are extracted. Preference is usually given to the longest possible segment of the SL sentence in the example base that matches the input sentence; longer segments are more likely to have unique TL translations. The segments obtained are commonly called *fragments*. The next step is the *alignment* step, sometimes also called *adaptation*. During this step, the TL segments corresponding to the SL fragments are extracted. This is done by looking at the words of the SL fragment and a potential TL correspondence. From there, at the very least the number of word correspondences between the two units and their difference in length are taken into consideration. In the final step, the *recombination* step (sometimes referred to as *synthesis* or *decoding*, a term from SMT), the TL fragments are recombined to form the final output sentence.

2.2 EBMT Approaches

2.2.1 EBMT Runtime Approach

Lepage and Denoual (2005) introduced an EBMT system that adheres to the runtime approach. It is based on the concept of proportional analogies. Proportional analogies are analogical equations: they consist of four sentences in the same language that follow the pattern "A is to B as C is to D". Example 2.2 shows a proportional analogy in English.

(2.2) *I'd like to open these windows.* · *Could you open a window?* :
I'd like to cash these traveler's checks. · *Could you cash a traveler's check?*

The idea behind the approach of Lepage and Denoual (2005) is simple: given three out of four sentences that together form an analogical equation, the fourth sentence can be obtained by solving the equation. As an example, let the fourth sentence in Example 2.2 (*Could you cash a traveler's check?*) be an input sentence to be translated into French. Let us assume that the two English–French sentence pairs displayed in Example 2.3 are found in the example base.

(2.3) *I'd like to open these windows.* · *Ces fenêtres, là, je peux les ouvrir?*
Could you open a window? · *Est-ce que vous pouvez m'ouvrir une fenêtre?*

Together with the input sentence, the two SL sentences in Example 2.3 form the SL analogical equation shown in Example 2.4.

(2.4) *I'd like to open these windows.* · *Could you open a window?* : x · *Could you cash a traveler's check?*

Solving this equation yields the sentence *I'd like to cash these traveler's checks.* for *x*. Assume that this sentence is part of a sentence pair in the example base, as shown in Example 2.5.

(2.5) *I'd like to cash these traveler's checks. · Ces chèques de voyage, là, je peux les échanger?*

Together with the two TL sentences in Example 2.3, the TL sentence in Example 2.5 forms the analogical equation shown in Example 2.6.

(2.6) *Ces fenêtres, là, je peux les ouvrir? · Est-ce que vous pouvez m'ouvrir une fenêtre? : Ces chèques de voyage, là, je peux les échanger? · x*

The solution to this equation is the final translation of the input sentence. The system solves analogical equations by means of an algorithm that determines the longest common subsequence and computes edit distance. Solving the equation in Example 2.6 leads to the following output sentence: *Est-ce que vous pouvez m'échanger un chèque de voyage?*.

It is easy to see that the approach brings about several problems. Firstly, it might be the case that no SL triple $A \cdot B \cdot C$ can be found in the example base to form a proportional analogy with an input sentence *D*. Secondly, multiple triples might be found. Thirdly, multiple solutions to the equation might exist. To illustrate this, consider a different case in which English is the TL and the TL equation displayed in Example 2.7 was produced.

(2.7) *May I have some tea, please? · May I have a cup of coffee? : I'd like some strong tea, please. · x*

Intuitively, it is clear that the output sentence *x* has to incorporate the elements [*I'd like, a cup of, strong, coffee*]. The most obvious solutions are *I'd like a cup of strong coffee.* and *I'd like a strong cup of coffee.* However, because the approach is string-based, there are many more possibilities as to where the string *strong* can be inserted: *I'd like a cstrongup of coffee.*, *I'd like a custrongp of coffee.*, and so on. An approach that is based on word forms would not encounter this sort of difficulty, but it entails a different problem: it is not capable of capturing the different inflectional forms of a word. For example, the quadruple *It walks across the street. · It walked across the street. : It floats across the river. · It floated across the river.* could not be established with such an approach, simply because the word forms *walks/walked* and *floats/floated* would not be recognized as analogous. Somers et al. (2009) proposed to look at strings of morphemes as a possible solution. This implies moving away from a pure EBMT approach to one that relies on morphological analysis during a preprocessing step.

2.2.2 EBMT Compiled Approach

Generalization over Sequences of Words

Cicekli and Güvenir (2001) proposed to generalize over sequences of words. The underlying assumption is that given two SL–TL sentence pairs, if the two SL sentences have certain word form sequences in common, the corresponding TL sentences are expected to exhibit the same similarities among each other. The similar parts of the SL sentences are then assumed to be translations of the similar parts of the TL sentences, and the same applies for the differing parts. Consider the following two SL sentences: *I will drink orange juice.* and *I will drink coffee.* They share the word form sequence *I will drink* and differ in the sequences *orange juice* and *coffee*. Assume that the two sentences are part of an English–Turkish example base, a section of which is displayed in Example 2.8.

(2.8) ***I will drink*** *orange juice.* · *portakal suyu* ***içeceğim****.*
I will drink *coffee.* · *kahve* ***içeceğim****.*

The TL sentences in Example 2.8 share the word form sequence *içeceğim* and differ in the beginning parts of the sentences. To generalize the two sentence pairs, the similar and the differing sequences are replaced with variables. Cicekli and Güvenir (2001, 58) referred to this as "similarity template learning" and "difference template learning". The resulting generalized templates are displayed in Example 2.9.

(2.9) *I will drink* $\boldsymbol{X^S}$. · $\boldsymbol{X^T}$ *içeceğim.*
$\boldsymbol{X^S}$ *orange juice* · *portakal suyu* $\boldsymbol{X^T}$
$\boldsymbol{X^S}$ *coffee* · *kahve* $\boldsymbol{X^T}$

These templates are added to the example base, along with the two "atomic templates" *orange juice* · *portakal suyu* and *coffee* · *kahve*. When translating a sentence like *I will drink tea.*, the system will look for the maximum number of matching word forms between the input sentence and the SL sentences of the example base and will attempt to generalize over the remaining parts. Assuming that it would find the template *I will drink* X^S. · X^T *içeceğim.*, it would then attempt to replace the variable of the TL template with an appropriate word form or word form sequence. Assuming that our example base contains the atomic template *tea* · *çay*, the following translation would be produced: *çay içeceğim.*

Up to this point, the approach is one of pure word form matching. One problem which it is likely to encounter is that there might not be a one-to-one word correspondence between the similarities and differences of the SL and TL sentences. Consider the English–Turkish sentence pairs shown in Example 2.10.

(2.10) *they are walking* · *yürüyorlar*
they are running · *koşuyorlar*

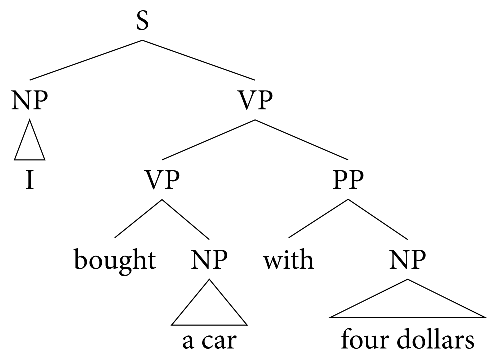
Turkish is an agglutinative language, where verbs can contain multiple types of information. In Example 2.10, the verb contains both the actual third person plural subject as well as the information that it has progressive tense. The same information is distributed across multiple word forms on the English side. Cicekli and Güvenir (2001) proposed to look at such sentences as combinations of word form and stem-morpheme sequences (called "lexical-level representations"). Analyzing the sentence pairs of Example 2.10 in this way yields the sequences displayed in Example 2.11.

(2.11) *they are walk+PROG · yürü+PROG+*3PL
they are run+PROG · koş+PROG+3PL

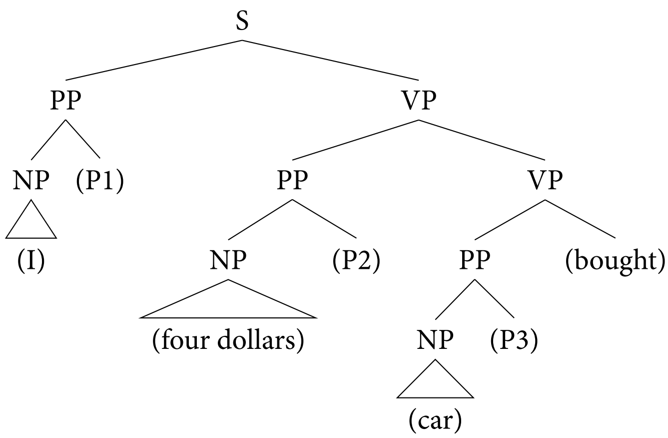
The sequences that can be generalized in this example are *walk*/*run* and *yürü*/*koş*. This leads to the following generalized template: *they are* X^S*+PROG ·* X^T*+PROG+3PL*.

Syntactic Generalization

Kaji et al. (1992) proposed an approach to syntactic generalization with semantic constraints. Their approach relies on a bilingual dictionary as its only resource. We assume that the dictionary contains knowledge of semantic classes, since such information is required during both training and recombination. Naturally, the approach also relies on a syntactic parser for the SL and the TL. The system parses each sentence of the training data. No syntactic disambiguation takes place at this stage: all possible parse trees for a sentence are extracted. Example 2.12 displays sample parse trees for the English–Japanese sentence pair *I bought a car with four dollars.* · 私は4ドルで車を買った。. For the sake of better understanding, the Japanese words have been replaced with their English translations.



(2.12)



After parsing, the system aligns the words on the basis of the bilingual dictionary. Only content words are considered at this stage. An SL–TL word correspondence is established if a content word pair appears in the bilingual dictionary. For the above sentence pair, this results in an alignment of the SL words *I*, *four*, *dollars*, *car* and *bought* with the corresponding Japanese translations. To align the syntactic phrases, a bottom-up searching procedure is initiated, i. e., the shortest SL phrases are considered first. For each SL–TL phrase combination, the algorithm checks whether every content word of the SL phrase has a correspondence in the TL phrase and vice versa. If this is the case, the two phrases are coupled. Since phrases are represented as sets of content words, several phrases might share the same set of content words. In this case, the shortest SL phrase is always aligned with the shortest TL phrase, where length is determined on the basis of all words contained in the phrase.

The phrase pairs are then generalized: aligned subphrase pairs are replaced with variables in phrase pairs. The syntactic categories of the subphrases are also recorded. For example, assume that an adverb phrase pair *if the path name is omitted* · パス名を省略すると contains a noun phrase pair *the path name* · パス名. This adverb phrase pair would be generalized to: *if X[NP] is omitted* · *X[NP]*を省略すると. The templates obtained are subsequently refined: the system determines how many TL correspondences an SL template has. If it has exactly one correspondence, it is considered effective. If it has many correspondences, it is considered useless and is discarded. If it has more than one but only a few correspondences, the word form sequences from which it was derived are taken into consideration. For example, assume that an SL template *play X[NP]* has two TL correspondences, called *T1* and *T2* to simplify matters. Looking into the parallel corpus, we find that the word form sequences *play baseball* and *play tennis* led to the generation of *T1*, and the sequences *play the piano* and *play the violin* yielded *T2*. We now make use of the semantic categories of the nouns: since the dictionary tells us that both *baseball* and *tennis* belong to the category *sport* and both *violin* and *piano* belong to the category *instrument*, we can introduce semantic constraints to obtain two different templates: *play X[NP/sport]* and *play X[NP/instrument]*.

Morphological Generalization

Phillips et al. (2003) proposed to generalize over morphological features when translating from Arabic to English. They acknowledged that this is essentially equal to lemmatizing, but stated that their approach differs from traditional lemmatization in that it deliberately allows for ambiguity in morphological analysis. In written Arabic text, short vowels are commonly omitted. A surface form can therefore be expanded into several different voweled forms. For example, the transliterated Arabic surface form *ktb* can be expanded to *kAtib* ("writer") and *kitAb* ("book"). Voweled forms often originate in different stems and allow for multiple morphological analyses. For example, the surface form *wktAby* has nine possible voweled forms, which originate in three different stems and allow for a total of 13 morphological analyses. Table 2.1 shows these possibilities.

Voweled form	Stem	Morphological analysis
wakitAbiy˜	*kitAbiy˜*	"and" + "writing/written"
wakitAbiy˜u		"and" + "writing/written" + [def. nom.]
wakitAbiy˜a		"and" + "writing/written" + [def. acc.]
wakitAbiy˜N		"and" + "writing/written" + [indef. nom.]
wakitAbiy˜i	*kitAbiy˜*	"and" + "writing/written" + [def. gen.]
		"and" + "writing/written" + [indef. gen.]
wakitAbayo	*kitAb*	"and" + "book" + "two"[acc.]
		"and" + "book" + "two"[gen.]
wakitAbayoya	*kitAb*	"and" + "book" + "two"[acc.] + "my"
		"and" + "book" + "two"[gen.] + "my"
wakitAbiy	*kitAb*	"and" + "book" + "my"
wakut˜Abiy	*kut˜Ab*	"and" + "village school" + "my"
		"and" + "authors/writers" + "my"

Table 2.1: Analyses of the Arabic surface form *wktAby*

Phillips et al. (2003) used a morphological analyzer that assigns stems with similar meanings the same lemma ID. Hence, if a word form corresponds to different stems that are not similar in meaning, it receives multiple lemma IDs. Phillips et al. (2003) proposed to cluster lemma IDs with the aim of having all lemma IDs that can be derived from a specific word form in the same cluster; subsequently, the word forms of the corpus can be generalized on the basis of these clusters, which preserves the morphological ambiguity inherent in them. In the course of their experiments, Phillips et al. (2003) found that they had to rephrase their original goal in such a way that a cluster contains *most* of the lemma IDs that can be derived from a specific word form.

The system splits both the training and the test set into phrases[3] and generalizes the phrases: every word form in a phrase is replaced with the name of the cluster to which most of its lemma IDs belong. The word form itself and all of its morphological analyses are recorded for later use. During matching, the system checks for every generalized phrase whether it is present in the example base. If the phrase is found, the system looks at each token, i. e., cluster tag, of the phrase in turn. For each tag, it calls upon the cluster associated with it. If the cluster contains the word form from which the cluster tag was derived, a perfect match is obtained. This is equivalent to matching without generalization. If no perfect match is obtained, the system retrieves the word forms of the cluster that have the same lemma ID as the word form from which the cluster tag was derived. It then looks at the morphological features of the extracted word forms. For a match to be established, some morphological features (like part of speech (POS) and person) have to be the same, others can be different. From here, a translation into the TL is obtained through rewrite rules.

So far we have presented three different approaches to generalized templates in EBMT: generalization over sequences of words, syntactic generalization and morphological generalization. In the following two sections we describe the two EBMT systems which we used for our experiments. Both systems started out as lexical EBMT systems, i. e., they did not make use of generalized templates. We first describe their original approach and then explain how they were extended to become EBMT systems that apply generalized templates. For the first system, *Marclator*, the extension consists of applying generalization over function words. The second system makes use of semantic and, to some extent, syntactic generalization.

2.3 EBMT at DCU

2.3.1 *Marclator*

One of the two EBMT systems we used for our experiments is *Marclator*.[4] The system was developed at Dublin City University (DCU) and is part of the Machine Translation Using Examples (*MaTrEx*) architecture (Stroppa and Way, 2006). *Marclator* does not apply the greedy matching strategy that is typical of many EBMT systems. Instead, the system segments both the training and the test data into chunks. Therefore, matching is reduced to a binary decision as to whether or not each chunk of an input sentence is found in the example base.

[3] The authors do not enlarge upon the nature of these phrases. We assume them to be n-grams.
[4] http://www.openmatrex.org/marclator/marclator.html

Marclator chunks data based on the Marker Hypothesis (Green, 1979). This is a psycholinguistic hypothesis stating that every language has a closed set of elements that are used to mark certain syntactic constructions. The set of elements includes function words and bound morphemes, such as *-ing* as an indicator of English progressive-tense verbs and *-ly* as an indicator of English adverbs. For some syntactic constructions, both a marked and an unmarked variant exist. The Marker Hypothesis affirms that in such cases, the unmarked form creates "perceptual complexity", while the marked form does not (Green, 1979, 484). Examples are *that*-complement sentences in which the complementizer can be eliminated (marked: *He said that he will come later.*; unmarked: *He said he will come later.*) and English relative sentences in which the relative pronoun can be dropped (*He sold the book (that) he had bought in the United States.*).[5] One way to find evidence for the Marker Hypothesis is to confront readers with the marked and unmarked form of such syntactic constructions to find out whether they deem the unmarked variant more difficult to read. Experiments of this kind have shown that the unmarked variants are indeed more difficult to process.

Marclator's chunking module solely considers function words as indicators of chunk boundaries. Each function word (subsequently called *Marker word*) triggers the opening of a new chunk, e. g., *He was* | ***on** the bus*. For English and German, this leads to left-marking chunks, with one exception: some German prepositions can also function as postpositions, in which case they are right-marking (e. g., *wegen*: *der Sache* ***wegen***). However, most prepositions in German are consistently left-marking. Bound morphemes, on the other hand, are right-marking (e. g., the morpheme *-ing* in *We are com**ing***). Considering left-marking and right-marking elements at the same time would considerably increase the complexity of the *Marclator* chunking algorithm.

A problem to be considered is that not all languages exhibit the same function word categories. Like many languages in the world, Czech has no overt definite articles. Furthermore, there are languages like Spanish, Portuguese or Irish in which certain preposition-pronoun combinations are contracted. Example 2.13 shows some of these instances.

(2.13) Spanish: *con* + *mi* → *conmigo* ("with me")
Portuguese: *com* + *nós* → *connosco* ("with us")
Portuguese: *de* + *isso* → *disso* ("of that")
Irish: *le* + *mé* → *liom* ("with me")

Ideally, such contractions should constitute single-word chunks. However, this is not the case, due to a restriction in the chunking algorithm according to which each chunk has to contain at least one non-Marker word. This restriction is necessary to ensure that, for example, articles always get translated together with the nouns that they accompany.

[5] More extreme examples are the so-called 'garden-path' sentences, e. g., *The horse raced past the barn fell.*

Category	Example
determiner	*den*
personal pronoun	*euch*
demonstrative pronoun	*jenem*
possessive pronoun	*seine*
interrogative pronoun	*welch*
indefinite pronoun	*andere*
relative pronoun	*denen*
preposition	*abseits*
coordinative conjunction	*aber*
subordinative conjunction	*falls*
cardinal numeral	*eins*
numeric expression	*ninety-nine*
auxiliary/modal verb	*darf*
punctuation	*!*

Table 2.2: Marker categories and examples

Lists of Marker words can be obtained from a lexical resource annotated with POS information. The underlying POS tagset has to be of a certain granularity: at the very least, it has to distinguish between different kinds of pronouns as well as between coordinative and subordinative conjunctions. For English and German, 14 categories are used; they are displayed in Table 2.2. For English, *Marclator* relies on a list derived from the dictionary of the MT platform *Apertium*.[6] The German Marker words were extracted from the database of the Centre for Lexical Information (*Celex*).[7]

The lists thus obtained contain a total of 450 Marker words for English and 550 for German. Table 2.2 also lists a sample Marker word for each category. The examples show that entries are included in their inflected forms. A Marker word can belong to multiple categories. For example, the German word *denen* is both a demonstrative and a relative pronoun. This does not cause any problems during chunking or the subsequent chunk alignment, as the chunk alignment module in its current form does not take into account Marker word categories. However, disambiguation of Marker words becomes an issue when dealing with generalized templates, an approach described later in this section. Stroppa and Way (2006) found that treating the punctuation marks *! ? , . : ;* as additional Marker elements improved performance in their experiments.

The first few open-source releases of *Marclator* did not include a German Marker file, as *Celex*, the source from which the original file was derived, is not a freely available resource. We created a new German Marker file for the third release of the system. As a basis we used the English–German word list contained in the dictionary lookup program *Ding*.[8] The list contained the following POS (sub-)categories relevant to our Marker file: determiner, pronoun, personal pronoun, conjunction, preposition, numeric expression. To arrive at the granularity of the categories listed in Table 2.2, we marked each occurrence of a pronoun as demonstrative, possessive, interrogative, indefinite or relative pronoun.[9] Similarly, we determined whether a conjunction was coordinative or subordinative. In the end, we added auxiliary and modal verbs in all possible inflected forms as well as the punctuation marks *! ? , . : ;* to the file. The new German Marker file

[6] http://www.apertium.org/
[7] http://www.ldc.upenn.edu/Catalog/CatalogEntry.jsp?catalogId=LDC96L14
[8] http://www-user.tu-chemnitz.de/~fri/ding/
[9] Note that personal pronouns constituted a separate category.

contains 707 entries. We performed an experiment to determine its quality in comparison to the original file, the results of which are reported in Chapter 5. The bulk of our experiments reported in the same chapter were performed using the original file.

We expected the higher number of Marker words (+157 compared to the original file) to lead to longer chunks: in German, auxiliary verbs are often found between two Marker words, where, if they are themselves Marker words, they prevent a chunk from being opened at the beginning of the following Marker word. For example, in the German interrogative sentence *Was ist dein Traum?*, Marker words according to the original German Marker file are *was* and *dein*. Hence, chunking according to this file yields the sequence *was ist* | *dein traum*. The new Marker file contains *ist* in addition to the Marker words of the original file. With this file, no new chunk is created at the beginning of *dein*, due to the restriction that every chunk has to contain at least one non-Marker word. This leads to the chunking output *was ist dein traum*. Note that in this sentence, the word *ist* does not function as an auxiliary but rather as a copula verb. An example of a sentence in which a true auxiliary verb is surrounded by two Marker words is: *Was* ***hast*** *du gemacht?* The *Marclator* chunking module does not distinguish between these two cases: it classifies each occurrence of *ist* or *hast* as a Marker word.

After chunking the training data, *Marclator* aligns the SL and TL chunks of a sentence pair. For this, the system first aligns the words: it invokes the first four steps of *Moses* (Koehn et al., 2007), which means that it relies on *Giza++* (Och and Ney, 2003) for word alignment. The chunk alignment algorithm is an edit-distance style algorithm in which the distances are replaced by opposite-log conditional probabilities. The algorithm allows for block movements (Leusch et al., 2006). This renders it suitable for alignments between languages with different constituent order, like English and German. For such language pairs, the chunks are not guaranteed to correspond pairwise sequentially. The chunk alignment algorithm also supports many-to-one alignments. The conditional probability of an SL–TL chunk alignment is computed from the values of two features (Stroppa and Way, 2006):

1 word translation probabilities: these are the translation probabilities extracted from the output of *Giza++*. The individual values are combined into a word-based lexicon model to produce the final value for this feature.

2 word cognates: the value of this feature is based on a combination of the Levenshtein distance, the longest common subsequence ratio (Hirschberg, 1975) and the Dice coefficient (Dice, 1945).

Both the word and the chunk alignment algorithm of *Marclator* rely on statistical knowledge. The integration of statistical knowledge in EBMT systems is discussed in Chapter 3, where we also raise the question of the extent to which such systems can still be considered pure EBMT systems, i. e., systems that follow the EBMT paradigm exclusively.[10]

The recombinator of *Marclator* is a left-to-right monotone recombinator. When translating an input sentence, it first looks for a matching sentence in the example base. If none is found, the sentence is chunked based on the Marker Hypothesis. Each chunk that is not found in the example base is then split into single words. If several TL correspondences for an SL chunk or word are found in the example base, the one with the highest probability is chosen.[11] Thus, for each input sentence, the recombinator outputs a single hypothesis.

2.3.2 Extension to *Marclator*

A problem inherent in the approach described above is that the chunks of an input sentence often cannot be found in the example base. Since translating a chunk as a whole is likely to yield a better translation than translating it word by word, it is desirable to increase the chunk coverage of a system. Gough and Way (2003) extended the precursor to *Marclator* by including an additional layer of abstraction: they produced generalized chunks from word form chunks by replacing the Marker word at the beginning of a word form chunk with the name of its category, e. g., *of a marathon* → *<PREP> a marathon*.

In its original form, the system aligned the generalized chunks sequentially: an aligned sentence pair had to contain the same number of chunks so that the chunks could be aligned pairwise, subject only to the Marker category labels (subsequently called *Marker tags*) matching. Gough and Way (2003, 75) noted that "this seemingly naïve approach prove[d] quite effective". In Gough and Way (2004), they refined their algorithm by taking into account the words contained in the SL–TL chunks and their positions as well as their cognates. This led to a larger amount of chunk alignments.

During the actual translation step, the system of Gough and Way (2003) first looks for matching sentences, then reverts to word form chunks and subsequently to generalized chunks. To instantiate a TL Marker tag, the system traces back the corresponding SL Marker word and gathers all possible translations for this word from the set of word alignments. It then calculates the relative frequency for each of these TL words and assigns each translation a score. The generalized template extension is not part of the current *Marclator* system. We therefore reimplemented it for our experiments. This process is described in Section 5.2.

[10] Note that the notion of pure *EBMT* systems introduced here is different from that of *pure* EBMT approaches, which refers to EBMT approaches that do not incorporate a training stage (as discussed earlier in this chapter).

[11] This is a common procedure for decoders that do not incorporate a language model. Language modeling in EBMT is discussed in Section 3.2.

2.4 EBMT at CMU

2.4.1 *CMU-EBMT*

The second EBMT system we worked with for our experiments is *CMU-EBMT*.[12] The system forms part of *PanLite* (Frederking and Brown, 1996), an MT architecture developed at Carnegie-Mellon University (CMU). Apart from *CMU-EBMT*, *PanLite* contains a transfer-based and a knowledge-based MT engine. *CMU-EBMT* can be invoked on its own from within *PanLite*. In what follows, we describe the system's training and translation process.

CMU-EBMT requires a parallel corpus and a bilingual dictionary. Optionally, it can be provided with a list of synonyms in the TL, a list of words that may be elided during alignment (e. g., the Spanish words *se*, *su*, *el* or *una*) and a list of words that may be inserted (e. g., determiners). Brown (1996) used entries from a commercial bilingual dictionary for his experiments in translation from Spanish to English, along with a list of synonyms obtained from *WordNet*. In Brown (1997), he proposed to compile a bilingual dictionary from the parallel corpus itself, thereby exploring the lexical knowledge implicitly contained in the aligned sentences. The algorithm is divided into two steps: creating a correspondence table and filtering the table according to a threshold scheme. The correspondence table is a two-dimensional table in which the SL words form one dimension and the TL words the other. For each SL–TL sentence pair in the parallel corpus, the set of unique words on either side is determined. For every possible SL–TL word combination within the two sets, the respective count in the correspondence table is incremented. Word pairs that do not occur in a similar position receive an increment of 1, while word pairs whose positions in the respective half of the sentence pair differ only by a certain value receive double increment. This bias is introduced to take account of the fact that in languages with similar word order, words occurring in similar positions are likely to be mutual translations. The monolingual word counts are also recorded at this stage; they are required during the subsequent filtering step.

The correspondence table is filtered with two ratio tests: a symmetric and an asymmetric co-occurrence ratio test. Only word pairs that pass both tests are added to the final dictionary. The symmetric co-occurrence ratio test assumes one predetermined threshold. The test is passed if the co-occurrence count of the word pair meets the following two conditions:

1 It is equal to or greater than the product of the threshold and the number of occurrences of the SL word (which was recorded during the generation of the correspondence table).

2 It is equal to or greater than the product of the threshold and the number of occurrences of the TL word.

[12] http://sourceforge.net/projects/cmu-ebmt/

The asymmetric co-occurrence ratio test assumes two predetermined thresholds. The test is passed if the co-occurrence count of the word pair meets *either* of the following conditions:

1 It is equal to or greater than the product of threshold one and the number of occurrences of the SL word, and it is equal to or greater than the product of threshold two and the number of occurrences of the TL word.

2 It is equal to or greater than the product of threshold one and the number of occurrences of the TL word, and it is equal to or greater than the product of threshold two and the number of occurrences of the SL word.

Brown (1997) found that obtaining dictionary entries in this manner produced defective translations for high-frequency words. He therefore extended the algorithm by an additional extraction step. In this step, only sentence pairs containing words that appeared in at least 20 % of the SL sentences are considered. This leads to a second dictionary containing improved translations for the high-frequency words. The two dictionaries are merged in the final step, with the values obtained during the second step overriding those of the first step in case of duplicate entries.

Unlike *Marclator*, *CMU-EBMT* does not require subsentential units to be compiled before the actual translation step. Training in *CMU-EBMT* consists solely of training a language model and, optionally, tuning the system parameters to the data used. Both language modeling and parameter tuning are techniques borrowed from SMT; language modeling is discussed in Section 3.2. To render the subsequent matching step efficient, the parallel corpus is indexed offline. The matching step resembles closely that of a traditional EBMT system: *CMU-EBMT* extracts every substring of the input sentence with a minimum length of two tokens that appears in the SL half of the example base. Example 2.14 shows a Spanish input sentence. A set of substrings that could result from matching the sentence with an example base is shown in Table 2.3.

(2.14) *El Banco de Santander habia sido elegido el lunes per las autoridades monetarias espanolas para comprar el Banco Espanol de Credito (Banesto), cuarto banco espanol.*

The system tries to find a translation for each fragment of the kind shown in Table 2.3. It identifies the smallest and the largest possible segment in the TL sentence that correspond to an SL fragment. Every possible substring of the largest segment that contains at least the minimal segment receives a score. The best alignment is the one with the lowest score. The alignment score is the weighted sum of the values of eight features, which include: the number of SL words with no correspondences in the TL segment,[13] the number of TL words with no correspondences

[13] SL–TL word correspondences are established with the help of the bilingual dictionary and the (optional) TL synonym list: an SL and a TL word correspond if the TL word itself or one of its synonyms appears in the set of translations for the SL word.

El Banco de	*el lunes por*	*de Credito*
El Banco de Santander	*por las*	*de Credito (*
Banco de	*por las autoridades*	*Credito (*
Banco de Santander	*por las autoridades monetarias*	*, cuarto*
de Santander	*las autoridades monetarias*	*banco espanol*
habia side	*comprar el*	*espanol*
elegido el	*Espanol de Credito*	

Table 2.3: *CMU-EBMT*: matched fragments

in the SL fragment, the number of SL words with a correspondence in the TL sentence but not in the relevant TL segment, and the difference in length between the SL and the TL segment. Each translation is passed on to the recombination step as long as its score does not exceed five times the length of the SL fragment.

2.4.2 Extensions to *CMU-EBMT*

Manually Created Semantic and Syntactic Generalized Templates

Brown (1999) proposed an extension to *CMU-EBMT* that makes use of semantic and syntactic generalized templates. He referred to the generalized template categories as *equivalence classes*. Examples of semantic and syntactic equivalence classes are given in Table 2.4. The table shows that members of equivalence classes can in turn contain equivalence classes; in particular, semantic classes can be part of syntactic classes. This becomes obvious in the last instance (shown in bold): The SL template *a* <*color*> <*noun-f*> and the TL template *une* <*noun-f*> <*color*> both contain the semantic class <*color*> and are members of the syntactic class <*np-f*> (noun phrase, feminine).

Brown (1999) pointed out that including syntactic equivalence classes holds the potential of implicitly disambiguating word forms. This happens, for example, when translating the English phrase *the affordable painters* into Spanish: when looking at the English word *affordable*, a system cannot tell without further knowledge whether it is a singular or a plural adjective (<*adj-s*> vs. <*adj-p*>, cf. Table 2.4). The system therefore has to keep track of both options. Subsequently, it is able to identify *painters* as a plural noun (<*noun-m-p*>, cf. Table 2.4). Since there exists a replacement rule that contains both <*noun-m-p*> and <*adj-p*> (class <*np-m*>), *affordable* is correctly disambiguated as a plural adjective in this case.

The system generalizes both the training and the test set: it recursively replaces words and phrases that are part of an equivalence class with the corresponding equivalence class tag (subse-

Class	Sample member
<city>	*Vienna · Wien*
<continent>	*Europe · Europa*
<religion>	*Christianity · Christentum*
<month>	*December · Dezember*
<frequency>	*bimonthly · halbmonatlich*
<company>	*BASF · BASF*
<company-suffix>	*GmbH · GmbH*
<company>	*<company> <company-suffix> · <company> <company-suffix>*
<firstname-m>	*Aleksander · Aleksander*
<lastname>	*Zimmermann · Zimmermann*
<fullname-m>	*<firstname-m> <lastname> · <firstname-m> <lastname>*
<fullname-m>	*George Washington · George Washington*
<adj-s>	*affordable · accesible*
<adj-p>	*affordable · accesibles*
<noun-m-p>	*painters · pintores*
<np-m>	*<adj-p> <noun-m-p> · <noun-m-p> <adj-p>*
<noun-m>	*book · livre*
<np-m>	*<poss> <adj-n> <noun-m> · <poss> <noun-m> <adj-m>*
<np-m>	*<poss> <noun-m> · <poss> <noun-m>*
<np-f>	*the <noun-f> · la <noun-f>*
<np-f>	***a <color> <noun-f> · une <noun-f> <color>***

Table 2.4: Semantic and syntactic equivalence classes for English–Spanish in Brown (1999)

quently called *class tag*). Syntactic classes are applied before semantic classes, and disambiguation numbers are introduced to distinguish between multiple occurrences of the same class tag in a sentence. In the training data, generalization is performed only if a member of a particular equivalence class is found in both the SL and the corresponding TL sentence. In the case of single-word replacements, the word forms that are replaced are retained as alternatives during the matching process. This does not apply to replacements of more than one word, due to the difference in length to the (single-token) class tags.

In the sentences of the test set, all members of an equivalence class are replaced recursively. To demonstrate this process, consider the English input sentence in Example 2.15.

(2.15) *John Miller flew to Frankfurt on December 3rd.*

Assume that the English–German equivalence class rules shown in Example 2.16 exist. Note that in the last replacement rule, the order of the class members is different for the two languages (*<month> <ordinal>* vs. *<ordinal> <month>*, shown in bold).

(2.16) *<firstname-m> : John · John*
<lastname> : Miller · Miller
<city> : Frankfurt · Frankfurt
<month> : December · Dezember
<ordinal> : 3rd · 3.
<person-m> : <firstname-m> <lastname> · <firstname-m> <lastname>
<date> : <month> <ordinal> · ***<ordinal> <month>***

Applying the first five rules in Example 2.16 to the input sentence of Example 2.15 yields: *<firstname-m> <lastname> flew to <city> on <month> <ordinal>.* From there, the last two rules of Example 2.16 apply. This leads to the final generalized sentence: *<person-m> flew to <city> on <date>.*

The matching process is equivalent to that of the purely lexical *CMU-EBMT* system,[14] with the apparent difference that here, two matching levels – a lexical and a generalized one – exist. Alignment proceeds in the same way as in *CMU-EBMT*. Following this, the rules that were stored during the generalization of the input sentence are applied in reverse so as to transform the generalized TL fragments into word form TL fragments. To demonstrate this instantiation process, we assume that a single TL fragment was extracted for the input sentence of Example 2.15: *<person-m> flog am <date> nach <city>.* Applying the replacement rules backwards yields: *<firstname-m> <lastname> flog am <ordinal> <month> nach <city>. → John Miller flog am 3. Dezember nach Frankfurt.*

Automatically Created Generalized Templates

Brown (2000) proposed to apply word clustering methods to automatically obtain equivalence classes. His algorithm starts out by identifying SL–TL word correspondences in the parallel corpus. The unique word pairs (i. e., the pairs of SL and TL words that maintain no other word correspondences) are extracted. The term vectors required for the clustering process are built from the SL words' contexts.[15] Brown (2000) determined the context to be three words to the left and three words to the right of an SL word. For the French–English word pair *cinq · five*, the context for the SL word *cinq* might consist of the two sequences displayed in Example 2.17 (where *<NUL>* is a placeholder to indicate that there was not enough context on either side).

[14] By *(purely) lexical* we refer to the non-generalized *CMU-EBMT* system.
[15] Note that this is equivalent to bilingual clustering using monolingual techniques (Brown, 2000).

(2.17) *<NUL> <NUL> Le **cinq** jours depuis la*
*elles commenceront en **cinq** jours. <NUL>*

Position-sensitive word counts are obtained from this, as shown in Table 2.5. For example, *Le(-1)* in the fifth line (shown in bold) indicates that the word *Le* has a position of *-1* with respect to the word *cinq* in one of the context lines of Example 2.17. The frequencies of the context words are combined into a term vector for the word pair *cinq · five*.

Context word	Count
<NUL> (-3)	1
elles (-3)	1
<NUL> (-2)	1
commenceront (-2)	1
***Le* (-1)**	**1**
en (-1)	1
jours (1)	2
depuis (2)	1
. (2)	1
la (3)	1
<NUL> (3)	1

Table 2.5: Clustering: context

Brown (2000) reported that his automatically created equivalence classes yielded a test set coverage comparable to that of the manually created classes. The results improved further when he seeded the clustering process with manually created classes. For the clustering algorithm, he experimented with different variants of group-average clustering and agglomerative clustering and found that single-link agglomerative clustering performed best. This algorithm introduces a new cluster for each term vector and then repeatedly merges the two most similar clusters until no two clusters have a similarity score above a predetermined threshold. Similarity between two clusters is determined by computing the maximal cosine similarity between any pair of members of these two clusters. Gangadharaiah et al. (2006) referred to the approach of Brown (2000) and suggested to apply a spectral clustering algorithm instead. Spectral clustering algorithms make use of the eigenvalues of distance matrices. Gangadharaiah et al. (2006) compared spectral clustering to group-average clustering rather than agglomerative clustering, which was found to perform better by Brown (2000). They observed that their clusters yielded more "natural and intuitive word classes" than those obtained by group-average clustering (Gangadharaiah et al., 2006, 42). The empirical evaluation which they conducted shows spectral clustering to perform better than group-average clustering only in cases where there were moderate amounts of data. The algorithm performed worse when evaluated over small amounts of data and equally good when evaluated over large amounts of data.

Brown (2003) combined his approach to generalizing the equivalence classes automatically with the transfer-rule induction approach of Cicekli and Güvenir (2001) which was described in Section 2.2.2. Recall the assumption behind the transfer-rule induction approach: if two SL sentences in an example base exhibit certain similarities and certain differences, the corresponding TL sentences are expected to feature the same similarities and differences. Consider the similar and differing parts in the two sentence pairs listed in Example 2.18. The similar parts are shown in bold.

(2.18) ***nous regardons*** *les approvisionnements en énergie* **.** · ***we are watching*** *energy supplies* **.**
nous regardons *les produits chimiques agricoles* **.** · ***we are watching*** *agricultural chemicals* **.**

The generalized template pair produced from these two sentence pairs is: *nous regardons <cl_1>.* · *we are watching <cl_1>*. An equivalence class *<cl_1>* is created which receives two SL–TL members: *les approvisionnements en énergie* · *energy supplies* and *les produits chimiques agricoles* · *agricultural chemicals*. Furthermore, the pairs displayed in Example 2.19 are added to the example base as new subsentential units.

(2.19) *nous regardons <cl_1>.* · *we are watching <cl_1>*
nous regardons · *we are watching*
les approvisionnements en énergie · *energy supplies*
les produits chimiques agricoles · *agricultural chemicals*

The process of retrieving quadruples of sentences that exhibit similarity/difference patterns and of generalizing them repeats on the updated corpus until no more replacements can be performed.

At this point, the two approaches are combined: clustering is performed over word pairs (Brown, 2000) and pairs of equivalence class tags like the one above, *<cl_1>* · *<cl_1>*. The class tags are clustered because the induction process produces many classes with only a few members each. Clustering serves to merge classes that are used in similar contexts.[16] The input to the clustering process is the final updated training corpus. The context of the SL half of each word pair or class tag pair is established based on this corpus. Term vectors are then built from the context, just like in the approach of Brown (2000). This results in equivalence classes that contain either word pairs only, class tag pairs only or combinations of word pairs and class tag pairs. These are the final equivalence classes over which the system generalizes. The names of the classes are the numbers of the clusters. Table 2.6 gives some examples.

In this chapter we introduced the two main groups of approaches that exist in EBMT: runtime and compiled approaches. We gave examples of systems that adhere to these approaches. In particular, we introduced several systems that rely on generalized templates. In the next chapter we will show that these systems are among the best-performing systems in EBMT. Nevertheless, they are still outperformed by SMT systems in many cases. We discuss the reasons for this, i. e., the differences between EBMT and SMT. We also show how the two paradigms have developed since their introduction and present examples of systems in which they have been combined.

[16] Note that a similar approach was pursued by Phillips et al. (2003), who clustered the lemma IDs output by an Arabic morphological analyzer (as described in Section 2.2.2).

Class	Members
1776	*absurde · nonsense*
	<cl_18> · <cl_18>
2158	*pêcheurs · fisheries*
	pénuries · shortages
	officiers · officers
2609	*<cl_54> · <cl_54>*
	<cl_98> · <cl_98>
	<cl_375> · <cl_375>

Table 2.6: Sample clusters

3 Example-Based vs. Statistical Machine Translation

We stated in the introductory chapter that both EBMT and SMT rely on a parallel corpus. In the last chapter we outlined how EBMT systems make use of the corpus: by consulting the training set (its example base) directly at runtime. In contrast, SMT systems consult the probabilities of SL–TL word or phrase pairs which they have learned from the training data offline. Hence, the main feature that distinguishes the two paradigms is the type of knowledge used during the translation step. This corresponds to the difference between symbolic and numeric knowledge (Carl and Way, 2003) and has one important impact: whenever an input segment is found in the example base in an EBMT system and the SL half of the corresponding example has only one translation, the system is guaranteed to render that translation, i. e., to "be able to reproduce the training set" (Way, 2010b, 3). This is not the case with an SMT system, since SMT systems operate solely on probabilities.

EBMT systems have often performed worse than SMT systems in the past. Groves and Way (2005a) compared an EBMT system that uses generalized templates with an SMT system that is based on *Pharaoh* (Koehn, 2004a). They experimented with English-to-French and French-to-English translation on a test set of 322,000 sentences and found that, on average, the SMT system received 7 BLEU points more than the EBMT system.[1] The absolute scores were on the order of 21.02 (SMT) vs. 14.27 (EBMT) for the French-to-English system, and 19.3 (SMT) vs. 14.9 (EBMT) for the English-to-French system. Purely lexical EBMT systems typically perform still worse.

The biggest shortcoming of EBMT is that it does not combine translations of phrases well. This problem is known as *boundary friction* (Way, 2001, 2). It is especially frequent when translating into a morphologically rich language. As an example for translating from English into German, assume that the sentence pairs listed in Example 3.1 are contained in the example base (Way, 2001).

(3.1) ***A big dog*** *eats a lot of meat.* · ***Ein großer Hund*** *frisst viel Fleisch.*
I have *two ears.* · ***Ich habe*** *zwei Ohren.*

[1] BLEU and other MT evaluation metrics are discussed in Chapter 4.

An EBMT system might make use of the phrases shown in bold to translate a sentence like *I have a big dog.* into: *Ich habe ein großer Hund.* In doing so, it would neglect the fact that German uses different inflectional forms to mark grammatical case: the German phrase *ein großer Hund* in the first sentence is a nominative noun phrase and therefore a legitimate choice as the subject of this sentence, but *Ich habe* requires an accusative object (*einen großen Hund*). To solve problems like these, EBMT systems have started incorporating probabilistic knowledge (Bangalore et al., 2002; Groves and Way, 2005a). This development is discussed in Section 3.2.

Generally speaking, EBMT systems are capable of outperforming SMT systems in three cases:

1 when the training set and test set are sufficiently similar: EBMT systems can exploit the examples contained in the training data directly and possibly in full length

2 when they can exploit long phrases: this reduces the likelihood of boundary friction

3 when there are only small amounts of data available: EBMT systems are less prone to the problem of distorted probabilities when dealing with sparse data. This problem is known to SMT systems and is mentioned in Section 3.3.2.

Cases 1 and 2 are closely related in that a high similarity between the training set and the test set is a precondition for an EBMT system's ability to explore longer phrases. The cases are listed separately because each has its own implication: case 1 has encouraged system developers to look for domains in which a high internal similarity of data exists *per se*. For example, Gough and Way (2003) applied their EBMT system to controlled language translation. Controlled languages are subsets of natural languages. They are designed so as to reduce the amount of ambiguity and complexity inherent in language. For this reason, they have a restricted grammar and vocabulary. Hence, they exhibit a high degree of internal lexical and syntactic similarity. Case 2 has consequences for the matching technique: EBMT approaches that pursue a greedy matching strategy are likely to yield better translations than approaches that partition the example base and the test data according to a predefined scheme. As a consequence of case 3, EBMT systems have been preferred over SMT systems for translation tasks that involve minority languages; for such language pairs, often only small amounts of parallel data exist.

3.1 MT Systems in a Three-Dimensional Space

Earlier in this work, we referred to EBMT runtime approaches as pure approaches because they come closest to EBMT in its earliest days. Wu (2005) pointed out that EBMT systems pursuing a runtime approach perform *lexical*, *logical* and *example-based* MT. According to the author, each of these terms (or rather, the concepts behind them) has a counterpart with which it forms an axis

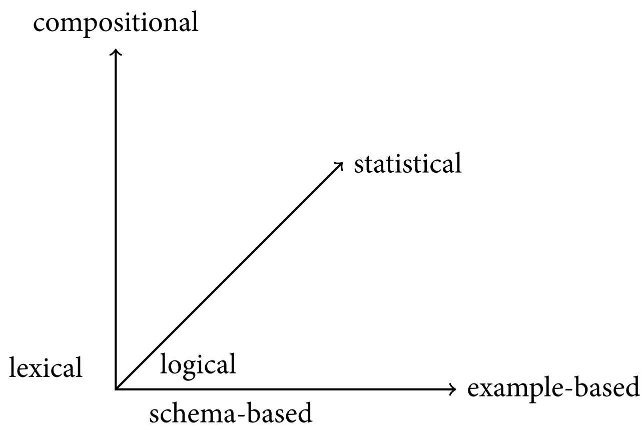


Figure 3.1: MT space as a three-dimensional space

in a three-dimensional space. Lexical MT stands in opposition to *compositional* MT: it does not make use of recursive transfer rules that describe how the translation of a chunk can be obtained by combining the translations of smaller component chunks. Logical MT forms the counterpart of *statistical* MT and as such does not make use of statistics and probabilities. Example-based MT derives its meaning from a contrast to *schema-based* MT: the former uses the examples at runtime, thereby performing memorization, while the latter relies on abstract schemata at runtime.

Every MT system can be assigned a place within the three-dimensional space opened up by the three axes lexical–compositional, logical–statistical and example-based–schema-based (cf. Figure 3.1). It is clear that not all of today's EBMT systems are located at the intersection of lexical, logical and example-based MT. Wu (2005) pointed out that since the introduction of the paradigm in 1984, EBMT systems have moved in three directions: towards compositional, schema-based and statistical MT. The movement towards statistical MT is the result of an increased use of probabilities to compute similarities during matching, score alignments and rerank n-best lists of translations. N-best lists are commonly reranked on the basis of a language model. We introduce language models in the following section and take a closer look at their application in EBMT.

3.2 Probabilistic Knowledge in EBMT

Several approaches have been proposed to provide a solution for the boundary friction problem described earlier. Somers et al. (1994) suggested to include information about the context in which a fragment appears, e. g., to record the words and POS tags that may precede or follow it. Brown et al. (2003) proposed to enable EBMT recombinators to combine overlapping fragments, with overlaps consisting of one or several words. The underlying assumption is that an output sentence composed of overlapping TL fragments is likely to be of better quality than an output sentence composed of non-overlapping TL fragments. Consider the input sentence in Example 3.2 from a French-to-English translation task.

(3.2) *Je doute qu'il soit nécessaire de lancer une enquête complète pour l'instant.*

Assume that the fragment pairs displayed in Examples 3.3 to 3.8 have been obtained from the example base based on a longest-match with the input sentence of Example 3.2.

(3.3) *Je doute qu'il · I do not think it is*

(3.4) *Je doute qu'il soit · I doubt whether that will be*

(3.5) *qu'il soit nécessaire de · not think it is necessary to*

(3.6) *nécessaire de lancer · necessary to start*

(3.7) *une enquête complète · a full investigation*

(3.8) *pour l'instant. · for the moment.*

Recombinators that are not capable of combining overlapping fragments are forced to combine the TL fragments of Examples 3.4, 3.6, 3.7 and 3.8, leading to the following translation: *I doubt whether that will be necessary to start a full investigation for the moment.* This translation is deficient; a better one would be *I doubt whether it is necessary to start a full investigation for the moment.* The recombinator developed by Brown et al. (2003), which considers overlaps, can make use of the TL fragments of Examples 3.3, 3.5, 3.6, 3.7 and 3.8, thereby producing a better translation: *I do not think it is necessary to start a full investigation for the moment.* The recombinator assigns translations produced from overlapping fragments a higher score than those produced from non-overlapping fragments. It does so by introducing a weight that measures the amount of overlap. The recombinator also makes use of a language model.

Language modeling is a technique adopted from speech recognition (Katz, 1987). Language models measure the likelihood of appearance of a sequence of words in a particular language. With respect to MT, they provide a means of estimating the fluency of an MT output. They have found wide use in SMT and have also been introduced to EBMT. Typically, the units considered are n-grams (sequences of contiguous words). In SMT, language models are used to form a translation initially (*hypothesis recombination*) or to rerank a set of already produced translations (*hypothesis reranking*). For the latter purpose, n-grams of higher order are usually applied. In EBMT, language models are traditionally used for hypothesis reranking. Bangalore et al. (2002) and Groves and Way (2005a) included language models in their EBMT systems in this way. Groves and Way (2005a) reported that doing so yielded an improvement over a traditional EBMT system. In contrast, the EBMT system of Brown et al. (2003) described earlier uses a language model for hypothesis recombination. This is presumably due to the fact that the system can be invoked as part of a larger MT architecture, which combines engines of multiple paradigms. Each of the engines partitions an input sentence according to its own scheme and outputs its own fragment translations. The translations are then gathered in a common lattice, and a language model is applied to combine them into an output sentence.

3.3 Compositionality in SMT

Wu (2005) posited that SMT started out as statistical, lexical and schema-based MT and has since undergone one major shift: towards compositional MT. The shift was caused mainly by the abandonment of word-based (Brown et al., 1990) in favour of phrase-based models. SMT systems also achieved compositionality by including generalized templates. In what follows, we describe how this was done. We then discuss the integration of syntactically motivated phrases as a special case of phrases in SMT. We then introduce the concept of hybrid SMT/EBMT systems. We show an example of a hybrid system in which the EBMT subsystem contributes syntactically motivated phrases to the overall system.

3.3.1 Generalized Templates

Few SMT approaches have made use of generalized templates so far. The most important one is that by Och and Ney (2004). Their "alignment template" approach is best known for laying the groundwork for phrase-based SMT. The part of the work that deals with generalized templates was initiated by Och and Weber (1998). They acknowledged that the procedure they proposed adds "an 'example-based' touch to the statistical approach" (Och and Weber, 1998, 986). Their alignment templates, also called "translation rules", are generalized phrase pairs. Generalization is performed by replacing the words of a phrase by the names of the word classes to which they belong. The word classes are derived from a bilingual clustering process. Hence, words that are members of the same class are contextually similar.[2] Och and Ney (2004) reported a significant improvement over a word-based SMT system when applying their system for French-to-English and Chinese-to-English translation. They offered that "it would be possible to employ parts-of-speech or semantic categories instead of the automatically trained word classes used here" (Och and Ney, 2004, 425).

3.3.2 Syntactically Motivated Phrases

Due to their restriction to word n-grams, language models cannot ensure the grammaticality of an entire output sentence. In particular, common language models are not capable of capturing the long-range dependencies typical of German syntax. Enabling them to do so would require increasing their n-gram order. This is not a viable option, as it is likely to lead to the problem of *data sparseness*: the longer the n-grams, the less likely they are to appear in the TL corpus over which the language model is built. Sparse instances of n-grams lead to unreliable probability estimates.

A way to capture the syntactic differences between an SL and a TL is to use syntactically motivated phrases. In a traditional phrase-based SMT system, the phrases are not linguistically motivated: like the units used in a language model, they are just n-grams of varying lengths. There has been much debate as to whether SMT can benefit from exchanging its concept of n-gram-based phrases with that of syntactic constituents as phrases. Koehn et al. (2003, 51) trained several phrase-based SMT systems on data between 10,000 and 320,000 sentence pairs and came to the conclusion that "requiring phrases to be syntactically motivated does not lead to better phrase pairs, but only to fewer phrase pairs, with the loss of a good amount of valuable knowledge". They argued that the phrase pair *there is* · *es gibt* is not a constituent pair and would thus not be recognized as a (syntactically motivated) phrase pair, although it should be.

3.4 Hybrid SMT/EBMT Systems

The chunking module of *Marclator* is capable of identifying *there is* and *es gibt* as chunks. In experiments performed prior to the release of *Marclator*, Groves and Way (2005b) investigated the effect of including Marker-based chunks in *Moses*'s predecessor *Pharaoh*. They found that doing so led to an improvement over a baseline phrase-based SMT system that relied solely on SMT phrase pairs. In the case of *there is* · *es gibt*, the pair is not only expected to be a Marker-based

[2] Note that a similar approach was pursued by Brown (2000), who automatically generated equivalence classes for his EBMT system, albeit with monolingual clustering techniques (as described in Section 2.4.2). The word classes described here can be generated with the *mkcls* program (http://www.fjoch.com/mkcls.html). The program takes as parameters the number of classes to be built and the number of optimization runs to be performed. The same word classes are output as a by-product of *Giza++* in the *.vcb.classes.cats* files.

chunk pair; the SMT system is likely to produce it as well. It is the fact that the pair is contained in both the EBMT 'chunk' table and the SMT phrase table that leads to an increased probability in the combined table and might cause the phrase pair to be preferred over other pairs with the same SL phrase.

3.4.1 *OpenMaTrEx*

The system just described is a precursor to the Machine Translation Using Examples (*MaTrEx*) architecture of which *Marclator* is a part. A reduced open-source version of *MaTrEx* was released as *OpenMaTrEx* (Dandapat et al., 2010).[3] We used *OpenMaTrEx* as a baseline system in our experiments. Despite its name referring to EBMT exclusively, the system also heavily relies on SMT techniques. In fact, it has been termed a *hybrid* SMT/EBMT system (Groves and Way, 2005a). MT subsystems can be combined to form either hybrid or *multi-engine* systems. In a hybrid system, the individual subsystems, which may or may not belong to different MT paradigms, serve to master different tasks of the overall translation process. In order to discuss to which extent this is true for *OpenMaTrEx*, a short overview of the system's training and translation process is given:

1 A 5-gram language model is trained with the *IRST LM* toolkit (Federico and Cettolo, 2007).[4] Modified Kneser-Ney smoothing (Chen and Goodman, 1996) is applied.

2 The SL and TL side of the training data are chunked according to the Marker Hypothesis. The resulting chunks are aligned. Both the chunking and the chunk alignment module are components of *Marclator*.

3 The system runs steps 1 to 5 of *Moses*. In step 1, the data is prepared. A vocabulary file, a sentence-aligned corpus file and a vocabulary class file are output for each of the two languages. Step 2 invokes *Giza++* for each language direction. In the actual alignment step, step 3, one of several possible heuristics is applied. The default heuristic, *grow-diag-final*, starts with the intersection of the bidirectional alignments and includes additional alignment points from the union of the two alignments if they connect at least one previously unaligned word. In step 4, two lexical translation tables that give maximum likelihood estimates for each SL–TL word correspondence are created. In step 5, phrase pairs are extracted based on the previously established alignments.

4 The EBMT chunk pairs obtained in step 2 of *OpenMaTrEx* are merged with the SMT phrase pairs obtained in the preceding step. For the EBMT chunk pairs to receive the same format

[3] http://www.openmatrex.org/
[4] http://hlt.fbk.eu/en/irstlm/

as the SMT phrase pairs, an attempt is made to assign a word alignment to every aligned chunk pair based on the output of *Giza++*. Chunk pairs for which this attempt fails are discarded. A lexicalized reordering model is also created at this stage based on the word alignment.

5 The remaining training steps of *Moses* are invoked. Optionally, an additional score is computed in which each phrase pair receives the value 0 if it is only an SMT phrase pair and 1 if it is an EBMT chunk pair (and possibly also an SMT phrase pair).[5]

6 The system's parameters are tuned using Minimum Error Rate Training (MERT) (Och, 2003).

7 The test set is translated with the *Moses* decoder.

It is evident from this description that the tasks performed by the EBMT and the SMT components within *OpenMaTrEx* are not mutually distinct: both the EBMT and the SMT subsystem perform phrase extraction and phrase alignment. In fact, these two tasks are the EBMT subsystem's only contribution. For the classification of the overall system as a hybrid system it is crucial that the EBMT subsystem does not produce any output by itself; the actual decoding is performed by the SMT subsystem only. In a multi-engine system, each subsystem performs all of the tasks and produces its own translations. The best output is then commonly chosen via a language model. *PanLite*, the MT architecture which we mentioned in Section 2.4 and to which *CMU-EBMT* belongs, is an example of a multi-engine system.

The case of *OpenMaTrEx* shows that it is sometimes difficult to tell where hybridity begins. Can a component that merely contributes syntactically motivated chunk pairs be considered a fully-fledged subsystem within a hybrid architecture? In the case of *OpenMaTrEx*, the EBMT component receives its justification as a subsystem from the fact that it is part of a stand-alone EBMT system (*Marclator*). This in turn raises the question of to what extent systems like *Marclator* can still be considered pure EBMT systems: *Marclator* relies on SMT techniques for word alignment and chunk alignment, as does *CMU-EBMT* for language modeling and parameter tuning. We propose to look at these two systems as EBMT systems that exemplify the shift of EBMT towards statistical MT.

3.4.2 Other Hybrid SMT/EBMT Systems

Liu et al. (2006) developed a hybrid SMT/EBMT system to translate from English to Chinese. Their system parses the SL side of each example and stores the example as a triple containing the

[5] A more direct way to weight the EBMT chunk pairs would be to multiply their count in the combined phrase table. However, Srivastava et al. (2009) found that this option was too weak: applying multiplication factors between 2 and 16 did not lead to any improvement over a baseline SMT system.

SL parse tree, the TL string and the links between the leaf nodes of the SL tree and the substrings of the TL string. The system first parses an input sentence and then searches the example base for matching subtrees. Each match receives a score, which, among other features, takes into account the semantic similarity between the head words of two corresponding nodes, determined from *WordNet*. The recombination module is a statistical generation model that considers the matching score, the translation probability of an SL–TL word and the language model probability of the TL string. The language model used is a trigram model. Liu et al. (2006) compared their system with *Pharaoh* and observed that it achieved slightly higher evaluation scores.

Smith and Clark (2009) built a hybrid SMT/EBMT architecture in which the EBMT subsystem translates only those parts of a sentence for which it is reasonably confident. The SMT subsystem then fills the gaps and produces the final overall translation. Smith and Clark (2009) used *Moses*, taking advantage of the fact that its decoder accepts do-not-translate segments embedded in XML tags. For their EBMT subsystem, they experimented with string-based and syntax-based matching. They explored three distance/similarity measures for string-based matching:

1 Levenshtein distance

2 a common substring measure in which each common substring contributes the square of its length to the similarity score

3 Levenshtein distance enhanced with the semantic knowledge contained in *WordNet*

For syntax-based matching, the EBMT subsystem analyzes the training set and test set syntactically with a dependency parser. Matches are obtained by retrieving the largest connected subtrees of the SL half of the example base that contain the same words and dependency types as the subtrees of the input sentence. The subtrees are then aligned word by word with their TL halves to extract the corresponding TL sequences. Smith and Clark (2009) found that the syntax-based matching technique performed best, followed by string-based matching based on Levenshtein distance. Inclusion of semantic similarity in the Levenshtein distance led only to a minor improvement. Overall, the hybrid system performed approximately 3 BLEU points worse than the baseline *Moses* system. Smith and Clark (2009, 7) observed that "*Moses* is not as good at ordering the EBMT-translated phrases correctly as it is with those it produces itself" but insisted that they found many cases in which their hybrid system performed better than *Moses*.

Another recent hybrid system is *Cunei* (Phillips and Brown, 2009).[6] We are inclined to call it a hybrid EBMT/SMT system because it contains a considerable amount of EBMT knowledge. Most importantly, it accesses the parallel corpus directly at runtime. When translating an input sentence, *Cunei* searches the SL half of the parallel corpus for phrases that match any segment of

[6] http://www.cunei.org/

the input sentence. Matching can take place not only on the basis of word forms, but also on the basis of lemmas, POS tags or statistical cluster labels. Each SL match is stored in a lattice along with a record of the input span it covers. The system then aligns each match with every phrase in the corresponding TL sentence of the parallel corpus. Each *instance* of a resulting TL phrase is scored with a log-linear model. The model includes functions for features that are common to all instances of the same TL phrase (e. g., the TL phrase's overall frequency in the corpus and a phrase penalty) as well as functions for features that are specific to the given instance. The instance-specific features are based on information obtained during matching and phrase alignment. The matching features take into account the similarity between the input sentence and the SL phrase that corresponds to the TL phrase under consideration as well as the context of the SL phrase. The phrase alignment features measure the likelihood that the SL phrase aligns to the TL phrase in the respective sentence pair. They are modeled as feature functions over the word alignment, which is computed offline.

The system generates an n-best list of phrase alignments. During decoding, all instances of a TL phrase that result in the same phrase pair are combined into a single log-linear model, which can be optimized just like in a traditional SMT system. Phillips and Brown (2009) reported their evaluation results for experiments on Finnish-to-English, German-to-English and French-to-English translation tasks: the results are on the order of those of the baseline *Moses* system. Phillips and Brown (2009) emphasized that their system exhibited better word selection, while *Moses* exhibited better word ordering.

In this chapter we discussed the difference between SMT and EBMT. We showed that EBMT underwent a shift towards statistical MT by incorporating probabilistic knowledge to compute similarities during matching, score alignments and rerank n-best lists of translations. We also introduced the concept of hybrid and multi-engine systems. We showed that *Marclator* is part of a hybrid SMT/EBMT system, while *CMU-EBMT* belongs to a multi-engine system. We problematized the classification of the two systems as EBMT systems, as they rely on a considerable amount of SMT knowledge. In the next chapter we show how the performance of MT systems can be evaluated. We introduce the most common metrics for automatic evaluation.

4 MT Evaluation

We mentioned in our introductory chapter that when a CBMT system is built, a small part of the parallel corpus is reserved for measuring the system's performance. The process of determining the quality of an MT system's output is known as MT *evaluation*. The output to be evaluated is commonly called the *candidate translation* or *hypothesis*. Evaluation can be performed either manually (human evaluation) or automatically. In automatic evaluation, the MT system's output is compared with one or multiple *reference translations*. Common variables analyzed in human evaluation are fluency and adequacy (White et al., 1993). Fluency denotes the well-formedness of an output, while adequacy refers to the extent to which the semantic content of a candidate translation matches with that of a reference translation. Human evaluation is subjective and time-consuming. In contrast, automatic evaluation is guaranteed to be objective. Furthermore, it is cheap and fast to apply. Thus, what is desired from the MT development point of view is an automatic evaluation metric that comes as close as possible to human judgement, while still maintaining objectivity.

Owczarzak et al. (2007) pointed out that at the time of their writing there was no automatic evaluation metric available that correlated highly with human judgements on both fluency and adequacy.[1] Nevertheless, it is possible to establish a list of properties of automatic evaluation metrics that have led to high correlations with human judgements. In what follows, we present such a list. Note that we do not posit that an ideal evaluation metric should include all of these features.

1 The metric can take into account the difference in length between the candidate and the reference translation, and it can penalize large discrepancies.

2 It can penalize the omission of certain words more severely than the omission of others. In particular, the omission of a content word receives a greater penalty than that of a function word.

3 The metric can allow for semantic variation.

4 It can allow for morphological variation.

[1] The correlation is commonly measured with Pearson's product-moment correlation coefficient, known as Pearson's r correlation.

5 It can check for grammatical well-formedness.

6 It can allow for grammatical variation.

7 It can provide meaningful results both at the sentence level and the document level.

8 It can allow for multiple reference translations to be provided, while still yielding meaningful results when provided with only a single reference translation. In particular, evaluation results obtained with different numbers of reference translations can be compared.

9 The metric has an intuitive interpretation.

In what follows, we introduce the most common automatic MT evaluation metrics. The above features serve as guidelines for us to discuss these metrics. We divide the metrics into three groups: distance-based, n-gram-based and syntax-based metrics.

4.1 Distance-Based Evaluation Metrics

The first metrics for automatic evaluation that were introduced in the early 1990s are distance-based metrics, where distance means edit distance (Estrella, 2008). The metrics compute the minimum number of edit operations (substitutions, insertions and deletions) necessary to transform a candidate translation into a reference translation. For example, Word Error Rate (WER) (Tillmann et al., 1997), an evaluation metric from speech recognition, calculates edit distance based on tokens. The final score is computed by dividing the sum of all necessary edit operations by the number of tokens in the candidate translation. The metric has two important shortcomings. Firstly, all tokens receive the same weight; hence, there is no distinction between (say) a punctuation mark and a content word. Secondly, since candidate–reference token pairs are compared sequentially, the metric does not allow for variation in word order. This can be detrimental for a language with relatively free word order, like German. In order to overcome this deficiency, the Position-Independent Error Rate (PER) (Nießen et al., 2000) was introduced. PER treats translations as bags of words and thus computes edit distance irrespective of position. Both WER and PER have a variant that allows for comparison with multiple reference translations: mWER and mPER (Nießen et al., 2000).

A more recent distance-based evaluation metric is the Translation Edit Rate (TER) (Snover et al., 2006). It differs from WER in that it allows for shifts of tokens/phrases in addition to the basic edit operations. A drawback unique to this metric is that neither the length of the token sequences that are shifted nor the distance across which they are shifted are taken into consideration. TER also inherits the shortcomings of WER: i. e., punctuation marks are treated as regular tokens, and all operations have a uniform cost of 1. In addition, case corrections (lowercase to uppercase or vice versa) count as regular edit operations.

The drawbacks of TER led to the introduction of two adapted versions: the Human-Targeted Translation Edit Rate (HTER) and TER-Plus (TERp) (Snover et al., 2009). HTER takes into account semantics by first allowing a human to edit a candidate translation in such a way that it obtains the same meaning as the reference translation. This edited translation is then used to compute TER in the familiar way. HTER has shown to correlate better with human judgements on single sentences than BLEU and METEOR (Snover et al., 2009), two n-gram-based metrics that are discussed in the following section. TERp is a further enhancement of TER in which the cost of edit operations can be tuned. The metric also includes three additional operations: word stem matches, *WordNet* synonym matches and multiword matches using a table of scored paraphrases.

4.2 N-Gram-Based Evaluation Metrics

N-gram-based metrics are the most widely used automatic evaluation metrics. In their simplest form, they do not take semantics into account. Most of them only compute n-gram precision. N-gram precision refers to the number of correctly translated n-grams (the number of n-grams in the candidate translation that appear in the reference translation(s)) divided by the total number of n-grams in the candidate translation. N-gram recall is the ratio of the number of correctly translated n-grams to the total number of n-grams in the reference translation(s). N-gram recall has shown to correlate more strongly with human judgements than n-gram precision (Banerjee and Lavie, 2005). However, it is rarely computed in a metric; it is difficult to compute when multiple reference translations are considered simultaneously, as we will show in our discussion of BLEU.

The problem with both precision and recall is that they can be inflated so as to produce spuriously high values. For example, a candidate translation that contains only one word, say, *the*, will receive a precision of 1 as soon as one of the reference translations contains the word *the* also. Likewise, recall is 1 if there is one reference translation consisting of a single word that is also contained in the candidate translation. A solution to this problem is to combine the two measures. For example, the General Text Matcher (GTM) metric shown in equation 4.1 (Turian et al., 2003) uses the F measure, where P denotes precision and R recall.

$$F = \frac{2 \cdot P \cdot R}{P + R} \tag{4.1}$$

An alternative solution that has been suggested to circumvent the problem of spurious inflation is to modify the measure itself. This was the solution chosen in the Bilingual Evaluation Understudy (BLEU) metric (Papineni et al., 2002). BLEU is the most widely used of all the automatic MT evaluation metrics. It includes a modified n-gram precision score: modification consists of limiting the number of n-gram matches to the maximum number of occurrences of this n-gram in a single reference translation. In other words, for each n-gram in a candidate translation, its number of occurrences in each of the reference translations is determined. The maximum value is chosen and is divided by the total number of n-grams in the candidate translation. Modified n-gram precision is calculated separately for each n-gram order. The n-grams computed by BLEU usually range from 1 to 4. N-grams of higher order to some extent capture grammatical well-formedness. However, this does not mean that BLEU takes into account syntactic structure explicitly. It also does not allow for syntactic variation.

As a result of computing n-gram precision, BLEU automatically penalizes candidate translations that are longer than their reference translations. A related question remains: how to penalize candidate translations that are shorter than their reference translations. One way to do this is by considering recall. However, BLEU allows for multiple reference translations to be provided, which are likely to contain different translation variants for the same SL word or phrase. If recall were calculated on the basis of words, those candidate translations that contain all of the possible translations (instead of just one) would be rewarded the most. In order to prevent this effect, different translations of the same SL word would have to be recognized as synonymous. The synonyms would then have to be gathered to form a semantic concept, and recall would have to be computed on the basis of such concepts.

Papineni et al. (2002) argued that such computation is costly. They therefore introduced a brevity penalty score (*BP*) to penalize candidate translations that are shorter than the corresponding reference translations. *BP* is computed over the entire corpus.[2] It is defined as in equation 4.2, where c is the length of the candidate translation and r the length of the reference corpus.

$$BP = \begin{cases} 1 \text{ if } c > r \\ e^{(1 - \frac{r}{c})} \text{ if } c \leq r \end{cases} \tag{4.2}$$

The overall BLEU score is computed as the geometric mean of the modified n-gram precisions, p_n, multiplied by the exponential brevity penalty score, *BP*, as shown in equation 4.3. N is the maximum n-gram length and w_n a positive weight (the weights together sum up to one).

$$BLEU = BP \cdot exp(\sum_{n=1}^{N} w_n \, log \, p_n) \tag{4.3}$$

[2] Papineni et al. (2002) posited that computing it over individual sentences would lead to a penalty that is too severe.

Equation 4.3 shows that the BLEU score is zero if one of its factors is zero. BLEU is, therefore, not a suitable metric for evaluations at the sentence level. The metric has also been shown to penalize small differences in length between a candidate and a reference translation too severely. Furthermore, Callison-Burch et al. (2006, 251) concluded that it permits "a tremendous amount of variation". They pointed out that there are often "millions of variations on a hypothesis translation that receive the same Bleu score", stressing that "[a]s the number of identically scored variants goes up, the likelihood that they would all be judged equally plausible goes down" (Callison-Burch et al., 2006, 249ff.).

Some of the drawbacks of BLEU were tackled by the introduction of a metric closely related to it, NIST (Doddington, 2002). NIST applies the arithmetic rather than the geometric mean, thus providing a relief from the problem of zero-value scores. Furthermore, NIST includes a modified brevity penalty score that assigns less of a penalty to small differences in length between a candidate and a reference translation. Its major conceptual improvement is the introduction of n-gram weights: less frequent n-grams are assigned a higher weight than more frequent ones, as they are considered to be more informative. Like BLEU, NIST is insensitive to lexical and syntactic variation. A problem unique to NIST is that its score increases with the amount of text used for the evaluation. This means that the metric contains no upper bound, which makes comparisons of NIST scores obtained with different amounts of test data virtually impossible.

The Metric for Evaluation of Translation with Explicit Ordering (METEOR) (Banerjee and Lavie, 2005) matches unigrams only. Because of this alone, it would not qualify as an n-gram-based metric. However, the metric also calculates a penalty score that penalizes non-contiguous unigram matches. The penalty score is computed by searching for the longest common n-grams between a candidate and a reference translation: the longer the n-grams, the smaller the number of resulting chunks and the smaller the penalty.[3] Hence, METEOR is effectively an n-gram-based metric. Other than BLEU and NIST, it allows for lexical variation and, to a limited extent, morphological variation. Lexical variation is optional; if included, it is captured by considering synonym information obtained from *WordNet*. Morphological variation is captured by considering stemmed words. For each sentence of a candidate and a reference translation, METEOR first tries to establish an exact unigram matching. The unigrams that could not be matched are subsequently stemmed, and an attempt is made to match the stemmed unigrams. If lexical variation is included, the remaining unmatched unigrams are substituted by their synonyms, and another level of matching is attempted. Each matching is scored based on the formula shown in Equation 4.4.

$$score \;=\; f\text{-}measure \;\cdot\; (1 - penalty) \tag{4.4}$$

[3] The possible number of chunks ranges between 1 and the number of unigrams matched.

f-measure considers unigram precision and unigram recall. Calculating recall is possible because METEOR considers one reference translation at a time:[4] the score of Equation 4.4 is calculated for each reference translation and each matching (exact, morphological and semantic matching) in turn. The highest value is then chosen as the score for the sentence under consideration. This score is also used to calculate the final system score, which ranges between 0 and 1.

4.3 Syntax-Based Evaluation Metrics

We showed that some of the n-gram-based metrics introduced in the previous section include an implicit analysis of grammatical well-formedness. However, for a metric to arrive at a judgement regarding the grammaticality of an entire sentence and, more importantly, to allow for grammatical variation, full syntactic parsing is required. The demand for syntax-based evaluation metrics arose with the integration of syntactic knowledge in the MT systems themselves. Liu and Gildea (2005) came to the conclusion that BLEU and other n-gram-based metrics were not capable of adequately rewarding the output of syntax-based MT systems. They pointed out that a sentence can achieve a high BLEU score even if it is not a grammatically correct or complete sentence, e. g., if it is missing a verb. They were the first to investigate the use of syntactic information in MT evaluation: they experimented with constituency structures and unlabelled dependency structures and concluded that "adding syntactic information to the evaluation metric improves both sentence-level and corpus-level correlation with human judgements" (Liu and Gildea, 2005, 25).

Based on these observations, Owczarzak et al. (2007) developed an automatic evaluation metric that makes use of labelled dependency structures, as opposed to the unlabelled structures used by Liu and Gildea (2005). Dependency structure labels indicate the grammatical relation between a head and its modifier. Owczarzak et al. (2007) posited that taking them into account makes it possible to reward partial matches, e. g., if a lexical item is assigned the correct grammatical relation but an incorrect partner. Using dependency structures is usually motivated by the fact that these structures abstract over constituent order. The metric of Owczarzak et al. (2007) derives flat sets of dependency triples from LFG f-structures. It computes two series of f-scores: one for all dependency structures and one for only those structures whose paths end in a predicate–value pair denoting a grammatical relation.[5] The metric also includes an option whereby *WordNet* synonyms can be added so as to capture lexical variation. Owczarzak et al. (2007) observed that their metric correlates strongly with human judgements on fluency.

He and Way (2009) extended the metric of Owczarzak et al. (2007) by introducing parameter tuning of dependency labels. They proposed the assumption that not all dependency labels have

[4] Remember that the problem with calculating recall in BLEU was that BLEU considers all of the reference translations at the same time.

[5] Examples of predicate–value pairs that do not end in a grammatical relation are *focus* or *person*. They are atomic.

the same importance but instead should be weighted,[6] and that machine learning techniques can be applied to search for the appropriate weights. They used regression SVMs to perform this task and achieved high correlations with human judgements on fluency.

4.4 Evaluation Metrics and MT Paradigms

The choice of evaluation metric strongly depends on the type of system(s) whose output is to be evaluated. BLEU has been shown to be inappropriate for evaluations of translation tasks that involve systems of different paradigms, especially when comparing phrase-based SMT systems with "systems that do not employ similar n-gram-based approaches" (Callison-Burch et al., 2006, 255). SMT systems that include tuning via Minimum Error Rate Training (MERT) (Och, 2003) are virtually optimized towards BLEU. It is commonly accepted that systems should be tuned to the same metric that is used for evaluating their output. For a long time, BLEU was the only metric available for MERT. This changed with the introduction of Z-MERT (Zaidan, 2009), a software tool which, in principle, is capable of supporting any evaluation metric.[7] Cer et al. (2010) investigated the use of metrics other than BLEU for tuning and the impact of tuning a system to one metric and evaluating it with another. They built two SMT systems, a Chinese-to-English and an Arabic-to-English one, and applied BLEU (1-grams up to 5-grams), TER, NIST, TERp, WER and METEOR for tuning and evaluation. They observed that systems which were tuned to BLEU and NIST produced better results when evaluated with TERp, WER and METEOR than vice versa. Therefore, they concluded that BLEU and NIST "are still the best general choice for training model parameters" (Cer et al., 2010, 562). They also observed that when using BLEU to tune and evaluate a system, the best results were not achieved with identical n-gram orders: tuning to 3-gram BLEU and evaluating with 4-gram BLEU yielded the best results for their Chinese-to-English system.

In this chapter we presented the most common metrics for automatic MT evaluation. We demonstrated that these metrics suffer from different shortcomings: e. g., no distinction is made between words and punctuation symbols (WER), no syntactic variation is permitted (BLEU, NIST, METEOR), sentence-level scores can be zero (BLEU), and scores obtained with different sizes of test data cannot be compared (NIST). In addition, we showed that SMT systems that apply MERT are optimized towards BLEU. In the next chapter we present our experiments centering around generalized templates in EBMT. We first show the statistics of our experimental data set and discuss its characteristics. We then present our own approach to EBMT and the results of our experiments with this approach. Despite its shortcomings and its bias in favour of SMT, we used BLEU to evaluate our EBMT systems, as the metric is widely used in the MT

[6] A similar assumption was made by Doddington (2002) for n-grams when developing NIST.
[7] http://www.cs.jhu.edu/~ozaidan/zmert/. Z-MERT is available in *Moses* through the Perl script *zmert-moses.pl*.

community. We also applied NIST and METEOR. Since the metric of He and Way (2009) is only available for evaluation of translations that have English as the TL, we could not use it to evaluate our English-to-German experiments. The baseline systems we used that apply MERT (i. e., *Moses* and *OpenMaTrEx*) were tuned to BLEU.

5 Experiments

5.1 Data

In the introductory chapter we presented the most common parallel corpora for use in MT systems. For our experiments in English-to-German translation, we did not rely on one of these corpora. Instead, our experimental data set consisted of subtitles that were kindly provided to us by *SDI Media*,[1] a commercial subtitling company. The subtitles had been produced and translated manually for different U.S. television series. In what follows, we give an overview of the size of our data set and the general characteristics of subtitles. We also discuss the suitability of subtitles as data for EBMT and SMT systems.

5.1.1 Corpus Profile

Our corpus contained 1,051,056 subtitles. Table 5.1 shows the statistics of word form tokens, word form types and lemma types. The relation between subtitles and sentences was of one of the following three kinds:

- one-to-one, with one subtitle corresponding to one sentence, e. g.:
  Here's a toast to the happy couple.
- one-to-many, with one subtitle containing multiple sentences, e. g.:
  Don't get too close. He may ask you to squeeze something.
- many-to-one, with a sentence extending over more than one subtitle, e. g.:
  You think anyone ever looked back on their life and said, "Gee, I wish...
  ...I hadn't taken a year off to be with the baby"?

To improve the expected translation quality, we ensured that each subtitle contained at least one full sentence. For this, we merged many-to-one correspondences so that sentences that had initially extended over more than one subtitle were now contained in a single subtitle on both the SL and the TL side. In the case of one-to-many correspondences, the sentences contained in a subtitle were sometimes uttered by different speakers. This was indicated by leading dashes (e. g., –

[1] http://www.sdimedia.com/

Yeah, who says I'll be in the mood? – You're always in the mood.). We observed that the order of the speaker turns in the German translations was consistent with that of the English originals. Therefore, we split the subtitles at turn-taking markers (leading dashes) to obtain a more fine-grained parallel text. We did not perform sentence splitting after that. Hence, our basic translation units were subtitles.

	English		German	
	+ punct.	– punct.	+ punct.	– punct.
Subtitles	1,051,056		1,051,056	
Tokens	10,058,473	8,063,922	9,003,672	7,135,631
Types	103,654	103,628	169,939	169,923
Lemma types	76,803	76,783	125,627	125,612

Table 5.1: Subtitle corpus: profile

The merging and turn-splitting procedure resulted in 1,133,063 subtitle pairs which consisted of on average 8.9 tokens (after splitting punctuation marks) for English and 7.9 for German. These are short translation units. The average length of units (sentences) in the German side of the Europarl corpus is 26 tokens (Sennrich, 2009).[2] Way and Gough (2005) used an English–French parallel corpus with an average sentence length of 13.1 for English and 15.2 for French. We also found that 1.75 % of the subtitle pairs in our corpus occurred more than once. The mean number of occurrences of a subtitle pair was 1.07; the standard deviation was 3.65, which suggests that a number of subtitle pairs occurred with a very high frequency. In Section 5.4.1, we discuss how this influenced the performance of our EBMT systems.

5.1.2 Characteristics of Subtitles

Subtitles can serve two different purposes: they can either substitute or supplement the audio output of a video clip. In the former case, the viewers are usually hearing-impaired. Hence, the subtitles should not only render the on-screen speech but also descriptions of other sound events that are crucial to the story line, e. g., song lyrics or noises. The present work deals with subtitles of the second category: they are aimed at viewers who do not know the primary language of a video clip and who therefore need a translation. The viewers are confronted with three different input channels: a visual channel (on-screen image), an auditive channel (audio output) and a verbal channel (subtitle).[3] The subtitles are usually displayed on screen no longer than eight seconds

[2] Sennrich (2009) discarded sentences that were longer than 40 tokens, arriving at a mean sentence length of 20 tokens.
[3] Arguably, the on-screen presentation of subtitles is essentially visual, too, but the point to be emphasized here is that subtitles consist of linguistic units.

and are assigned a set amount of space (Volk, 2008). If the text of a screenplay passage exceeds the given spatial window, it has to be shortened. In fact, Cintas and Remael (2007) pointed out that it is normally the case with a subtitle to be a shortened version of a screenplay passage. In addition, it can be a revised version. Examples of revisions are (Cintas and Remael, 2007, 145–171):

- simplifying verbal periphrases, e. g., ***I'm gonna have*** *this place* ***fixed****.* → ***I'll fix*** *this place.*
- using shorter (near-)synonyms
- using simple instead of compound tenses, e. g., ***I have stopped*** *smoking exactly 134 days ago.* → ***I stopped*** *smoking exactly 134 days ago.*
- changing verbs and adjectives into nouns, e. g., *I don't want it to be too* ***transparent****.* → *I don't want* ***transparency****.*
- changing adjectives into adverbs, e. g., *I was in a* ***deep*** *sleep.* → *I slept* ***soundly****.*
- using pronouns and other deictics to replace nouns, noun phrases or other phrases, e. g., *I am a hairdresser.* ***The only thing*** *I know is* ***how to do hair****.* → *I'm a hairdresser.* ***It's all*** *I know.*

5.1.3 Annotations

Apart from the primary text, subtitles also contain special characters and tags conveying meta information. For example, our data contained expressions enclosed in *<i>...</i>* tag pairs. Such expressions (henceforth called *italicized expressions*) are used on two occasions:

1 to mark off-screen dialogue or instances of voice-over, e. g., a loudspeaker announcement in a shopping mall (*<i> Attention, Big Q shoppers. Another day, another dollar sale. </i>*) or the part of a telephone conversation that is uttered by a speaker who is not shown on screen

2 to emphasize an utterance. The emphasis either serves to underline the tone in which a statement was made or to distinguish an expression (e. g., a foreign language expression or the name of an institution) from its context (*Should we hum the theme from <i> Jeopardy </i> ?*)

In both of these cases, the usage of the *<i>...</i>* tag pair is semantically motivated. Thus, italicized expressions could be expected to be useful subsentential units for an EBMT system. The results of an experiment exploring this idea are shown in Section 5.3 of this chapter.

Depending on their length, subtitles are segmented into several lines, usually into one or two.[4] A line break tag indicates the beginning of a new line on screen. Ideally, its position is syntactically motivated. Cintas and Remael (2007) listed several placement rules: line breaks should be inserted before a coordinating or subordinating conjunction. They should not separate an adjective from a noun which it modifies or an article from a noun. A preposition should not be separated from a noun phrase with which it forms a prepositional phrase, and auxiliary verbs should not be separated from their main verbs. The five most frequent POS tags[5] following a line break in the SL half of our subtitle corpus were:

1 *:* · sentence-internal punctuation

2 *IN* · preposition, subordinating conjunction

3 *PP* · personal pronoun

4 *DT* · determiner

5 *CC* · coordinating conjunction

This shows that many of the rules introduced by Cintas and Remael (2007) were observed in our subtitles: for example, line breaks were placed before the preposition (*IN*) in a prepositional phrase or before the determiner (*DT*) in a noun phrase. The tags appearing most frequently after the line break tag in the German subtitles largely corresponded with the English originals:

1 *$(* · sentence-internal punctuation (corresponding to the English tag *:*)

2 *APPR* · left-marking preposition (corresponding to the English tag *IN*)

3 *PPER* · personal pronoun, irreflexive (corresponding to the English tag *PP*)

4 *NN* · noun (no corresponding English tag)

5 *ART* · definite or indefinite article (corresponding to the English tag *DT*)

Hence, attempts were made to split the SL and TL subtitles in the same places. This raises the possibility of making use of line break information in the chunking process, i. e., opening a new chunk at a line break tag. When applied in isolation, such an approach would lead to chunks that are too long. It would, therefore, have to be combined with an alternative chunking scheme. We did not perform any experiments that explored this idea. Therefore, we just discarded the line break tags. In an industrial setting, the translation task would include recording their positions

[4] The maximum number of lines in our data was four.

[5] We tagged our data with the *TreeTagger* (Schmid, 1995). The English tags used by this tagger largely follow the Penn Treebank tagset (Santorini, 1990). The German tags rely on the Stuttgart-Tübingen-Tagset (STTS) (Schiller et al., 1999).

in the input sentences and inserting them in the translation hypotheses during a postprocessing step.

5.1.4 Suitability of Subtitles for EBMT and SMT

We showed that due to their co-existence with other input channels as well as due to the spatial and temporal constraints which they underlie, subtitles have to be composed as "short textual units with little internal complexity" (Volk, 2008, 208). Hardmeier (2008) posited that both of these features are advantageous to CBMT systems. Short translation units are particularly beneficial for EBMT systems: they increase the likelihood of a complete match between an input sentence and a sentence of the example base. Hardmeier (2008, 15) listed as further characteristics of subtitles the presence of "stutterings, word repetitions or renderings of non-standard pronunciations". However, he emphasized that these phenomena are not expected to be highly frequent, as they stand in direct opposition to the principles of unobtrusiveness and readability that guide the creation of subtitles. In any case, MT systems that do not incorporate explicit syntactic knowledge, like standard phrase-based SMT systems, are not expected to have problems with stutterings, word repetitions and non-standard expressions. Such phenomena do not pose a problem for an EBMT system like *Marclator*, either, since the system does not perform full syntactic parsing.[6] For the same reason, *Marclator* is capable of dealing with fragmentary input. Example 5.1 shows a German input sequence consisting of an elliptic sentence followed by a full sentence. The chunked sequence is also shown along with it.

(5.1) *Nein, es ist nur… Es braucht eine Weile, sich dran zu gewöhnen.* → *Nein | es ist nur | Es braucht | eine Weile | sich dran | zu gewöhnen*

We can see from the chunked sequence that the second occurrence of the German pronoun *es* triggered the opening of a new chunk after the ellipsis (*es ist nur…)*.

Subtitles represent suitable data for EBMT systems for yet another reason: we mentioned earlier that EBMT systems are capable of directly exploiting a high similarity between a training set and a test set. Controlled language EBMT (Gough and Way, 2003) takes advantage of this fact. Although subtitles are not instances of controlled language, they have also undergone a normalization process, which renders them, if nothing else, at least more syntactically similar among each other than the screenplays from which they are derived.

[6] On a related note, we discovered that the token *'ne*, a substandard version of the German indefinite article *eine*, appeared frequently in our data. We therefore included it as a Marker word in the German Marker file that we developed (see Section 2.3).

5.2 Our Approach

For our experiments, we divided the subtitle data into a training set consisting of 1,130,717 subtitles and a test set and development set consisting of 1173 subtitles each. We performed a second set of experiments using only half of the training data (565,358 subtitles) to see the effect of data size on MT quality.

5.2.1 Generalized Templates in EBMT

Our approach to EBMT consisted of combining the two systems described in Sections 2.3.2 and 2.4.2. Recall that these are the generalized template extensions to the EBMT systems *Marclator* (DCU) and *CMU-EBMT* (CMU). Combining the systems meant building a new system that applies both the DCU and the CMU generalization scheme. Our goal was to see whether our combined system could outperform the two individual systems. For this, we ran an experiment with the combined system as well as one with each individual system. We (re-)implemented the three approaches on top of *Marclator*: we included the word alignment, Marker-based chunking and chunk alignment module of *Marclator*. These components were described in Section 2.3. We also used the *Marclator* recombinator and adjusted it separately for each of the three systems so as to make it capable of dealing with the particular generalization scheme.

Apart from the three generalized EBMT systems, we built a fourth one in which we included italicized expressions as subsentential units. The motivation behind this approach was described in Section 5.1.3. For this experiment, we used the purely lexical *Marclator* system.[7] In summary, we built four systems: *Marclator* with DCU generalized templates, *Marclator* with CMU generalized templates, *Marclator* with DCU & CMU generalized templates and *Marclator* with italicized expressions. In what follows, we describe these systems. For each of the three generalized systems, we end the description with a translation example. In Section 5.3, we present the results of our experiments in running the four systems. In Section 5.4, we discuss the results and provide further analyses of our systems' behaviours.

System 1: *Marclator* with DCU generalized templates

This system includes the generalized template extension to *Marclator* that was described in Section 2.3.2. Recall that *Marclator* is based on Marker words, which are function words. Hence, the extension generalizes over the Marker words at the beginning of Marker-based chunks. We re-implemented it by using the *Marclator* components mentioned above (word alignment, Marker-based chunking and chunk alignment module) and adding a module that generalizes the aligned

[7] The system is listed along with the generalized EBMT systems in this section for the sake of simplicity.

SL–TL chunk pairs. We also extended the *Marclator* recombination module: in its original form, the recombination module checks for the presence of sentences and word form chunks[8] before reverting to word-by-word translation. We added an additional matching step to follow the chunk matching: in this step, the system replaces the Marker word at the beginning of a chunk by its corresponding Marker tag and searches for the resulting generalized chunk in the example base. Where this attempt fails, the system reverts to word-by-word translation.

The only difference remaining to the approach described in Section 2.3.2 is that the system of Gough and Way (2004) outputs all possible hypotheses for an input sentence, while the *Marclator* recombinator only outputs the one-best hypothesis. This means that once our system has established a generalized chunk match with the SL side of the example base and has extracted the corresponding TL generalized chunk, it has to make a decision as to which Marker word to insert for the Marker tag. For this, it identifies the SL Marker word underlying the SL generalized chunk that was matched. It gathers the word alignment links that contain the SL Marker word and chooses the alignment with the highest frequency, provided that the resulting TL word is also a Marker word. For example, assume that in an English-to-German translation task, the SL chunk *on the bus* could not be found in the example base. Hence, the system generalizes it, producing the SL generalized chunk *<PREP> the bus*.[9] Assume that the system subsequently finds the following generalized chunk pair in the example base: *<PREP> the bus* · *<PREP> dem bus*. The system gathers all of the word alignments that contain the underlying SL Marker word *on*. It chooses the TL word that belongs to the alignment with the highest frequency (e. g., *auf* from an alignment *on* · *auf*) and checks whether it is a TL Marker word. If it is, the system replaces the TL Marker tag *<PREP>* with that word. If the TL word retrieved is not a Marker word, the system reverts to the alignment with the second highest frequency, and so on. If none of these attempts is successful, the system reverts to word-by-word translation.

Henceforth, this system is called *System 1*. Figure 5.1, adapted from Armstrong et al. (2006), visualizes its training and translation process. Shown in rectangles are the resources that are required: training data, test data (input), Marker files, Marker-based chunks, aligned sentences, aligned chunks, aligned generalized chunks and aligned words. Shown in ovals are the modules that constitute our system, i. e., the word alignment, chunking, chunk alignment, chunk generalization and recombination module. The two components which we added to *Marclator*, i. e., the chunk generalization and the extended recombination module, are displayed in grey. The numbers attached to the arrows (I to IV) specify the matching order (from sentences to chunks to generalized chunks to words).

[8] We subsequently refer to word form chunks (as opposed to generalized chunks) simply as chunks.

[9] We adopt the convention of writing the DCU Marker tags in uppercase letters. The CMU equivalence class tags will be written in lowercase letters.

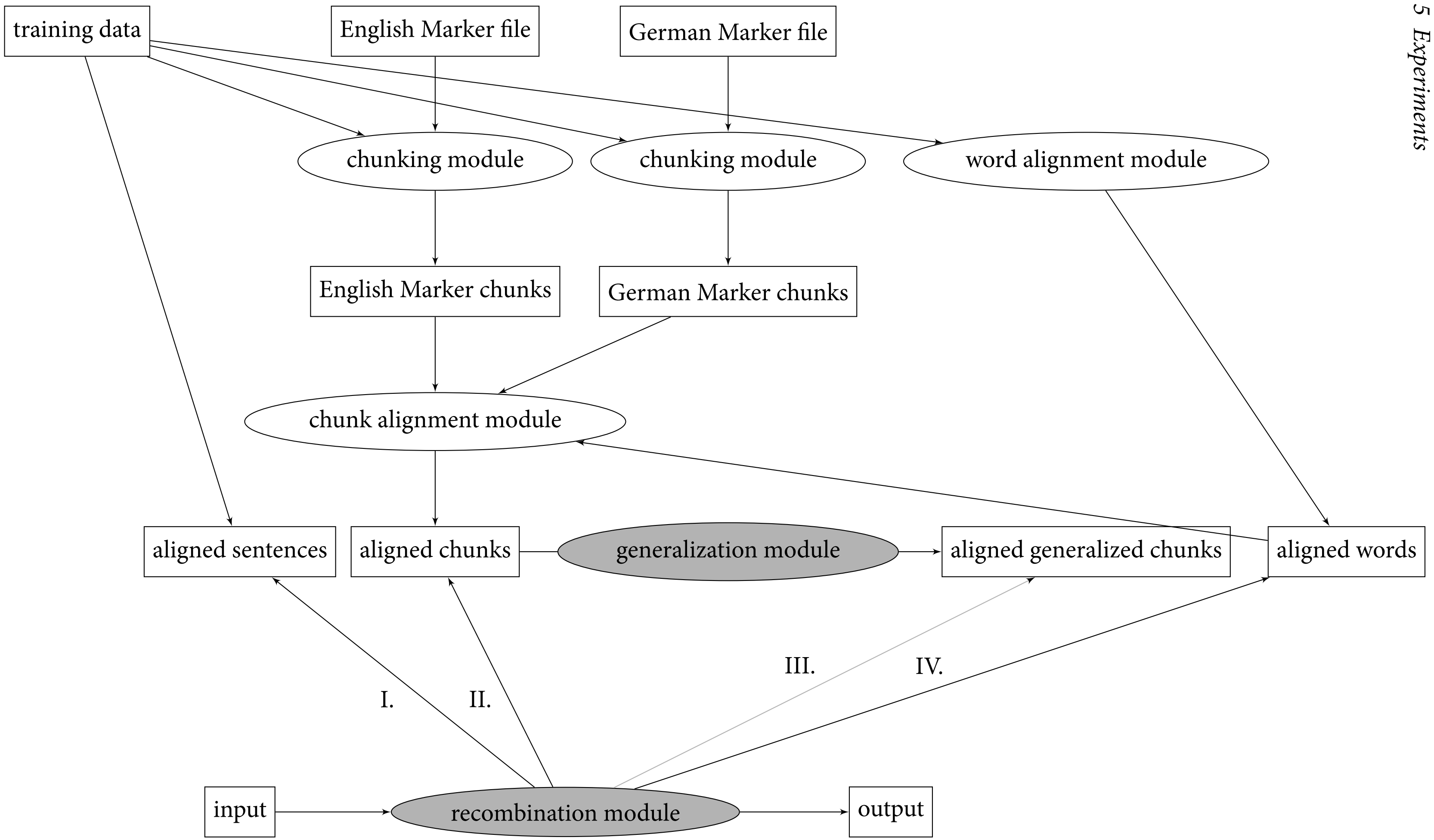


Figure 5.1: *System 1*: training and translation process

In what follows, we demonstrate the translation process of *System 1* shown in Figure 5.1 by means of a sample sentence from the test set. Assume that the following English sentence is to be translated into German: *and i guess i 've finally got to accept that he is gone from my life . System 1* first checks whether the sentence is contained in the example base as a whole. If this is not the case, it chunks the sentence. The chunked sequence is shown in Example 5.2.

(5.2) *and i guess | i 've finally got | to accept | that he is gone | from my life .*

For each of the chunks in Example 5.2, the system checks whether it is part of the example base. This is the case for the first and the third chunk (*and i guess* and *to accept*). For both of these chunks, the system retrieves the corresponding TL chunks from the chunk alignments; they are *und ich glaube* (for *and i guess*) and *dass* (for *to accept*).[10]

The second chunk (*i 've finally got*) is not contained in the example base. Therefore, the system generalizes it, producing *<PERS_PRON> 've finally got*. For this generalized chunk, it finds a match in the example base. The corresponding TL generalized chunk is: *<PERS_PRON> haben*. The system subsequently searches for a German translation for the SL Marker word *i* (underlying the SL Marker tag *<PERS_PRON>*) in the word alignments. Assuming that it finds *ich*, it produces the TL chunk *ich haben*. The fourth chunk (*that he is gone*) is also not contained in the example base, and neither is its generalized variant. Hence, the system translates the chunk word by word: *dass er ist weg*. The last chunk (*from my life*) produces a generalized chunk match, the generalized chunk being *<PREP> my life*. The system retrieves the corresponding TL generalized chunk: *<PREP> meinem leben*. From this, it produces the TL chunk *von meinem leben*. Together, the above steps lead to the following sentence translation: *und ich glaube ich haben dass er ist weg von meinem leben .* Note that this translation is deficient. We discuss the problems inherent in the approach of *System 1* in Section 5.4.

System 2: *Marclator* with CMU generalized templates

This system incorporates the CMU semantic and syntactic equivalence classes described in Section 2.4.2. Of the 81 classes for the language pair English–German that were provided to us by the developer of the *CMU-EBMT* extension, the majority are semantic classes. The classes contain a total of 5545 replacement rules. Recall that a replacement rule specifies an equivalence class tag and a (lexical or generalized) SL–TL pair whose two halves may be replaced by the tag; e. g., the rule *<time-s> : minute · minute* contains a lexical SL–TL pair (*minute · minute*), and the rule *<frequency> : every <time-s> · jede <time-s>* contains a generalized SL–TL pair (*every <time-s> · jede <time-s>*).

[10] Note that *dass* (*that*) is not a good translation of *to accept*.

Like *System 1*, *System 2* has *Marclator* at its core, which means that it relies on the word alignment, chunking, chunk alignment and (extended) recombination module of *Marclator*. We implemented an additional module that generalizes Marker-based chunk pairs on the basis of the CMU generalized templates. In the training data, our system generalizes symmetrically: it only replaces an equivalence class member in the SL chunk if a member of the same class appears in the corresponding TL chunk. For example, assume that the two English–German replacement rules displayed in Example 5.3 exist.

(5.3) *<time-s> : minute · minute*
<frequency> : every <time-s> · jede <time-s>

In the chunk pair *appreciate every minute · jede minute genießen*, both the SL and the TL chunk contain a member of the equivalence class *<time-s>* (*minute* and *minute*). Hence, the system generalizes the chunk pair to *appreciate every <time-s> · jede <time-s> genießen*. From there, another replacement rule applies, as both the SL and the TL chunk contain a member of the class *<frequency>* (*every <time-s>* and *jede <time-s>*). The final generalized chunk pair is *appreciate <frequency> · <frequency> genießen*.

At runtime, the system generalizes an SL chunk by replacing a class member with the corresponding class tag one at a time and storing the intermediate results. Consider the English chunk *mr. benjamin meyers* along with the (monolingual) replacement rules listed in Example 5.4.

(5.4) *<fname-m> · benjamin*
<honorific> · mr.
<person-m> · <honorific> <fname-m>

The generalized chunks produced from these rules are shown in Example 5.5.

(5.5) *mr. <fname-m> meyers*
<honorific> benjamin meyers
<honorific> <fname-m> meyers
<person-m> meyers

To match the generalized chunks with the example base, the system starts with the most general chunk (*<person-m> meyers* in Example 5.5) and checks whether it is part of the example base. If it is, the corresponding TL generalized chunk with the highest alignment score is identified. The TL word form chunk is produced by applying the rules that were stored during the generalization of the SL chunk in reverse. If the most general chunk is not found in the example base, the system reverts to the second most general chunk, and so on. If none of the generalized chunks are found in the example base, the system reverts to word-by-word translation.

The resulting system is called *System 2*. As an example for its translation process, assume that the following English sentence is to be translated into German: *maude had an affair with tom jones ? System 2* first checks whether the sentence is contained in the example base as a whole. If this is not the case, it chunks the sentence. The chunked sequence is shown in Example 5.6.

(5.6) *maude | had an affair | with tom jones ?*

For each of the chunks in Example 5.6, the system checks whether it is part of the example base. This is the case for the first and the second chunk (*maude* and *had an affair*). The system obtains the corresponding TL chunks from the chunk alignments: *maude* and *eine affäre*.[11] The third chunk (*with tom jones ?*) cannot be found in the example base. Therefore, the system generalizes it to *with <firstname-m> <lastname> ?* and from there to: *with <fullname> ?*. The corresponding TL generalized chunk is: *mit <fullname>*. By applying the replacement rules in reverse, the system obtains the chunk *mit tom jones*. Together, the above steps yield the following final translation: *maude eine affäre mit tom jones*.

System 3: *Marclator* with DCU and CMU generalized templates

This is the system that combines Systems *1* and *2*. Accordingly, it generalizes over DCU Marker words as well as CMU semantic and syntactic equivalence classes. Like Systems *1* and *2*, the system has *Marclator* at its core. This makes it possible to directly compare the effectiveness of the generalization schemes. However, the DCU and the CMU generalization scheme are not mutually exclusive. There are a number of overlaps, i. e., the CMU classes contain 50 words that are also Marker words for English (e. g., *after*, *and*, *before*), and 19 for German (e. g., *aber*, *allen*, *er*). We prompted the system to generalize over the Marker words first, thereby implicitly giving preference to the DCU scheme in case of overlaps.

Henceforth, this system is called *System 3*. To see how the two generalization schemes interact, consider the following example: assume that *System 3* is presented with the English sentence *i 'll go see if she 's awake* to translate into German. If the sentence is not part of the example base as a whole, the system chunks it: *i 'll go see | if she 's awake*. It finds the first chunk (*i 'll go see*) in the example base and obtains the following translation for it: *ich sehe mal*. The second chunk (*if she 's awake*) is not part of the example base. Hence, the system generalizes it according to the DCU scheme first, producing the intermediate generalized chunk *<SUB_C> she 's awake*. From there, it applies the CMU generalization scheme. *she* is a member of the equivalence class *<perspro>*.[12] Hence, the system produces the generalized chunk *<SUB_C> <perspro> 's awake*.

[11] Note that the second chunk does not contain a translation of the verb *had* that is part of the SL chunk.

[12] This is a case of an overlap between the DCU and the CMU generalization scheme: *she* is also a DCU Marker word. However, it does not appear in initial position in the chunk at hand.

This generalized chunk is found in the example base, where it is aligned with the TL generalized chunk *<SUB_C> <perspro> wach*. The system instantiates *<perspro>* with the TL side of the replacement rule *<perspro> : she · sie*, i. e., with *sie* . To obtain a Marker word for *<SUB_C>*, it invokes the familiar Marker tag instantiation process. This leads to the Marker word *wenn* and, hence, to the following final translation: *ich sehe mal wenn sie ist wach* .

System 4: *Marclator* with italicized expressions

The motivation behind extending the example base with italicized expressions was described in Section 5.1: such expressions are manually annotated and semantically motivated. We added italicized expression pairs that were longer than one token to the set of chunk pairs in the purely lexical *Marclator* system. Note that only pairs whose SL halves started with a Marker word were actual candidates for matching. Since we assumed that the italicized expression pairs were of high quality (due to their manual composition), we assigned them a score greater than that of any other pair in the chunk alignment. The resulting system is called *System 4*.

Baselines

We established three baselines: *Marclator*, *OpenMaTrEx* and *Moses*. The *Marclator* baseline was the purely lexical system described in Section 2.3. For the *Moses* baseline, we used the default system included in *OpenMaTrEx*. The system uses a 5-gram language model and modified Kneser-Ney smoothing. Training is performed according to the default options and thus includes tuning via MERT. In addition, a lexicalized reordering model (*msd-bidirectional-fe*) is learnt. The *OpenMaTrEx* baseline system makes use of EBMT chunk pairs from *Marclator* and SMT phrase pairs from *Moses*. We used the default configuration described in Section 3.4.1, which includes a 5-gram language model with modified Kneser-Ney smoothing and tuning via MERT. We included the optional binary feature that records whether a phrase pair is an EBMT chunk pair or not. To train the language models for *Moses* and *OpenMaTrEx*, we used the TL side of the training data.

5.2.2 Generalized Templates in SMT

We also performed an experiment with generalized templates in SMT. We used the DCU generalization scheme. Our goal was to obtain generalized sentences from *Moses* which we could then use as input for a module that context-sensitively instantiates Marker tags. Recall that the Marker tag instantiation process in Systems *1* and *3* is context-insensitive: for a given TL Marker tag, the systems always insert the translation of the corresponding SL Marker word which has the highest frequency in the set of word alignments. The approach proposed here is further different from the one pursued in Systems *1* and *3* in that here, Marker tags are instantiated after (as opposed to during) decoding.

We used half of the data for our experiment. We ran the Marker-based chunker over the training as well as the test data and replaced each word at the beginning of a chunk with its corresponding Marker tag. We then discarded the chunk boundaries, so that the result of the generalization process were sentences rather than chunks. We trained two language models, one on the TL half of the original (purely lexical) training set, and the other on the TL half of the generalized training set. We ran the *Moses* decoder on the generalized test set and applied the generalized language model. The result were generalized TL sentences like the German sentence displayed in Example 5.7.

(5.7) *und er umarmte <PERS_PRON> und er habe ihn geküsst <COORD_C> er sagte ihm , wie stolz <PERS_PRON> war mit ihm .*

We then initiated a *post-hoc* instantiation process. The aim was to choose the best Marker word to be inserted in the place of a given Marker tag with the help of the purely lexical language model. For each generalized output sentence like the one in Example 5.7, this process consists of the following six steps:

1 Get all Marker tags contained in the output sentence
e. g., for the sentence in Example 5.7: *<PERS_PRON>, <COORD_C>, <PERS_PRON>*

2 For each Marker tag: get all Marker words that belong to this category from the Marker file
e. g., for *<PERS_PRON>*: *deiner, dich, dir, mich*, etc.

3 For each Marker word: get all n-grams from the purely lexical language model that contain this word and that are part of the output sentence at the relevant position.
e. g., for the Marker word *mich*:
-2.377654 mich -0.958914
-0.613872 umarmte mich -0.068652
-0.666969 er umarmte mich -0.058926
-2.175931 mich und -0.240510

4 For each n-gram: determine its length n, extract its language model probability e^{LM} and compute a score that captures the n-gram's relevance. The score should be higher for longer n-grams. We experimented with two scores ($s1$ and $s2$) shown in equations 5.8 and 5.10.

$$s1 = e^{\mathrm{LM}} \cdot BP \tag{5.8}$$

BP in equation 5.8 is a brevity penalty similar to the one applied in BLEU (Papineni et al., 2002). It penalizes n-grams that are shorter than the maximum n-gram length of the language model (i. e., 5). Hence, *BP* is higher for shorter n-grams and lower for longer n-grams. It is defined as:

$$BP = \begin{cases} 1 \text{ if } n = 5 \\ e^{(1 - \frac{5}{n})} \text{ if } n < 5 \end{cases} \tag{5.9}$$

e. g., for the n-gram match *umarmte mich*: $e^{\text{LM}} = 0.541$; $n = 2$; $BP = 0.223$; hence $s1 = 0.121$. The corresponding system is called *Moses GT1*.

$$s2 = \frac{1}{(e^{\text{LM}})^{\text{n}}} \tag{5.10}$$

The score shown in equation 5.10 was established experimentally: e. g., for the n-gram *umarmte mich*: $e^{\text{LM}} = 0.541$; $n = 2$; hence $s2 = 3.413$. The corresponding system is called *Moses GT2*.

5 For each Marker word: get the n-gram scores for this word (i. e., the values of $s1$ or $s2$) and sum them. The result is the score for the Marker word.

6 For each Marker tag: get the Marker word with the highest score and replace it with that word.

The result of this process were instantiated sentences. The emphasis of this experiment was on the Marker tag instantiation process and not on the overall performance of the resulting systems (*Moses GT1* and *Moses GT2*). Using generalized sentences opens up the possibility for the SMT decoder to segment an input sentence into longer phrases, but at the same time it implies losing information about those SL word forms that were generalized. For such a system to yield an improvement over a traditional SMT system, the advantage gained by exploring longer phrases has to outweigh the disadvantage of losing SL information.

5.2.3 New German Marker file

As mentioned in Section 2.3, we created a new German Marker file. The new file contains the complete inflectional paradigms of pronouns, auxiliary verbs and modal verbs. We compared the difference in effectiveness between the original and the new file by running *OpenMaTrEx* on half of the data.

5.3 Evaluation Results

5.3.1 Generalized Templates in EBMT

Data	System	BLEU	NIST	METEOR
	1 · *Marclator* DCU templates	0.1274	**4.3948**	**0.4052**
	2 · *Marclator* CMU templates	0.1269	4.3815	0.4047
	3 · *Marclator* DCU & CMU templates	**0.1277**	4.3937	0.4051
All Subtitles	4 · *Marclator* with <*i*> tags	0.0985	4.2248	0.3981
	Marclator (baseline)	0.0995	4.2411	0.3990
	OpenMaTrEx (baseline)	0.2763	5.7880	0.4914
	Moses (baseline)	0.2709	5.7472	0.4854
	1 · *Marclator* DCU templates	**0.1135**	4.2511	0.3927
	2 · *Marclator* CMU templates	0.1130	**4.2527**	**0.3931**
	3 · *Marclator* DCU & CMU templates	0.1133	4.2471	0.3924
½ Subtitles	4 · *Marclator* with <*i*> tags	0.0898	4.1067	0.3878
	Marclator (baseline)	0.0910	4.1293	0.3892
	OpenMaTrEx (baseline)	0.2450	5.4573	0.4654
	Moses (baseline)	0.2474	5.4855	0.4666

Table 5.2: Evaluation scores: generalized templates in EBMT

Table 5.2 shows the results of our experiments with generalized templates in EBMT. The results for *System 4*, a lexical EBMT system, are also listed. The best of our systems (Systems *1* to *4*) with regard to each of the three evaluation metrics (BLEU, NIST and METEOR) is shown in bold. The table shows that for both data sizes, there was no agreement among all three metrics as to which system performed best.

All Subtitles

On the full amount of data (upper half of Table 5.2, "All Subtitles"), *System 3* performed best according to BLEU, while *System 1* performed best according to NIST and METEOR. *System 4* performed the worst according to all three metrics, followed by *System 2*. The three generalized EBMT systems (Systems *1* to *3*) outperformed the lexical EBMT system *Marclator* according to all three evaluation metrics. *System 4*, a lexical EBMT system, performed worse than the *Marclator* baseline. *Marclator* was in turn outperformed by the SMT system *Moses* and the hybrid SMT/EBMT system *OpenMaTrEx*. *OpenMaTrEx* performed better than *Moses*.

We measured statistical significance by bootstrap resampling (Koehn, 2004b) on BLEU.[13] The improvement of *System 3* over *System 2* is statistically significant, while the improvement of *System 3* over *System 1* is not. The improvements of Systems *1*, *2* and *3* over the baseline *Marclator* system are all significant, as are the improvements of the baseline *OpenMaTrEx* and the baseline *Moses* system over Systems *1* to *4*. *System 4* did not perform significantly worse than the baseline *Marclator* system. The improvement of *OpenMaTrEx* over *Moses* is also not significant.

1/2 Subtitles

On half of the data (lower half of Table 5.2, "½ Subtitles"), *System 1* performed best according to BLEU, while *System 2* performed best according to NIST and METEOR. As on the full data, *System 4* performed the worst according to all three metrics; in particular, the system's scores were worse than those of the *Marclator* baseline. Again, systems *1* to *3* performed better than the *Marclator* baseline, while the two other baselines (*OpenMaTrEx* and *Moses*) outperformed *Marclator*. In contrast to the results on the full data, *Moses* achieved higher scores than *OpenMaTrEx*.

The improvement of *System 1* over System *2* is significant according to BLEU, while that of *System 1* over *System 3* is not. Furthermore, as on the full data, the improvements of Systems *1*, *2* and *3* over the baseline *Marclator* system are significant. Again, *System 4* did not perform significantly worse than the baseline *Marclator* system. The improvements of the baselines *OpenMaTrEx* and *Moses* over Systems *1* to *4* are significant (as on the full data). The improvement of *Moses* over *OpenMaTrEx* is not significant.

In summary, the evaluation results for the two data sizes exhibit several common features: *System 4* performed the worst and was outperformed by the lexical EBMT baseline system *Marclator*. *Marclator* was outperformed by the two other baseline systems as well as by our three generalized EBMT systems; all of these improvements were significant. Among our three generalized EBMT systems, there were two significant improvements: that of *System 3* over *System 2* on the full amount of data, and that of *System 1* over *System 2* on half of the data. We discuss these results in Section 5.4.1.

5.3.2 Generalized Templates in SMT

The results of our experiments in applying generalized templates to SMT are displayed in Table 5.3. They show that the baseline *Moses* system outperformed both *Moses GT1* and *Moses GT2*. Recall that *Moses GT1* and *Moses GT2* differ in the way in which n-grams are scored during the *post-hoc* Marker tag instantiation process: one method uses a score similar to the brevity penalty

[13] An approximate randomization with 500 shuffles was performed for the significance tests. All further statements about significance refer to a significance level of α=5 % and to BLEU.

applied in BLEU, and the other is an experimental score. The improvement of *Moses* over *Moses GT1* and *Moses GT2* is significant in both cases. *Moses GT1* also performed significantly better than *Moses GT2*, which means that the n-gram score that is similar to the BLEU brevity penalty worked better than the experimental score. Our emphasis was not on the overall performance of the system; instead, our main goal for this experiment was to develop a context-sensitive Marker-word instantiation algorithm. We investigate the performance of the algorithm in Section 5.4.2.

System	BLEU	NIST	METEOR
Moses GT1	0.2253	5.1361	0.4259
Moses GT2	0.1976	4.8733	0.3979
Moses (baseline)	**0.2474**	**5.4855**	**0.4666**

Table 5.3: Evaluation scores: generalized templates in SMT

5.3.3 *OpenMaTrEx*: original vs. new Marker file

Table 5.4 displays the results of our experiment that compared the effectiveness of the two German Marker files on half of the data. Recall that the two files differ in that the new file contains the complete inflectional paradigms of pronouns, auxiliary verbs and modal verbs. Table 5.4 shows that *OpenMaTrEx* performed slightly better with the new Marker file according to BLEU and NIST and slightly worse according to METEOR. The improvement achieved with the new file according to BLEU is not significant. In Section 2.3, we hypothesized that the new file would generate longer chunks than the original file. Longer chunks are desirable since they are more likely to have unique TL translations. We investigate our hypothesis in Section 5.4.3.

System	BLEU	NIST	METEOR
OpenMaTrEx original file	0.2450	5.4573	**0.4654**
OpenMaTrEx new file	**0.2480**	**5.4619**	0.4623

Table 5.4: Evaluation scores: original vs. new German Marker file in *OpenMaTrEx*

5.4 Discussion

5.4.1 Generalized Templates in EBMT

The evaluation results in Table 5.2 show that our generalized EBMT systems achieved higher scores than the lexical EBMT system *Marclator*. This observation supports earlier findings according to which EBMT systems benefit from an additional layer of abstraction, i. e., from generalized templates. We believe that it is reinforced by the performance results of *System 1* (*Marclator* with DCU generalized templates) and *System 2* (*Marclator* with CMU generalized templates). *System 1* performed better than *System 2* on the full data according to all three evaluation metrics as well as on half of the data according to BLEU. We investigated the generalized chunk coverage of the two systems, i. e., the number of successful generalized chunk matches with respect to the total number of attempts made at matching a generalized chunk.[14] The coverage was 8.26 % for *System 1*. For *System 2*, it was 2.14 %, which is very low. We conclude from this that the higher generalized chunk coverage of *System 1* was the reason why this system performed better than *System 2* in most cases. In broader terms, this means that in order to perform better, an EBMT system has to exhibit a higher generalized chunk coverage.

Table 5.2 also shows that combining *System 1* and *System 2* into *System 3* (*Marclator* with DCU & CMU generalized templates) did not yield a clear improvement over the individual performances of these two systems. We think that this is due to minor differences in the way in which chunks are generalized in our systems as well as to overlaps in the generalization schemes: recall that the two schemes have certain class members in common. The results might also indicate that *System 3* overgeneralized.[15] However, we think that this explanation is not valid in our case: we demonstrated that the CMU generalization scheme led to a very low generalized chunk coverage in *System 2*. Hence, it also did not contribute many generalized chunks to the translation process of *System 3*.

We believe that the low generalized chunk coverage of *System 2* demonstrates the problem inherent in the use of semantic word classes, which form the majority of the CMU equivalence classes. Example 5.11 shows a sample of equivalence classes.

(5.11) *animal · city · color · company · compass · continent · country · croatian cities · currency · date · direction · disease · drug · first name · flower · first name male · frequency · fruit · full name · honorific · last name · measure · money · month · newspaper · number · nut · ocean · organization · religion · rivers · rock size · size · sports team · stock name · time of day · tree · tv · us state · weekday*

[14] Recall that a generalized chunk match is attempted after every unsuccessful word form chunk match.
[15] We mentioned the risk of overgeneralization in our introduction to EBMT in Chapter 2.

The sample shows that the classes are very specific; many of them (e. g., *city, company, country*) have proper name members. On average, each class contains 69 members. To improve the generalized chunk coverage, this number would have to be increased. Compiling lists of equivalence class members manually for multiple languages is time-consuming. Nevertheless, we think it can be useful to apply semantic generalized templates: these templates do not interfere with the grammar of a sentence, which is an important property when translating between languages with different syntactic structure (like English and German).

This property is not met with the DCU generalized templates, as they are based on a hypothesis that makes syntactic assumptions. When investigating the output of the system that applies these templates, *System 1*, we observed one major source of errors, which we call *chunk-internal boundary friction*. Boundary friction (cf. Section 3) is normally caused by the juxtaposition of two separate translation units that do not agree in grammatical case. With the introduction of generalized templates, boundary friction can also take place in a single chunk, i. e., when a Marker word is inserted that does not accommodate the grammatical properties of the rest of the chunk. In the case of English-to-German translation, inserting TL Marker words context-insensitively (as is done in *System 1*) is error-prone: due to the morphological richness of German, an English Marker word can correspond to multiple word forms of the same lemma on the German side. For example, the English Marker word *are* can be translated into the German Marker words *bist*, *sind* and *seid*. Example 5.12 shows an English input sentence and the corresponding German output sentence, translated by *System 1*. The section where chunk-internal boundary friction occurred is shown in bold.

(5.12) *are you sure that superman was hypnotized last night ? · **sind du** sicher dass das superman war hypnotisiert gestern nacht ?*

To translate the English sentence in Example 5.12 into German, our system made use of a German generalized chunk *<AUX> du sicher*. It then instantiated *<AUX>* with *sind*. *sind* is a German verb in the first or third person plural, while *du* is a second-person singular pronoun and the subject of the sentence. In German, the subject and the verb of a sentence have to agree in person and number. Therefore, the combination of *du* and *sind* is grammatically incorrect.

A similar case arose when *System 1* instantiated prepositional Marker tags (*<PREP>*). In German, prepositions have to be followed by noun phrases in specific cases. For example, the preposition *von* requires a dative noun phrase to follow it. The preposition *über* can be followed by either a dative or an accusative noun phrase, depending on whether it indicates a location or a direction: e. g., *wir befinden uns über **dem Meer*** (location, dative) vs. *wir fliegen über **das Meer*** (direction, accusative). Example 5.13 shows an English input sentence translated by *System 1*. The section where chunk-internal boundary friction occurred is shown in bold.

(5.13) *we went through the academy together . · wir haben **durch der** akademie .*

To translate this sentence, our system made use of the following generalized chunk pair: *<PREP> the academy together* · *<PREP> der akademie*. It then instantiated *<PREP>* in the TL half of the chunk pair with *durch*, producing the TL chunk *durch der akademie*. The German preposition *durch* requires an accusative noun phrase to follow it. *der akademie* can be either genitive or dative, but not accusative. Hence, the phrase *durch der akademie* is not grammatical; a correct phrase would be *durch die akademie*.[16]

System 1 also produced good translations, both with respect to grammar and meaning. However, most of these instances were due to the high sentence-repetition ratio in our corpus, as mentioned in Section 5.1. Example 5.14 shows a case in which our system was capable of exploiting an entire sentence from the example base to translate an input sentence.

(5.14) Input: *where 's the baby ?*
Output: *wo ist die kleine ?*

Translating an input sentence as a whole is the most trivial case for an EBMT system.

Table 5.2 shows that *System 4* (*Marclator* with italicized expressions) performed worse than the baseline *Marclator* system, i. e., than *Marclator* without italicized expressions, on both data sizes. There are two possible reasons for this: the first is that we gave *a priori* preference to an italicized expression pair whenever it competed with a chunk pair derived from the *Marclator*-own chunk alignment module. The second possible reason is that the quality of the italicized expression annotations was not as high as we assumed. In order to verify this hypothesis, we checked whether there was a corresponding tag on the German side for each *<i>* tag in the English subtitles.[17] Of the 72,299 tag instances which we found in the English side of the corpus, this was true for 96.2 % of the tags. This is a high agreement score. We concluded from this that the bad performance of *System 4* was at least in part due to the matching strategy which we had chosen.

The evaluation results further show that the difference in performance with regard to the two data sizes was small for our systems (Systems *1* to *4*) as well as for the baseline *Marclator* system (between 0.008 and 0.013 in BLEU). On both data sizes, the EBMT systems performed much worse than the SMT system *Moses* and the hybrid SMT/EBMT system *OpenMaTrEx*. This is in accordance with earlier findings according to which evaluation scores are generally lower for EBMT systems than for SMT or hybrid SMT/EBMT systems (cf., for example, Groves and Way (2005a)). We think that in our case, the performance gap is largely due to the recombination module of *Marclator*: the recombinator is monotone in nature and outputs only the one-best hypothesis.

16 Apart from this, the system also selected the wrong German verb (*haben* instead of *gehen*).
17 Note that we only checked for the presence of a tag; we did not examine whether an SL and a TL italicized expression spanned the same content.

No language model is applied for hypothesis recombination or reranking. Both *OpenMaTrEx* and *Moses* apply a language model for hypothesis recombination. We believe that it is essential for an EBMT system to make use of a language model to reward output sentences that are more fluent with respect to the TL. We stated in Section 3.2 that this is currently not the case with most EBMT systems.

5.4.2 Generalized Templates in SMT

We stated in Section 5.2.2 that for *Moses GT1* and *Moses GT2* to outperform the *Moses* baseline, the advantage gained by exploring longer phrases has to outweigh the disadvantage of compromising SL information. It is evident from the evaluation results shown in Table 5.3 that this was not the case, i. e., that the lack of SL information was not compensated: *Moses GT1* and *Moses GT2* performed worse than the baseline *Moses* system. Table 5.5 shows that the two systems were indeed capable of exploring longer phrases during decoding: the mean length of phrases used by the baseline *Moses* system was 1.91, while it was 2.07 for *Moses GT1* and *Moses GT2*.[18]

Unit	*Moses*	*Moses GT1/GT2*
Phrases	5020	4677
Phrase length: mean	**1.9133**	**2.0748**
Phrase length: std. deviation	1.1228	1.2147

Table 5.5: *Moses* vs. *Moses GT1* and *Moses GT2*: phrases used

As previously mentioned, our emphasis was on the instantiation algorithm. When looking at the output of *Moses GT1*, we observed several instances in which a Marker word had been inserted that was semantically incorrect in the given context. One of these instances is displayed in line 3 of Example 5.15 (the relevant Marker word is shown in bold). Also shown are the corresponding generalized input sentence (line 1), the segmented generalized output (line 2) and the reference translation (line 4).

(5.15) *excellency , give me <DET> few minutes <PREP> them .*[19]
exzellenz , |0-1| ich brauche <DET> paar minuten |2-6| <PREP> ihnen . |7-9|
*exzellenz , ich brauche ein paar minuten **nach** ihnen .*
*geben sie mir ein paar minuten **mit** ihnen .*

[18] The numbers refer to both *Moses GT1* and *Moses GT2*, since the two systems differ only in the way in which TL Marker tags are instantiated, which takes place after decoding.
[19] The ungeneralized input is: *excellency , give me a few minutes **with** them .*

The example shows that *Moses GT1* instantiated <*PREP*> with *nach* instead of *mit* (as in the reference translation), since it lacked knowledge of the corresponding SL word, *with*. Nevertheless, *nach ihnen* is a syntactically correct German prepositional phrase. This shows that the instantiation algorithm itself worked well in this case.

To establish the overall quality of the algorithm, we compared the output of the *Moses* baseline system with the instantiated output of *Moses GT1* on a sentence-wise basis. Because BLEU often does not yield meaningful results at the sentence level, we used METEOR as an evaluation metric. Of the 1171 test sentences, *Moses GT1* received:

- equal scores for 47.74 % of the sentences (559 sentences, of which 246 were mutual zero scores)
- worse scores for 34.07 % of the sentences (399 sentences)
- better scores for 18.19 % of the sentences (213 sentences)

Note that not every sentence output by the *Moses* decoder in *Moses GT1* necessarily contains a Marker tag that has to be instantiated. To determine which of the above sentences had, in fact, undergone an instantiation process, we looked at their uninstantiated correspondences. For each uninstantiated sentence, we counted the number of Marker tags and checked to which of the above three categories (equal, worse or better) its instantiated version belonged. We observed that the average number of instantiated Marker tags per sentence was higher for the 213 sentences that received better scores (1.80 Marker tags) than for the 958 sentences that received equal or worse scores (1.42 Marker tags). This shows that the instantiation process itself worked sufficiently well.

5.4.3 *OpenMaTrEx*: original vs. new Marker file

In Section 2.3, we hypothesized that our new German Marker file would generate longer chunks than the original file. To validate this hypothesis, we investigated the lengths of the chunks that were produced with each of the two files. The results are displayed in Table 5.6. They confirm our hypothesis: the average number of tokens per chunk was approximately 2.5 when using the original Marker file. These are short chunks. With the new Marker file, the average chunk length was only slightly higher: 2.6. Longer chunks are expected to lead to translations of higher quality. However, the new file did not lead to a significant improvement in translation quality. This suggests that the increase in chunk length achieved with the new file was too small.

5.4.4 *OpenMaTrEx* vs. *Moses*

The evaluation results in Table 5.2 show that the difference in performance with regard to the two data sizes was small for the EBMT systems. For *Moses* and *OpenMaTrEx*, it was larger (0.023

Unit	Old	New
Chunks	1,537,452	1,445,766
Chunks per subtitle (mean)	2.562	2.409
Chunks per subtitle (std. deviation)	1.340	1.305
Tokens per chunk (mean)	**2.543**	**2.619**
Tokens per chunk (std. deviation)	1.344	1.442

Table 5.6: Chunking: original vs. new Marker file

and 0.031 in BLEU). While *Moses* outperformed *OpenMaTrEx* on half of the data, *OpenMaTrEx* scored better than *Moses* on the full data. An improvement of *OpenMaTrEx* over *Moses* was also reported by Dandapat et al. (2010), who used *OpenMaTrEx* with the optional binary feature included (as in our experiments) for Spanish–English translation on a training set of 200,000 sentences and a test set of 2000 sentences;[20] they found that the system achieved higher BLEU scores than the baseline *Moses* system (30.75 % vs. 30.59 %).

In general, there are two possible reasons for an improvement by *OpenMaTrEx* over *Moses*:

1 There were unique EBMT chunk pairs (EBMT chunk pairs that were not produced by SMT) in the *OpenMaTrEx* phrase table that were used during decoding.

2 No unique EBMT chunk pairs were actually used during decoding, but adding the EBMT chunk pairs to the *OpenMaTrEx* phrase table led to an increased phrase translation probability for certain SMT phrase pairs and caused them to be favoured over other phrase pairs with the same SL side. These phrase pairs are expected to be of higher quality since they were produced by both phrase/chunk pair extraction techniques.

In order to determine which of the two reasons was true for our case, we extracted the SMT phrase pairs and the EBMT chunk pairs from the combined phrase table of *OpenMaTrEx*. We then reran the *Moses* decoder in *verbose* mode and traced back the origin of each TL phrase used in the output. Table 5.7 displays the result of this analysis. It shows that the number of EBMT chunk pairs used was equal to the number of common chunk/phrase pairs used, i. e., 1754 (shown in bold). This means that no unique EBMT chunk pairs were used during decoding. Based on this observation, we can exclude the first reason given above. Note that this does not necessarily imply that there were no unique EBMT chunk pairs in the entire phrase table; however, this was also the case in our experiment.

[20] The sentences were taken from the Europarl corpus (Koehn, 2005).

	EBMT		SMT		Common		Unknown	
System	Absolute	%	Absolute	%	Absolute	%	Absolute	%
Moses	1211	**25.61**	4647	98.30	1211	25.61	80	**1.69**
OpenMaTrEx	**1754**	**38.15**	4522	98.36	**1754**	38.15	75	**1.63**

Table 5.7: *Moses* vs. *OpenMaTrEx*: phrase pairs used during decoding

Naturally, the baseline *Moses* system did not actually make use of EBMT chunk pairs, but computing the hypothetical numbers allowed us to compare the percentage of EBMT chunks used in *OpenMaTrEx* with that (theoretically) used in *Moses*. Table 5.7 shows that the percentage was higher in *OpenMaTrEx* than in *Moses* (38.15 % vs. 25.61 %, shown in bold). This confirms that the second reason given above was responsible for the improvement of *OpenMaTrEx* over *Moses*. Table 5.7 also shows that the EBMT chunks helped to reduce the percentage of unknown (untranslated) words/phrases: there were 1.63 % unknown words/phrases in *OpenMaTrEx*, while there were 1.69 % in *Moses* (shown in bold). Example 5.16 shows an English input sentence that contained a word which was left untranslated by *Moses* but not by *OpenMaTrEx* (shown in bold).

(5.16) *your only reason for* ***loaning*** *me that money in the beginning was because you were sure i 'd never be able to repay it .*

Neither the *Moses* nor the *OpenMaTrEx* phrase table contained an entry that consisted of only the word *loaning* as its SL side. However, the *OpenMaTrEx* phrase table contained an SL phrase that included the word and matched its context in the input sentence of Example 5.16: *for loaning me*. Hence, when translating the input sentence, *Moses* had to leave the word *loaning* untranslated, while *OpenMaTrEx* made use of the phrase *for loaning me* and translated it as *dass*.[21] Table 5.8 shows how the two systems translated the input sentence. The unknown word in *Moses*'s output is indicated by square brackets. Note how *OpenMaTrEx* performed reordering by moving the word *dass* to the beginning of the sentence (the original position of the word is indicated by a blank field).

In this chapter we presented the results of our experiments, most of which involved EBMT systems using generalized templates. We built three generalized EBMT systems. We showed that our combined system did not perform significantly better than the two individual systems. However, all three systems outperformed the purely lexical *Marclator* baseline. This led us to conclude that EBMT systems benefit from generalized templates. We demonstrated that a high

[21] Note that *dass* (*that*) is a deficient German translation for the English phrase *for loaning me*.

input		*your only*	*reason*	*for*	*loaning*	*me*	*that money*	*in the beginning was…*
Moses		*ihre einzige*	*grund*	*für*	[*loaning*]	*mir*	*das geld*	*im anfang war…*
OpenMaTrEx	*dass*	*ihre einzige*	*grund*				*das geld*	*im anfang war…*

input	*because you were*	*sure*	*i 'd never be able*	*to repay*	*it .*
Moses	*denn du hast*	*sicher ,*	*ich könnte nie*	*zurückzahlen*	*.*
OpenMaTrEx	*denn du hast*	*sicher*	*ich könnte nie*	*zurückzahlen*	*.*

Table 5.8: *OpenMaTrEx* vs. *Moses*: phrase segmentation

generalized chunk coverage is more difficult to achieve with semantic generalized templates than with generalized templates that are based on function words. Semantic generalized templates have the advantage that they leave the grammar of a sentence intact, which is particularly important when translating between languages with different syntactic structure.

This property is not met with generalized templates that are based on function words, which is why our system that applies such templates suffered from chunk-internal boundary friction. A remedy to the problem of chunk-internal boundary friction would be to apply a context-sensitive (instead of a context-insensitive) Marker tag instantiation algorithm. We developed such an algorithm and applied it to an SMT system. We found that while the overall system did not perform well (as it was missing SL information), the performance of the algorithm itself was promising. We also demonstrated that our new German Marker file yielded longer chunks but did not lead to a significant improvement in performance compared to the original file. We assume that for a significant improvement to be achieved, the new chunks would have to be even longer.

We further showed that all of the EBMT systems which we used performed significantly worse than both the SMT and the SMT/EBMT baseline system. We expressed our belief that it is essential for EBMT systems to incorporate a language model for hypothesis recombination. The hybrid SMT/EBMT system which we used as a baseline performed better than the SMT system on the full amount of data. The difference in performance was not significant; however, it might become significant as the data size is increased further. In the next chapter we summarize our work on EBMT using generalized templates and propose future research questions based on the above findings.

6 Conclusion

6.1 Summary

In the present work we showed that EBMT, one of the three major paradigms of Machine Translation, unites a variety of approaches. These approaches have in common that they adhere to three phases: matching, alignment and recombination. One group of approaches provides an additional level for the matching phase by using generalized templates. This level can prevent a system from having to revert to word-by-word translation. Generalized templates can be based on syntactic, semantic or morphological classes. Generalization over sequences of words and over combinations of class types is also applied. In cases where there are too many classes with too few members each, a possible solution is to merge individual classes by clustering them.

The key strength of EBMT systems is their ability to exploit longer phrases during matching. Longer segments are more likely to have unique TL translations. Furthermore, they decrease the number of phrases that have to be combined in order to produce a translation hypothesis and, thus, reduce the likelihood of boundary friction. Boundary friction – the lack of grammatical agreement between two adjacent translation units – is the main problem from which EBMT systems suffer. As a remedy for this problem, EBMT systems have been equipped with language models. Language models are a manifestation of probabilistic knowledge in EBMT. As such, they illustrate the shift towards statistical MT which EBMT has undergone since its introduction. Statistical methods have also been applied for the matching and the alignment phase in EBMT.

Since many of today's EBMT systems make use of statistical knowledge, it has become increasingly difficult to draw a clear line between EBMT and SMT. Nevertheless, there remains one important distinguishing feature: at runtime, EBMT systems consult the training data directly, while SMT systems consult the probabilities thereof. As a consequence, EBMT systems do not run into the problem of distorted probabilities. For this reason, EBMT is often preferred over SMT in cases where there are only small amounts of data available, e. g., in translation tasks involving minority languages. EBMT has also been favoured in cases where the training set and test set are highly similar *per se*, e. g., in controlled language translation. We argued that subtitle translation is similar to controlled language translation in that its data has also been normalized, albeit to a lesser extent. Subtitles are suitable for EBMT for yet another reason: they are usually shorter than the translation units of other domains. Shorter translation units increase the likelihood of retrieving an input sentence as a whole from the example base, which is the most trivial case for an EBMT system. However, short units are also likely to be covered by an SMT system.

We carried out a number of experiments with EBMT systems. In particular, we combined two systems that employ generalized templates. The combined system did not yield a significant improvement in translation quality compared to the individual performances of the two systems; it still exhibited a low generalized chunk coverage. However, our generalized EBMT systems consistently outperformed the lexical EBMT baseline. This shows that generalized templates are advantageous to an EBMT system's performance. For both generalization schemes that we worked with, applying them to a new language potentially means that a new list of generalized template classes has to be compiled.[1] We consider this to be one of the major drawbacks of the two approaches.

We demonstrated that it is more difficult to achieve a high generalized chunk coverage with semantic generalized templates than with generalized templates based on function words. Semantic generalized templates have the advantage that they do not interfere with the grammar of a sentence. In contrast, generalized templates based on function words are relatively easy to compile. However, we showed that a system which relies on such templates can suffer from chunk-internal boundary friction. Chunk-internal boundary friction occurs when a Marker word inserted for a Marker tag in a generalized chunk does not agree syntactically with the rest of the chunk. This phenomenon is frequent when translating from an information-poor into an information-rich language. In our English-to-German experiments, for example, it occurred when German preposition or auxiliary verb Marker tags were instantiated. This was because German, like English, demands that prepositions be followed by noun phrases in specific grammatical cases and requires the subject and the verb of a sentence to agree in person and number.

To reduce the chunk-internal boundary friction problem, we proposed an algorithm that context-sensitively instantiates TL Marker tags by using a language model. We applied the algorithm to an SMT system: we ran the system's decoder on a generalized test set and a generalized language model and initiated a *post-hoc* instantiation process. During this process, the best TL Marker word to be inserted in the place of a Marker tag is determined on the basis of our instantiation algorithm. The overall system did not perform well, as it lacked information about the SL words that had been generalized. However, the evaluation results relating to the instantiation algorithm alone were promising.

We reported that all of our EBMT systems were outperformed by the SMT and the hybrid

[1] Marker word lists currently exist for Catalan, Czech, English, Spanish, French, Irish, German, Italian and Portuguese. To our knowledge, the CMU equivalence classes have only been used for English-to-Spanish and English-to-German translation.

SMT/EBMT baseline system. We emphasized the importance of incorporating a language model into the recombination process of an EBMT system. The results of our experiments also showed that hybrid SMT/EBMT systems like *OpenMaTrEx* have the potential to outperform SMT systems like *Moses*.

6.2 Outlook

We expect more hybrid SMT/EBMT systems to be capable of outperforming state-of-the-art SMT systems in the future. In order to combine "the best of both worlds" (Groves and Way, 2005b, 183), such systems will have to take full advantage of the strengths of EBMT. Above all, this means that they have to allow the EBMT subsystem to explore long phrases and possibly also provide a generalization layer. A technique that has been applied with good results in *OpenMaTrEx* is the introduction of a feature function to weight EBMT chunk pairs. Another successful example of a hybrid system – both conceptually and in terms of the quality of its output – is the *Cunei* system, where fragments are extracted, aligned and scored at runtime; they receive a score that depends, among other features, on the amount of context they share with the relevant portion of the input sentence. A further promising strategy is to take advantage of *Moses*' capability of dealing with XML markup and to specify EBMT output segments as do-not-translate units for the *Moses* decoder.

A way to overcoming the high adaptability cost of the generalized EBMT system that relies on the CMU generalization scheme is to generate the equivalence classes automatically, i. e., through clustering. The Marker words for the DCU generalization scheme could be obtained automatically by taking advantage of the fact that function words are among the most frequent words in every corpus. Following this assumption, the n-most frequent words can be considered as Marker words.[2] Generalization could then be performed over the POS categories of these words. This could result in classes similar to those of the present Marker approach. However, we investigated the distribution of function words in the TL half of our corpus and found that when extracting the top 0.1 % words (152 words), only 63.82 % (97 words) were contained in our new German Marker file. The words that were not contained in the Marker file were not specific to the subtitle domain; they were words like *nicht* (English: *not*), *ja* (*yes*), *hier* (*here*) or *nein* (*no*). This suggests that it would be difficult to achieve both a high precision and a high recall with such an approach.

One way to increase the generalized chunk coverage for the DCU generalized template approach is to generalize chunks recursively, i. e., not only generalize over the first word of a chunk but also over all subsequent Marker words. Contiguous Marker words appear, for example, in

[2] For each of these words, all other possible word forms of the same lemma would also have to be included in the Marker list.

prepositional phrases, e. g., ***of a** marathon*. Recursively generalizing this chunk would lead to the generalized chunk *<PREP> <DET> marathon*, which would have a greater matching likelihood than the original generalized chunk, *<PREP> a marathon*. However, recursively generalized chunks also increase the likelihood of chunk-internal boundary friction. They should therefore not be applied without a prior adjustment of the Marker tag instantiation algorithm. We plan to incorporate our algorithm that context-sensitively instantiates Marker tags into our generalized template extension of *Marclator*. When doing so, we will also experiment with different n-gram orders and with global (as opposed to local) instantiation, where we will consider all Marker tags of a sentence simultaneously.

Bibliography

Stephen Armstrong, Declan Groves, Marian Flanagan, Yvette Graham, Bart Mellebeek, Sara Morrissey, Nicolas Stroppa, and Andy Way. The MaTreX system: Machine translation using examples. In *TC-STAR OpenLab Workshop on Speech Translation*, page not numbered, Trento, Italy, 2006.

Satanjeev Banerjee and Alon Lavie. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In *Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization, Proceedings of the ACL-05 Workshop*, pages 65–72, University of Michigan, Ann Arbor, Michigan, USA, 2005.

Srinivas Bangalore, Vanessa Murdock, and Giuseppe Riccardi. Bootstrapping bilingual data using consensus translation for a multilingual instant messaging system. In *Proceedings of the 19th International Conference on Computational Linguistics (COLING 2002)*, pages 1–7, Taipei, Taiwan, 2002.

Peter F. Brown, John Cocke, Stephen A. Della Pietra, Vincent J. Della Pietra, Fredrick Jelinek, John D. Lafferty, Robert L. Mercer, and Paul S. Roossin. A statistical approach to machine translation. *Computational Linguistics*, 16(2):79–85, 1990.

Ralf D. Brown. Example-based machine translation in the Pangloss system. In *COLING-96: The 16th International Conference on Computational Linguistics, Proceedings*, pages 169–174, Center for Sprogteknologi, Copenhagen, Denmark, 1996.

Ralf D. Brown. Automated dictionary extraction for “knowledge-free” example-based translation. In *TMI-97: Proceedings of the 7th International Conference on Theoretical and Methodological Issues in Machine Translation*, pages 111–118, St. John’s College, Santa Fe, New Mexico, USA, 1997.

Ralf D. Brown. Adding linguistic knowledge to a lexical example-based translation system. In *8th International Conference on Theoretical and Methodological Issues in Machine Translation (TMI 99)*, pages 22–32, University College, Chester, England, 1999.

Ralf D. Brown. Automated generalization of translation examples. In *The 18th International Conference on Computational Linguistics, COLING 2000 in Europe, Proceedings of the Conference*, pages 125–131, Universität des Saarlandes, Saarbrücken, Germany, 2000.

Ralf D. Brown. Transfer-rule induction for example-based translation. In Michael Carl and Andy Way, editors, *Recent Advances in Example-Based Machine Translation*, pages 1–11. Kluwer Academic Publishers, Dordrecht, The Netherlands, 2003.

Ralf D. Brown, Paul N. Bennett, Jaime G. Carbonell, Rebecca Hutchinson, and Peter Jansen. Reducing boundary friction using translation-fragment overlap. In *MT SUMMIT IX: Proceedings of the Ninth Machine Translation Summit*, pages 24–31, New Orleans, USA, 2003.

Chris Callison-Burch, Miles Osborne, and Philipp Koehn. Re-evaluating the role of BLEU in machine translation research. In *EACL-2006: 11th Conference of the European Chapter of the Association for Computational Linguistics, Proceedings of the Conference*, pages 249–256, Trento, Italy, 2006.

Michael Carl and Andy Way. Introduction. In Michael Carl and Andy Way, editors, *Recent Advances in Example-Based Machine Translation*, pages xvii–xxxii. Kluwer Academic Publishers, Dordrecht, The Netherlands, 2003.

Daniel Cer, Christopher D. Manning, and Daniel Jurafsky. The best lexical metric for phrase-based statistical MT system optimization. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Proceedings of the Main Conference*, pages 555–563, Los Angeles, California, 2010.

Stanley F. Chen and Joshua Goodman. An empirical study of smoothing techniques for language modeling. In *34th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference*, pages 310–318, University of california, Santa Cruz, Santa Cruz, California, USA, 1996.

Ilyas Cicekli and Halil Altay Güvenir. Learning translation templates from bilingual translation examples. *Applied Intelligence*, 15(1):57–76, 2001.

Jorge Díaz Cintas and Aline Remael. *Audiovisual Translation: Subtitling*. St. Jerome Publishing, Manchester, UK, 2007.

Michael Collins, Philipp Koehn, and Ivona Kučerová. Clause restructuring for statistical machine translation. In *ACL-05: 43rd Annual Meeting of the Association for Computational Linguistics*, pages 531–540, University of Michigan, Ann Arbor, Michigan, USA, 2005.

Sandipan Dandapat, Mikel L. Forcada, Declan Groves, Sergio Penkale, John Tinsley, and Andy Way. OpenMaTrEx: A free/open-source marker-driven example-based machine translation system. In *Proceedings of IceTAL*, pages 121–126, Reykjavík, Iceland, 2010.

Lee R. Dice. Measures of the amount of ecologic association between species. *Ecology*, 26(3): 297–302, 1945.

George Doddington. Automatic evaluation of machine translation quality using n-gram co-occurrence statistics. In *HLT 2002: Human Language Technology Conference, Proceedings of the Second International Conference on Human Language Technology Research*, pages 138–145, San Diego, California, 2002.

Jinhua Du and Andy Way. A discriminative latent variable-based de classifier for chinese–english smt. In *Proceedings of the 23rd International Conference on Computational Linguistics (COLING 2010)*, pages 286–294, Beijing, China, 2010.

Paula Estrella. *Evaluating Machine Translation in Context: Metrics and Tools*. PhD thesis, University of Geneva, Geneva, Switzerland, 2008.

Marcello Federico and Mauro Cettolo. Efficient handling of n-gram language models for statistical machine translation. In *ACL 2007: Proceedings of the Second Workshop on Statistical Machine Translation*, pages 88–95, Prague, Czech Republic, 2007.

Robert E. Frederking and Ralf D. Brown. The Pangloss-Lite machine translation system. In *Expanding MT Horizons, Proceedings of the Second Conference of the Association for Machine Translation in the Americas*, pages 268–272, Montreal, Quebec, Canada, 1996.

Rashmi Gangadharaiah, Ralf Brown, and Jaime Carbonell. Spectral clustering for example based machine translation. In *HLT-NAACL 2006: Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics, Proceedings of the Main Conference*, pages 41–44, New York, New York, USA, 2006.

Nano Gough and Andy Way. Controlled generation in example-based machine translation. In *MT SUMMIT IX: Proceedings of the Ninth Machine Translation Summit*, pages 133–140, New Orleans, USA, 2003.

Nano Gough and Andy Way. Robust large-scale EBMT with marker-based segmentation. In *TMI-2004 Conference: Proceedings of the 10th International Conference on Theoretical and Methodological Issues in Machine Translation*, pages 95–104, Baltimore, Maryland, USA, 2004.

Thomas Green. The necessity of syntax markers. two experiments with artificial languages. *Journal of Verbal Learning and Behavior*, 18:481–496, 1979.

Declan Groves and Andy Way. Hybrid data-driven models of machine translation. *Machine Translation*, 19(3-4):301–323, 2005a.

Declan Groves and Andy Way. Hybrid example-based SMT: the best of both worlds? In *ACL-05: Building and Using Parallel Texts: Data-Driven Machine Translation and Beyond, Proceedings of the Workshop*, pages 183–190, University of Michigan, Ann Arbor, Michigan, USA, 2005b.

Halil Altay Güvenir and Aysegül Tunc. Corpus-based learning of generalized parse tree rules for translation. In *Proceedings of the 11th Biennial Conference of the Canadian Society for Computational Studies of Intelligence on Advances in Artificial Intelligence (AI'96)*, pages 121–132, London, UK, 1996.

Christian Hardmeier. Using linguistic annotations in statistical machine translation of film subtitles. Master's thesis, Universität Basel, 2008.

Yifan He and Andy Way. Learning labelled dependencies in machine translation evaluation. In *EAMT-2009: Proceedings of the 13th Annual Conference of the European Association for Machine Translation*, pages 44–51, Universitat Politècnica de Catalunya, Barcelona, Spain, 2009.

Daniel S. Hirschberg. A linear space algorithm for computing maximal common subsequences. *Communications of the ACM*, 18(6):341–343, 1975.

Hiroyuki Kaji, Yuuko Kida, and Yasutsugu Morimoto. Learning translation templates from bilingual text. In *Proceedings of the fifteenth [sic] International Conference on Computational Linguistics, Actes du quinziéme colloque international en linguistique informatique, COLING-92*, pages 672–678, Nantes, 1992.

Slava M. Katz. Estimation of probabilities from sparse data for the language model component of a speech recognizer. In *IEEE Transactions on Acoustics, Speech and Signal Processing*, pages 400–401, 1987.

Mihoko Kitamura and Yuji Matsumoto. A machine translation system based on translation rules acquired from parallel corpora. In *International Conference: Recent Advances in Natural Language Processing, Proceedings*, pages 27–36, Hotel "Orpheus", Tzigov Chark, Bulgaria, 1995.

Philipp Koehn. Pharaoh: A beam search decoder for phrase-based statistical machine translation models. In *Machine Translation: From Real Users to Research, 6th Conference of the Association for Machine Translation in the Americas, AMTA 2004, Proceedings*, pages 115–124, Washington, DC, USA, 2004a.

Philipp Koehn. Statistical significance tests for machine translation evaluation. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pages 388–395, Barcelona, Spain, 2004b.

Philipp Koehn. Europarl: a parallel corpus for statistical machine translation. In *MT Summit X: The Tenth Machine Translation Summit*, pages 79–86, Phuket, Thailand, 2005.

Philipp Koehn, Franz Josef Och, and Daniel Marcu. Statistical phrase-based translation. In *HLT-NAACL 2003: conference combining Human Language Technology conference series and the North American Chapter of the Association for Computational Linguistics conference series*, pages 48–54, Edmonton, Canada, 2003.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondřej Bojar, Alexandra Constantin, and Evan Herbst. Moses: Open source toolkit for statistical machine translation. In *ACL 2007, Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics*, pages 177–180, Prague, Czech Republic, 2007.

Yves Lepage and Etienne Denoual. The 'purest' EBMT system ever built: No variables, no templates, no training, examples, just examples, only examples. In *MT Summit X Workshop: Second Workshop on Example-based Machine Translation*, pages 81–90, Phuket, Thailand, 2005.

Gregor Leusch, Nicola Ueffing, and Hermann Ney. Cder: Efficient mt evaluation using block movements. In *EACL-2006: 11th Conference of the European Chapter of the Association for Computational Linguistics, Proceedings of the Conference*, pages 241–248, Trento, Italy, 2006.

Vladimir I. Levenshtein. Binary codes capable of correcting deletions, insertions, and reversals. *Soviet Physics Doklady*, 10:707–710, 1966.

Ding Liu and Daniel Gildea. Syntactic features for evaluation of machine translation. In *Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization, Proceedings of the ACL-05 Workshop*, pages 25–32, University of Michigan, Ann Arbor, Michigan, USA, 2005.

Zhanyi Liu, Haifeng Wang, and Hua Wu. Example-based machine translation based on tree-string correspondence and statistical generation. *Machine Translation*, 20(1):25–41, 2006.

George A. Miller. WordNet: a lexical database for English. In *Human Language Technology, Proceedings of a workshop held at Plainsboro, New Jersey*, pages 409–409, Plainsboro, New Jersey, 1993.

Makoto Nagao. A framework of a mechanical translation between Japanese and English by analogy principle. In *Proceedings of the International NATO Symposium on Artificial and Human Intelligence*, pages 173–180, New York, NY, USA, 1984.

Sonja Nießen, Franz J. Och, Gregor Leusch, and Hermann Ney. An evaluation tool for machine translation: Fast evaluation for MT research. In *LREC-2000: Second International Conference on Language Resources and Evaluation, Proceedings*, Athens, Greece, 2000.

Sergei Nirenburg, Constantine Domashnev, and Dean J. Grannes. Two approaches to matching in example-based machine translation. In *TMI-93: The Fifth International Conference on Theoretical and Methodological Issues in Machine Translation with special emphasis on: MT in the Next Generation*, pages 47–57, Kyoto International Community House, Kyoto, Japan, 1993.

Franz Josef Och. Minimum error rate training in statistical machine translation. In *41st Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference*, pages 160–167, Sapporo Convention Center, Japan, 2003.

Franz Josef Och and Hermann Ney. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51, 2003.

Franz Josef Och and Hermann Ney. The alignment template approach to statistical machine translation. *Computational Linguistics*, 30(4):417–449, 2004.

Franz Josef Och and Hans Weber. Improving statistical natural language translation with categories and rules. In *COLING-ACL '98, 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics, Proceedings of the Conference*, pages 985–989, Université de Montréal, Montreal, Quebec, Canada, 1998.

Karolina Owczarzak, Josef van Genabith, and Andy Way. Labelled dependencies in machine translation evaluation. In *ACL 2007: Proceedings of the Second Workshop on Statistical Machine Translation*, pages 104–111, Prague, Czech Republic, 2007.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. BLEU: a method for automatic evaluation of machine translation. In *40th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference*, pages 311–318, Philadelphia, PA, USA, 2002.

Aaron B. Phillips and Ralf D. Brown. Cunei machine translation platform: System description. In *Proceedings of the 3rd International Workshop on Example-Based Machine Translation*, pages 29–37, Centre for Next Generation Localisation, Dublin City University, Dublin, Ireland, 2009.

Aaron B. Phillips, Violetta Cavalli-Sforza, and Ralf D. Brown. Improving example based machine translation through morphological generalization and adaptation. In *MT SUMMIT IX: Proceedings of the Ninth Machine Translation Summit*, New Orleans, USA, 2003.

Alexandre Rafalovitch and Robert Dale. United Nations General Assembly Resolutions: A six-language parallel corpus. In *MT Summit XII: Proceedings of the Twelfth Machine Translation Summit*, pages 292–299, Ottawa, Ontario, Canada, 2009.

Philip Resnik. Mining the web for bilingual text. In *37th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference*, pages 527–534, University of Maryland, College Park, Maryland, USA, 1999.

Philip Resnik and Noah A. Smith. The web as a parallel corpus. *Computational Linguistics*, 29(3): 349–380, 2003.

Beatrice Santorini. *Part-of-Speech Tagging Guidelines for the Penn Treebank Project*. Department of Computer and Information Science, University of Pennsylvania, 1990.

Anne Schiller, Simone Teufel, Christine Stöckert, and Christine Thielen. Guidelines für das Tagging deutscher Textcorpora mit STTS. Technical report, Institut für maschinelle Sprachverarbeitung, Stuttgart, Germany, 1999.

Helmut Schmid. Improvements in part-of-speech tagging with an application to German. In *Proceedings of the EACL'95 SIGDAT Workshop*, pages 47–50, Dublin, Ireland, 1995.

Rico Sennrich. Syntactically enriched statistical machine translation from English to German. Master's thesis, Universität Zürich, Switzerland, 2009.

James Smith and Stephen Clark. EBMT for SMT: A new EBMT-SMT hybrid. In *Proceedings of the 3rd International Workshop on Example-Based Machine Translation*, pages 3–10, Centre for Next Generation Localisation, Dublin City University, Dublin, Ireland, 2009.

Matthew Snover, Bonnie Dorr, Richard Schwartz, Linnea Micciulla, and John Makhoul. A study of translation edit rate with targeted human annotation. In *AMTA 2006: Proceedings of the 7th Conference of the Association for Machine Translation in the Americas: Visions for the Future of Machine Translation*, pages 223–231, Cambridge, Massachusetts, USA, 2006.

Matthew Snover, Nitin Madnani, Bonnie J. Dorr, and Richard Schwartz. Fluency, adequacy, or HTER?: Exploring different human judgments with a tunable MT metric. In *EACL 2009: Fourth Workshop on Statistical Machine Translation, Proceedings of the Workshop*, pages 259–268, Megaron Athens International Conference Centre, Athens, Greece, 2009.

Harold Somers. Review article: Example-based machine translation. *Machine Translation*, 14(2): 113–157, 1999.

Harold Somers. An overview of EBMT. In Michael Carl and Andy Way, editors, *Recent Advances in Example-Based Machine Translation*. Kluwer Academic Publishers, Dordrecht, The Netherlands, 2003.

Harold Somers, Ian McLean, and Danny Jones. Experiments in multilingual example-based generation. In *Proceedings of the 3rd Conference on the Cognitive Science of Natural Language Processing (CSNLP'94)*, page not numbered, Dublin, Ireland, 1994.

Harold Somers, Sandipan Dandapat, and Sudip Kumar Naskar. A review of EBMT using proportional analogies. In *Proceedings of the 3rd International Workshop on Example-Based Machine Translation*, pages 53–60, Centre for Next Generation Localisation, Dublin City University, Dublin, Ireland, 2009.

Ankit Srivastava, Sergio Penkale, Declan Groves, and John Tinsley. Evaluating syntax-driven approaches to phrase extraction for MT. In *Proceedings of the 3rd International Workshop on Example-Based Machine Translation*, pages 19–28, Centre for Next Generation Localisation, Dublin City University, Dublin, Ireland, 2009.

Ralf Steinberger, Bruno Pouliquen, Anna Widiger, Camelia Ignat, Tomaz Erjavec, Dan Tufis, and Daniel Varga. The JRC-Acquis: A multilingual aligned parallel corpus with 20+ languages. In *LREC-2006: Fifth International Conference on Language Resources and Evaluation, Proceedings*, pages 2142–2147, Genoa, Italy, 2006.

Nicolas Stroppa and Andy Way. Matrex: DCU machine translation system for IWSLT 2006. In *IWSLT 2006: Proceedings of the 3rd International Workshop on Spoken Language Translation*, pages 31–36, Palulu Plaza, Kyoto, Japan, 2006.

Nicolas Stroppa, Declan Groves, Andy Way, and Kepa Sarasola. Example-based machine translation of the Basque language. In *AMTA 2006: Proceedings of the 7th Conference of the Association for Machine Translation in the Americas: Visions for the Future of Machine Translation*, pages 232–241, Cambridge, Massachusetts, USA, 2006.

Eiichiro Sumita. Example-based machine translation using DP-matching between word sequences. In *Proceedings of the Workshop on Data-driven Methods in Machine Translation*, pages 1–8, Toulouse, France, 2001.

Jörg Tiedemann and Lars Nygard. The OPUS corpus – parallel and free. In *LREC-2004: Fourth International Conference on Language Resources and Evaluation, Proceedings*, Lisbon, Portugal, 2004.

Christoph Tillmann, Stephan Vogel, Hermann Ney, Alex Zubiaga, and Hassan Sawaf. Accelerated DP-based search for statistical translation. In *Proceedings of the 5th European Conference on Speech Communication and Technology (Eurospeech'97)*, pages 2667–2670, Rhodes, Greece, 1997.

Jesús Tomás, Jordi Bataller, Jaime Lloret, and Francisco Casacuberta. Mining Wikipedia as a parallel and comparable corpus. *Language Forum*, 34(1):123–136, 2008.

Joseph Turian, Luke Shen, and I. Dan Melamed. Evaluation of machine translation and its evaluation. In *MT SUMMIT IX: Proceedings of the Ninth Machine Translation Summit*, pages 386–393, New Orleans, USA, 2003.

Martin Volk. The automatic translation of film subtitles. a machine translation success story? In *Festschrift in Honor of Anna Sågvall Hein*, volume 7 of *Studia Linguistica Upsaliensia*, pages 202–214, 2008.

Andy Way. Translating with examples. In *MT Summit VII: Workshop on Example-Based Machine Translation, Proceedings of the Workshop*, Santiago de Compostela, Spain, 2001.

Andy Way. Machine translation. In Alex Clark, Chris Fox, and Shalom Lappin, editors, *Handbook of Computational Linguistics and Natural Language Processing*, pages 591–641. Wiley-Blackwell, 2010a.

Andy Way. Panning for ebmt gold, or "remembering not to forget". *Machine Translation (to appear)*, 2010b.

Andy Way and Nano Gough. Comparing example-based and statistical machine translation. *Natural Language Engineering*, 11(3):295–309, 2005.

John S. White, Theresa A. O'Connell, and Lynn M. Carlson. Evaluation of machine translation. In *Human Language Technology, Proceedings of a workshop held at Plainsboro, New Jersey*, pages 206–210, Plainsboro, New Jersey, 1993.

Dekai Wu. MT model space: Statistical versus compositional versus example-based machine translation. *Machine Translation*, 19(3-4):213–227, 2005.

Dekai Wu. Toward machine translation with statistics and syntax and semantics. In *Automatic Speech Recognition Understanding, 2009. ASRU 2009. IEEE Workshop on*, pages 12–21, 2009.

Omar F. Zaidan. Z-MERT: A fully configurable open source tool for minimum error rate training of machine translation systems. *The Prague Bulletin of Mathematical Linguistics*, 91:79–88, 2009.

Curriculum Vitae

Sarah Ebling

Lerchenberg 1
8046 Zürich

Phone: +41 79 520 58 49
E-mail: sarah.ebling@sunrise.ch

Born: April 23, 1984 (Baden, Switzerland)
Nationality: Swiss

Education

2004–2011 **Undergraduate/Graduate Studies**
German Linguistics and Literature, Computational Linguistics
University of Zurich, Switzerland
University of Heidelberg, Germany
Dublin City University, Ireland

2000–2004 **High School**
Kantonsschule Baden, Switzerland
Tamalpais High School, Mill Valley, CA, USA

1996–2000 **Middle School**
Bezirkschule Baden, Switzerland

1992–1996 **Elementary School**
Primarschule Baden, Switzerland
Nesbit Elementary School, Belmont, CA, USA

Jobs/Internships

Jan/Feb 2010 **Student Research Assistant**
Sentiment analysis using machine learning techniques
HITS (formerly EML Research), Heidelberg, Germany

2008–2010 **Student Research Assistant**
Project semtracks
Data mining, linguistic and statistical analyses, media relations, data visualization
Heidelberg, Germany; Zurich, Switzerland

2007–2008 **Administrative Assistant**
Homburger Law Firm, Zurich, Switzerland

2006 **Three-Month Journalism Internship**
Neue Zürcher Zeitung, Zurich, Switzerland

2003–2008 **Freelance Journalist**
Aargauer Zeitung, Baden, Switzerland

Presentations

14.09.2010 Computational Methods of Discourse Analysis. Full-day workshop at "Empirische Methoden bei der Untersuchung von SprachRäumen" (EMUS), University of Leipzig (with Klaus Rothenhäusler).

05.08.2010 Sprachgebrauchsmuster in politischen Diskursen: Maschinelle Methoden der datengeleiteten Analyse. IVG, Sektion 53, "Diskurslinguistik im Spannungsfeld von Deskription und Kritik", University of Warsaw (with Noah Bubenhofer and Saskia Vola).

19.02.2010 Flirting with the Fringe: Constructing Arenas by Language Usage. EU workshop "Arenas of Contestation", University of Heidelberg (with Joachim Scharloth and David Eugster).

05.09.2009 Politische Sprache an den Rändern. Korpusgeleitete Zugänge zur Sprache extremistischer Parteien im Bundestagswahlkampf 2009. AG Sprache in der Politik, "Die da oben… Politik und Partizipation im Wahljahr", University of Leipzig (with Joachim Scharloth).

Publications

2010 (to appear) Sarah Ebling. Korpusgeleitete Zugänge zur Rhetorik deutscher und schweizerischer Politiker am Beispiel von Peer Steinbrück und Hans-Rudolf Merz. In Kersten Sven Roth und Christa Dürscheid, editors, *Wahl der Wörter – Wahl der Waffen? Politische Sprache und Kommunikation in der Schweiz*, Sprache – Politik – Gesellschaft. Hempen, Bremen.

(to appear) Joachim Scharloth, Christian Gerber, Balthasar Glättli, Michel Studer, Noah Bubenhofer, Sarah Ebling, and Saskia Vola. Die Schweiz in der Krise: Korpuspragmatische Untersuchungen zur sprachlichen Konstruktion und Diffusion von Krisensemantiken. *Aptum*, 2.

2009 Noah Bubenhofer, Tobias Dussa, Sarah Ebling, Martin Klimke, Klaus Rothenhäusler, Joachim Scharloth, and Saskia Vola (Forschergruppe semtracks). So etwas wie eine Botschaft". Korpuslinguistische Analysen der Bundestagswahl 2009. *Sprachreport*, 4:2–10.