Coping with Data-sparsity in Example-based Machine Translation

Rashmi Gangadharaiah

CMU-10-020

Language Technologies Institute
School of Computer Science
Carnegie Mellon University
5000 Forbes Ave., Pittsburgh, PA 15213
www.lti.cs.cmu.edu

Thesis Committee:
Jaime Carbonell, Chair (Carnegie Mellon University)
Ralf D. Brown, Co-chair (Carnegie Mellon University)
Stephan Vogel (Carnegie Mellon University)
Altay Güvenir, Bilkent University, Turkey.

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy
In Language and Information Technologies.

Keywords: Data-Sparsity, Example-based Machine Translation (EBMT) , Generalized-Example-based Machine Translation (G-EBMT), Template Induction, Clustering, Class-based Language Modeling.

To my family

Abstract

Data-driven Machine Translation (MT) systems have been found to require large amounts of data to function well. However, obtaining parallel texts for many languages is time-consuming, expensive and difficult. This thesis aims at improving translation quality for languages that have limited resources by making use of the available data more efficiently.

Templates or generalizations of sentence-pairs where sequences of one or more words are replaced by variables are used in the translation model to handle data-sparsity challenges. Templates are built from clusters or equivalence classes that group related terms (words and phrases). As generating such clusters can be time-consuming, clusters are automatically generated by grouping terms based on their semantic-similarity, syntactical-coherence and context. Data-sparsity is also a big challenge in statistical language modeling. In many MT systems, sophisticated tools are developed to make the translation models better but they still rely heavily on a restricted-decoder which uses unreliable language models that may not be well suited for translation tasks especially in sparse-data scenarios. Templates can also be used in Language Modeling.

Limited training data also increases the number of out-of-vocabulary words and reduces the quality of the translations. Many of the present MT systems either ignore these unknown words or pass them on as is to the final translation assuming that they could be proper nouns. Presence of out-of-vocabulary words and rare words in the input sentence prevents an MT system from finding longer phrasal matches and produces low quality translations due to less reliable language model estimates. Approaches in the past have suggested using stems and synonyms of OOV words as replacements. This thesis uses an algorithm to find possible replacements which are not necessarily synonyms to replace out-of-vocabulary words as well as rare words based on the context in which these words appear.

The effectiveness of each of the template-based approaches both in the translation model and in the language model are demonstrated for English $\rightarrow$Chinese and English$\rightarrow$French. The algorithm to handle out-of-vocabulary and rare words are tested on English $\rightarrow$French, English $\rightarrow$Chinese and English $\rightarrow$Haitian. A *Hybrid* approach combining all the techniques is also studied in English $\rightarrow$Chinese.

Acknowledgments

This thesis work would not have been possible without the guidance and support of my advisors, Jaime Carbonell and Ralf Brown, and I would like to thank them. My advisors have been great mentors throughout my research. I was given the freedom to pursue areas that I found interesting and my advisors were always there to guide me whenever I found myself stuck with a problem. I was fortunate to have advisors who helped me think practically and as well as theoretically.

I would also like to thank my committee members, Stephan Vogel and Altay Güvenir for their comments during my thesis proposal. Their valuable comments made the thesis more concrete and complete.

I have learned a lot from Prof. Raj Reddy from the few discussions that we have had. His view of research with the ultimate goal of improving the society has changed my way of thinking. His vision was also a motivation for me to work on data sparse languages.

I am grateful to my mentor (who is also now my father-in-law), Prof. N. Balakrishnan at the Indian Institute of Science, for all the interesting discussions and for introducing me to Natural Language Processing (especially Speech Recognition and Machine Translation) and Electromagnetics. He is one of the few people actively involved in speech and language related research in India and I was privileged to work under him.

I also like to thank Robert Frederking (Bob) and Carolyn Rose for their comments on some of my chapters in this thesis. Their comments helped me strengthen the motivations for some of the techniques used in this thesis. I appreciate Bob for his support during one of the conferences during which I was rejected a Visa and he enthusiastically agreed to present the paper and also gave very useful feedback.

I like to thank Liang Chenmin, Qiao Li and Yang Weng for providing feedback on the Chinese translations and examples used in my conference papers.

During my studies at CMU, I had the chance to work on a number of interesting research problems. I like to thank Lori Levin who helped me understand various issues in

Natural Language. I also like to thank Bhiksha Raj and Rita Singh for sharing their knowledge in building Speech Recognition systems and for their support during my job hunt in India. I would also like to thank the students in LTI, especially the MT community and Alon Lavie for providing feedback during my presentations at LTI.

There are many others who have also influenced my student life during the five years at CMU. All my friends at CMU: Sivaraman Balakrishnan (aka Kannan), Satashu Goel, Abhishek Jajoo, Sonia Singhal, Emil Albright, Aaron Philips, Jae Dong Kim, Peter Jansen, Hetunandan Kamisetty, Varun Gupta, Vivek Seshadri, Srivatsan Narayanan, Kaushik Lakshminarayanan, Ravishankar Krishnaswamy, Swapnil Patil, Gaurav Veda, Ashwini Balakrishnan, Debabrata Dash, Suyash Shringarpure, Gopal Siddharth, Aditya Prakash, Shilpa Arora, Rohit Kumar, Mohit Kumar, Kishore Prahallad, Ayesha Bhargava, Mudit Bhargava, Lakshmi, Sharath Rao, Abhimanyu Lad, Satanjeev Banerjee, Vamshi Ambati, Sanjika Hewavitharana, ThuyLinh Nguyen, Amar Phanishayee, Tad Thomas Merryman, Jason Thornton, Eui Seok Hwang, my officemates: Jonathan Elsas, Pinar Donmez, Agha Ali Raza and John Kominek, thank you all for all the exciting and fun filled moments at CMU.

Special thanks to the Administrative staff at LTI: Radha Rao, Stacey Young, Linda Hager, Mary Jo Bensasi, Corinne Meloni and Dana Houston for their timely help.

I like to thank my family: parents, sister and in-laws for their moral support and encouragement throughout my stay in the US. I am extremely grateful to my father who had faith in me and sent me to the US for higher education. I should also thank my daughter, Advika, whose cheerful attitude and innocent smile has helped me strive through difficult times- I know I haven't spent enough time with you, but I will definitely make up for it. Last but not the least, I like to thank my husband, Balakrishnan Narayanaswamy (aka Murali), for his infinite support in every aspect of my life.

Once again, Thank You All.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Machine Translation (MT) refers to the task of translating text or speech from one language to another using a machine. There are many 'classes' of MT systems such as rule based MT, statistical MT, syntax-based MT, context-based MT and example-based MT [Hutchins, 2001]. In this thesis we will look at data driven approaches in general and EBMT in particular.

Before the introduction of data-driven approaches, manually built transfer rules were used to generate translations. Developing a system using manually built rules to translate new language-pairs took several years. With the increase in the number and size of data sources and with the exponential increase in the computational power available to process the data, data-driven approaches became extremely popular. Data-driven approaches require just a parallel corpus and almost no other sizable knowledge sources. As a result, these systems are not only easy to build but also can be quickly adapted to new language pairs.

Data-driven MT systems work surprisingly well when large amounts of training data are available even without incorporating much language specific knowledge. These systems work very well for predictable texts which have closed/limited vocabulary [Hutchins, 2005] and are widely used as an aid to human translators and for translating large manuals [Kay, 1982]. However, these systems today are far from perfect when dealing with limited data and their performance drops substantially in data sparse conditions. This is because computers do not have the ability to deal with language complexities that help humans generalize and require large amounts of data [Munteanu and Marcu, 2005] to capture such complexities.

Sparse training data, test and training data from different domains and even simple

typographical errors all negatively affect performance of MT systems. All these situations produce Out Of Vocabulary (OOV) words that are the bane of many NLP tasks [Woodland et al., 2000] and much work in the MT community has been directed at ameliorating these adverse effects [Habash, 2008].

The Example-based Machine Translation (EBMT) system used in this thesis like other data-driven approaches uses a parallel corpus to translate new input source sentences. EBMT systems (like most MT systems) consists of two parts - a Translation Model and a Language Model. In the Translation Model, the input sentence to be translated is matched against the source sentences in the bilingual training corpus. When a match is found somewhere in the training data, the corresponding translation in the target language is obtained through sub-sentential alignment. Once these fragments are extracted, they need to be stitched together to form a coherent target language sentence. In the EBMT system used in this thesis, the final target translation is obtained from these partial target translations with a beam-search decoder using a target Language Model.

In this thesis, we will look at how data sparsity in general and rare/out-of-vocabulary words in particular effect the translation model and the language model. Once we have shown that the effects are in fact substantial, we describe our contributions toward improving corpus based MT in data sparse conditions.

1.1 Why is *data-sparsity* a big challenge?

The main drawback of Data-driven approaches is the need for large amounts of data to function well [Munteanu and Marcu, 2005]. Since they do not use language specific knowledge they require data to be able to generalize to new test sentences or alternatively, to improve coverage. Obtaining such a large bilingual corpus is both expensive as well as time-consuming especially if humans are required to manually translate the source language half. The task can become extremely difficult when building translation systems between rare languages since such languages naturally have very few bilingual speakers. In order to understand how important data sparsity is, we refer to Figure 1.1 (from `http://en.wikipedia.org/wiki/List_of_languages_by_number_of_native_speakers`). Among the known languages (Figure 1.1), many languages have less than 1 million speakers (actually about 49% of the known languages).

Another possible way to obtain bilingual data (that is both cheap and fast) is to mine for parallel texts on the world wide web. For example, news articles in two different languages on two different websites that describe the same event can be sentence-aligned to obtain

- Low coverage for new test sentences
- Increase in OOV rate
- Alignment errors
- Unreliable Language Model estimates
- Low quality translations

Table 1.1: Issues that arise in data sparse conditions.

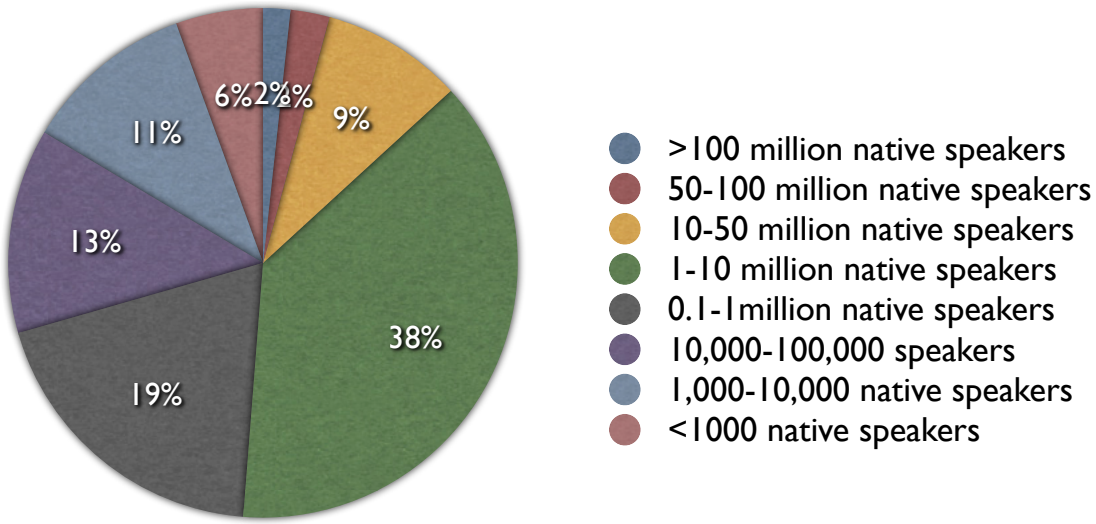

Figure 1.1: Percentage of languages based on number of native speakers

parallel texts. Unfortunately, not all languages have data on the web and rare languages are the most likely to fall into this category. Even languages that have more than 10 million speakers (like Hindi) are still considered as low-density languages as the amount of data on the web and the number of knowledge sources available are scarce. Thus, building MT systems for rare and low-density languages forces us to deal with issues that arise data sparse conditions (Table 1.1).

When OOV words are present, the translation model fails to obtain longer phrasal matches as it is forced to fragment the input sentence at OOV boundaries. The language model also produces unreliable estimates by backing off to the lower-order n-gram models while decoding. The degradation is caused not just by the OOV words but also by words that appear less frequently in the training data (or rare words). Since rare words appear in fewer contexts, the problems that exist with OOV words are also true for rare words. One way to handle such situations is to use other knowledge sources such as part-of-speech taggers, morphological analyzers (if a language is a highly inflected language) or synonyms. However, many low-density languages lack such resources.

In addition to OOV words, many other problems faced by conventional MT systems are exacerbated in data sparse conditions. For instance, when parallel data is limited, poor-alignments and typographical errors render the translation model incapable of obtaining good-quality target phrasal translations.

While we have so far discussed how data sparsity effects the translation model it turns out that having small test corpora degrades the performance of the MT system through the language model as well. Present statistical decoders place constraints on the amount of reordering that can be performed with the target fragments in order to reduce computational complexity. For example, if the output of the translation model has 10 fragments, then a complete decoder will have to find the best possible translation among $10!$ possible translations which is computationally infeasible. When training data is limited, even short test input sentences to be translated can slow down the target sentence generation. This is because the translation model may find only dictionary matches for each word in the test sentence or very short matches leading to many short fragments. To obtain reliable translation model and language model estimates under such conditions, finding long phrasal matches becomes crucial. Unfortunately, in general, longer phrasal matches can usually only be obtained by increasing the corpus size.

In this thesis, we will look at alternative techniques that allow us to obtain longer phrasal matches even in data sparse conditions. We modify the bilingual training corpus to enable longer source phrasal matches (and thus longer target phrasal matches) for new (or test) sentences from the training corpus. Table 1.2 shows the average length of the matching source phrases with respect to the training corpus (with English as the source language and Chinese as the target language) on a test set of 4000 sentences extracted with the EBMT system and using our technique (from Chapter 7) that enables retrieval of longer matches. For example, if we apply our technique on a 15k ($k = 10^3$) bilingual training data set, the average length increases to 2.32, which is even higher than what the baseline EBMT system would find with a larger data set of 30k (where the average length is 2.23)

Language-Pair	Training data size	Baseline	G-EBMT
Eng-Chi	15k	2.17	2.32
	30k	2.23	2.36
	200k	2.84	3.03

Table 1.2: Average length of the source phrases (length in terms of number of words) for various bilingual training data sets with English as the source language and Chinese as the target language.

1.2 Thesis Focus

This section briefly explains the focus of this thesis. The main goal of this thesis is to improve the translation quality in data-sparse conditions.

Find replacements for handling OOV and rare words Approaches in the past have suggested using synonyms [Marton et al., 2009], morphological analyzers [Habash, 2008] and part-of-speech taggers [Popovic and Ney, 2004] to handle OOV words. Previous approaches have also only concentrated on finding replacements for OOV words and not the rare words. In this thesis, Chapter 4 concentrates on finding replacements in situations where a language lacks such resources for both the OOV *as well as* the rare words.

Template Induction in the translation model Translation templates (or short reusable sequences) are generalizations of source and target sentences where sequences of one or more words are replaced by variables. Various methods have been proposed to create such templates in EBMT and differ in the way the templates are created. Templates were introduced in the early EBMT systems to handle target sentence generation in the absence of statistical decoders. We show that templates can still be used even in the presence of statistical decoders as they reduce the amount of pre-translated text required. Hence, templates are well suited for translation in data-sparse conditions (Chapters 5, 6 and 7).

For Template Induction: Unsupervised Clustering based on Context, semantic-similarity and syntax Template induction requires equivalence classes or clusters containing related phrase-pairs. A phrase here is not necessarily a linguistic unit. A phrase-pair contains a source phrase and its corresponding target phrasal translation. The best way to obtain reliable and good-quality clusters is to use human generated clusters. As with

obtaining parallel texts, obtaining these clusters can become time-consuming and difficult for a rare language where finding bilingual speakers is not easy. Instead, automatic clustering algorithms can be used to obtain these clusters. The quality of the templates generated depends on the quality of the clusters produced by the automatic clustering algorithm. Many powerful automatic clustering algorithms have been suggested in the past on simulated data (such as images in the field of Image Processing). This thesis focuses on identifying and clustering only the reliable units (words or phrases) for the purpose of generating templates. Contextual information (Chapter 5), semantic similarity (Chapter 7) and syntactical coherence (Chapter 6) are used as features for clustering.

For clustering: Automatic determination of number of clusters Another bottleneck in using automatic clustering tools is the determination of number of clusters (N). The number of clusters can be found empirically by evaluating the translation quality of a development set with each value of N. The value of N that gives the best translation quality score can then be chosen as the optimal value for the number of clusters. However, such an approach takes several days on corpus-based MT systems. Many approaches in the past have developed algorithms for automatically determining the number of clusters on simple simulated images. While applying these techniques to a real-world problem such as Machine Translation, various problems were encountered. This thesis provides algorithms to successfully obtain the optimum number of clusters (Chapters 5,6 and 7). We believe that these problems could arise in other practical systems and our algorithm would apply to those problems as well.

Template Induction in the language model Various class-based language models have been proposed in the past where words are grouped based on their POS tags or by automatically clustering words to create equivalences classes. Class-based models have shown to give better probability estimates for longer sequences of $n-$grams by making reasonable predictions for unseen histories by using the class information of all the words present in the histories. Conventional class-based language models require all words in the data to belong to a class. When errors (like, segmentation errors in Chinese, human translation errors, etc.) exist in the data, it is better not to cluster all the words in the data. This thesis finds unreliable words and does not consider them for the task of clustering. The model built with just these reliable data points will be referred to as the "template-based" model (Chapter 8) as the word sequences of $n-$grams are converted into sequences containing either the word or its class label (and not both). Note: the template-based model can be treated as a class-based model built by placing unreliable words that do not have a class (i.e., words that have not been included in the generated clusters) in unique classes

to form singleton clusters.

1.3 Organization of the Chapters in this Thesis

The organization of the chapters in this thesis is as shown in Figure 1.2. The thesis provides solutions to the negative effects that data sparsity has on EBMT. The EBMT system used to test the methods provided in this thesis is described in Chapter 3.

Chapter 4 explains the procedure adopted to handle out-of-vocabulary words and rare words. Chapter 5 explains how to obtain templates for the translation model by clustering word-pairs. This is extended to handle segment-pairs (or phrase-pairs) in Chapters 6 and 7. Chapter 8 applies templates to improve the language model (template-based language model: an extension of the Class-based language model).

In Chapter 5, word-pairs are clustered using the Spectral Clustering algorithm with contextual features. Chapter 6 clusters segment-pairs based on syntactic information obtained by chunking the source and target languages. Chapter 6 clusters segment-pairs based on their semantic-relatedness using a hierarchical clustering approach called the Weighted Single Linkage clustering algorithm. We then proceed to combine all these techniques from Chapters 4, 5, 6, 7 and 8, and build a combined model in Chapter 9 to see the overall benefit in translation quality in data sparse conditions.

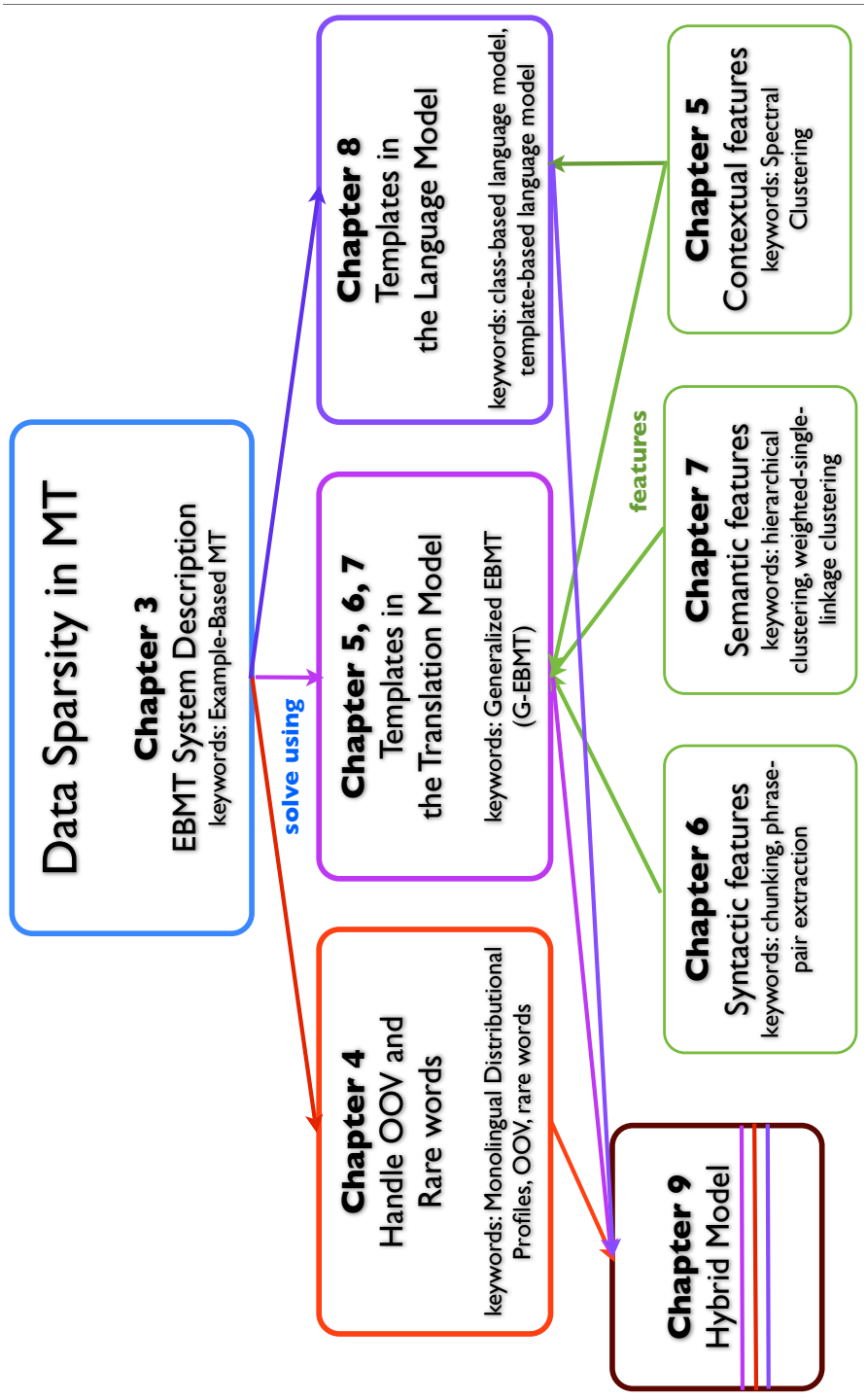

Figure 1.2: Organization of this thesis.

Chapter 2

Related Work

2.1 Current Paradigms in Machine Translation

Different machine translation techniques have emerged over time and vary in the way the problem is handled and modeled. They can be classified as linguistic-based or nonlinguistic-based. Techniques that combine linguistic and nonlinguistic approaches also exist. Today, there is no clear boundary between these techniques as all these techniques borrow ideas from each other.

2.1.1 Linguistics-based Paradigm

These approaches use strong linguistic knowledge for modeling the process. The analysis, transfer and generation is based solely on the knowledge that humans posses about a language. Transformation based linguistic approaches were built with the idea that a language has a basic structure for sentences which can be created by context free grammar rules and a lexicon. Constraint based linguistic approaches impose constraints on context free grammar rules. Rule-based Machine Translation is the most popular linguistic-based approach.

Rule-based Machine Translation (RBMT)

These systems use different levels of linguistic rules for translation. This paradigm includes transfer-based machine translation, interlingual machine translation and dictionary-based machine translation techniques. Transfer-based and interlingua-based techniques

have an intermediate representation that represents the meaning of the original sentence. The intermediate representation in interlingua-based MT is independent of the language pair, while in transfer-based MT, the representation has some dependence on the language pair. RBMT systems follow a series of steps to generate the target sentence. First, the input source text is analyzed morphologically, syntactically and semantically. Then, a series of structural transformations are applied based on structural rules or interlingua to generate the target text. A bilingual dictionary and grammar rules are used in these transformations which are developed by linguists. Hence, developing and maintaining these systems is time-consuming and expensive.

2.1.2 Nonlinguistic-based Paradigm

Existence of large bilingual-parallel corpora for many languages and powerful machines led to the development of these approaches. These corpus-based techniques derive knowledge from the parallel corpora to translate new sentences.

Statistical Machine Translation (SMT)

The idea was borrowed from Speech Recognition and is based on statistical prediction techniques. A translation model is learned from a bilingual-parallel corpus and a language model from the target monolingual corpus. The best translation is found by maximizing the translation and language model probabilities. These systems perform very well when large amounts of data are available. In word-based SMT [Brown et al., 1990], the translation elements are words and in phrase-based SMT ([Marcu and Wong, 2002];[Koehn, 2004];[Vogel et al., 2003];[Och and Ney, 2001]), the translation elements are phrases. Phrase-based SMT systems are more widely used than the word-based systems. In phrase-based SMT, sequences of source words or phrases are translated to sequences of target words. These phrases are not linguistic phrases and are extracted using statistical methods from the parallel corpus.

Since phrase-based SMT extracts all possible phrase-pairs (source phrase and corresponding target translation) from the bilingual training corpus as an off line process, storage space as well as time to retrieve the phrase-pairs during run time increases with larger amounts of training data. Hence, these systems place restrictions on the length of the phrase-pairs extracted by the translation model although longer phrases capture more context and improve translation quality. Others [Vogel et al., 2003] subsample the training corpus based on the test data.

Early SMT systems used the source-channel approach to find the best translation [Brown et al., 1990]. Other SMT systems modeled the posterior probability (or an inversion of the translation model) using the maximum entropy (log-linear models) approach ([Berger et al., 1996];[Och and Ney, 2001]) as an alternative to the source-channel approach which was also later suggested for other natural language understanding tasks [Papineni et al., 1998]. The advantage of such an approach is that many features can be used in the inverted translation model.

Example-based Machine Translation (EBMT)

These systems use bilingual-parallel texts as their main knowledge source and translate by analogy based on the belief that humans translate by first decomposing sentences into phrases and then join the target phrases to form the target sentence. Phrases are translated by analogy to previous translations and the translation unit in these systems is the sentence. A few EBMT systems use the parallel corpora directly during translation. If the translation of an identical source sentence is not present in the corpus, these systems find best matching sentence pairs and modify them to generate the target sentence. Many EBMT systems convert the bilingual corpora into templates or rules (Generalized EBMT) in order to help in target sentence generation where some words or phrases are replaced by variables on both the source and target sides. EBMT systems can also extract the implicit knowledge from a bilingual corpora in advance and then use a decoder to find the translations for new input source sentences from this knowledge. As many EBMT systems combine rule-based and data driven approaches, EBMT lies somewhere between RBMT and SMT (Figure 2.1).

Hence, all EBMT systems find examples (or sentence-pairs) that are similar to the test/input sentence from the training corpus during run time and perform some manipulation to obtain the best translation.

Remarks

Many current EBMT systems, including the system used in this thesis, now use statistical decoders (like SMT systems) to join partial candidate translations.

Like static phrase-based SMT, many EBMT systems [Brown, 1998] extract phrase-pairs once examples similar to the input test sentence are found and apply discriminative training approaches. However, static Phrase-based SMT extracts the phrase-pairs as an offline process and EBMT [Brown, 1998] performs the extraction dynamically during run time. The EBMT system used in this thesis finds examples that have source phrasal

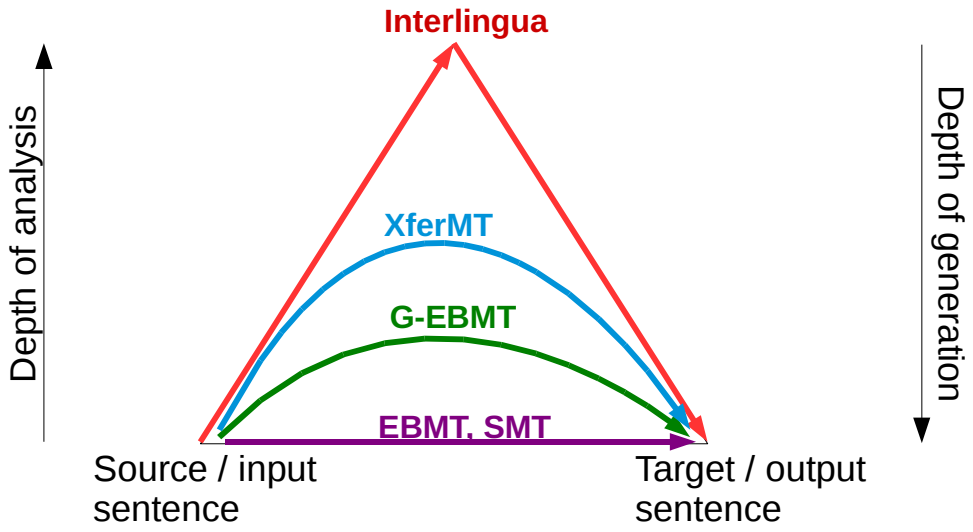

Figure 2.1: The Translation Pyramid

matches with the test sentence, extracts the corresponding translations (to form phrase-pairs) and then scores them. To overcome the space and time issues that Phrase-based SMT faces with large phrase-tables, a few Phrase-based SMT approaches now perform the extraction during run time for each test sentence ([Zhang and Vogel, 2005];[Callison-Burch et al., 2005]).

Unlike static phrase-based SMT, since the extraction of the phrase-pairs is done on the fly in EBMT, example-specific features can also be extracted i.e., it is possible to have the same phrase-pair extracted from different examples but with different scoring features. [Phillips, 2010] incorporates example-specific features into phrase-based SMT in an efficient way. Hence, as mentioned earlier, the boundary between phrase-based SMT and EBMT is shrinking.

2.1.3 Combining Linguistic-based and Non-linguistic-based Approaches

This paradigm involves mixing of the MT paradigms described above. To overcome many of the problems faced by nonlinguistic-based methods (like long range dependencies), linguistic knowledge (in the form of parse trees, part-of-speech tagging, etc.) is being adapted into these systems.

2.2 Generalized Example-based Machine Translation (G-EBMT)

The G-EBMT system uses templates as its basic unit as opposed to sentences in EBMT. Translation templates are generalizations of sentence-pairs where sequences of one or more words are replaced by variables. These templates bear resemblance to transfer rules used in Rule-Based Machine Translation systems [Lavie, 2008] but use fewer constraints.

Wang [1998] used shallow structures (sequences of non-terminal labels or $word$ class labels that give rise to terminals or words only) to improve the alignment model in SMT. Viterbi alignment is first performed on the training corpus. Then the source half of the training corpus is decomposed into ordered list of structures. With the help of the source sequence of structures and the alignment information, the target sentence is decomposed into a list of structures. The probability of the Viterbi alignment is then calculated with the structural information. The alignment with the greatest likelihood according to the structure-based model is chosen for collecting counts in their EM learning. Wang [1998] also shuffles words and phrases in the top K hypotheses of the decoder to generate more hypotheses with better scores. These new hypotheses are ordered based on their scores and the shuffling continues until there are no changes to the top K hypothesis. Thus, structural information is $only$ applied for shuffling phrases in the final K hypothesis generated by the decoder. Their system uses words as units of translation and not phrases, because of which adding structural information in to the decoder is not straight forward.

Generalization creating 'alignment templates' has been applied in Phrase-Based SMT as well [Och and Ney, 2004] using only part of speech tags or automatically clustered $words$ where all source and target words need to be clustered. In Och and Ney [2004], the probability of using an alignment template is calculated for every source phrase of a consistent phrase-pair in the training corpus in terms of its source and target class label sequences. During runtime, the test sentence is segmented into groups of consecutive words (or phrases) and the alignment templates are used to determine the target class sequence and hence the target candidate translation by replacing the target class labels by the translations of the words in the source phrase. Finally, the target candidate translations are reordered to find the best translation.

Syntax-based SMT [Yamada and Knight, 2001] also makes use of transfer rules (or channel operations) on parsed input sentences. However, in Syntax-based SMT, these transfer rules containing only non-terminal symbols are used for reordering the input parse tree (parse tree of the input sentence) to match that of the target language structure and lexical transfer rules are used for translating source words to their corresponding target words

to generate the translation. In EBMT, templates provide a way to order target phrasal matches that are not necessarily linguistics-based syntactic phrases and to increase coverage. In EBMT, lexical transfer rules consists only of terminal symbols and a generalization template contains variables and terminal symbols.

The G-EBMT also bears resemblance with the Hierarchical Phrase-based model in SMT [Chiang, 2005] which differs from the Syntax-based SMT approach in that the Hierarchical Phrase-based SMT uses synchronous Context-Free-Grammar which is not necessarily linguistic-based. The G-EBMT system has a shallow (or flat) structure and is not hierarchical as hierarchical models in our EBMT system tend to overgeneralize the system. Over-generalization not only produces huge target phrasal matches increasing the decoding time but can also reduce the quality of target translations produced. Hierarchical Phrase-based models [Chiang, 2005] place restrictions on the number of non-terminal labels that can be present in a phrase-pair, whereas, our G-EBMT system does not place any such constraints. Also, the G-EBMT system is more constrained than the Hierarchical model in Chiang [2005] which allows any non-terminal to be replaced by any phrase. Hence, G-EBMT lies somewhere between the Transfer-based MT approaches and EBMT (Figure 2.1).

2.3 Earlier Approaches

2.3.1 Handling out-of-vocabulary (OOV) and rare words

One of the main reasons for low quality translations is the presence of large number of OOV and rare words (low frequency words in the training corpus). Many of the present translation systems either ignore these unknown words or leave them untranslated in the final target translation. When data is limited, the number of OOV words increases.

The main reasons for OOV words are: limitations in the amount of training data, domain changes, morphology and spelling mistakes. Data-driven MT systems give surprisingly good results when made to work with languages that are not morphologically rich and perform poorly with highly inflected languages. If the number of OOV words in a sentence is either one or two, the output produced may still be understandable. When data is limited, the number of OOV words increases leading to the poor performances of the translation models and the language models due to the absence of longer sequences of source word matches and less reliable language model estimates.

Orthographic and morpho-syntactic preprocessing techniques on the training and test data has been shown to reduce OOV word rates. Popovic and Ney [2004] demonstrated this on Spanish, Catalan, and Serbian which are rich morphological languages in their SMT system. They introduced two types of transformations to the verbs to reduce the number of unseen word forms, one was the Base-POS representation and the other, Stem-suffix representation. In the Base-POS representation, the full form of the verb is replaced with its base form and the sequence of relevant POS tags. In the Stem-suffix representation, a list of suffixes that correspond to the set of relevant POS tags is defined and are split from the stem. Both their methods reduced the vocabulary size and the number of OOV words in the development and test corpus. Their experiments showed that the use of morphemes improves translation quality.

Yang and Kirchhoff [2006] proposed a backoff model for Phrase-based Statistical Machine Translation (PBSMT) that translates word forms in the source language by hierarchical morphological word and phrase level abstractions. They evaluated their model on the Europarl corpus for German-English and Finnish-English and showed improvements over the state-of-the-art phrase based models. A standard phrase table with full word forms is trained. If an unknown word is found in the test data, the word is first stemmed and the phrase table entries for words sharing the same root are modified by replacing the words with their stems. If a phrase entry or a single word phrase can now be found, the corresponding translation is used, otherwise the model backs off to the next level and applies compound splitting to the unknown word, splitting it into two parts. If there are no match-

ing phrase entries for either of the two parts, stemming is applied again and a match with the stemmed phrase table entries is carried out. If there are no matches found even in this stage, the word is passed on verbatim in the translation output. Since an off-the-shelf decoder was used, backoff was implicitly enforced by providing a phrase-table that included all backoff levels i.e., the phrase table included phrasal entries based on full word forms as well as stemmed and split counterparts. Their method showed improvements on small training sets.

Vilar et al. [2007], performed the translation process treating both the source and target sentences as a string of letters. Experiments were carried out on a small data set to mimic scarce conditions from Catalan to Spanish. The vocabulary size for a letter-based system is much smaller (around 70 including digits, whitespace, punctuation marks, upper and lowercase letters) when compared to a word-based system. Hence, there are no unknown words when carrying out the actual translation of a test corpus. The difference in BLEU score [Papineni et al., 2002] between the word-based and the letter-based system remained fairly constant. They also performed experiments by combining the word-based system and the letter-based system, where the word-based system did most of the translation work and the letter-based system translated the unknown words. The combined system led to improvements over their word-based system.

Habash [2008] addresses spelling, name-transliteration OOVs and morphological OOVs in an Arabic-English Machine Translation system. In MORPHEX, the OOV word is matched to an in-vocabulary (INV) word that could be a possible morphological variant of the OOV word. In SPELLEX, the OOV word is matched to an in-vocabulary (INV) word that could be a possible correct spelling of the OOV word. Phrases with the INV token in the phrase table of their PBSMT system are "recycled" to create new phrases in which the INV is replaced with the OOV word. Four types of spelling correction are used: letter deletion, insertion, inversion and substitution. In DICTEX, the phrase table is extended with entries from a manually created dictionary that contains English glosses of the Buckwalter Arabic morphological analyzer. In TRANSEX, English transliteration hypotheses are generated if the OOV word is assumed to be a proper name. The method is similar to the method used by Freeman et al. [2006] to select the best match from a large list of possible names in English. The list of possible transliterations was added to the phrase table as translation pairs. These pairs are assigned very low translation probabilities so that they don't interfere with the rest of the phrase table. The method was found to successfully produce acceptable translations in 60% of the cases. The results showed improvement over a state-of-the-art PBSMT system.

Outline of our approach in this thesis and Comparison to other work

This thesis presents a method in Chapter 4 inspired by the Context-based MT approach [Carbonell et al., 2006] that improves translation quality by extracting larger number of overlapping target phrasal candidates. Context-based MT does not require parallel text but it requires a large monolingual target language corpus and a full form bilingual dictionary. The main principle is to find those n-gram candidate translations from a large target corpus that contain as many potential word and phrase translations of the source text from the dictionary and fewer spurious content words. The overlap decoder combines the target n-gram translation candidates by finding maximal left and right overlaps with the translation candidates of the previous and following n-grams. Hence, only contextually confirmed translations are kept and scored by the overlap decoder. When the overlap decoder does not find coherent sequences of overlapping target n-grams, more candidate translations are obtained by substituting words or phrases in the target n-grams by their synonyms. The idea behind their approach is based on the distributional hypothesis which states that words with similar meanings tend to appear in similar contexts [Harris, 1954]. Their synonym generation differs from others ([Barzilay and McKeown, 2001];[Callison-Burch et al., 2006]) in that it does not require parallel resources. First, a list of paired left and right contexts that contain the desired word or phrase are extracted from the monolingual corpus. Next, the list is sorted and unified where a long paired context that occurs multiple times is ranked higher than the one without repeated occurrences. The same corpus is used to find other words and phrases that fit the paired contexts in the list. Finally, the new middles (or replacements) are ranked according to some criteria. Hence, their approach adopted synonym generation to find alternate translation candidates that would provide maximal overlap during decoding. Our method uses the same idea of clustering words and phrases based on their context but uses the clustered words as replacements for OOV and rare words on the source language side.

Marton et al. [2009] proposed an approach similar to Carbonell et al. [2006] to obtain replacements for OOV words, where monolingual distributional profiles for OOV words were constructed. Hence, the approach was applied on the source language side as opposed to Carbonell et al. [2006] which worked on the target language. Only similarity scores and no other features were used to rank the paraphrases (or replacements) that occurred in similar contexts. The high ranking paraphrases were used to augment the phrase table of PBSMT.

All of the previously suggested methods only handle OOV words (except Carbonell et al. [2006] which handles low frequency target phrases) and no attempt is made to handle rare words. Many of the methods explained above directly modify the training corpus

(or phrase table in PBSMT) increasing the size of the corpus. Our method clusters words and phrases based on their context as described by Carbonell et al. [2006] but uses the clustered words as replacements for not just the OOV words but also for the rare words on the source language side. Our method does not make use of any morphological analyzers, POS taggers or manually created dictionaries as they may not be available for many rare or low-resource languages. The translation of the replacements in the final decoded target sentence is replaced by the translation of the original word (or the source word itself in the OOV case), hence, we do not specifically look for synonyms. The only condition for a word to be a candidate replacement is that its left and right context need to match with that of the OOV/rare-word. Hence, the clustered words could have different semantic relations. For example,

$(cluster1)$:"laugh, giggle, chuckle, cry, weep"
where "laugh, giggle, chuckle" are synonyms and "cry, weep" are antonyms of "laugh".

Clusters can also contain hypernyms (or hyponyms), meronyms (or holonyms), troponyms and coordinate terms along with synonyms and antonyms. For example,

$(cluster2)$:"country, region, place, area, district, state, zone, United States, Canada, Korea, Malaysia".
where "country" is a hypernym of "United States/Canada/Korea/Malaysia". "district" is a meronym of "state". "United States, Canada, Korea, Malaysia" are coordinate terms sharing "country" as their hypernym.

2.3.2 Generalized Templates in the Translation Model

Three well known major problems that exist in EBMT systems are coverage [Brown, 2000], boundary definition and boundary friction ([Carl, 2001];[Carl et al., 2004]). Coverage is a measure of how well the system generalizes to unseen sentences. Boundary definition refers to the problem of deciding how to segment the source sentence into fragments, while boundary friction refers to the problem of deciding how to join and smooth the translations of the source fragments. Translation templates are generalizations of sentence pairs where sequences of one or more words are replaced by variables. Various methods have been proposed in the past to create such templates in EBMT [Carl et al., 2004] and differ in the way the templates are created. This section gives a brief survey of approaches adopted in the past to create generalized templates.

Earlier EBMT systems adopted templates to address the boundary friction (like, Ci-

cekli and Güvenir [1996]; Gangadharaiah and Balakrishnan [2006]) problem. The input source sentence is generalized and matching templates are found. If the system finds templates that match the input source template, the target language variables are replaced by the target translations of the words that were generalized in the input sentence. While some of the systems require a full template to match the input source sentence for target sentence generation (like, Güvenir and Cicekli [1998]; Cicekli and Güvenir [1996]), others adopted a fragmentary translation algorithm (like Kaji et al. [1992]) where the target sentence generation is similar to the generation approaches adopted in Transfer-based MT systems. Somers et al. [1994] suggest using 'hooks' using alignment information and Block [2000] uses a simple translation algorithm that can join only single variable target fragments.

One might wonder if templates are useful in EBMT systems that use statistical decoders to join partial target fragments. Since finding the best translation among all possible reordering of the target fragments is expensive, reordering constraints are placed on decoders. So templates can be used to improve translation quality with computationally restricted decoders. For language-pairs that have very different word orders, it is beneficial to extract longer phrasal matches from the translation model [like, [Gangadharaiah et al., 2010a];[Zhang and Vogel, 2005];[Callison-Burch et al., 2005]] and templates provide a way to generate longer target phrasal matches.

It is also crucial to get the boundary definition right if one needs to obtain good generalized templates. Most presently-used word alignment models are asymmetric i.e., a source word is aligned to at most one target word and one target word can be aligned to many source words. To make the models symmetric, alignment is done in both directions: source to target and target to source. These alignments are later combined to extract phrases. PESA Vogel [2005] treats the extraction process as a sentence splitting task where the best splitting is the one that maximizes the overall sentence alignment probability. Pharaoh [Koehn, 2004] extracts phrase pairs that have no words aligned to words outside the phrase pair boundary. Ma et al. [2007] suggested a chunker that is trained on bilingual information to obtain good source chunks using POS tags. Phrase-pairs generated with these phrase extraction models could be clustered based on context to create generalized templates. Since present phrase extraction techniques create a huge number of phrase-pairs due to many null alignment mappings (Eg. 9,466,096 phrase pairs from 200k data with Pharaoh), clustering phrase pairs is expensive. Hence, a selection criteria is required to select the highly reliable phrase-pairs. Also, finding the optimum number of clusters (N) [Brown, 2000, Gangadharaiah et al., 2006] is expensive where many translation experiments need to be carried out on different N on a development set, and additionally, this expense increases with the number of phrases to be clustered.

Outline of the template-based approaches in this thesis and Comparison to other work

In this thesis, we use knowledge about (i) phrase structure using chunk boundaries for the source and target languages (iii) semantic-relatedness of phrase-pairs (iii) contextual information, to create templates.

In Chapter 5, we use an automatic clustering algorithm previously applied in image segmentation tasks to cluster word-pairs that appear in similar contexts and show how the clustering algorithm is more powerful to other clustering techniques applied in natural language processing. Although the technique is powerful when the number of clusters is known beforehand, finding the number of clusters is not trivial in any clustering technique. As mentioned earlier, finding the number of clusters empirically is computationally expensive. We suggest a method to automatically find the number of clusters. We now proceed to outline our methods to cluster phrase-pairs as well.

We use a phrase extraction method in Chapter 6 that incorporates knowledge about source and target languages by using chunks (a group of words forming a linguistic unit) extracted from sentences. We use word alignment information to align the chunks and obtain consistent phrase-pairs. We call such phrase-pairs, $segment-pairs$, to distinguish them (phrases) from their usual definition as being made up of one or more words. In our work, segments are defined to be made up of one or more chunks. Using knowledge that chunks can be a unit of sentences and alignment information, we reduce the search space of possible phrase-pairs that can be extracted and this allows us to extract much longer consistent phrase-pairs.

Once these consistent segment-pairs have been extracted, they are clustered using their chunk label information. These clusters are then used for template induction. Güvenir and Cicekli [1998] used similar and dissimilar portions of sentences that addressed boundary definition to create templates. The method proposed by McTait [2001] is similar to that of Güvenir and Cicekli [1998], except that his method allows *m:n* mappings. These methods limit the amount of generalization while creating templates as they only depend on similar and dissimilar portions of sentence pairs and do not use any other information like statistical word-alignments or hand-made bilingual dictionaries and syntactic structures of sentence pairs. Kaji et al. [1992] proposed an approach that used phrase labels from parse trees to create templates, where the source and target phrases were coupled with the help of a bilingual word dictionary. In Block [2000], translation templates are generated by replacing a chunk pair (created from word-alignments) in another chunk pair by a variable if the former is a substring in the latter. As an alternative to generating statistical word alignments, [Carl, 2001] used bracketing to extract chunk pairs. The template generation

process is similar to that of Block [2000] except that more than one chunk pair can be replaced in a translation template.

Gaijin [Veale and Way, 1997] performs phrase chunking based on the marker hypothesis for boundary definition which states that natural languages are marked for grammar at the string level by a closed set (prepositions, determiners, quantifiers, etc.). The best target segment for each source segment is found based on a matching criteria. All well-formed segment mappings between the source and target sentences are variabilized to create a translation template. An input source sentence is translated with the template that matches the segment structure of the input sentence. "Grafting" is performed for phrasal mismatches between the template and the input source sentence to be translated where an entire phrasal segment of the target is replaced with another from a different example. "Keyhole surgery" is performed for lexical mismatches where individual words in a target segment are replaced or morphologically fine-tuned to suit the current translation.

Brown [2001] proposes a recursive transfer-rule induction process which combines the idea put forward by Güvenir and Cicekli [1998] and word clustering [Brown, 2000] to create templates. However, in Brown [2000], experiments need to be carried out on a tuning set to determine the optimum value of number of clusters (N), where the value of N that gives the highest translation score on the tuning set is regarded as the best value. Finding N is expensive as for every value of N, the system has to be tuned and this can take several days for each value of N.

In Phillips [2007], structural templates made up of sequences of POS tags are used just before decoding a lattice of phrasal target translations to obtain new partial phrasal translations. All partial POS sequences that match the input sentence to be translated are retrieved. Their corresponding target POS sequences determined with the help of alignment links between lexical source and target tokens form templates for new phrasal translations. Lexical translations present on a lattice are substituted into the structural templates to form new phrasal translations which are then decoded by an SMT-like decoder. Hence, the method uses a more general structure made up of just POS tags than Veale and Way [1997]. It requires POS tagging for all the source and target sentences present in the training corpus and a structural source index for retrieving partial POS sequences of the input sentence that needs to be translated. Kim et al. [2010] not only extracts phrasal translations from sentence-pairs containing partial source phrasal matches of the test sentence but also extracts additional new partial phrasal translations that are limited to chunk boundaries.

The translation algorithm in Block [2000] can only join single variable target fragments. Although the method proposed in Kaji et al. [1992] makes use of syntactic phrases from parse trees, the templates created are less controllable as the method collapses words and phrases only by POS and linguistic phrase labels.

Templates in the EBMT system can also be generated by grouping paraphrases to form equivalence classes. Many data-driven approaches have been proposed in the past to generate paraphrases which can be used in a number of Natural Language Processing tasks, like question answering, summarization, etc. In Barzilay and McKeown [2001], multiple translations were used for generating paraphrases where a corpus containing two or more English translations of five classic novels was used. The sentences were first aligned by applying sentence alignment techniques. Paraphrases were then extracted from the sentence-aligned corpus by equating phrases which were surrounded by identical words. One disadvantage with this technique is that it relies on multiple translations which can be a rare resource for many languages.

Quirk et al. [2004] applied statistical machine translation tools to generate paraphrases of input sentences in the same language. Sentences which were paired using string edit distance were treated as a parallel corpus for monolingual MT. The procedure used in SMT on bilingual corpora was applied on monolingual corpora containing English sentences aligned with other English sentences. An automatic word aligner generated correspondences between words. Non-identical words and phrases that were connected by word alignments were treated as paraphrases.

Callison-burch [2007] described a technique for automatically generating paraphrases using a bilingual parallel corpora. To extract English paraphrases, they look at the foreign language phrases the English phrase translates to, find all occurrences of those foreign phrases and then all the English phrases they originated from. These extracted English phrases were treated as potential paraphrases. Paraphrases were applied to source language phrases that were unseen in the training data. If the translation of a phrase was not previously learned but its synonymous phrase was learned, then the unseen phrase was replaced by its paraphrase and translated. The number of paraphrases is increased by using many other parallel corpora to create the source language paraphrase model. Spanish-English and French-English translation tasks were also tested. Spanish paraphrases were created using, Spanish-Danish, Spanish-Dutch, Spanish-Finnish, Spanish-French, Spanish-German, Spanish-Greek, Spanish-Italian, Spanish- Portuguese and Spanish-Swedish parallel corpora. French paraphrases were also created in a similar way. Augmenting a PB-SMT system with paraphrases led to improved coverage and translation quality.

In this work, we extend the method proposed by Callison-burch [2007] and use it in a different aspect. We use paraphrases in Chapter 7 to generate equivalence classes and hence create templates in an EBMT system.

2.3.3 Generalized Templates in the Language Model (Template-based Language Models)

Short reusable sequences ("templates") whose elements are words, POS tags or equivalence classes can also be used in the Language Model (Chapter 8). In many MT systems, sophisticated tools are developed to make the translation models stronger but rely on unreliable word-based language models that may not be well suited for translation tasks.

Various class-based language models have been proposed in the past where words are grouped based on their POS tags or by automatically clustering based on their contextual information. Class-based models make reasonable predictions for unseen histories by using the class information of all the words present in the histories.

The conventional n-gram model is based on the assumption that the i^{th} word w_i depends on only the $(n-1)$ preceding words. The n-gram word-based language model can be described as follows,

$$p(w_i|h) = p(w_i|w_{i-1}, ..., w_{i-n+1}) \tag{2.1}$$

The above probabilities can be easily estimated using Maximum Likelihood [Jelinek, 1997]. Although higher values of n capture the underlying characteristics of a language, higher order n grams do not occur frequently, leading to inaccurate probability estimates [Stanley and Goodman, 1996]. Generally, a small value of n (typically 3 or 4, and occasionally up to 5 or 6) is chosen based on the size of the data available. To overcome sparseness in the n-gram probability estimates, smoothing or discounting strategies [Stanley and Goodman, 1996] are applied to give a better estimate for previously unseen n-grams. Even with smoothing, probability estimates are not always reliable.

In order to make reasonable predictions for previously unseen histories, words are assigned to classes [Brown et al., 1992] as some words are related to each other either in their meaning or syntactic information. An n-gram class-based model where the words are clustered into classes using a function Π which maps w_k to class c_k can be described as,

$$p(w_i|h) = p(w_i|c_i) \ \times \ p(c_i|c_{i-1}, ..., c_{i-n+1}) \tag{2.2}$$

Class-based models require all words present in the training data to be clustered. For languages such as Chinese where words are not separated by spaces segmentation has to be performed as a preprocessing step. Present segmenters report accuracies between 85% to 90% [Peng et al., 2004]. Clustering words present in the inaccurately segmented data can lead to unreliable clusters and thus inaccurate probability estimates which inturn lead to low quality translations.

Outline of the our approach in this thesis

We modify the class-based approach so that,
(i) it does not require all words to be clustered into classes, which helps in situations where a small set of highly reliable clusters (for example, days of the week, months of the year, etc.) are present in hand,
(ii) good quality templates are obtained as only reliable clusters are used otherwise the template-based system grounds out to word-based modeling and hence, the probability estimates are better.

It should be noted that the template-based model is equivalent to a class-based model formed by placing each of the words that were not clustered for building the template-model in a unique class, leading to singleton clusters for unreliable words. However, the template-based approach does not require a list of all these singleton clusters to build the model. Hence, the template-based model does not require re-creating the list of clusters when data for building the model changes. In the template-based model, word sequences of $n-$grams are converted into short reusable sequences containing either the word or its class label but not both. When hand-made clusters are not available, automatic clustering tools can be used to obtain clusters, but our task here is to only cluster reliable words. This work addresses the question of how to automatically generate clusters that contain only (or mostly) reliable words that are well suited for MT decoding tasks.

Our method is similar to the factored language models (FLMs) [Kirchhoff and Yang, 2005], where a word can be represented by features like POS tags or other linguistic information. This results in a model space that is extremely large as there are $m!$ possible backoff paths with m conditioning factors. FLMs use Genetic algorithms to learn the structure in this huge search space. Our model is a much simpler version of FLMs where we use one extra feature other than the word itself and the backoff procedure adopted is fixed and not learnt.

In this thesis, the algorithms adopted to improve MT in data-sparse conditions and their corresponding experiments are done in an EBMT system, but can be easily extended to other data-based methods (like PBSMT).

Chapter 3

System Description

3.1 EBMT System Description (Panlite)

This chapter gives a brief description of the EBMT system (generalized-EBMT in Figure 3.1) during training and runtime. The EBMT system, `Panlite` [Brown, 1998], used in this thesis (Figure 3.3), like other EBMT systems, requires only a large bilingual/parallel corpus of the language-pair under consideration. The system can use other sources like a root/synonym list, a set of equivalence classes (like days in a week, months in a year, etc) for template induction, a list of words that can be inserted or ignored during alignment. These lists are provided in a configuration file.

3.2 During Training

A dictionary is first obtained from the training corpus using a correspondence table [Brown, 1997] and then a thresholding filter is used to filter out the less likely translations (Figure 3.3). The correspondence table is a two-dimensional matrix with one of the dimensions corresponding to all the source words and the other dimension corresponding to all the target words in the corpus. From each sentence-pair, counts for every possible source-target word-pairing is incremented in the correspondence table. Monolingual occurrence counts for the unique source ($count(W_s)$) and unique target words ($count(W_t)$) are also incremented. The monolingual counts will be used for filtering.

After all the sentence-pairs are processed, the correspondence table is filtered using symmetric and asymmetric tests. The tests use two threshold functions: the first threshold

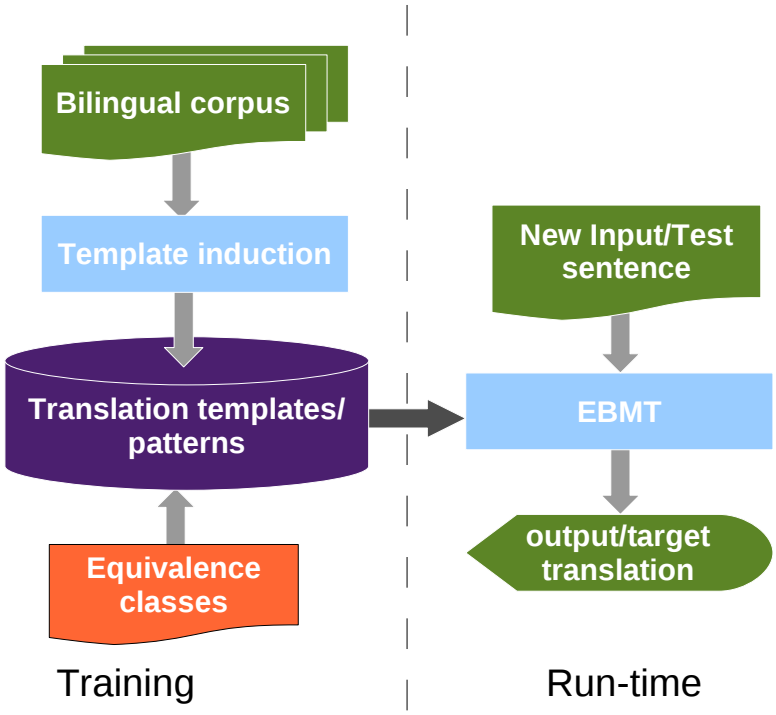

Figure 3.1: Generalized Example based Machine Translation.

function is a step function which gives a high threshold for co-occurrence counts below some value and a constant threshold for the others. The second function is a decaying function which sets a threshold of 1 for co-occurrence count of 1 and decays rapidly with increasing co-occurrence count. Any elements in the matrix that do not pass the two tests are set to zero. All elements (or word-pairs) that have a non-zero value are later added to the dictionary. The symmetric test is passed if:

$$\begin{aligned} C(W_s, W_t) &\geq threshold(C) * count(W_s) \;\&\\ C(W_s, W_t) &\geq threshold(C) * count(W_t) \end{aligned} \tag{3.1}$$

where, $C(W_s, W_t)$ is the co-occurrence count for (W_s, W_t) word-pairing. The asymmetric

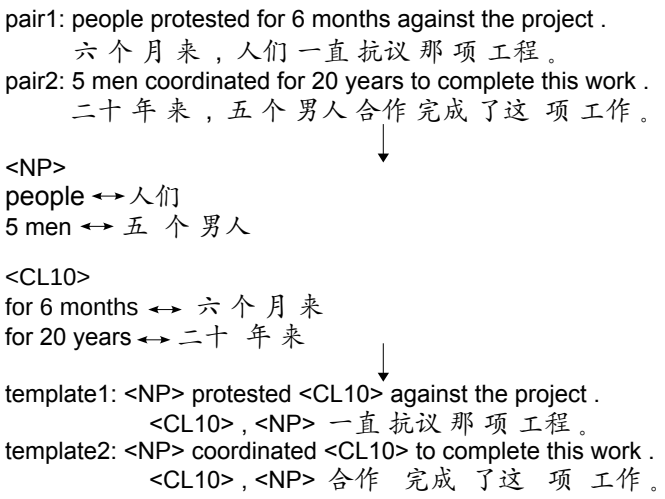

Figure 3.2: Usage of Templates.

test uses two thresholds ($thresh1$ and $thresh2$) and is passed if:

$$\begin{aligned}
&\Big[C(W_s, W_t) \geq thresh1(C) * count(W_s)\ \&\\
&\qquad C(W_s, W_t) \geq thresh2(C) * count(W_t)\Big]\\
&\qquad\qquad OR\\
&\Big[C(W_s, W_t) \geq thresh2(C) * count(W_s)\ \&\\
&\qquad C(W_s, W_t) \geq thresh1(C) * count(W_t)\Big]
\end{aligned} \tag{3.2}$$

The asymmetric test is used to handle words that are polysemic in one language.

The word-alignment algorithm then creates a correspondence table for every sentence-pair with the help of a bilingual dictionary. A matrix is constructed by looking up a bilingual dictionary for the translations of each source word on the source side of the sentence-pair. If any translation of a source word (say s_i) exists on the target side (say t_j) of the sentence-pair, then the corresponding entry in the matrix (i.e., $M_{i,j}$) is filled. The matrix is then processed to remove unlikely correspondences using a few heuristics. For example, typically a linguistic unit (like a noun phrase, prepositional phrase, etc.) in the source language (or target language) will correspond to a single unit in the target language (or source language) even when the source and target languages have very different word orders, hence, the words that make up a linguistic unit tend to appear together rather

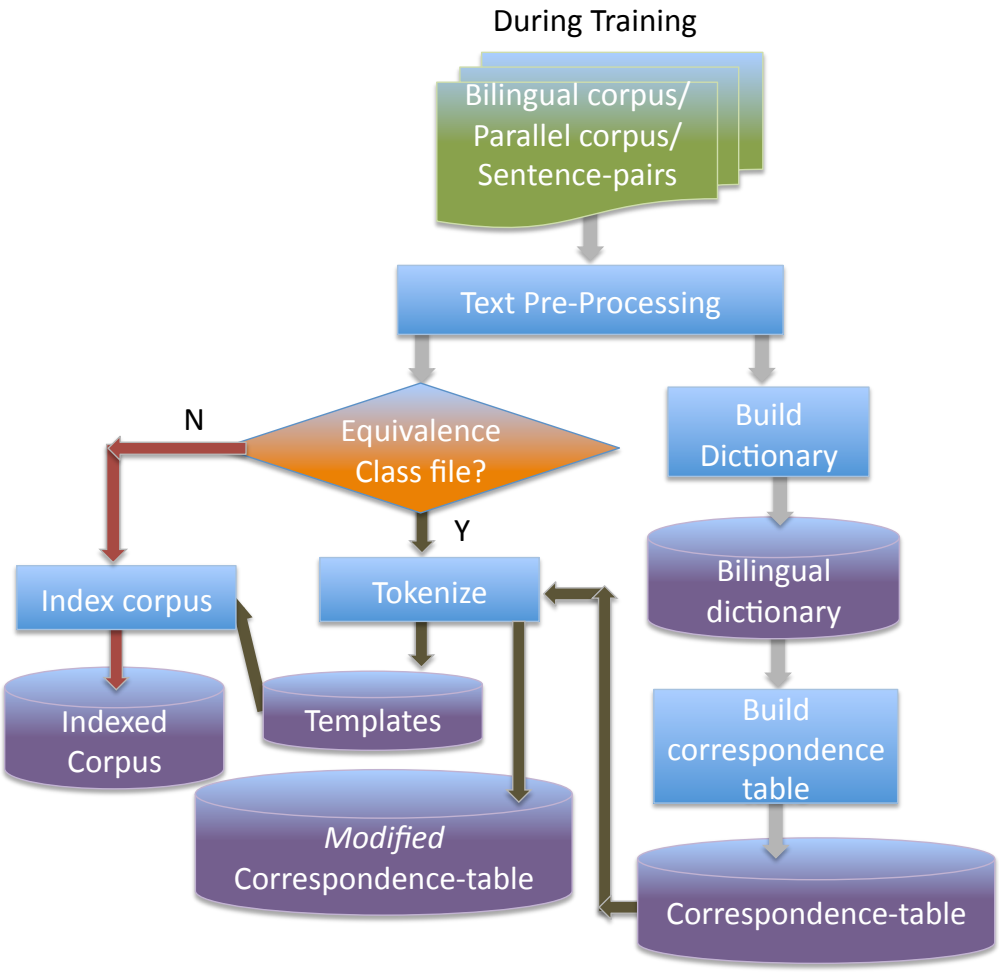

Figure 3.3: System Description: During Training.

than spread around. For each word, an expected range for the translation is computed and correspondences outside the range are removed (for more details [Brown, 1997]). The computation of the correspondence table is done while indexing the corpus. Indexing is performed using a variant of the Burrows-Wheeler-Transform [Brown, 2004]. BWT is an algorithm that was originally used in data compression. It is a block-sorting transformation which groups elements of its input lexically. Lookups are faster with the BWT index than an inverted index as BWT groups all instances of an $n-$gram together. Once the training

data is indexed, the BWT contains the necessary information to retrieve training instances that have matching phrases of the input sentence (sentence to be translated) without requiring any additional space and can be instantiated in a compressed form. BWT is applied to a suffix array to turn it into a self-index, which allows the original text to be discarded thus saving space.

If the configuration file is provided with a list of equivalence classes (for example, <NP> and <CL10> in Figure 3.2), the sentence-pairs in the corpus are tokenized during the indexing process. While tokenizing the corpus, if a word or phrase on the source side of the sentence-pair belongs to any of the equivalence classes, it is replaced by the class label only if the translation of the word in the equivalence class appears on the target side of the sentence-pair. Hence, each sentence-pair is converted into the most general template (*template1* and *template2* in Figure 3.2) where all those phrase-pairs in the sentence-pair that have been clustered are replaced by their class labels. The replacement is usually done in the reverse order of length starting from longer phrases. If two overlapping phrases (overlapping partially or if one of the phrases subsumes the other shorter phrase either on the source side or target side of the sentence-pair) need to be generalized, then the phrase-pair with the most number of alignment correspondences (between the source and target half of the phrase-pair) is generalized. If these overlapping phrases have the same number of correspondences, then the phrase-pair that appears first (from left to right) is generalized. The correspondence table for the sentence-pair is also modified by collapsing the word-alignment scores for every generalized source and its corresponding target phrase by an alignment score of 1.0.

The resulting indexed corpus and the correspondence table are used during run time.

3.3 During Run-Time

Next, we move on to the online phase where a new input sentence needs to be translated. The system fetches all sentence-pairs that have partial source phrasal matches with the input test sentence from the indexed corpus. If the test sentence matches an indexed source sentence completely, the corresponding target sentence is submitted as the translation. If not completely found, the phrase extractor extracts partial target phrases from sentence-pairs that contain partial source phrasal matches with the test sentence.

The phrase extractor in the system finds consistent minimum and the maximum possible segments of the target sentence that could correspond to the source fragment and gives a score to every possible sub-segment in the maximum segment containing the minimum segment. The system tries to find anchor words. An anchor word is a source word that

has only one possible translation listed and is also the only translation of its translation. If no anchors are found, the sentence-pair is discarded from consideration as un-alignable and the system continues with the rest of the sentence-pairs that contain source phrasal matches with the test sentence. If anchors have translations that are too far apart in the target sentence, the sentence-pair is discarded from further consideration. The target phrase containing all the anchors forms the minimum translation segment. The substring that contains the minimum segment and left and right adjacent words that have no correspondences outside the matched source fragment forms the maximal translation segment. The sub-segments with the highest score are selected as the target phrasal matches for the source fragment. The scoring function uses a weighted sum of a few test functions. The weights can be changed for differing lengths of the source fragments. The test functions include number of source (or target) words that have correspondences in the target (or source) fragment, words in the source or target that have no correspondences, difference in the lengths of the source and the target fragments, etc.

The above step may result in a large number of target fragments for source phrases that occur very frequently in the training corpus. Hence, this huge search space can slow down the decoder. A parameter called 'Maximum Alternatives' is used to limit this search space where only the most likely target fragments are sent to the decoder. 'Maximum Alternatives' is typically set to 25. The decoder then selects 'Max-Alts' alternatives while decoding from the target fragments received. The decoder selects the 'Max-Alts' alternatives using overall scores assigned by the engine, contextual features (sentence as well as phrase level), quality features and future utility estimates. 'Max-Alts' is a parameter in the system that can be tuned. In our experiments, the optimal value for 'Max-Alts' was between 6 to 8.

If the configuration file is provided with a list of equivalence classes, the input sentence is processed from left to right, looking at all possible ways of generalizing and not just the most general form. Hence, the input sentence is converted into a lattice of all possible generalizations. An example is given in Figure 3.4. The example shows sample clusters and the resultant lattice. If the extension involves generalization, the value of the label (translation obtained from the target side of the equivalence-class member) is stored. Partial source-target matches (or phrase-pairs) for this generalized test sentence are obtained as was done with the ungeneralized test sentence.

These target fragments that contain generalizations can be viewed as the right-hand-sides of context free grammar production rules (R). The grammar ($G =< T, PT, R >$) consists of terminal symbols (T: which can be words or phrases of the target language), pre-terminal symbols (PT: which are like the non-terminal symbols except that they always rewrite as terminals) and production rules (R).

Test/Input Sentence:
the session opened at 2 p.m.

Sample Clusters:
(format : source phrase ↔ target phrase)

<C1>:
2 ↔ 2
3 ↔ 3
….

<C2>:
2 p.m. ↔ 2 heures
4 p.m. ↔ 4 heures
…....

<C3>:
session ↔ sèance
meeting ↔ réunion
….....

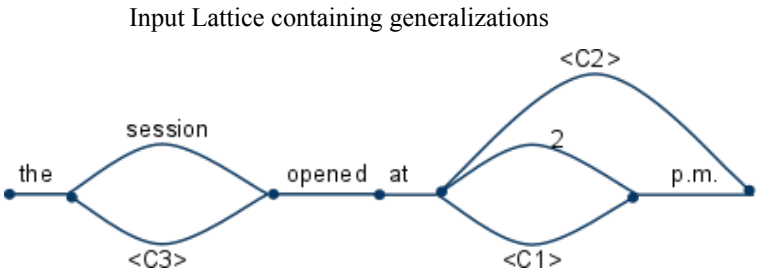

Figure 3.4: Tokenizing the test sentence

Once the target fragments are obtained for all the selected source phrases from the indexed training corpus they are further processed to put back the actual values (i.e., translations of the generalized source words) of the class labels that were stored earlier. The lexicalized translation candidates are then placed on a common decoding lattice which are then translated by a statistical decoder using a target language model. It should be noted that the extraction of target fragments for all possible source phrases is similar to the phrase extraction in Phrase-based Statistical Machine Translation, however, in the latter, the extraction is performed offline and the extracted phrase-pairs are stored as a phrase-table. The best translation is then determined by a statistical decoder which joins the partial translations with the help of a target language model. The decoder performs a multi-level beam search based on the weights on candidate translations and a target language model to select the best translation. The decoder maintains stacks of best translations based on the number of source words translated. A limit (which is tuned) is also placed on the stack

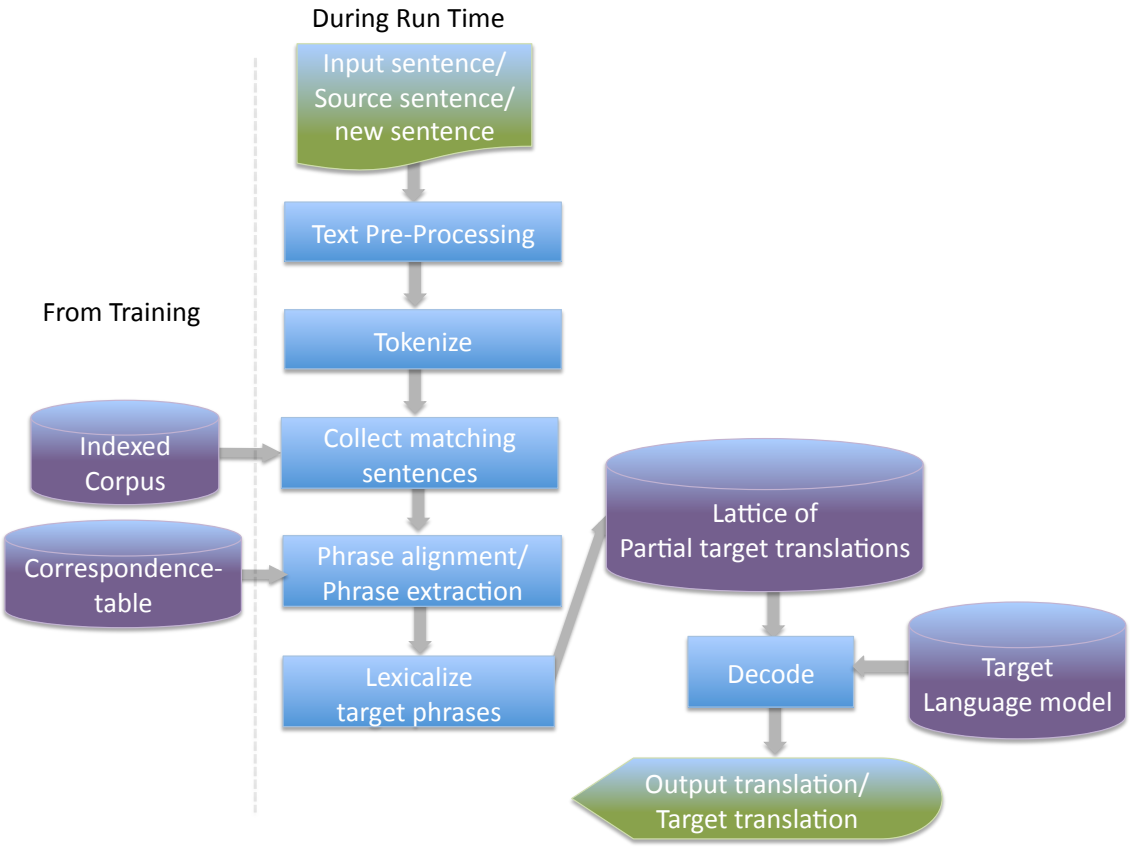

Figure 3.5: System Description: During Testing or Run Time.

size (Beam) to speed up the decoder. There is also a limit on the amount of reordering that can be performed with the fragments. The reordering window (which is also tuned) is defined as the number of source words skipped when joining two translation fragments of the source phrases out of order. The total score for a path is given by,

$$\begin{aligned} total\ score &= \frac{1}{n}\sum_{i=1}^{n}[wt_1 * log(b_i) + wt_2 * log(pen_i) + wt_3 * log(q_i) \\ &+ wt_4 * log(P(w_i|w_{i-2}, w_{i-1})] \end{aligned}$$

where n: number of target words in the path, wt_j: importance of each score, b_i: bonus factor, pen_i: penalty factor, $P(w_i|w_{i-2}, w_{i-1})$: Language Model (LM) score.

The Translation Model (TM) assigns a quality score (q) to each candidate translation which is computed as a log-linear combination of its alignment score and translation probability. The alignment score indicates the engine's confidence that the right translation has been generated. The translation probability is calculated as the proportion of times each alternative translation was generated while aligning all matches retrieved for that particular source phrase. Each candidate translation is weighted by giving bonuses for longer phrases and penalties for length mismatches between the source phrase and the candidate translation. Bonuses are also given for paths that have overlapping fragments. Generalization penalties based on the proportion of words generalized in a path are also used. Generalization penalties are used for the following reason. If there are two candidate translations: one generated by a lexical source phrase and the other by a source phrase containing generalizations resulting in the same source phrase, then the translation extracted for the lexical source phrase is favored. The weights are tuned using coordinate ascent to maximize the BLEU score on a tune set.

3.4 Data

Two language-pairs, English-French (Eng-Fre) and English-Chinese (Eng-Chi) are used to perform the experiments. As our aim is to increase coverage and translation quality when small amounts of data are available, the training data chosen for both the language-pairs are small. For Eng-Chi, three sets of size 15k, 30k and 200k sentence pairs from the FBIS data (NIST, 2003) were selected as training data. Two sets of size 30k and 100k from the Hansard corpus [LDC, 1997] were selected for the experiments with Eng-Fre. To tune the EBMT system, a tuning set of 500 sentences was chosen for both the language-pairs. The test data consisting of 4000 sentences were selected randomly from the corpus. As the test data was extracted from the parallel corpus, only one reference file was used. There was no overlap between the test, training and tune data. The target half of the training data was used to build 4-gram language models with Kneser-Ney smoothing. The value of n for the n-gram language models was not tuned, instead the same value of n was used for building all the language models. The data was segmented using the Stanford segmenter [Tseng et al., 2005]. Additionally, English-Haitian [CMU, 2010] language-pair (Eng-Hai) of 1619 sentence-pairs from the medical domain is used in Chapter 5. The tune and the test sets contained 200 sentence-pairs only, the rest of the data was used for training. Chapter 4 also uses Eng-Hai with 15,136 sentence-pairs from the newswire data. The tune set contained 500 sentence-pairs, the test set contained 4000 sentence-pairs and the remaining data was used for training. Any changes to the test or the tune data will be mentioned in the appropriate chapters.

3.5 Evaluation Methodology

To assess the translation quality, automated evaluation metrics are used in thesis. 4-gram word-based BLEU (BiLingual Evaluation Understudy) is used for Eng-Hai and Eng-Fre. 4-gram word-based and character-based (in Chapter4) BLEU [Papineni et al., 2002] are used for Eng-Chi.

The BLEU metric was chosen since it is widely used in the machine translation community. It requires one or more reference files to evaluate the candidate translations. Number of matching $n-$grams between the candidate and the reference translations are obtained and used to compute the precision score. While computing the modified precision (p_n), every $n-$gram match with respect to the reference translations, is truncated to not exceed the largest count of the $n-$gram in any reference file. Since the modified $n-$gram precision decays exponentially with n, a weighted (w_n) average of the logarithm of the modified precisions is used (this is equivalent to the geometric mean of the modified $n-$gram precisions). Since multiple reference translations are used, finding recall is not straight forward. Instead a brevity penalty (BP) as a multiplicative factor is used to penalize the obtained precision score when the candidate translation (with length $= c$) is shorter than the references (with length $= r$). The brevity penalty is computed as,

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

The final BLEU score is given by,

$$\begin{aligned} BLEU &= BP * e^{(\sum_{n=1...N} w_n * log(p_n))} \\ log(BLEU) &= min(1 - \frac{r}{c}, 0) + \sum_{n=1...N} w_n * log(p_n) \end{aligned}$$

Results of most of the experiments are reported in terms of the BLEU score. Translation Edit Rate (TER) and NIST [Doddington, 2002] metrics are also occasionally used. TER [Snover et al., 2006] is defined as the number of edits needed to change a candidate translation to exactly match one of the reference translations, normalized by the average length of the references. Edits include: insertions, deletions, substitutions of words and shifts of word sequences. All the edits have equal cost.

$$TER = \frac{Number\ of\ edits}{Number\ of\ reference\ words} \tag{3.3}$$

The NIST metric also uses modified $n-$gram precisions. The NIST metric weights by inverse conditional probability (last word of $n-$gram given all previous words in $n-$gram).

It should be noted that the NIST metric puts much more weight on $uni-$grams and $bi-$grams than BLEU. The information weights are computed over the reference translations as,

$$Info(w_1, w_2, ..., w_n) = log_2\Big[\frac{Number\ of\ occurrences\ of\ w_1, w_2, ...w_{n-1}}{Number\ of\ occurrences\ of\ w_1, w_2, ..., w_n}\Big] \tag{3.4}$$

The NIST score is given by,

$$NIST = \sum_{n=1,..N}\Big[\frac{\sum_{all\ w_1,..w_n\ that\ co-occur} Info(w_1, ..w_n)}{\sum_{all\ w_1,..w_n\ in\ candidate}(1)}\Big] * exp\Big[\beta * log_2(min(\frac{c}{L_{ref}}, 1))\Big] \tag{3.5}$$

where, β is used to make the BP = 0.5 when the number of words in the candidate translation (c) is $2/3^{rds}$ the average of the number of words in the reference translation. L_{ref} is the average number of words in a reference translation, averaged over all the references.

In the experiments, the test sentences are further split into 10 files and the Student's paired t-test and the Wilcoxon Signed-Rank test [Wilcoxon, 1945] are used to find the statistical significance.

The Student's t-test uses the Student's t-distribution for finding the statistical significance when the sample size is small. In our experiments the sample size chosen is 10. Since the data used to perform the test has to be sampled independently (otherwise t-tests give misleading results), we divide our test data of 4000 test sentences into non-overlapping 10 individual groups of 400 sentences each.

Our null hypothesis is that the difference in translation scores between System A (Baseline) and System B (System B is obtained with some modification to System A) has a mean of zero (i.e., there is no improvement to the system with the modification). The $t-$value is calculated as follows:

$$t = \frac{\bar{X}_D - \mu_o}{SD/\sqrt{(n)}}$$

where, $\bar{X}_D$ is the average of the differences of the BLEU scores of System A and System B obtained on the 10 (n) files. Similarly, SD is the standard deviation of the differences of the two systems. μ_o is a non-zero value for checking if the difference is larger than μ_o (i.e., significantly larger than zero). With the computed t, a p value can be found from the $t-$table. If the p value is lower than a threshold chosen (like, 0.001, 0.05 etc.) then the null hypothesis is rejected.

The sample means need to be normally distributed and the sample variance has to follow the χ^2 distribution for the t-test. Even if the random variable corresponding to the BLEU scores are not normally distributed, from the Central Limit Theorem, the sample means (with large number of samples) can be approximated by a normal distribution. However, the variance may not follow a χ^2 distribution if the random variable is not normally distributed. It has been shown that if the sample size is large, the sample variance does not affect the test. Hence, if the random variable is not normally distributed and the if the sample size is small, the t-test can give misleading results. So we also perform a non-parametric statistical hypothesis test called the Wilcoxon Signed rank test which does not make assumptions about the distribution of the data.

In the Wilcoxon Signed rank test, the null hypothesis being tested remains the same i.e., the difference in translation scores between System A (Baseline) and System B (System B is obtained with some modification to System A) has a mean of zero. The absolute values of the difference in BLEU scores between System A and System B on all the 10 test files are obtained and ranked (smallest difference receives a rank of 1). If the difference is tied for a few test files, then the mean rank is assigned. Also, differences that are equal to zero are not ranked. W_+ holds the sum of all the ranks of positive deviations (i.e., Bleu Score(B) $>$ Bleu Score (A)) and W_- holds the sum of all the ranks of negative deviations (i.e., Bleu Score(B) $<$ Bleu Score (A)). S is defined as the value of W_+. The value of S is compared to the Wilcoxon Table to obtain the p value. p : is defined as the probability of attaining S from a population of scores that is symmetrically distributed around the central point. The central point under the null hypothesis is expected to be zero. The table gives a critical value for different sample sizes and their p values for one-tailed and two-tailed tests. If the value of S is smaller than the critical value under a particular threshold (typically used One-tailed significance levels are: 0.025, 0.01 and 0.005; Two-tailed significance levels are: 0.05, 0.02 and 0.01) then the conclusion can be made that the improvements of System B over System A was unlikely to occur by chance.

Chapter 4

Handling Out-of-Vocabulary and Rare words

Out-of-vocabulary (OOV) words present a significant challenge for Machine Translation and other Language Technology tasks such as, Speech Recognition, Text mining, etc. Presence of OOV words and rare words in the input sentence prevents a machine translation system from finding longer source and target phrasal matches and produces low quality translations due to less reliable language model estimates. For low-resource languages, limited training data increases the frequency of OOV words and this degrades the quality of the translations.

Past approaches have suggested using stems or synonyms for OOV words. Unlike the previous methods, we show how to handle not just the OOV words but *rare* words as well. Our method requires only a monolingual corpus of the source language to find candidate replacements. The replacements found are not necessarily synonyms. A new framework is introduced to score and rank the replacements by efficiently combining features extracted for the candidate replacements. A lattice representation scheme allows the decoder to select from a beam of possible replacement candidates.

The main idea for adopting our approach is the belief that the EBMT system will be able to find longer phrasal matches and that the language model will be able to give better probability estimates while decoding if it is not forced to fragment text at OOV and rare-word boundaries. The new framework does show this behavior and gives statistically significant improvements in English-Chinese and English-Haitian translation systems.

4.1 Motivation for using semantically-related words as candidate replacements

As mentioned earlier, approaches suggested in the past handled OOV words by replacing the OOV words by synonyms or stems (from morphological analyzers). Transliteration hypotheses can be generated if the OOV word is assumed to be a proper name. Typically, finding morphological replacements involves a set of rules for breaking down an inflected word (or the surface form) to its stem and morphemes. For languages that lack such resources, closely related words that share at least n contiguous characters (for example, "activating" and "activated" share 7 contiguous characters "a,c,t,i,v,a,t") can be obtained from the available monolingual source data. We performed an analysis to see how well the above described methods would handle OOV words.

For this, we simulated sparsity by obtaining a small set of 30,000 randomly selected English sentences from the FBIS corpus. We chose English as the source language to perform this analysis for two main reasons (i) the examples can be easily illustrated (ii) all the translation experiments in this thesis including this chapter use English as the source language. Words in a test set of 4000 randomly selected sentences that did not appear in the 30,000 set were treated as OOV words. 200 OOV words were randomly chosen to perform the analysis.

Each of the 200 OOV words were classified manually as one of the following (i) Numbers (ii) Typos (iii) Abbreviations (iv) INF(F): indicating an inflected OOV for which at least one replacement that shared the same stem was found in the 30,000 sentences (v) INF: indicating an inflected OOV for which no replacements (i.e., no word shared the same stem) were found in the 30,000 sentences (vi) PN: proper nouns (vii) Other: indicating an adjective or a noun. Figure 4.1 shows the number of OOVs placed in each of the seven categories.

A few examples of OOVs placed in each of the seven categories are given below:
(i) Numbers: `1932` (year), `7.32%`, `170,000`
(ii) Typos: `tajikstan` (should have been `tajikistan`)
(iii) Abbreviations: `legco` (for `legislative council`), `afb` (for `Air Force base`)
(iv) INF(F): `activating`
(v) INF: `intimidation`
(vi) PN: `jianzhong`
(vii) Other: `basketball`, `humankind`

A reminder, our aim is to prevent the translation model and the language model from

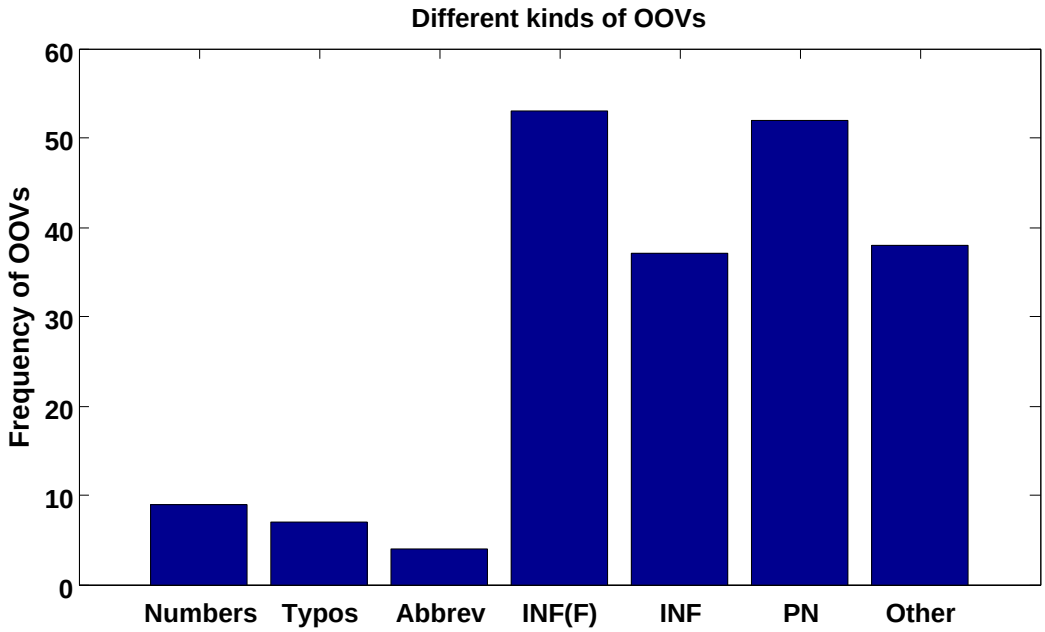

Figure 4.1: Kinds of OOV words.

fragmenting the test sentence at OOV boundaries and hence, our approach is to pre-process the test sentences replacing OOV words by their replacements. If a language lacks a morphological analyzer or a synonym generator, replacements can only be found for 56% of the OOVs which includes: *category iv* containing INF(F), *category i* can be replaced by any other number, *category vi* by any other frequently appearing proper noun, hoping that the replacement may appear in the same context as that of the OOV to obtain longer matches for the translation task. However, none of the previously suggested approaches can be used to find replacements for the OOVs in the rest of the categories (i.e., 44% of the OOVs). Hence, we use an approach to find replacements that are not necessarily synonyms or related in stem and perform post-processing on the final target translation to incorporate the actual OOV word or rare word [Gangadharaiah et al., 2010b].

4.2 OOV and Rare words

Words in the test sentence (new input sentence to be translated) that do not appear in the training corpus are called OOV words. Words in the test sentence that appear less than k

times in the training corpus are considered as rare words ($k = 3$).

4.3 Finding candidate replacements

The method presented in the following sections holds for both OOV as well as rare words. In the case of rare words, the final translation is post-processed (Section 4.3.7) to include the translation of the rare word. Only a large monolingual corpus is required to extract candidate replacements. To retrieve more replacements, the monolingual corpus is pre-processed by first generalizing numbers, months and years by NUMBER, MONTH and YEAR tags, respectively.

The procedure adopted will be explained with a real example T (the rest of the sentence is removed for the sake of clarity) with `hawks` as the OOV word,

T:a mobile base , *hitting three* **hawks** *with one arrow* over the past few years ...

4.3.1 Context

As the goal is to obtain longer target phrasal translations for the $test\ sentence$ before decoding, only words that fit the left and right context of the OOV/rare-word in the test sentence are extracted. Unlike Marton et al. [2009], where a context list for each OOV is generated from the contexts of their replacements, this thesis uses only the left and right context of the OOV/rare-word. So, a single left and right context is used in our framework knowing that a far smaller number of replacements will be extracted making the approach computationally cheaper.

The default window size for the context is five words (two words to the left and two words to the right of the OOV/rare-word). If the windowed words contain only function words, the window is incremented until at least one content word is present in the resulting context. This enables one to find sensible replacements that fit the context well. The contexts for T are:

Left-context (L): hitting three
Right-context (R): with one arrow

The above contexts are further processed to generalize the numbers by a $NUMBER$ tag to produce more candidate replacements. The resulting contexts are now:

Left-context (L): hitting $NUMBER$
Right-context (R): with $NUMBER$ arrow

As a single $L - R$ context is used, a far smaller number of replacements are extracted.

4.3.2 Candidate replacements

The monolingual corpus (ML) of the source language is used to find words and phrases (X_k) that fit LX_kR i.e., with L as its left context and/or R as its right context. The maximum length for X_k is set to 3. The replacements are further filtered to obtain only those replacements that contain at least one content word. As illustrated earlier, the resulting replacement candidates are not necessarily synonyms.

4.3.3 Features

A local context of two to three words to the left of an OOV/rare-word ($word_i$) and two to three words to the right of $word_i$ contain sufficient clues for the word, $word_i$. Hence, local contextual features are used to score each of the replacement candidates ($X_{i,k}$) of $word_i$. Each $X_{i,k}$ extracted in the previous step is converted to a feature vector containing 11 contextual features. Certainly more features can be extracted with additional knowledge sources. The framework allows adding more features, but for the present results, only these 11 features were used.

As our aim is to assist the translation system in finding longer target phrasal matches, the features are constructed from the occurrence statistics of $X_{i,k}$ from the bilingual training corpus (BL). If a candidate replacement does not occur in the BL, then it is removed from the list of possible replacement candidates.

Frequency counts for the features of a particular replacement, $X_{i,k}$, extracted in the context of $L_{i,-2}L_{i,-1}$ (two preceding words of $word_i$) and $R_{i,+1}R_{i,+2}$ (two following words of $word_i$) (the remaining words in the left and right context of $word_i$ are not used for feature extraction) are obtained as follows:

f_1: frequency of $X_{i,k}R_{i,+1}$
f_2: frequency of $L_{i,-1}X_{i,k}$
f_3: frequency of $L_{i,-1}X_{i,k}R_{i,+1}$
f_4: frequency of $L_{i,-2}L_{i,-1}X_{i,k}$

f_5: frequency of $X_{i,k}R_{i,+1}R_{i,+2}$
f_6: frequency of $L_{i,-2}L_{i,-1}X_{i,k}R_{i,+1}$
f_7: frequency of $L_{i,-1}X_{i,k}R_{i,+1}R_{i,+2}$
f_8: frequency of $L_{i,-2}L_{i,-1}X_{i,k}R_{i,+1}R_{i,+2}$
f_9: frequency of $X_{i,k}$ in ML
f_{10}: frequency of $X_{i,k}$ in BL
f_{11}: number of feature values $(f_1, ..f_{10}) > 0$

f_{11} is a vote feature which counts the number of features (f_1 ... f_{10}) that have a value greater than zero. Once these feature vectors have been obtained for all the replacements of a particular OOV/rare word, the features are normalized to fall within $[0, 1]$. The sentences in ML, BL and test data are padded with two begin markers and two end markers for obtaining counts for OOV/rare-words that appear at the beginning or at the end of a test sentence.

4.3.4 Representation

Before we go on to explaining the lattice representation, we would like to make a small clarification in the terminology used. In the MT community, a lattice usually refers to the list of possible partially-overlapping target translations for each possible source $n-$gram phrase in the input sentence. Since we are using the term lattice to also refer to the possible paths through the input sentence, we will call the lattice used by the decoder, the "*decoding lattice*". The lattice obtained from the input sentence representing possible replacement candidates will be called the "*input lattice*".

An input lattice (Figure 4.2) is constructed with a beam of replacements for the OOV and rare words. Each replacement candidate is given a score (Eqn 4.1) indicating the confidence that a suitable replacement is found. The numbers in Figure 4.2 indicate the start and end indices (based on character counts) of the words in the test sentence. In T, two replacements were found for the word `hawks`: `homers` and `birds`. However, `homers` was not found in the BL and hence, it was removed from the replacement list.

The input lattice also includes the OOV word with a low score (Eqn 4.2). This allows the EBMT system to also include the OOV/rare-word during decoding. In the Translation Model of the EBMT system, this test lattice is matched against the source sentences in the bilingual training corpus. The matching process would now also look for phrases containing `birds` and not just `hawks`. When a match is found, the corresponding translation in the target language is obtained through sub-sentential alignment. The scores on the input lattice are later used by the decoder. Each replacement $X_{i,k}$ for the OOV/rare-word

T : a mobile base , hitting three hawks with one arrow .....

input lattice:

0	0	(" a ")
1	6	(" mobile ")
7	10	(" base ")
11	11	(",")
12	18	(" hitting ")
13	17	(" three ")
18	22	(" *hawks* " 0.0026)
18	22	(" **birds** " 0.9974)
23	26	(" with ")
27	29	(" one ")
30	34	(" arrow ")

Figure 4.2: Lattice of the input sentence T containing replacements for OOV words.

($word_i$) is scored with a logistic function [Bishop, 2006] to convert the dot product of the features and weights ($\vec{\lambda} \cdot \vec{f_{i,k}}$) to a score between 0 and 1 (Eqn 4.1 and Eqn 4.2).

$$p_\lambda(X_{i,k}|word_i) = \frac{exp(\vec{\lambda}\vec{f_{i,k}})}{1+\sum_{j=1...S} exp(\vec{\lambda}\vec{f_{i,j}})} \tag{4.1}$$

$$p_\lambda(word_i) = \frac{1}{1+\sum_{j=1...S} exp(\vec{\lambda} \cdot \vec{f_{i,j}})} \tag{4.2}$$

where, $\vec{f_{i,j}}$ is the feature vector for the j^{th} replacement candidate of $word_i$, S is the number of replacements, $\vec{\lambda}$ is the weight vector indicating the importance of the corresponding features.

4.3.5 Tuning feature weights

We would like to select those feature weights ($\vec{\lambda}$) which would lead to the least expected loss in translation quality (Eqn 4.3). Negative logarithm of the $BLEU$ score [Papineni et al., 2002] is used to calculate the expected loss over a development set. As this objective function has many local minima and is piecewise constant, the surface is smoothed using the L2-norm regularization. Powell's algorithm [Powell, 1964] is used to find the best weights. 7 different random guesses are used to initialize the algorithm.

$$\min_{\lambda} E_{\lambda}[L(t_{tune})] + \tau * ||\lambda||^2 \tag{4.3}$$

The algorithm assumes that partial derivates of the function are not available. Approximations of the weights ($\lambda_1, ..\lambda_N$) are generated successively along each of the N standard base vectors. The procedure is iterated with a stopping criteria based on the amount of change in the weights and the change in the loss. A cross-validation set (in addition to the regularization term) is used to prevent over-fitting at the end of each iteration of the Powell's algorithm. This process is repeated with different values of τ, as in Deterministic Annealing [Rose, 1998]. τ is initialized with a high value and is halved after each process.

4.3.6 Decoding

The target translations of all the source phrases are placed on a common decoding lattice. An example of a decoding lattice for example T is given in Figure 4.3. The system is now able to find longer matches (`three birds with one arrow` and `three birds`) which was not possible earlier with the OOV word, `hawks`. The local ordering information between the translations of `three birds` and `with one arrow` is well captured due to the retrieval of the longer source phrasal match, `three birds with one arrow`. Our ultimate goal is to obtain translations for such longer $n-$gram source phrases boosting the confidence of both the translation model and the language model.

As mentioned in Section 3.1, the total score (TS) for a path (Eqn 4.4) through the translation lattice is the arithmetic average of the scores for each target word in the path. If the path includes a candidate replacement, the log of the score, $p_{\lambda}(w_i)$, given for a candidate replacement is incorporated into TS as an additional term with a weight wt_5.

$$\begin{aligned} TS = \frac{1}{t} \sum_{i=1}^{t} [& wt_1 \log(b_i) + wt_2 \log(pen_i) \\ & + wt_3 \log(q_i) + wt_4 \log(P(w_i|w_{i-2}, w_{i-1})) \\ & + \mathbb{1}_{(w_i = replacement)} \ wt_5 \log(p_{\lambda}(w_i)) \ ] \end{aligned} \tag{4.4}$$

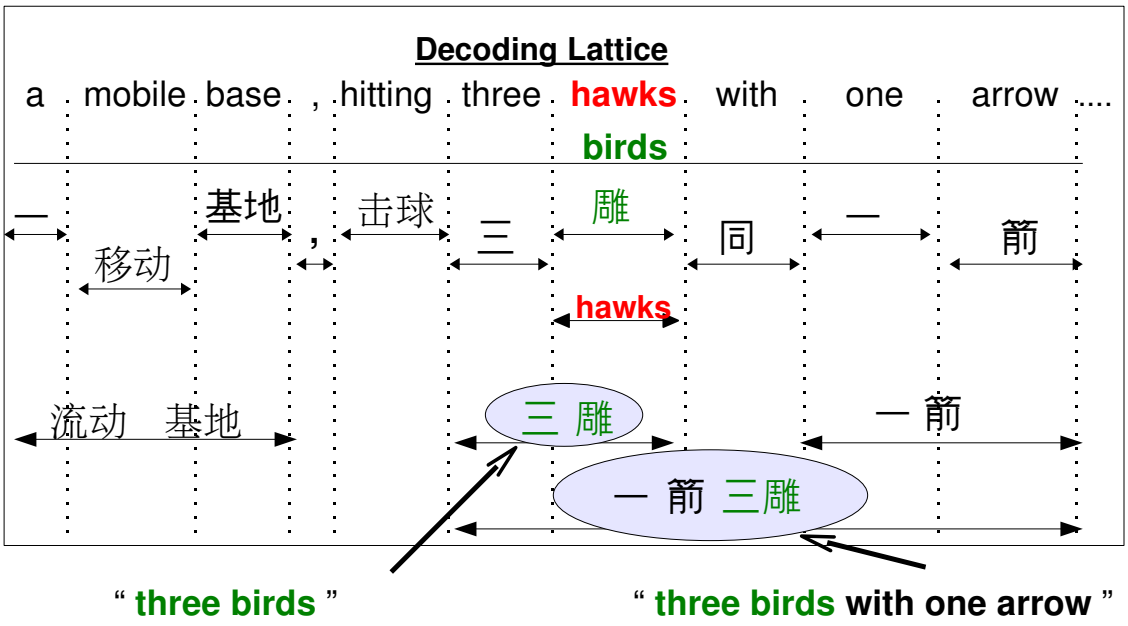

Figure 4.3: Lattice containing possible phrasal target translations for the test sentence T. Shows a long phrase found by the TM: for the source phrase `three birds with one arrow`, the translation in Chinese *lit.* means `one arrow three birds`

.

where, t is the number of target words in the path, wt_j indicates the importance of each score, b_i is the bonus factor given for long phrasal matches, pen_i is the penalty factor for source and target phrasal-length mismatches, q_i is the quality score and $P(w_i|w_{i-2}, w_{i-1})$ is the language model score. The parameters (wt_j) are tuned on a development set.

4.3.7 Post-processing

The target translation is post-processed to include the translation of the OOV/rare-word with the help of the best path information from the decoder. In the case of OOV words, since the translation is not available, the OOV word is put back into the final output translation in place of the translation of its replacement. In the output translation of the test example T, the translation of `birds` is replaced by the word, `hawks`.

For our example T, the translation obtained was (the rest of the sentence is removed for the sake of clarity): . . . 击球 一 箭 三 雕 . . .

The best path from the decoder tells us that 雕 came from the replacement, birds. Hence, this translation is then replaced by the OOV word itself, hawks.
... 击球 一 箭 三 hawks ...

For rare words, knowing that the translation of the rare word may not be correct (due to poor alignment statistics), the target translation of the replacement is replaced by the translation of the rare word obtained from the bilingual dictionary (details on dictionary extraction is in Chapter 3). If it was known that the replacement used for the rare or the OOV word is a synonym, we could keep the translation of the replacement as it is in the output translation. However, since we do not make use of any sources to indicate whether the replacement is a synonym, we replace the translation of the replacement by the translation of the rare word. If the rare word has multiple translations, the translation with the highest score is chosen.

4.4 Training and Test Data sets

The data sets used in this chapter differ from the data sets explained in Chapter 3. The performance was tested on two language-pairs, English-Chinese (Eng-Chi) and English-Haitian (Eng-Hai). Two training data sets of 30,000 and 200,000 sentence-pairs were used for Eng-Chi (details in Section 3.4). The (Eng-Hai) newswire data (Haitian Creole, CMU, 2010) containing 15,136 sentence-pairs was also used. For the monolingual English corpus, 9 million sentences were collected from the Hansard Corpus [LDC, 1997] and the FBIS data. The EBMT system that did not handle OOV/rare-words is chosen as the Baseline system. The parameters of the EBMT system are tuned with 500 sentence pairs for both Eng-Chi and Eng-Hai. The tuned EBMT parameters are used for the Baseline system and the system with OOV/rare-word handling. The feature weights for the method are then tuned on a separate development set of 200 sentence-pairs with source sentences containing at least 1 OOV/rare-word. The cross validation set for this purpose is made up of 100 sentence-pairs. In the OOV case, 500 sentence pairs containing at least 1 OOV word are used for testing. For the rare word handling experiments, 500 sentence pairs containing at least 1 rare word are used for testing.

4.5 Results

To assess the translation quality, 4-gram word-based BLEU is used for Eng-Hai and 3-gram word-based BLEU is used for Eng-Chi. Since the tune sets (200 sentence-pairs)

OOV/Rare	system	TER	BLEU	NIST
OOV	Baseline	0.7789	0.1861	4.8525
	Handling OOV	0.7695	0.1932	4.9664
Rare	Baseline	0.7423	0.2284	5.3803
	Handling Rare	0.7402	0.2312	5.4406

Table 4.1: Comparison of translation scores of the Baseline system and the system handling OOV and Rare words for Eng-Hai. Statistically significant improvements with $p < 0.0001$. *Note*: The test sets for handling OOV words is different from that used to handle rare words.

were very small, 4-gram BLEU scores for Eng-Chi were very low due to very few 4-gram matches with respect to the reference, hence we chose 3-grams for BLEU. The test data used for comparing the system handling OOV words and its Baseline (without OOV word handling) is different from the test data used for comparing the system handling rare words and its Baseline system (without rare word handling). In the former case, the test data handles only OOV words and in the latter, the test data only handles rare words. Hence, the test data for both the cases do not completely overlap. As we are interested in determining whether handling rare words in test sentences is useful, we keep both the test data sets separate and assess the improvements obtained by only handling OOV words and by only handling rare words over their corresponding Baselines. In Section 9.1.4, we use one test set of 4000 test sentences to handle both OOV and rare words to see the overall gain.

For both Eng-Chi and Eng-Hai experiments, only the top C ranking replacement candidates were used. The value of C was tuned on the development set and the optimal value was found to be 2. Translation quality scores (TER, NIST and BLEU) obtained on the test data with 30k and 200k Eng-Chi training data sets are given in Table 4.2. Table 4.1 shows the results obtained on Eng-Hai. Statistically significant improvements ($p < 0.0001$) were seen by handling OOV words as well as rare words over their corresponding baselines.

4.6 Analysis

4.6.1 Sample Replacement Candidates

Sample replacements found are given in Figure 4.4. As mentioned earlier, the replacements are not necessarily synonyms.

OOV/Rare	Training data size	system	TER	BLEU	NIST
OOV	30k	Baseline	0.8203	0.1412	4.1186
	30k	Handling OOV	0.8097	0.1478	4.1798
	200k	Baseline	0.7941	0.1990	4.6822
	200k	Handling OOV	0.7766	0.2050	4.7654
Rare	30k	Baseline	0.8209	0.1536	4.3626
	30k	Handling Rare	0.8002	0.1603	4.4314
	200k	Baseline	0.7804	0.2096	4.9647
	200k	Handling Rare	0.7735	0.2117	5.0122

Table 4.2: Comparison of translation scores of the Baseline system and system handling OOV and Rare words for Eng-Chi. Statistically significant improvements over the Baseline with $p < 0.0001$ on all three metrics.

4.6.2 Number of OOV words

The test set described in Chapter 3 was chosen for this analysis (instead of selecting only those test sentences that contain at least one OOV or rare word). We counted the number of OOV words present in the test sets (of 4000 sentences) for the Eng-Fre and Eng-Chi language-pairs (subplots A and C in Figure 4.5). We also counted the number of test sentences that contained at least one OOV word (subplots B and D in Figure 4.5). A large number of OOV words are seen in the Eng-Fre test set (with 72065 number of words) as the corpus is more diverse than the Eng-Chi test set (with 137478 number of words). 30% of the Eng-Chi test sentences contain OOV words when 15k training data is chosen. In the Eng-Fre case, about 70% of the test sentences contain OOV words when 30k training data is chosen. This gets worse if we choose French as the source language with 87% of the test sentences containing OOV words. We would expect more number of OOV words when highly inflected source languages (such as, Arabic, Urdu, Hebrew, etc) are chosen.

4.6.3 Length of target phrases

As the goal of the approach was to obtain longer target phrasal matches, we counted the number of target phrases of length n for each value of n present on the decoding lattice in the 30k Eng-Chi case. The subplots: A and B in Figure 4.6, show the frequencies of target phrases for higher values of n (for $n > 5$) when handling OOV and rare words. The plots clearly show the increase in number of longer target phrases when compared to the

OOV/Rare word	Candidate Replacements
Spelling errors	
krygyzstan	kyrgyzstan,...
yusukuni	yasukuni,..
kilomaters	kilometers, miles, km, ...
	Coordinate terms
somoa	india, turkey, germany, russia, japan,...
ear	body, arms, hands, feet, mind, car, ...
buyers	dealer, inspector, the experts, smuggler,.
	Synonyms
plummet	drop, dropped, fell, ...
	Synonyms and Antonyms
optimal	worse, better, minimal,....

Figure 4.4: Sample English candidate replacements obtained.

phrases obtained by the baseline systems.

Since the BLEU and NIST scores were computed only up to 3-grams, we further found the number of n-gram matches (for $n > 3$) in the final translation of the test data with respect to the reference translations (subplots: C and D). As expected, a larger number of longer $n-$gram matches were found. For the OOV case, matches up to 9-grams were found and the number of $n-$gram matches for n greater than 3 was increased by 26% (from subplot C of Figure 4.6).

A simple approach to improve translation quality by handling both OOV and rare words was adopted in this chapter. The framework allowed scoring and ranking each replacement candidate efficiently. We also showed how the method improved translation quality on two language-pairs with statistically significant improvements. The results also showed that rare words also need to be handled to see improvements in translation quality.

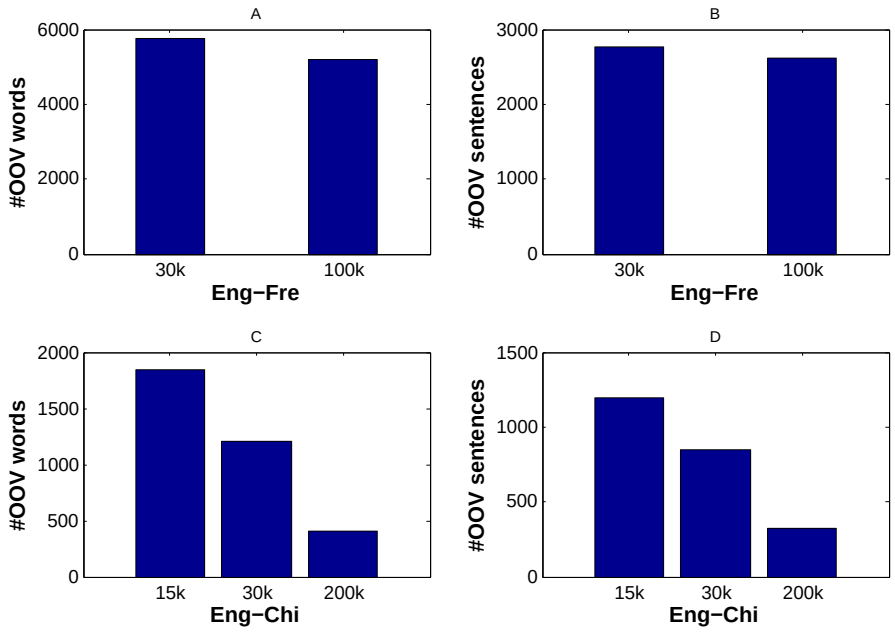

Figure 4.5: A: Number of OOV words in the Eng-Fre test set with 30k and 100k training data sets, B: Number of sentences containing at least one OOV word in the Eng-Fre test set with 30k and 100k training data sets, C: Number of OOV words in the Eng-Chi test set with 15k, 30k and 200k training data sets, D: Number of sentences containing at least one OOV word in the Eng-Chi test set with 30k and 100k training data sets.

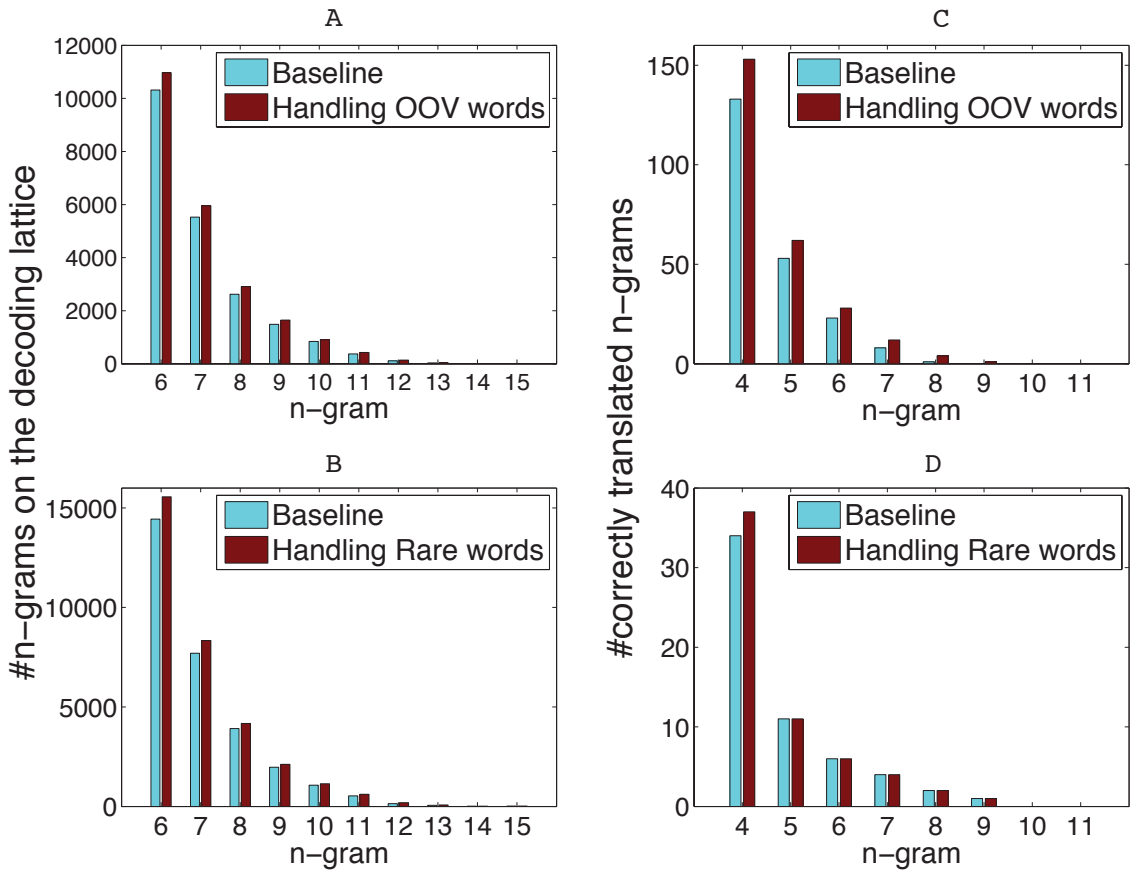

Figure 4.6: A, B: number of target phrases found for increasing values in length, n, on the decoding lattice. C, D: number of target n-gram matches for increasing values of n with respect to the reference translations.

Chapter 5

Templates in the Translation Model: using word-pairs

To generate word-generalized templates fully automatically, we need an automatic clustering algorithm to produce clusters. Several clustering algorithms have been proposed to group words into classes to obtain *word*-generalized templates. Ideally, one would like to use an algorithm that is simple in design, should be able to produce pure clusters, and should have some way of automatically determining the number of clusters. Many approaches in the past suggested grouping words/phrases that appeared in similar contexts between groups of sentence-pairs. While others adopted an approach that extracted syntactically related phrase-pairs (source phrase and its corresponding target phrase) with the help of a parser and an alignment tool. The first approach used context as its criterion to cluster words/phrases, where sequences of words in the contexts (left-hand sides and the right-hand sides) of the members to be grouped had to completely match in order and this limits the number of words that can be clustered and hence the amount of generalization that can be performed. The second approach only used the bracketing obtained by parsers and ignored any context information. Like Brown [2000], we use an approach similar to the first approach but our method relaxes the complete-match constraint by defining term vectors that capture frequency information for the words in the context of the word-pair under consideration. This relaxation enables us to cluster a large number of words. A combination of the two approaches (syntax and context) is suggested in Chapter 7.

Among the various well known clustering algorithms, spectral clustering based approaches [Ng et al., 2001] have been successful in areas such as image processing. They can be designed easily and are successful in many applications. In this chapter, we show that spectral clustering can be successfully applied to create templates in EBMT systems

as well. This chapter shows that Spectral clustering is superior to Group Average Clustering (GAC) [Brown, 2000] both in terms of semantic similarity of words falling in a single cluster, and overall BLEU score in an EBMT system. GAC examines each word-pair in turn computing a similarity measure to every existing cluster. If the best similarity measure is above a predetermined threshold, the new word is placed in the corresponding cluster, otherwise a new cluster is created if the maximum number of clusters has not yet been reached. We also show how to find the optimum number of clusters under noisy conditions. This chapter, explains the NJW algorithm (form of a spectral technique) and reports results obtained when this algorithm was used for clustering and creating templates in the translation model.

5.1 Motivation: Templates in the Translation Model

We first motivate the use of templates in the translation model. Assume the training corpus consists of just the following two sentence-pairs[1].

Example training corpus:
The Minister gave a speech on Wednesday .↔Le ministre a donné un discours mercredi .
The President gave a speech on Monday .↔Le président a donné un discours lundi .

Say the following two clusters are available to us, <CL0> and <CL1>. These clusters could be manually generated clusters (by a bilingual expert) or automatically found clusters by an unsupervised clustering algorithm.

Example word-pair Clusters:[2]
<CL0>: Minister-ministre,President-président,..
<CL1>: Wednesday-mercredi,Monday-lundi,..

Then generalized templates can be obtained by converting specific exemplars (in our case, the sentence-pairs) to general exemplars (in our case, template-pairs). With the above training corpus, a single generalized template (T) is formed by replacing, `Minister-ministre` *and* `President-président` by *<CL0>*, and `Wednesday-mercredi` *and* `Monday-lundi` by *<CL1>*.

[1] sentence-pair: source and its corresponding target sentence
[2] word-pair: source and corresponding target word

Generalized template (T):
The <*CL0*> gave a speech on <*CL1*> . $\leftrightarrow$ Le <*CL0*> a donné un discours <*CL1*> .

Say the following input, I, needs to be translated,

I:The President gave a speech on Wednesday .

If no templates are used, then the translation model would generate the following two phrasal matches from the corpus and place them on a common lattice: `The President gave a speech on` $\leftrightarrow$ `Le président a donné un discours` and `Wednesday` $\leftrightarrow$ `mercredi`. If a statistical decoder that uses a target language model is used, then the phrasal matches on the lattice can be reordered to generate the final translation. Although in this example, the target fragments can just be concatenated based on the order of appearance of their source phrases to get a legitimate translation in French, in many cases where we have many short target phrasal matches (especially with languages that have very different word orders), the best order in which the target phrases need to combined is decided by the decoder.

Many of the EBMT systems do not use a decoder and depend on the templates to combine and produce the output. In such systems, the input is converted to its template form (ITS) by replacing the words in the input sentence by variables (if the words belong to an equivalence class) and their translations - <CL0>: `président` and <CL1>: `mercredi` - are stored. If a matching template is not found then the sentence cannot be translated in these systems.

ITS: The <*CL0*> gave a speech on <*CL1*> .

The translation model looks for matches in the indexed corpus. ITS completely matches the source half of T and hence its target template is obtained as a candidate template (ITT) for the input sentence.

ITT: Le <*CL0*> a donné un discours <*CL1*> .

The translations that were stored are put back into the template to obtain the output(O).

O:Le président a donné un discours mercredi .

Templates are also useful in EBMT systems that do use statistical decoders. Present decoders have constraints on the amount they can reorder the target phrasal matches as it is computationally expensive to try all possible reorderings. For language-pairs that have very different word orders, extraction of longer phrasal matches from the translation model improves translation quality (Callison-Burch et al. [2005] and Zhang and Vogel [2005] showed this in Phrase-based SMT). Templates provide a way to generate longer target phrasal matches without requiring more training data and hence they are well suited for translation tasks in sparse data conditions. For the above input, ITT will be obtained from the translation model and the variables are replaced by the translations of the generalized words to produce a longer target phrasal match [(`Le président a donné un discours mercredi`) vs. (`Le président a donné` and `mercredi`)].

5.2 Spectral Clustering

This section describes the use of spectral clustering ([Ng et al., 2001]; [Zelnik-manor and Perona, 2004]) for automated extraction of equivalence classes based on context. A reminder, only $word$-pairs (word-pair: source word and its corresponding target word) are considered in this chapter. Spectral clustering is a general term used to describe a group of algorithms that cluster points using the eigenvalues of 'distance matrices' obtained from data. The algorithm described by Ng et al. [2001] is used with certain variations that were proposed by Zelnik-manor and Perona [2004] to compute the scaling factors automatically. For the k-means orthogonal treatment, the process described in Verma and Meila [2003] was used during the initialization. The scaling factors in Zelnik-manor and Perona [2004] help in self-tuning distances between points according to the local statistics of the neighborhoods of the points. Spectral Clustering is similar to kernel PCA (Principal Component Analysis) however, Spectral Clustering normalizes the affinity matrix and has been shown to perform better [Ng et al., 2001].

5.2.1 NJW Algorithm

The NJW algorithm is briefly described below.

1. Let S =$s_1, s_2, ...s_n$, denote the n term vectors to be clustered into k classes.
2. Form the affinity matrix A using a Gaussian Kernel defined by,
   $A_{ij} = exp(-d^2(s_i, s_j)/\sigma_i\sigma_j)$ for $i \neq j$
   $A_{ii} = 0$

Where
$d(s_i, s_j) = 1/(cosinesim(s_i, s_j) + eps)$,
$cosinesim(\vec{u}, \vec{v}) = \frac{\vec{u}.\vec{v}}{||\vec{u}||\times||\vec{v}||}$,
eps is used to prevent d from becoming infinity when $cosinesim = 0$,
σ_i is the local scaling parameter for s_i,
σ_i = $d(s_i, s_T)$ where s_T is the T^{th} neighbor of point s_i for some fixed T (7 for this work).

3. Define D to be the diagonal matrix given by,
$$D_{ii} = \Sigma_j A_{ij}.$$

4. Compute $L = D^{-1/2}AD^{-1/2}$.

5. Select k eigenvectors corresponding to k largest eigenvalues (k is presently an externally set parameter which is the number of clusters). Normalize eigenvectors to have unit length. Form matrix U by stacking all the eigenvectors in columns.

6. Form the matrix Y by normalizing $U's$ rows,
$$Y_{ij} = U_{ij}/\sqrt{(\Sigma_j U_{ij}^2)}.$$

7. Perform k-Means clustering treating each row of Y as a point in k dimensions, initializing either with random centers or with orthogonal vectors.

8. After clustering, assign the point s_i to cluster c if the corresponding row i of the matrix Y was assigned to cluster c.

5.2.2 Term vectors for clustering

Using a bilingual dictionary (created as explained in Chapter 3) and a parallel corpus, a rough mapping between source and target words is created. When there is only a single possible translation listed for a word by the mapping, a word-pair made up of the word and its translation is created. This word pair is then treated as an indivisible token for future processing. For each such word-pair, frequency counts are accumulated for each word in the surrounding context of its occurrences (N words, currently 3, immediately prior to and N words immediately following). As an example, consider the frequency counts accumulated for the word-pair: `minister` ↔ `ministre` (<*NULL*> is used when N crosses the beginning or the end of a sentence) in Figure 5.1.

These counts form a pseudo-document for each word-pair, which are then converted into term vectors with unit-norm normalization. A simple linear decay which gave higher

words in the context	Occurrence
<NULL>	2
the	1
gave	1
a	2
speech	1
le	1
donné	1

Table 5.1: for `minister`↔`ministre`

weights to words immediately prior to and immediately following the word-pair did not give any significant improvements over the above approach.

5.3 Motivation for using Spectral Clustering

Spectral Clustering algorithms have given high quality segmentation results in Image Processing. As an example, consider the segmentation results obtained by Spectral Clustering and another effective technique in segmentation (k-means) in Figure 5.1 (Subplots B and C, respectively) on an image that contains three circles (commonly used data for image segmentation). Intuitively, points lying along a circle are closer to each other (or high affinity) than to points in other circles, so, one would want all data points along a circle to be clustered together in one cluster. Clearly, Spectral Clustering is able to achieve better segmentation than k-means. Spectral Clustering algorithms use the eigen-structure of a similarity matrix to partition data points into clusters (Figure 5.1D) and hence are better at clustering non-convex regions.

Are there any cases where k-means performs worse than Spectral Clustering in natural language? If it does, Spectral Clustering can be a powerful tool for our clustering task to obtain purer clusters. As an example, consider three cases, case (i): where a few words almost always have the words, $context_1$ and $context_2$ appearing in their context;
case (ii): where a few words rarely appear with $context_2$ and always with $context_1$;
case (iii): where a few words rarely appear with $context_1$ and always with $context_2$.

One would ideally want words that belong to each case to be clustered together. For the purpose of illustration, only 2 dimensions of the word-pairs are considered. In general, the dimension is much larger than two. The three cases are simulated in Figure 5.2A. Each

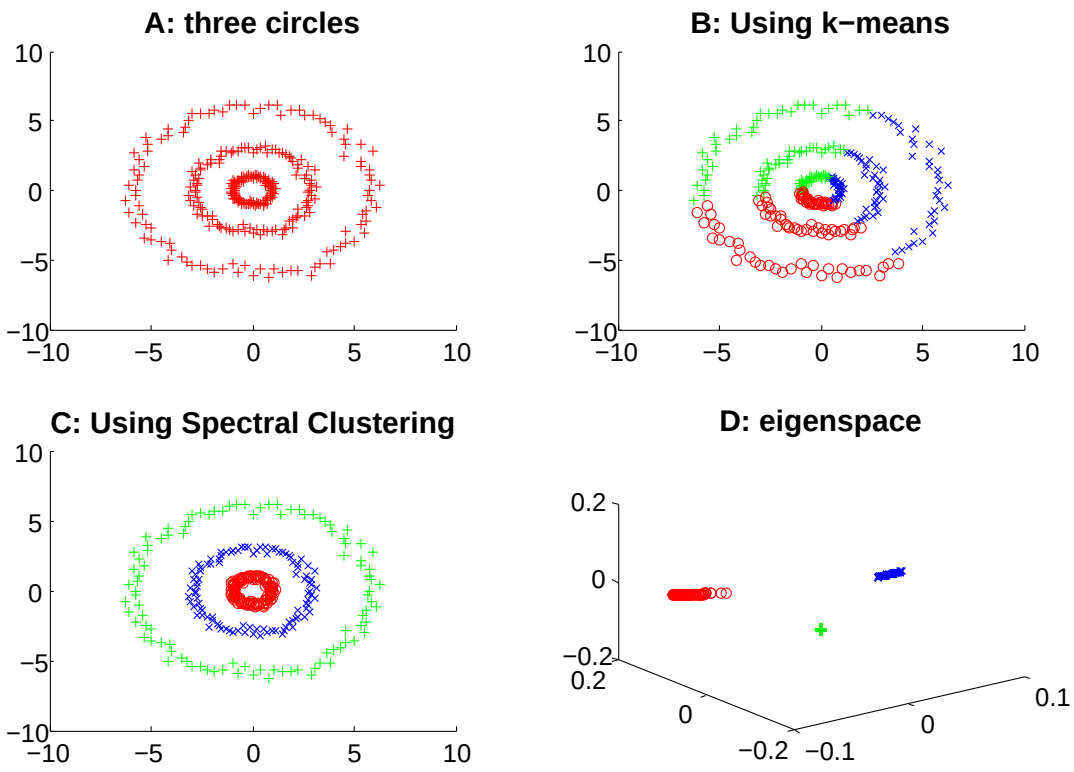

Figure 5.1: Image Segmentation results for the 3 circles data using Spectral Clustering and k-means Clustering.

data point represents a word (or a word-pair in the case of template- generation). The clusters obtained with Spectral Clustering and k-means are shown in Figure 5.2. Spectral Clustering clearly identifies the three regions/cases well.

5.4 Results: Clustering Algorithms

To show the effectiveness of the clustering methods in an actual evaluation, we set up the following experiment for an English to French translation task on the Hansard corpus [LDC, 1997]. The training data consisted of three sets of size 10k (set1, k:10^3), 20k (set2) and 30k (set3) sentence pairs chosen from the first six files of the Hansard Corpus (these data sets are different from the data described in Section 3.4). Only sentences of length 5

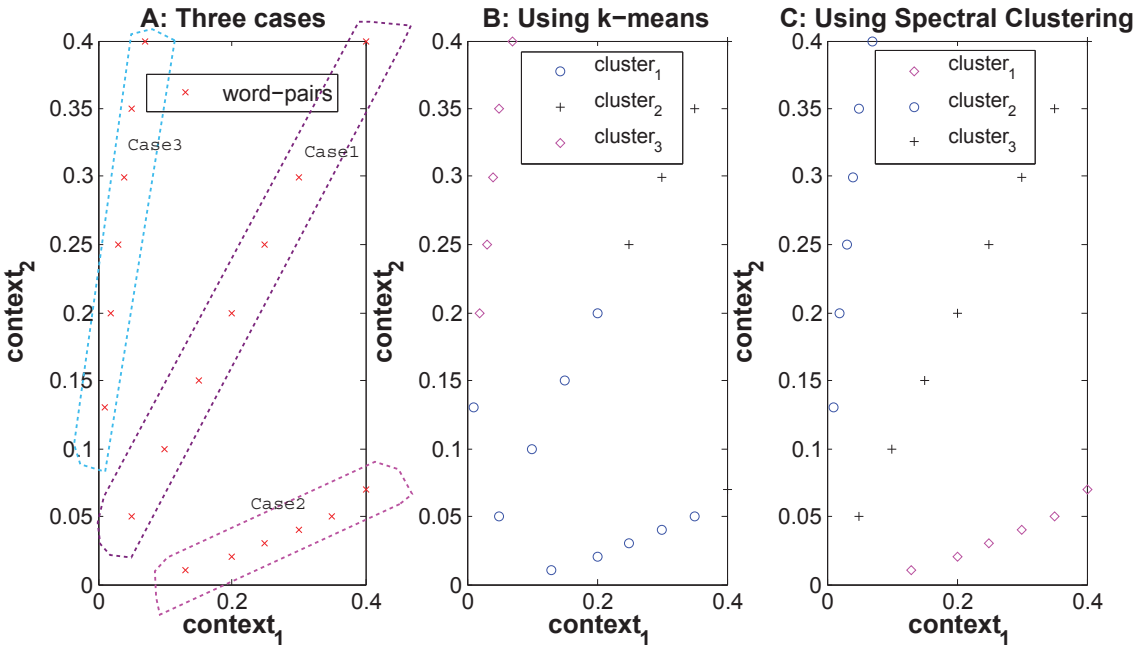

Figure 5.2: Clustering results for the 3 cases using Spectral Clustering and k-means Clustering.

to 21 words were taken. Only word-pairs with frequency of occurrence greater than 9 were chosen for clustering because more contextual information would be available when the word occurs frequently and this would help in obtaining better clusters (other thresholds are used later in Section 5.6). The test data was chosen to be a set of 500 sentences obtained from files 20, 40, 60 and 80 of the Hansard corpus with 125 sentences from each file. Each of the methods was run with different number of clusters and results are reported only for the optimal number of clusters in each case. The optimal number was empirically found on a tune set. The value that gave the highest improvement in translation quality was chosen as the optimal number.

5.4.1 Quality of Clusters

We compare the results obtained with the NJW algorithm against the incremental GAC algorithm [Brown, 2000] which was found to be more powerful than $k-$means in many of our experiments in obtaining clusters that improved translation quality. Some example

	GAC	GAC	Spectral Clustering	Spectral Clustering
Training data Size	% Rel imp	#clus	% Rel imp	#clus
10k	3.33	50	1.37	20
20k	8.11	300	12.73	80
30k	2.88	300	3.88	200

Table 5.2: % Relative improvement over baseline EBMT # clus is the number of clusters for best performance. Statistically significant improvements with $p < 0.0001$.

classes obtained with Spectral Clustering and GAC are shown in Table 5.3. Spectral clustering gives more natural and intuitive word classes than those obtained by GAC. Even though this is not guaranteed to improve the translation performance, it shows that maybe the increased power of spectral clustering to represent non-convex classes (non-convex in the term vector domain) could be useful in a real translation experiment. The first class in an intuitive sense corresponds to measurement units. We see that in the $< units >$ case, GAC misses some of the members and these missing members are actually distributed among many different classes. In the second class $< months >$, spectral clustering has mainly the months in a single class whereas GAC adds a number of seemingly unrelated words to the cluster. The classes were all obtained with the 20k sentence-pair subset of the Hansard Corpus. For spectral clustering, 80 clusters were chosen and 300 clusters for GAC since these gave the highest BLEU scores on the tune set.

5.4.2 Templates built from clusters

Templates are then generated from the resulting clusters (details in Chapter3, Section 3.2). The results in Table 5.2 show that spectral clustering requires moderate amounts of data to get a large improvement. For small amounts of data (10k) it is slightly worse than GAC.For 30k sentence-pairs, results with all three methods (Baseline, GAC and Spectral Clustering) are very similar, though spectral clustering is the best. For moderate amounts of data, when generalization is the most useful, spectral clustering gives a significant improvement over the baseline as well as over GAC.

Spectral clustering	GAC
adjourned ↔ hre	adjourned ↔ hre
cent ↔ %	
days ↔ jours	
families ↔ familles	families ↔ familles
hours ↔ heures	
million ↔ millions	million ↔ millions
minutes ↔ minutes	
o clock ↔ heures	o clock ↔ heures
p.m. ↔ heures	p.m. ↔ heures
p.m. ↔ hre	
people ↔ personnes	people ↔ personnes
per ↔ % ↔ per %	
times ↔ fois	times ↔ fois
years ↔ ans	
august ↔ août	august ↔ août
december ↔ décembre	december ↔ décembre
february ↔ février	february ↔ février
january ↔ janvier	january ↔ janvier
march ↔ mars	march ↔ mars
may ↔ mai	may ↔ mai
november ↔ novembre	november ↔ novembre
october ↔ octobre	october ↔ octobre
only ↔ seulement	only ↔ seulement
june ↔ juin	june ↔ juin
july ↔ juillet	july ↔ juillet
april ↔ avril	april ↔ avril
september ↔ septembre	september ↔ septembre
	page ↔ page
	per ↔ $
	recognize ↔ parole
	recognized ↔ parole
	recorded ↔ page
	section ↔ article
since ↔ depuis	since ↔ depuis
	took ↔ séance
	under ↔ loi

Table 5.3: Clusters for $< units >$ and $< months >$, comparing Spectral Clustering and Group Average Clustering.

5.5 Automatic determination of Number of Clusters

Although spectral clustering algorithms are powerful in forming pure clusters, in most applications, the number of clusters (N) is set manually. There has not been significant success in identifying the optimal number of clusters automatically for noisy real data. We suggest a method to automatically determine the number of clusters required for clustering data and we also try to remove incoherent words in clusters.

In our EBMT system, parameters are tuned based on the performance of the system on a development set using coordinate ascent. If N has to be found empirically, the MT parameters need to be re-tuned for every value of N. Tuning these parameters for each N is computationally expensive as the process can take several days. As mentioned at the beginning of this chapter, one would like to use an algorithm that is simple in design, should produce pure clusters, and should have some way of automatically determining N.

If all the data points in different clusters were infinitely far apart then one could easily find N for the spectral clustering algorithm by counting the number of eigenvalues that are equal to 1. However, clusters are not far apart in real world problems. An algorithm to automatically determine N was proposed in Sanguinetti et al. [2005] and tested on artificially constructed images. This method could not be applied directly to our EBMT system (Section. 5.5.1). We hypothesize that this is because of the noisy and imperfect nature of real data as opposed to the artificial data in Sanguinetti et al. [2005]. This thesis provides a solution: modify the algorithm to detect and remove outliers. We believe that these problems could arise in other practical systems and our modified algorithm would apply to those problems as well.

In essence, this work addresses the question of how to automatically generate clusters that contain mostly reliable words when hand-made clusters are not available to generate templates. The contribution of this section is three-fold. Firstly, an algorithm is developed to automatically find the optimum N on real data (Section. 5.5.2). Secondly, we detect incoherent points (that do not fit in any cluster) and show how the performance improves by removing these points (Section. 5.6.2). Finally, we show an increase in translation quality (Section. 5.6) in sparse data conditions by creating generalized templates in the translation model of an EBMT system.

The intuition behind the algorithm in Sanguinetti et al. [2005] is as follows. When the rows of the k eigenvectors are clustered along mutually orthogonal vectors, their projections will cluster along radial directions in a lower dimensional space. When q is less than the best number of clusters (N), meaning that $[N - q]$ eigenvectors are discarded, the points that are not close to any of the first q centers get assigned to the origin. Elongated

Lang-Pair	data	Manual	SangAlgo	Mod Algo
Eng-Fre(TM)	10k	0.1777	0.1641	0.1790

Table 5.4: BLEU scores with templates created using manually selected N, SangAlgo [Sanguinetti et al., 2005] and the modified algorithm to automatically find N.

k-means is initialized with $q = 2$ centers from the points and the $(q + 1)^{th}$ center as the origin. Their elongated k-means algorithm down-weighs distances along the radial direction and penalizes distances along the transversal direction. If points get assigned to the center which originated from the origin, the value of q is incremented and the procedure is repeated. The procedure is terminated when no points get assigned to the $(q + 1)^{th}$ center.

5.5.1 Problems encountered

To see the performance of the algorithm on different language pairs, we separately applied the clustering algorithm and the resulting templates to the translation model of an English-French (Eng-Fre) EBMT system. The analysis on the 10k set of Eng-Fre (data set from section 5.4) is as follows. As seen in Figure 5.3, the number of points assigned to the origin reaches zero in the 34^{th} iteration (when the number of clusters is 36). Hence, generalized templates were obtained with 35 clusters. These templates were used to translate the test data. With experiments performed using generalized templates obtained with 35 clusters, the average BLEU score was found to be much less (difference of 1.4 BLEU points on average) than the BLEU scores with generalized templates obtained using N that was set experimentally (Table 5.4). The automatically determined N was not the same as the experimentally determined N.

To study the nature of the problem, the value of q was increased beyond 3 for the artificial image data consisting of 3 circles in Sanguinetti et al. [2005] and the number of points assigned to the origin was analyzed (Figure 5.4A). When the value of q was increased beyond 3, the number of points assigned to the origin remained at zero (Figure 5.4B). However, for real data in our case, as the value of q was increased beyond 34, fluctuations in number of points assigned to the origin were observed (Figure 5.3). Intuitively, these fluctuations could be due to the presence of data points that are hard to classify.

To further analyze if the fluctuating points were incoherent points, we added a noisy data point in the three circles image data (star in Figure 5.4C). Fluctuations were now seen even with this simulated image data (Figure 5.4D). The algorithm predicted that 3 circles were sufficient, but, as the value of q was increased, there was one point that got

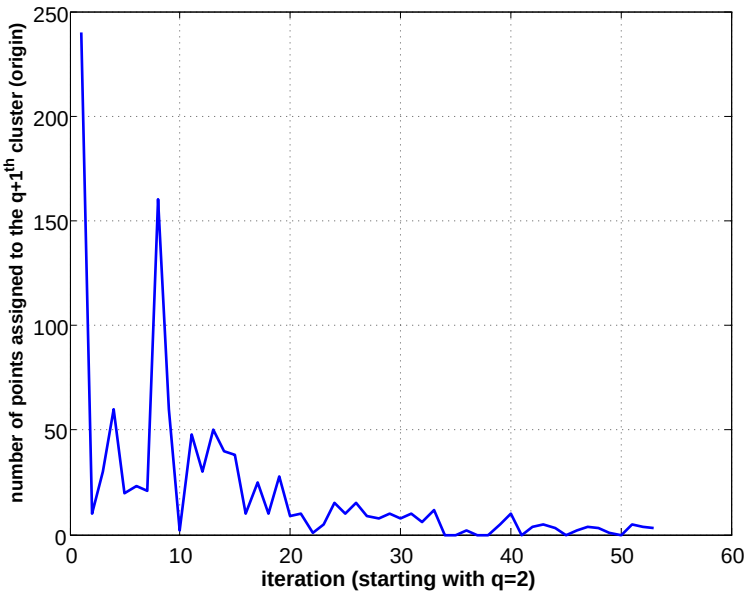

Figure 5.3: Plot of number of data points assigned to the origin in every iteration using the algorithm from Sanguinetti et al. [2005] in EBMT for Eng-Fre.

assigned to the origin when the number of clusters was 5 ($q = 4$ with $q + 1$ as origin). The fluctuating point was found to be the noisy data point that was added in Figure 5.4C. These incoherent points are points in reality that do not belong to any cluster. For the real data in our situation, these incoherent points arise when they appear in many different contexts or when the alignment-mapping between the source and the target of the word-pair is not correct and hence they clearly do not belong to any cluster. We will further see that these points not only make the process of finding the number of clusters difficult, they also reduce the quality of the clusters obtained (Impure clusters in Table 5.6). The algorithm given in Section 5.5.2 removes these unclassifiable points from the rows of the U matrix (containing eigenvectors with greatest eigenvalues stacked in columns) and reruns the procedure to determine the optimum N.

5.5.2 Modified Algorithm

The algorithm starts with q=2 centers and $(q+1)^{th}$ center as the origin. BP (Break Point) holds the first iteration number at which the number of points assigned to the origin was 0.

flag=1;
while *flag* **do**
    Initialization Step (INIT):
        Set q=2;
        Set BP=ϕ;
        Set it=0;
        Set i=0;
    Increment Step (INC):
        i=i+1;
        Compute U with q eigvectors with greatest eigvalues
        Initialize q centers from rows of U
        Initialize $q + 1^{th}$ center as origin
    Elongated k-means clustering(U, q+1):
    if *#points assigned to origin* > 0 **then**
        if *BP* $\neq \phi$ **then**
            remove rows from U;
            Goto INIT;
        else
            q=q+1;
            Goto INC;
        end
    else
        $BP = i$;
        it=it+1;
        if $it >= 4$ **then**
            flag=0;
        end
    end
end
N=BP-2

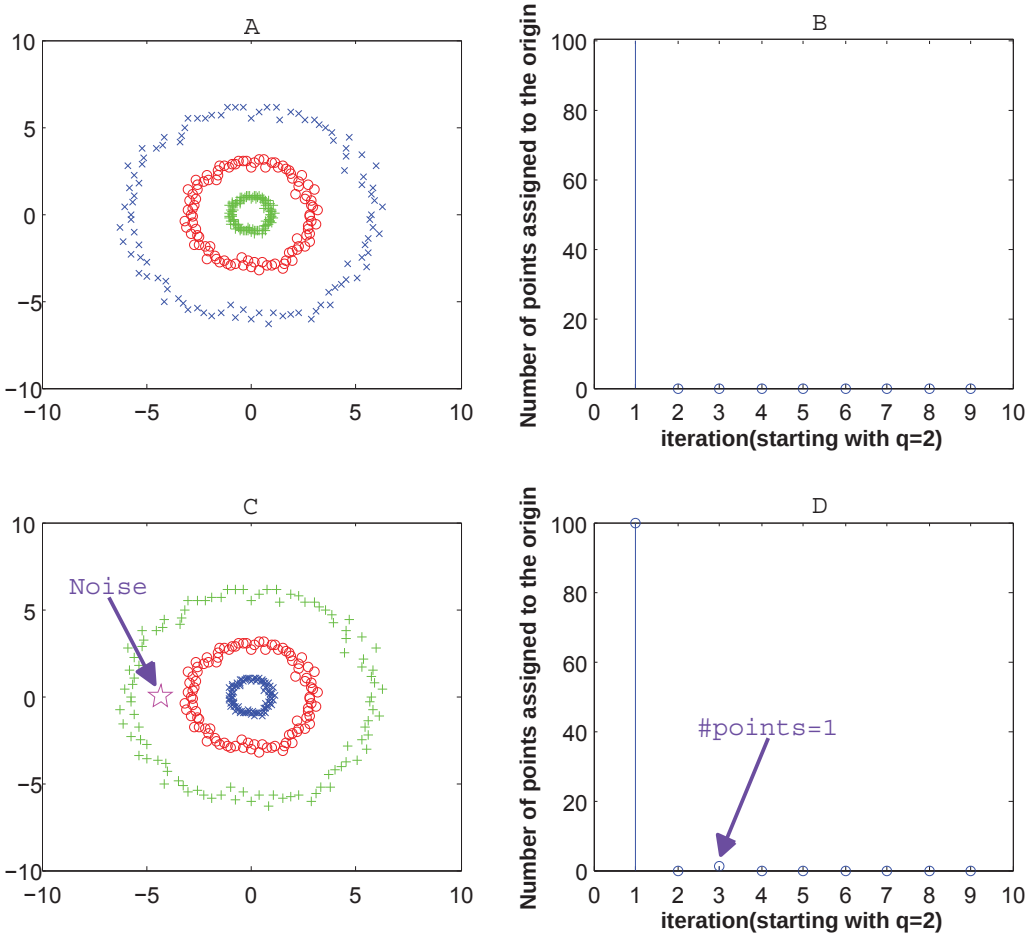

Figure 5.4: Plot of number of data points assigned to the origin in every iteration using SangAlgo [Sanguinetti et al., 2005] on the three circles image.

it holds the number of consecutive iterations for which the number of points assigned to the origin was 0. At the start of the algorithm, BP is empty and it is 0. Elongated k-means is performed with q+1 centers. If there are points assigned to the $(q+1)^{th}$ center, the value of q is incremented as in Sanguinetti et al. [2005]. Say for the first time at iteration i there are no points assigned to the origin, then BP is set to i. If BP has been set and the number of points assigned to the origin is greater than 0 in the following iteration, then the points assigned to the origin are removed from the U matrix and the algorithm is rerun starting with q=2 centers. If the number of points assigned to the origin remains

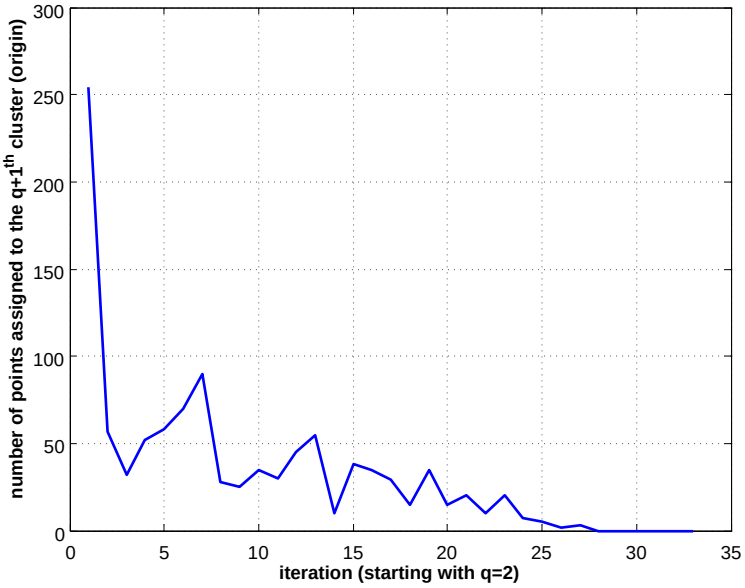

Figure 5.5: Plot of number of data points assigned to the origin in every iteration using algorithm 5.5.2 in EBMT for Eng-Fre.

at 0 for 4 consecutive iterations (Figure 5.5), the procedure is terminated and the best N is given by BP-2. We believe that if there are no points assigned to the origin for 4 consecutive iterations, then there will probably be no points assigned to the origin in the future iterations as was the case in many of the experiments and hence the procedure is terminated.

The modified algorithm was used to re-determine the number of clusters (Figure 5.5) and the BLEU score obtained with the new clusters with incoherent points removed is shown in Table 5.4 (column corresponding to "Mod Algo"). The score is closer to that obtained when the number of clusters was determined empirically. The automatically found N does not match the empirically found N due to the removal of incoherent points and also because the scoring function (BLEU score) used to find N empirically has many local maxima as we will discuss later in Section 5.6 (Figure 5.6).

	POS	Auto Clus
TM	0.1283	0.1296

Table 5.5: Average BLEU scores with templates created using POS and Automatically determined clusters on 30k Eng-Chi.

5.6 Results: Templates in the translation model with Spectral Clustering

The experiments in this section use the Eng-Chi, Eng-Fre and Eng-Hai data sets described in Section 3.4 without any limitations on the length of the sentences used for training. The experiments in the following subsections analyze the benefits of our word-clustering algorithm.

5.6.1 Equivalence classes

Part of speech (POS) tags are good candidates for equivalence classes. These tags can be obtained with semi-supervised learning [Tseng et al., 2005] techniques with training data. However, for languages with limited data resources (like Haiti), obtaining POS tags may not be possible. For such languages, unsupervised clustering techniques can be applied. Under these conditions, the question remains, *Are automatically found clusters as good as POS tags?* To answer this, we created templates based on POS tags and compared their performance with templates created using automatically found clusters on 30k Eng-Chi. The POS tags were obtained using Tseng et al. [2005] to create templates and were applied in the translation model. For the POS experiment, the word-pairs for the templates were grouped using the POS tags of the target word. For the comparison to be fair, we grouped only those word-pairs that were also used in the automatic clustering process. Target words with multiple POS tags were not considered. The BLEU scores with POS templates and templates created using automatic clusters on the 10 test files were almost the same (average BLEU scores over the test files in Table 5.5). It can be concluded that automatically found clusters are indeed good candidates for creating templates especially in sparse data conditions and for rapidly developing better MT systems for new languages.

Impure clusters	Pure clusters
(almost ↔ presque)	
(certain ↔ certains)	
(his ↔ sa)	(his ↔ sa)
(his ↔ son)	(his ↔ son)
(its ↔ sa)	(its ↔ sa)
(its ↔ ses)	(its ↔ ses)
(last ↔ hier)	
(my ↔ mes)	(my ↔ mes)
(my ↔ mon)	(my ↔ mon)
(our ↔ nos)	(our ↔ nos)
(our ↔ notre)	(our ↔ notre)
(their ↔ leur)	(their ↔ leur)
(their ↔ leurs)	(their ↔ leurs)
(these ↔ ces)	(these ↔ ces)
(too ↔ trop)	
(without ↔ sans)	
	(his ↔ ses)

Table 5.6: Cluster purity before and after removal of oscillating points. Word-pairs with frequency of occurrence greater than 9 were chosen to generate these clusters.

5.6.2 Oscillating points

Table 5.6 shows the changes in the cluster members of a cluster due to removal of oscillating data points. Words that oscillated, *(*almost ↔ presque*)*, *(*certain ↔ certains*)*, *(*last ↔ hier*)* and *(*without ↔ sans*)* were removed from the cluster. The member, *(*his ↔ ses*)* was added to the modified cluster, which is good since other versions of his are already present. The member, *(*too ↔ trop*)*, which did not fit well, got placed into a different cluster. A similar phenomenon was observed in Eng-Chi. Word-pairs with wrong alignments, data errors (spelling mistakes of words), words with multiple senses that fit in many contexts were found to be removed by the algorithm.

About 5 to 11 word-pairs were discarded as incoherent in the experiments. For Eng-Chi and Eng-Fre, the total number of word-pairs clustered were between 2000 to 12000, and 265 for Eng-Hai. For 200k Eng-Chi, 91 word-pairs were discarded as incoherent.

File	Man. worst	Man. best	Auto
tune	0.1240	0.1330	0.1339
test	0.1248	0.1298	0.1296

Table 5.7: Average BLEU scores on test and tune files with templates created using manually and automatically found N on 30k Eng-Chi.

5.6.3 Number of clusters (N)

A case can be made that fewer clusters should lead to longer phrases and hence, the translation quality should improve. We will take a case where this fails: consider placing all word-pairs into a single cluster, in other words, any member in the cluster can be replaced by any other member in the corpus. We would definitely get long source phrasal matches and hence long target phrasal matches with respect to the test set, but we clearly know that the translation quality will not be good. Hence, smaller number of clusters (with more longer phrases) does not necessarily mean that we would obtain good quality translations. On the other hand, having more number of clusters (fewer longer phrases) also does not mean that one will see improvement in translation quality, we could consider the case where all word-pairs are placed in unique classes, i.e., the number of clusters is equal to the number of word-pairs to be clustered, this will be equivalent to the baseline that uses no templates and hence we will not see any improvements in translation quality over the baseline. So finding the right number of clusters to improve translation quality to its best (with longer meaningful target phrases) is crucial.

Table 5.7 compares the average BLEU scores obtained on 10 test files from the empirically found best N and automatically found N applied in the translation model for 30k Eng-Chi. To find the best N empirically, the Spectral Clustering Algorithm in the previous section was run with different values of N and the value of N that gave the highest BLEU score on the tune file was chosen. Tuning the parameters for each value of N took on average 8 days (on a 2.9 GHz dual-core processor). The scores obtained with templates created from automatically found N versus empirically found N is almost the same. Finding the right N is important, `Man. worst` in Table 5.7 shows the test scores obtained with the worst value of N (worst N was also found based the tune set). The plot in Figure 5.6 shows more translation scores around the region that gave the highest score on both the tune and test file. Regions not shown in the plot were lower than 0.128 on the tune file. The plot in Figure 5.6 also shows the presence of many local maxima (many different values of N attain approximately the same highest BLEU score on the test set) and our goal was to find one of them automatically.

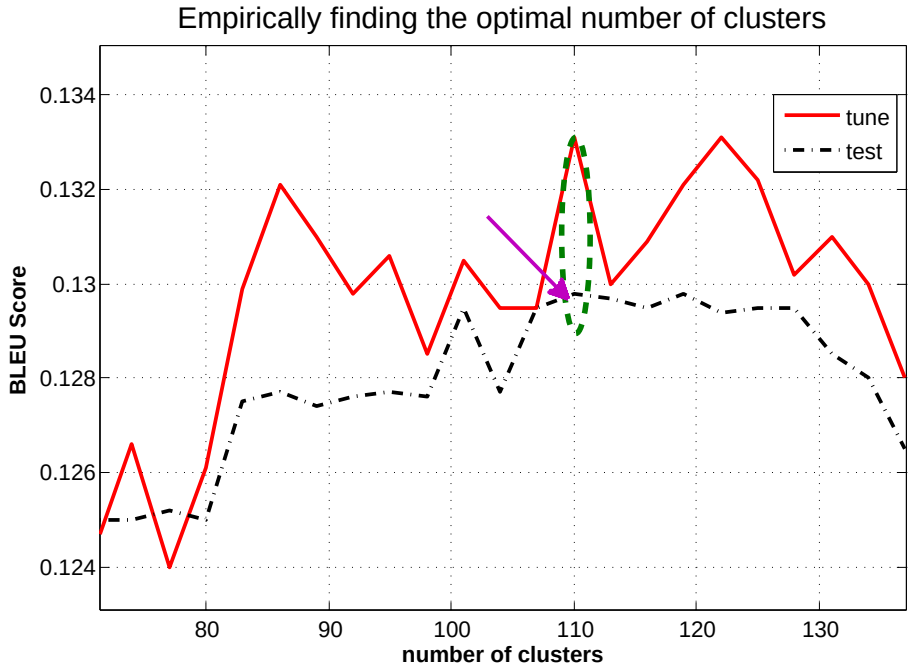

Figure 5.6: Plot of BLEU scores with different values of N on 30k Eng-Chi.

5.6.4 Selecting word-pairs based on frequency

Since the idea behind using templates is to obtain longer phrasal matches for phrasal units that contain less frequent words, the experiments in Sections, 5.6.1, 5.6.2 and 5.6.3 (except the illustration Table 5.6) were carried out on word-pairs that had their source and target words appearing at least 4 times (th_1) and not more than 15 times (th_2) in the training corpus. The idea behind not using a very low th_1 was that choosing an even lower threshold could result in clustering poorly aligned word-pairs. However, these may not be the best frequency thresholds. This section analyzes other frequency thresholds (thresholds thus chosen are different for each of the training data sets) to see if better performance can be achieved.

First, the frequency with which a word-pair occurs in the training data was obtained. The frequency for a word-pair is the number of times the source and the target words that make up a word-pair occur together in a sentence-pair. Next, the frequency-of-frequency curve (which has a power-law distribution) was obtained with the word-pairs in

Training Size	Baseline	low-frequency region	knee region	tail region
30k	0.1245	0.1272	0.1319	0.1302
200k	0.1785	0.1769	0.1807	0.1794

Table 5.8: Average BLEU scores with word-pairs from different frequency regions on 30k and 200k Eng-Chi.

30k Eng-Chi. The plot was then divided into three regions, low-frequency region (where the curve had a large negative slope), mid-frequency region (knee of the curve) and the high-frequency region (tail of the curve, where the slope is almost zero). Templates were created using word-pairs from different regions. The translation scores obtained with templates created from each of the three regions is given in Table 5.8. The percentage of word-pairs that fell into the low-frequency regions were around 75% to 80%, the mid-frequency regions had about 12% to 18% and the high frequency regions had about 5% to 11% of the total number of word-pairs extracted from the training corpus.

From the results, the mid-frequency (knee) region gave the best performance. The results obtained with the knee region gave statistically significant improvements (p <0.0001) over the low-frequency region but was not a significant improvement over the templates from the tail region. This could be attributed to the fact that a lot of word-pairs that appear less frequently tend to have alignment errors (especially with segmentation errors in Chinese) and hence, give low-quality templates. For word-pairs that appear very frequently in the training corpus, templates may not contribute much to obtaining longer target phrases as these word-pairs may appear frequently enough to obtain longer target phrases even when templates are not used. We also experimented by clustering word-pairs obtained by combining the mid-frequency region and the tail region but the results were lower than the scores obtained by either of the two regions. With Eng-Fre, the low-frequency region and the high-frequency region gave similar scores, however the mid-frequency region was the best overall. About 93% of the word-pairs (source halves) clustered from the mid-frequency region appeared in the test set at least once. More results on other data sets are shown in Table 5.9. We explore finer regions of threshold granularity in Chapters 6 and 7.

5.6.5 More Results: Templates in the translation model with Eng-Chi, Eng-Fre and Eng-Hai

Table 5.9 shows the average BLEU scores obtained by using templates and compares the scores obtained on a baseline system that used no templates on Eng-Chi, Eng-Fre and

Lang-Pair		Baseline	Templates in the translation model
Eng-Chi	15k	0.1076	0.1102
Eng-Chi	30k	0.1245	0.1319
Eng-Chi	200k	0.1785	0.1807
Eng-Fre	30k	0.1577	0.1652
Eng-Fre	100k	0.1723	0.1767
Eng-Hai		0.2182	0.2290

Table 5.9: Average BLEU scores with templates applied in the translation model. Statistically significant improvements with $p < 0.0001$.

Eng-Hai. The results clearly show the gain that can be obtained by using templates. The improvements over the baseline were statistically significant (p <0.0001) on all data sets. It is known that increasing the amount of training data in an EBMT system with templates in the TM will eventually lead to saturation in performance, where they perform about as well as the system with no templates. This is seen in the results obtained with Eng-Chi.

5.6.6 Further Analysis

Our goal of applying templates for translation was to obtain longer target phrases to improve the translation quality with today's constrained decoders. In Section 5.6.4, we showed why choosing word-pairs from the right frequency region was important to see improvements in translation quality. We suggested a method to find the optimum number of clusters and we also showed why it was crucial to find the optimum number of clusters in Section 5.6.3. At the same time we also showed that $more$ longer phrases (fewer clusters) does not necessarily improve translation quality (Table 5.6). For the purpose of illustration we will use an English example to show what we mean. As an example consider the following candidate translation,

`on a apple`

which was obtained from an $over - generalized$ candidate template (output of the TM, obtained for a generalized source phrasal match in the input sentence) by replacing the values of the class labels.

`on a <CL23>`

lets say, the cluster, <CL23> contains two members `table` and `apple`. Where, `table`

and `apple` may have appeared in similar contexts and the clustering algorithm may have put them in the same cluster instead of putting them in different clusters due to insufficient number of clusters. This example clearly shows that the phrase is ungrammatical. Hence, to obtain longer target phrases of good quality, the best partition of the word-pairs into clusters with the right number of clusters plays an important role.

In this section, we will further analyze the output of the translation model and the resultant translations from the decoder. The analyses in this section are only performed on the 30k Eng-Chi training corpus.

Coverage

To see how many source phrasal matches can be obtained from the training corpus for the test set, we first generalized the source half of the 30k Eng-Chi training corpus and the test data. We then found how many $n-$grams of the test data were present in the training corpus. Figure 5.7 shows the number of matching $n-$grams in the test set- with and without generalization. The plot clearly indicates the increase in the number of source phrasal matches with generalization when compared to the number of source phrasal matches without generalization. Of course, a single $n-$gram on the source side could be split into many non-contiguous $m-$grams. Hence, this plot can be considered as an upper bound on the number of target phrasal matches (while restricting the maximum target phrasal alternatives for every source phrase to 1) that can be obtained. In other words, if each and every source phrasal match has a contiguous translation on the target side, and every source word generated one target word, the number of target phrasal matches obtained could be the same as the number of source phrasal matches for each n. However, since the source and target languages considered here are very different from each other with different word orders and different fertilities (3 English words produce 1 Chinese word on average) we will expect to see fewer matches with 'Maximum Alternatives' equal to 1.

Percentage of words generalized

We were interested in knowing the number of words that were generalized in the training data. The more the number of generalized words, the longer will be the target phrases for the test set. However, this does not necessarily indicate improvement in the quality of the translation as many target phrases can be ungrammatical. Figure 5.8 shows the percentage of words generalized for each of the Eng-Chi training data sets. When the corpus has about 11% of its words generalized, the performance in translation quality is the best. So for the mid-frequency region, if 10 words are picked at random from the corpus, about

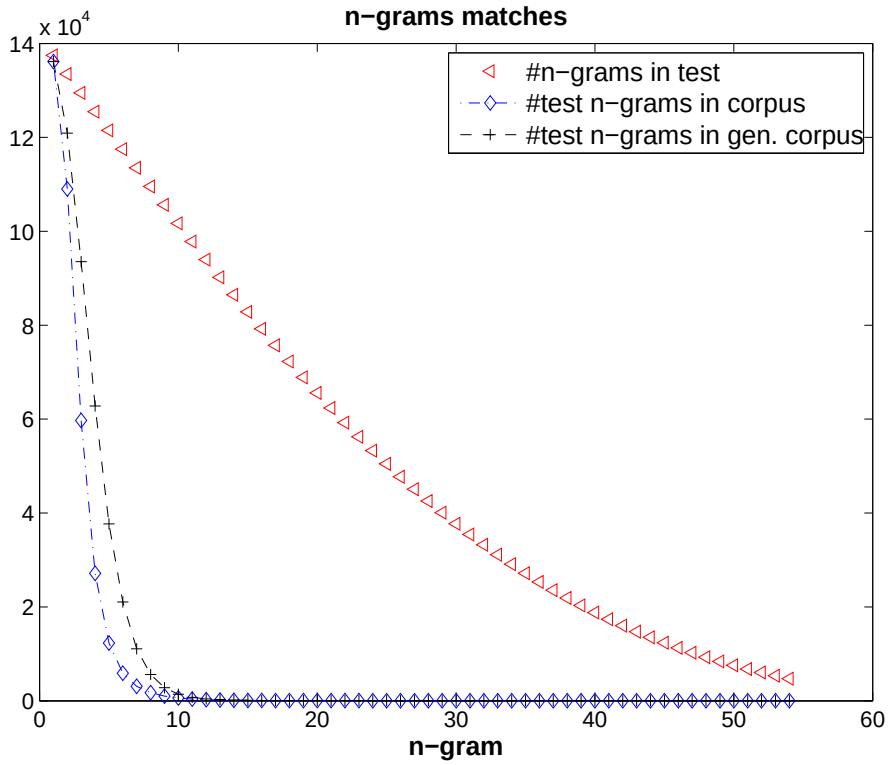

Figure 5.7: Number of n-grams (i) in the test set (ii) matches between the test set and source side of 30k Eng-Chi (iii) matches between the generalized test set and generalized source side of 30k Eng-Chi.

1.1 words would be generalized. From Figure 5.8 and 5.8, it can be concluded that even though fewer words are generalized when word-pairs are chosen from the mid-frequency region, the improvement in quality is the best when compared to the generalization obtained from the other regions. Word-pairs generalized from the high frequency regions also show improvements over the baseline. Since word-pairs from very high frequency regions (function words like, determiners, auxiliary verbs, etc.) appear in many contexts, the clustering algorithm can find the term vectors confusing to determine the best possible clusters, thus smearing the boundaries between clusters. Also, word-level alignments for word-pairs from very high frequency regions with respect to the target language may not be accurate, especially when the number of functions words differ between the source and the target sentence. If word-pairs appear rarely in a corpus whose source and target words

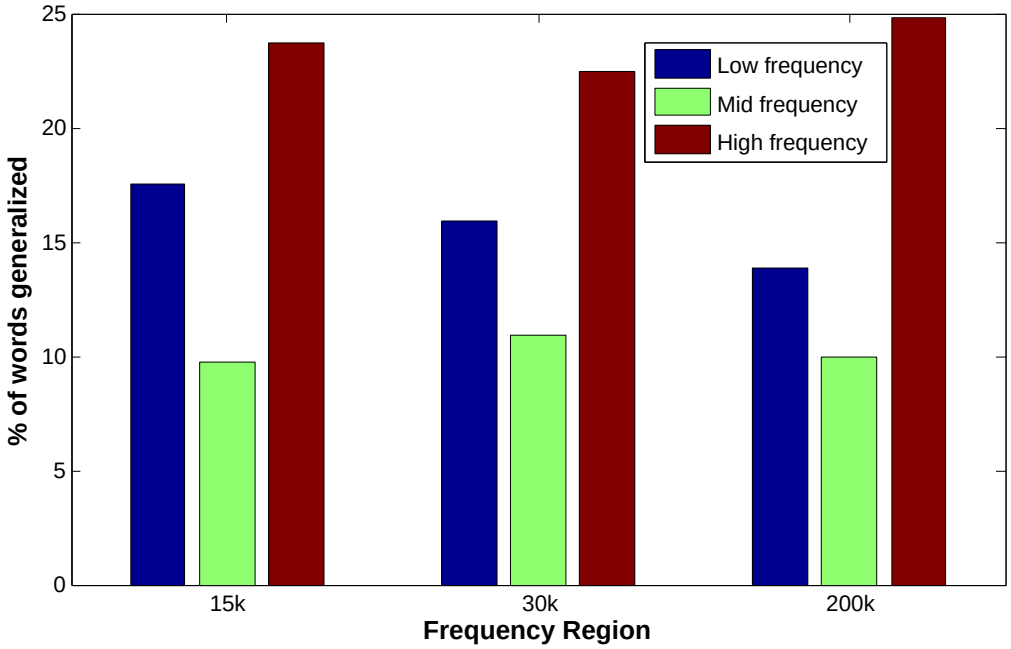

Figure 5.8: % of words generalized in each of the Eng-Chi training data sets. Low-frequency: generalization performed with word-pairs clustered only from the low frequency region, Mid-frequency: generalization performed with word-pairs clustered only from the mid frequency region, High-frequency: generalization performed with word-pairs clustered only from the mid frequency region.

appear together in all the sentence-pairs, their word level alignments can be obtained accurately. However, if these words are typographical errors (hence, appear infrequently in the corpus) in either of the two languages, the alignments will not be accurate. Also, since word-pairs from the low frequency region appear in fewer contexts, the term vectors for the clustering algorithm may not carry sufficient statistics about the word-pair, leading to poor clusters.

Output Analysis: translations and target phrases obtained from the translation model

As mentioned in Section 3.1, the output of the translation model (TM) is a lattice of possible candidate translations for $n-$grams in the test sentences (fragments/arcs/phrase-pairs: source fragment and its corresponding target fragment). An example of a phrase-pair that

was generated due to generalization is as follows:

From the TM:
the <CL31> party and the <CL89> have ↔ <CL31> 党 和 <CL89>

After putting back the values of the class labels we get:
the chinese party and the government have ↔ 中国 党 和 政府

The G-EBMT system was able to generalize 3999 test sentences out of the 4000 test sentences. The plot in Figure 5.9 shows the number of lexical (no generalizations) and new generalized phrase-pairs (whose target halves with class labels replaced by their corresponding values, were not present in the lexical phrase-pairs) with respect to the length of the target phrases present in the output of the G-EBMT's TM. From the best path information of the decoder- of the 3999 sentences, translations of 3038 test sentences contained partial target fragments that were generated due to generalization. The maximum alternatives (a complexity parameter in our EBMT system, details in Chapter 3) was 25.

We increased the 'Maximum Alternatives' to 200 to see if more new generalized phrase-pairs could be extracted and also to check whether generalization was really needed to generate new target fragments. For example, if a target phrase is generated by a lexical phrase-pair and also generated by the generalized phrase-pair (after replacing the values of the class labels), then the case could be made that generalization is not helping. The plot in Figure 5.10 shows the number of lexical phrase-pairs and new phrase-pairs generated due to generalization with respect to the length of the target $n-$grams. Even with the increase in the lexical target phrases, there still exists many new target phrasal matches obtained from templates that were not present in the lexical target phrases. Figure 5.11 shows a similar plot but with respect to the length of the source phrases of the new phrase-pairs which will ultimately increase the length of the target phrases. The plots in Figure 5.11 and Figure 5.10 clearly show an increase in the number of n-grams largely up to 10-grams.

Our ultimate goal is to check whether the target halves of the new phrase-pairs appear in the reference translations. Figure 5.12 shows the number of new target phrases that also appear in the reference translations. The 'Maximum Alternatives' was chosen as 200 in Figure 5.12 where a large number of lexical phrase-pairs are also extracted limiting the number of new target phrasal matches that can be obtained with templates. The plot in 5.13 still clearly indicates the increase in the number of useful (present in the reference translations) target phrasal matches.

Since there were additional target phrases added to the lattice when the 'Maximum

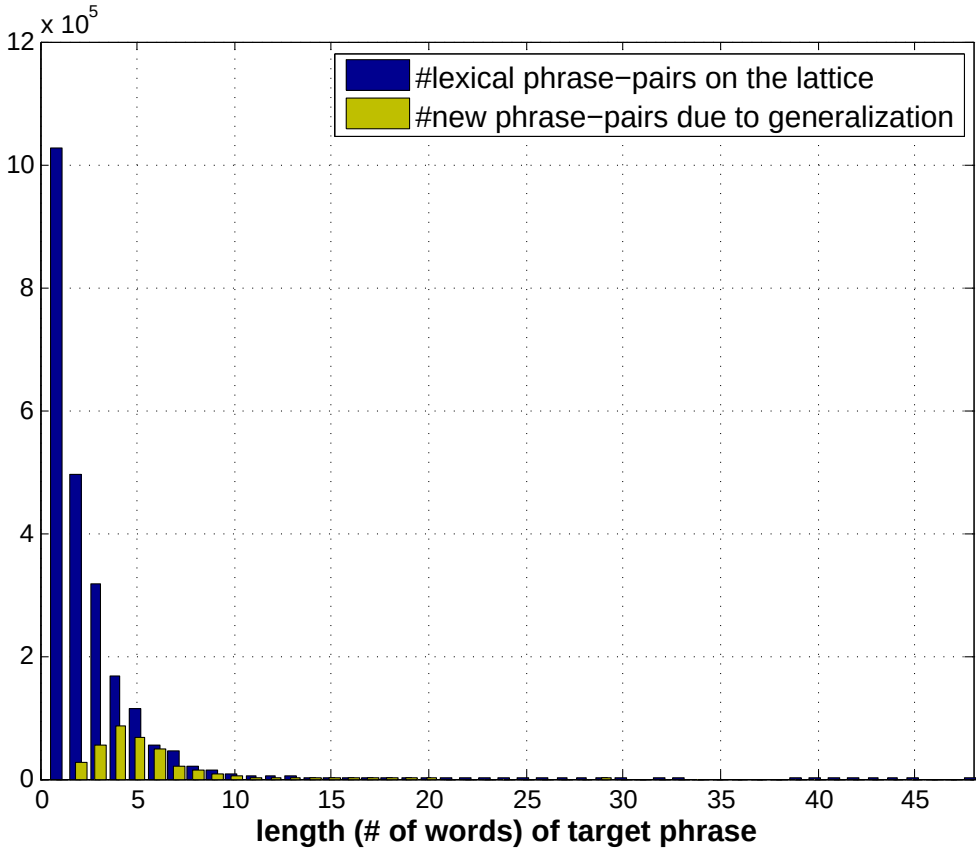

Figure 5.9: Number of phrase-pairs with increasing values of the length of the target halves (i) from lexical phrase-pairs only (ii) new phrase-pairs solely due to generalization. Max-Alternative=25.

Alternatives' was increased from 25, one might wonder if the performance of either (or both) the Baseline or the Generalized system can also be improved. For this, we experimented with 'Maximum Alternatives' set to 25 and 200. The difference in scores was not significant but with 'Maximum Alternatives' of 200, the BLEU score was worse on 3 test subfiles (out of 10 files). Hence, in the rest of the experiments in this thesis, we only use 'Maximum Alternatives' of 25.

A few sample translations produced by the baseline with no templates and the generalized system are given below. The highlighted text (in green) in the output translation of the G-EBMT system, indicates that the target phrase was generated by a source phrase that

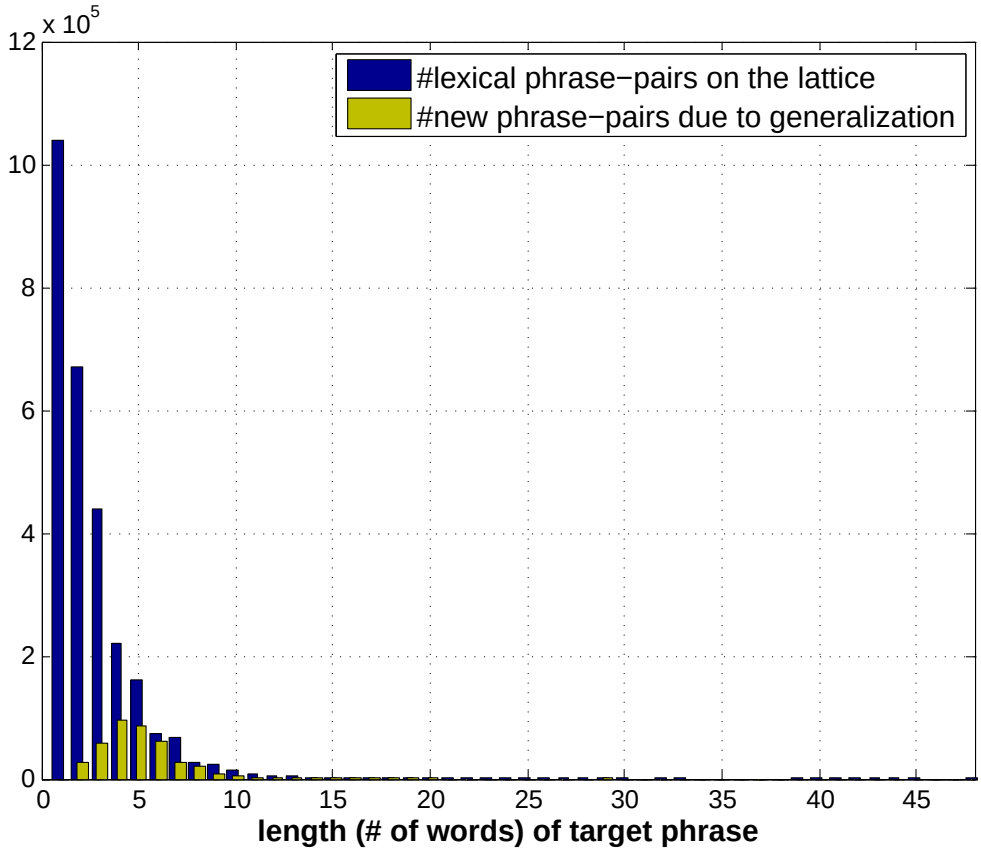

Figure 5.10: Number of phrase-pairs with increasing values of the length of the target halves (i) from lexical phrase-pairs (ii) new phrase-pairs due to generalization. Max-Alternative=200.

contained generalized tokens. The actual translation of its corresponding source phrase in the test sentence is also highlighted in the reference translation.

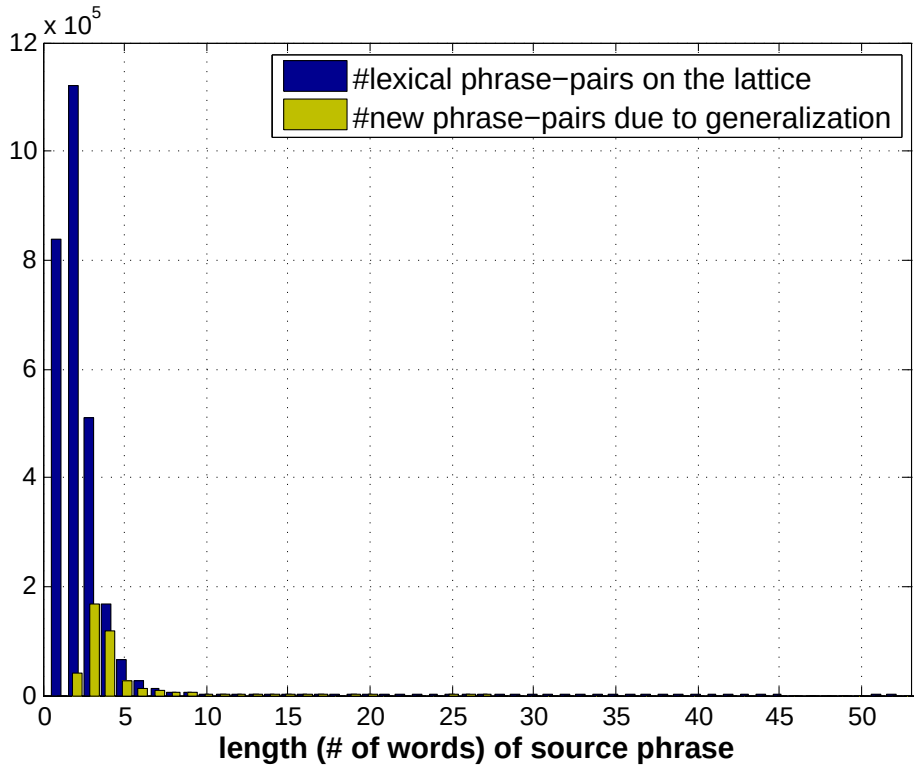

Figure 5.11: Number of phrase-pairs with increasing values of the length of the source halves (i) from lexical phrase-pairs (ii) new phrase-pairs due to generalization. MaxAlternatives=200.

Sample translations

Sample1:
Test sentence:the special reports **of the relevant state council departments** should be made under state council 's unified plan .
Reference: 国务院 有关 部门 的 专题 汇报 由 国务院 统一 安排 。
Baseline: 特殊 的 报告 ， 国务院 、 中央 军事 委员会 的 有关 部门 要 下 国家 的 统一 。
G-EBMT: 的 ， 国务院 有关 部门 的 报告 是 国家 的 统一 。

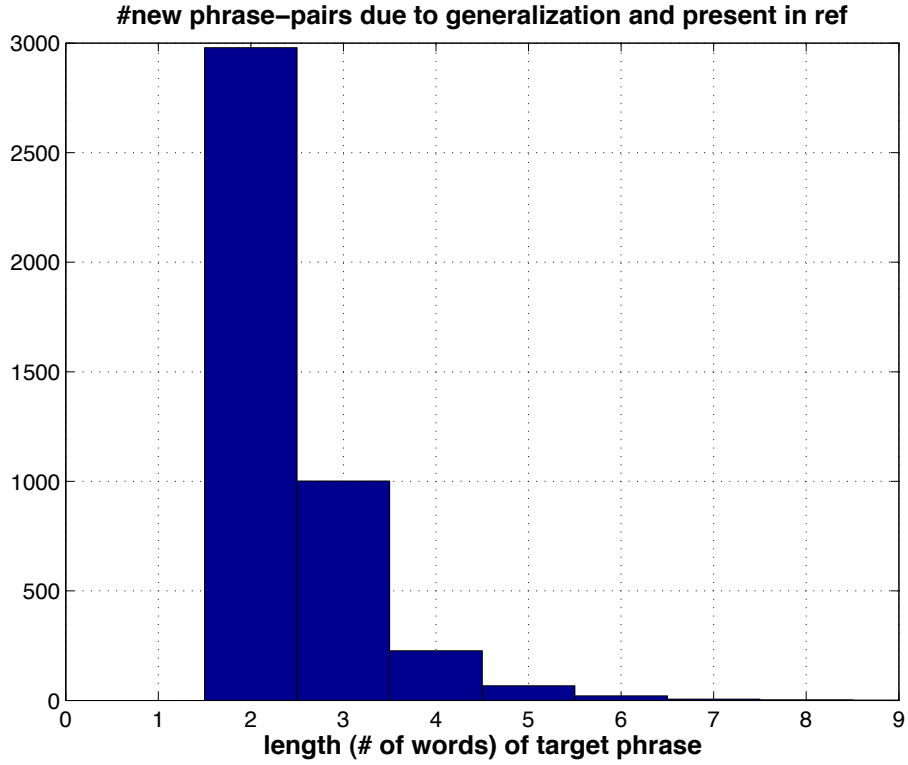

Figure 5.12: Number of new partial translations solely due to generalization and present in the reference translations. Max-Alternatives=200.

Sample2:

Test sentence:he said : " **if chen shui - bian assumes office** , there will be no peace on both sides of the strait , and the people in taiwan will face the disastrous consequences .

Reference: 他 说 ： " 如果 陈水扁 上台 ， 两 岸 关系 将 是 永无宁日 ， 台湾 民众 将 面临 灾难性 后果 。

Baseline: 他 说 ： " 如果 ， 得到 他 不 两 岸 和平 的 人民 ， 台湾 将 面临 灾难性 后果 。

G-EBMT: 他 说 ： " 如果 陈水扁 上台 ， 得到 不 是 两 岸 的 和平 ， 台湾 人民 。 面临

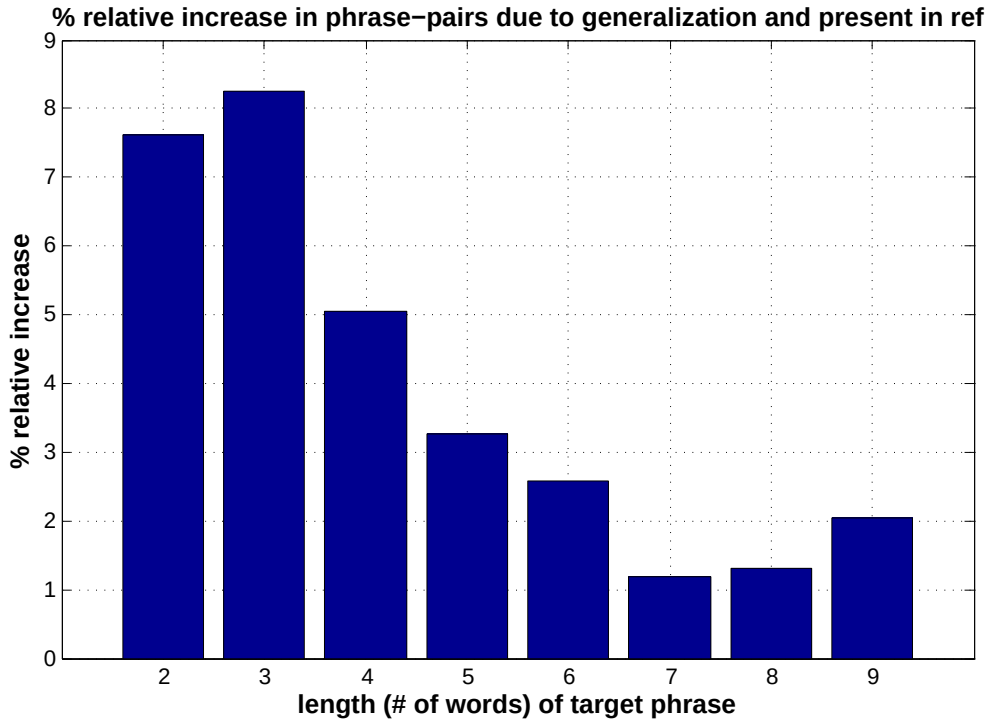

Figure 5.13: % Relative improvement in additional new (not found in the lexical phrase-pairs) partial translations solely due to generalization and present in the reference translations. Max-Alternatives=200.

Sample3:
Test sentence: we will strengthen the law enforcement ranks , strive to improve **the professional quality of law enforcement personnel** and the standard of law enforcement , improve work efficiency , and severely punish and get rid of those corrupt elements in the law enforcement ranks .
Reference: 加强 执法 队伍 建设 ， 努力 提高 执法 人员 的 业务 素质 和 执法 水平 ， 提高 工作 效率 ， 从严 惩治 并 清除 执法 队伍 中 的 腐败 分子 。
Baseline: 我们 ， 加强 执法 队伍 ， 努力 提高 执法 部门 的 质量 水平 ， 提高 工作 效率 ， 执法 、 第一 ， 这些 腐败 分子 的 执法 队伍 。
G-EBMT: 要 加强 执法 队伍 ， 努力 提高 执法 人员 的 业务 素质 和 执法 水平 的 提高 工作 效率 、 、 参与 的 腐败 分子 ， 执法 队伍 。

To summarize, this chapter applied an automatic clustering algorithm to create word-generalized-templates in EBMT and compared its performance and capabilities to other standard clustering algorithms that are applied for natural language. It also introduced a method to automatically find the number of clusters (N) for a real world problem- Machine Translation. The algorithm also refined the clustering process by removing incoherent points and showed that discarding these points boosts the translation quality many times above the best N found empirically. Statistically significant improvements (p <0.0001) were found on all data sets with templates over the baseline system with no templates by generating longer target phrases in data-sparse conditions.

Chapter 6

Templates in the Translation Model: using syntactically related phrase-pairs

In this chapter, we investigate another template-based approach that is useful when data available is limited. The previous chapter showed how to create templates using automatically generated equivalence classes that contained word-pairs only. In this chapter, we build equivalence classes containing phrase-pairs as well (will be called as segment-pairs in this chapter). While finding parallel (or bilingual) data for language-pairs is time-consuming and expensive, many languages do have moderate amounts of monolingual text available. Although efficient parsers for minority languages are still not available, robust (always outputs a solution) monolingual chunkers (shallow parsers) are being developed for many of the minority languages [Baskaran, 2006];[Dalal et al., 2006]. This chapter takes a step forward from just using word-pairs for clustering and utilizes information from independently developed monolingual chunkers to obtain segment-pairs [Gangadharaiah et al., 2011].

We use two approaches for clustering the segment-pairs, one based on syntactic structures of the segment-pairs (presented in this chapter), the other based on semantic-relatedness and structural-coherence of segment-pairs(presented in Chapter 7). A rigorous phrase-extraction model for extracting segment-pairs that shows how to incorporate information from chunks while extracting better segment-pairs by limiting the word-alignment boundaries to the chunk boundaries, is developed. We also automatically identify segment-pairs that contribute the most to improving translation quality. For the first time we also show how to obtain and use expert information from black box machine translation systems in the absence of human labelers when tuning features. Generalized templates created using our model with English-Chinese and English-French gave significant improvements over

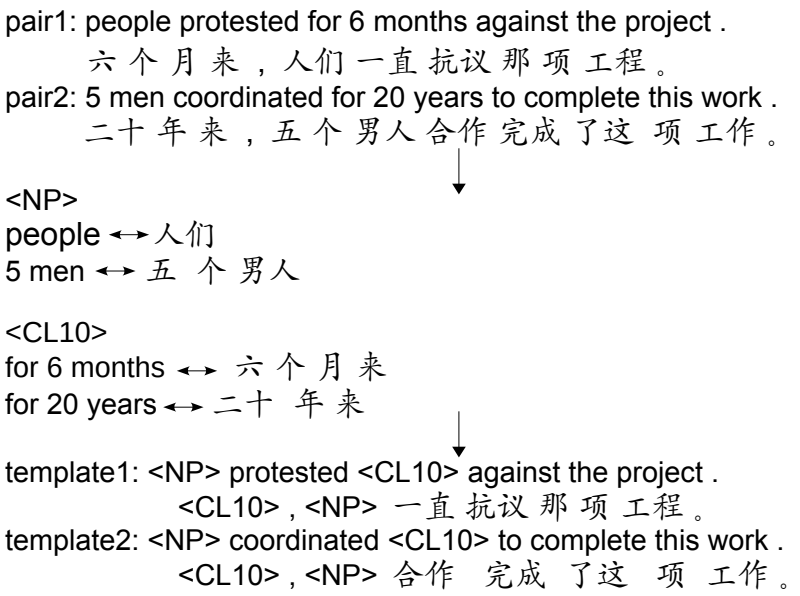

Figure 6.1: Phrase-generalized Templates.

a baseline with no templates.

6.1 Example: Phrase-generalization

Figure 6.1 shows an example of templates created from two sentence-pairs, `pair1` and `pair2`. `<NP>` and `<CL10>` are clusters generated by clustering the output of a phrase extraction model (details in Section 6.3.1). In a word-aligned training corpus, all occurrences of source-phrases and their corresponding target translations that belong to a cluster are replaced by their class labels to obtain *template-pairs* (`template1` and `template2`). The two template-pairs in Figure 6.1 for instance, can now be used to translate new input sentences like, `5 men protested for 20 years against the project`, even when the input sentence does not completely match the source-side of `pair1` or `pair2`.

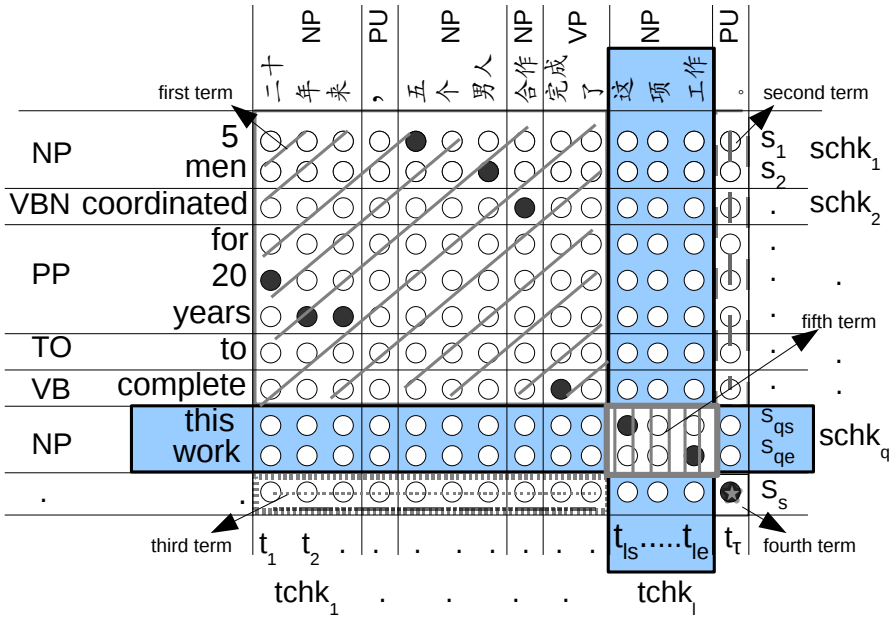

Figure 6.2: Sentence pair with chunks and chunk labels. Dark circles illustrate the primary alignments.

6.2 Motivation: for using Phrase Structure

In this chapter, we use knowledge about (i) phrase structure (ii) chunk boundaries for the source and target languages and (iii) alignment, to create templates. The purpose of including this knowledge is two fold: first, we use knowledge about the languages to reduce the search space of phrases to be generalized, second, we use knowledge about phrase structure to select and cluster phrases, allowing us to generalize only those phrases that will increase coverage when data available is small while not over-generalizing and decreasing the translation accuracy. Forcing the units to be structurally similar allows us to choose only those units that can be interchanged safely.

6.3 Procedure

The first phase of our processing is a phrase extraction method that incorporates knowledge about source and target languages by using chunks (a group of words forming a linguistic unit) extracted from sentences. Ideally we would want a chunker that chunks

the source and target sentences together [Ma et al., 2007]. However, many chunkers are mono-lingually developed. To utilize such chunkers, we combine a group of source chunks (called "source segment" in this work) and extract their corresponding group of target chunks (called "target segment"). We use word-alignment information in the chunk alignment model to align the chunks and obtain consistent segment-pairs as illustrated in Figure 6.2. The filled circles in Figure 6.2 represent the word-alignments (source-target word correspondences) between an English source sentence and its corresponding Chinese target. Hence, segment-pairs such as, `[5 men coordinated for]` and its corresponding target half that violate the chunk boundary condition are avoided. To find the chunk correspondence of $[5\ men]$, the chunk alignment model tries to align the chunk- $[5\ men]$ to the $chunks$ in the target sentence but not to individual words such as, [五].

Figure 6.4 provides examples of segment-pairs that could be extracted. This also has the added bonus of reducing the complexity of phrase-pair computation as the number of possible boundary combinations for a given sentence is drastically reduced. Thus, using knowledge that chunks can be a unit of sentences and alignment information, we reduce the search space and this allows us to extract much longer consistent phrases. A chunk in the source sentence does not necessarily correspond to one chunk in the target sentence, as a meaningful unit in one language is not represented in the same way in another language. In other words, it is likely that m chunks in the source sentence correspond to n chunks in the target sentence. `[NP(5 men) VBN(coordinated) PP(for 20 years)]` is made up of 3 chunks and its corresponding target, `[NP(二十 年 来) PU(,) NP(五 个 男人) NP(合作)]`, is made up of 4 chunks. The resulting syntactically coherent segment-pairs are then clustered based on their source and target chunk label sequences.

6.3.1 Formal description of the model

Suppose, the Source sentence has S words (as in Figure 6.2) : $s_1^S = s_1, s_2, s_3...s_S$ and Target sentence has τ words: $t_1^\tau = t_1, t_2, t_3, ...t_\tau$. For the sake of clarity, we define segments and chunks as follows: a chunk is a sequence of words and a segment is made up of one or more chunks. A segment-pair (or phrase-pair) is a source segment and its corresponding target translation. Our goal is to define a probability model P and then find the best possible segment boundaries $\hat{B}$ between s_1^S and t_1^τ,

$$\hat{B}(s_1^S, t_1^\tau) = \arg\max_b P(b|s_1^S, t_1^\tau) \tag{6.1}$$

The source sentence is chunked into m chunks $(schk_1^m)$: $schk_1, schk_2...schk_m$ and the target sentence is chunked into n chunks $(tchk_1^n)$: $tchk_1, tchk_2...tchk_n$, where m and

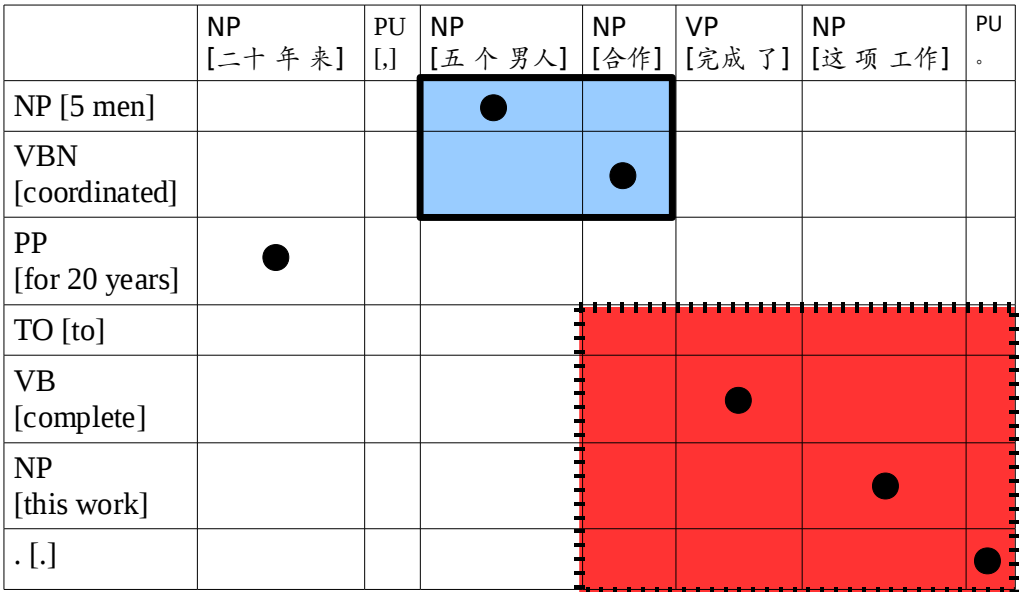

Figure 6.3: Union of chunk alignments

(5 men ↔ 五 个 男人),(for 20 years ↔ 二十 年 来),
(complete ↔ 完成 了),(this work ↔ 这 项 工作),(. ↔ 。),
(to complete ↔ 完成 了),(5 men coordinated ↔ 五 个 男人 合作),
(5 men coordinated ↔ , 五 个 男人 合作),(coordinated ↔ 合作),
(for 20 years ↔ 二十 年 来 ,),(this work . ↔ 这 项 工作 。),
(complete this work ↔ 完成 了这 项 工作),
(complete this work . ↔ 完成 了这 项 工作 。),
(to complete this work ↔ 完成 了这 项 工作),
(to complete this work . ↔ 完成 了这 项 工作 。),
(5 men coordinated for 20 years ↔ 二十 年 来 , 五 个 男人 合作),
(5 men coordinated for 20 years to ↔ 二十 年 来 , 五 个 男人 合作),
(5 men coordinated for 20 years to complete ↔ 二十 年 来 , 五 个 男人 合作 完成 了)

Figure 6.4: list of extracted segment-pairs.

n are random variables. ca represents alignments between the source and target chunks, wa represent alignments between the source and target words. Then, by marginalization,

$$\begin{aligned}
\hat{B}(s_1^S, t_1^\tau) &= \arg\max_b \sum_{ca,wa,schk_1^m,tchk_1^n} && P(b, schk_1^m, tchk_1^n, ca, wa | s_1^S, t_1^\tau) \\
&= \arg\max_b \sum_{ca,wa,schk_1^m,tchk_1^n} && P(wa|s_1^S, t_1^\tau)\ X\ P(schk_1^m|s_1^S, t_1^\tau, wa)\ X \\
& && P(tchk_1^n|s_1^S, t_1^\tau, wa, schk_1^m)\ X \\
& && P(ca|s_1^S, t_1^\tau, wa, schk_1^m, tchk_1^n)\ X \\
& && P(b|s_1^S, t_1^\tau, wa, schk_1^m, tchk_1^n, ca) \qquad (6.2)
\end{aligned}$$

In general, Eqn. (6.2) is computationally infeasible to compute and so we simplify and make a series of approximations.
Approximation1: The source chunks are obtained using a chunker trained only on the source language

$$P(schk_1^m|s_1^S, t_1^\tau, wa) = P(schk_1^m|s_1^S). \qquad (6.3)$$

Approximation2: Similarly, the target chunks are obtained using a chunker trained on the target language.

$$P(tchk_1^n|s_1^S, t_1^\tau, wa, schk_1^m) = P(tchk_1^n|t_1^\tau) \qquad (6.4)$$

Approximation3: The chunk alignment model align source and target chunks based on word-alignments,

$$P(ca|s_1^S, t_1^\tau, wa, schk_1^m, tchk_1^n) = P(ca|wa, schk_1^m, tchk_1^n) \qquad (6.5)$$

Approximation4: The segment extraction model produces segment pairs using information from the chunk alignments.

$$P(b|s_1^S, t_1^\tau, wa, schk_1^m, tchk_1^n, ca) = P(b|schk_1^m, tchk_1^n, ca) \qquad (6.6)$$

With the above approximations, Eqn. 6.2 can be re-written as,

$$\hat{B}(s_1^S, t_1^\tau) = \arg\max_b \sum_{ca,wa,schk_1^m,tchk_1^n} P(wa|s_1^S, t_1^\tau) \;\; X \;\; P(schk_1^m|s_1^S)\; X \;\; P(tchk_1^n|t_1^\tau) \;\; X \;\; P(ca|wa, schk_1^m, tchk_1^n)\; X \;\; P(b|schk_1^m, tchk_1^n, ca) \tag{6.7}$$

We approximate the above equation further by using only the most probable source and target chunk splittings instead of summing over all possible chunk splittings.

$$\begin{aligned} Approximation5 &: \hat{s}chk_1^m = \arg\max_{schk_1^m} P(schk_1^m|s_1^S) \\ Approximation6 &: \hat{t}chk_1^n = \arg\max_{tchk_1^n} P(tchk_1^n|t_1^\tau) \end{aligned} \tag{6.8}$$

A beam of different splittings can be obtained but is not performed in this thesis. With the above approximations, Eqn. 6.7 now becomes,

$$\hat{B}(s_1^S, t_1^\tau) = \arg\max_b \sum_{ca,wa} P(wa|s_1^S, t_1^\tau)\; X \; P(ca|wa, \hat{s}chk_1^m, \hat{t}chk_1^n)\; X \; P(b|\hat{s}chk_1^m, \hat{t}chk_1^n, ca) \tag{6.9}$$

with,

$$P(schk_1^m|s_1^S) = \begin{cases} 1, & \text{for } \hat{s}chk_1^m = \arg\max_{schk_1^m} P(schk_1^m|s_1^S) \\ 0, & \text{otherwise} \end{cases} \tag{6.10}$$

and,

$$P(tchk_1^n|t_1^\tau) = \begin{cases} 1, & \text{for } \hat{t}chk_1^n = \arg\max_{tchk_1^n} P(tchk_1^n|t_1^\tau) \\ 0, & \text{otherwise} \end{cases} \tag{6.11}$$

Jointly maximizing the three probabilities in the Eqn. 6.9 is computationally expensive. Hence, we model the three probabilities separately recognizing this may lead

to sub-optimal $\hat{B}(s_1^S, t_1^\tau)$. We first find the best word-alignments between the source and target sentences ($P(wa|s_1^S, t_1^\tau)$). Using the best word alignments, we further align the source and target chunks ($P(ca|wa, \hat{s}chk_1^m, \hat{t}chk_1^n)$) and finally find the best segment boundaries($P(b|\hat{s}chk_1^m, \hat{t}chk_1^n, ca)$) with these chunk alignments. We proceed to explain the chunk alignment and the segment extraction model.

6.3.2 Chunk Alignment Model

We need to find the best possible alignments, $\hat{ca}$, between the source and target chunks. Text chunking can be performed with tools such as Lafferty et al. [2002]. Say the source chunker generated m chunks and the target language chunker generated n target chunks.

$$\hat{ca} = \arg\max_{ca} P(ca|\hat{s}chk_1^m, \hat{t}chk_1^n, wa) \tag{6.12}$$

We divide the problem into two directions, $P(tchk_l|schk_q)$ and $P(schk_q|tchk_l)$ with $l = 1, 2, ...n$ and $q = 1, 2, ...m$. As a given source (target) chunk could be aligned to more than one target (source) chunk, rather than finding the best possible chunk correspondences, we select all target (source) chunks with positive alignment probabilities for the given source (target) chunk,

$$SA_q = \Big\{ l : P(tchk_l|schk_q) > 0 \Big\} \tag{6.13}$$

where, SA_q stores the chunk alignments for the source chunk ($schk_q$). Similarly,

$$TA_l = \Big\{ q : P(schk_q|tchk_l) > 0 \Big\} \tag{6.14}$$

TA_l stores the chunk alignments for the target chunk ($tchk_l$).
$P(tchk_l|schk_q)$ is modeled as:

$$\begin{aligned} Score(tchk_l|schk_q) = \quad & \left[P(t_1^{ls-1}|s_1^{qs-1})^\lambda\right] X \left[P(t_{le+1}^\tau|s_1^{qs-1})^\lambda\right] X \\ & \left[P(t_1^{ls-1}|s_{qe+1}^S)^\lambda\right] X \left[P(t_{le+1}^\tau|s_{qe+1}^S)^\lambda\right] X \\ & \left[P(t_{ls}^{le}|s_{qs}^{qe})^{1-4\lambda}\right] \end{aligned} \tag{6.15}$$

where, ls and le are the start and end indices of $tchk_l$, qs and qe are the start and end indices of $schk_q$ (see Figure 6.2). λ ($0 \leq \lambda \leq \frac{1}{4}$) indicates the importance of the five regions in Figure 6.2.

The idea behind using the first four terms in Eqn. 6.15 is to find a boundary that is agreed upon not just by the source and target chunks under consideration but also by the neighboring regions. The first term corresponds to the solid-slant line region, the second term to the dashed-vertical line region, the third term to the dotted-horizontal line region, the fourth term to the solid-star region and the fifth term to the solid-vertical line region.

Assuming that each target word was generated independently given its source chunk (i.e., t_i is independent of t_j given its source chunk, with $i \neq j$), we obtain Eqn. 6.16.

$$
\begin{aligned}
Score(tchk_l|schk_q) = \quad & \Big[\prod_{i=1}^{ls-1} P(t_i|s_1^{qs-1})\Big]^{\frac{\lambda}{ls-1}} \; X \; \Big[\prod_{i=le+1}^{\tau} P(t_i|s_1^{qs-1})\Big]^{\frac{\lambda}{\tau-le}} \; X \\
& \Big[\prod_{i=1}^{ls-1} P(t_i|s_{qe+1}^{S})\Big]^{\frac{\lambda}{ls-1}} \; X \; \Big[\prod_{i=le+1}^{\tau} P(t_i|s_{qe+1}^{S})\Big]^{\frac{\lambda}{\tau-le}} \; X \\
& \Big[\prod_{i=ls}^{le} P(t_i|s_{qs}^{qe})\Big]^{\frac{1-4\lambda}{ls-le+1}} \qquad (6.16)
\end{aligned}
$$

For each of the five terms in Eqn. (6.17), we assume that the generation of a target word by a source word is independent of other words in the same source chunk, i.e., if s_j generated t_i then, $P(t_i|s_1, s_2, ...) = P(t_i|s_j)$. It should be noted that, our method does allow multiple source words to generate the same target word (and vice versa), hence, a single target word can have multiple source word correspondences. We emphasize that $Score(tchk_l|schk_q)$ is set to zero if none of the source words in $schk_q$ have correspondences in $tchk_l$.

$$
\begin{aligned}
Score(tchk_l|schk_q) = \quad & \Big[\prod_{i=1}^{ls-1} \frac{1}{qs-1} \sum_{j=1}^{qs-1} P(t_i|s_j)\Big]^{\frac{\lambda}{ls-1}} X \\
& \Big[\prod_{i=le+1}^{\tau} \frac{1}{qs-1} \sum_{j=1}^{qs-1} P(t_i|s_j)\Big]^{\frac{\lambda}{\tau-le}} X \\
& \Big[\prod_{i=1}^{ls-1} \frac{1}{S-qe} \sum_{j=qe+1}^{S} P(t_i|s_j)\Big]^{\frac{\lambda}{ls-1}} X \\
& \Big[\prod_{i=le+1}^{\tau} \frac{1}{S-qe} \sum_{j=qe+1}^{S} P(t_i|s_j)\Big]^{\frac{\lambda}{\tau-le}} X \\
& \Big[\prod_{i=ls}^{le} \frac{1}{qe-qs+1} \sum_{j=qs}^{qe} P(t_i|s_j)\Big]^{\frac{1-4\lambda}{ls-le+1}} \qquad (6.17)
\end{aligned}
$$

Equation (6.17) looks similar to the equation for extracting phrase-pairs in Vogel [2005] (segment-pairs in our case), however, we weigh each of the five probability terms separately to normalize each term by the number of factors that contribute to them. Kim et al. [2010] (developed at the same time as this thesis work) also make use of chunkers to find better GIZA++ [Och and Ney, 2003] word-alignments, however, their score function (F) for aligning a source chunk ($schk_q$) and a target chunk ($tchk_l$), is given by: $F(tchk_l, schk_q) = \big(\frac{1}{qe-qs+1} \sum_{j=qs}^{qe} max_{i=ls}^{le} P(t_i|s_j)\big)$, where, they model $only$ the last term in Eqn. (6.17). They also make the assumption that each source word generates only one target word.

6.3.3 Segment extraction model

Errors caused by automatic text chunkers and mismatches in the number of source and target chunks are handled partly by this model. We take the union of possible chunk alignments (Figure 6.3) in $\chi_{m \mathbf{x} n}$ from Eqn. (6.13),

$$\chi_{i,j} = \begin{cases} \frac{1}{2}[Score(tchk_j|schk_i) + Score(schk_i|tchk_j)], & \text{if } j \in SA_i \text{ or } i \in TA_j \\ 0, & \text{otherwise} \end{cases} \tag{6.18}$$

We extract consistent segment-pairs of length less than $(S-1)$ words on the source side and $(\tau - 1)$ words on the target side. The procedure is similar to that of Zens et al. [2002] and Koehn et al. [2007] where the boundary (BP) of consistent pairs is defined over words but here we define them over chunks.

$$\begin{aligned} &BP(\hat{s}chk_1^m, \hat{t}chk_1^n, \hat{c}a) \quad = \\ &\quad \big\{(schk_j^{j+h}, tchk_i^{i+w}) : \exists (i', j') \in \hat{c}a, (j \leq j' \leq j+h, i \leq i' \leq i+w)\big\} \text{ and,} \\ &\quad \big\{(schk_j^{j+h}, tchk_i^{i+w}) : \nexists (i', j') \in \hat{c}a, (j' \notin \{j, ...j+h\}, i' \in \{i, ...i+w\})\big\} \text{ or,} \\ &\quad \big\{(schk_j^{j+h}, tchk_i^{i+w}) : \nexists (i', j') \in \hat{c}a, (j' \in \{j, ...j+h\}, i' \notin \{i, ...i+w\})\big\} \end{aligned} \tag{6.19}$$

Equation (6.19) implies that to form a consistent segment-pair, source chunks within a segment-pair need to be aligned to target chunks within the segment-pair boundary only and not to any target chunks outside the boundary and vice versa. For example, in Figure 6.3, the region in the solid-blue box is a consistent segment-pair, whereas, the region in the dotted-red box is not as the target chunk, `[NP 合作]` within the boundary is aligned to a source chunk `[VBN coordinated]` outside the boundary.

6.3.4 Filtering

An analysis of the segment-pairs extracted using the segment extraction model showed that many were not of good quality. A blame assignment analysis indicated that this was

due to poor word-alignments and chunking errors.
To counter this we introduced a filtration step to detect and remove such segment-pairs. Ideally, we would like a classifier that indicates whether a segment-pair should be included in our system or not to maximize the output BLEU score. However, this is a highly non-linear problem and would in general require re-creating the templates and indexing the corpus many times during the learning phase - a computationally infeasible approach.

Filtering as a Classification Task

Instead, we learn a simple to compute measure of 'goodness' of a segment-pair that serves as a computational surrogate for the output BLEU score of a translation system. We will then train a classifier that given a segment-pair will output a 1 if it is 'good' and 0 otherwise. In order to learn this measure we need an initial source of labelled data. For this a small set of segment-pairs can be chosen randomly and given to a bilingual expert who understands the language-pair. The expert then gives a label of 0 if at least one word needs to be changed in the segment-pair and 1 if there are no changes required. This data can then be used to train a classifier to classify the rest of the segment-pairs.

In the absence of Human labelers

This method can be extended to situations where an expert is not available by using another Machine Translation system trained on a large corpus as an expert black box. Since it would be expensive to translate all the segment-pairs, a small set can be randomly drawn and their source and target-halves can be translated by the MT system. If the translations match the segment-pairs perfectly then a label of 1 can be assigned (Figure 6.6).

We pause now for a moment to explain why we used the above procedure. The very existence of a good machine translation system would seem to indicate that the language does not suffer from data sparsity. However, in our experiments we did not have a human to translate the segment-pairs and since we were simulating sparsity by extracting small data sets from a larger corpus, we could treat the translations of the bigger system as translations of a human. In real data-sparse conditions, a human will be required in the loop to obtain the data for training a classifier. So, our method of using a black box MT system is intended to *simulate* a human labeler of segment-pairs. Our experiments show that this is a more efficient use of the expert resources. In addition, we feel that this is a very interesting method of extracting labels from an expert that may be useful in other cases as well. Consider a case where the phrase-table of an MT system needs to be mounted on a small memory device like a PDA. The above procedure can be used with either the

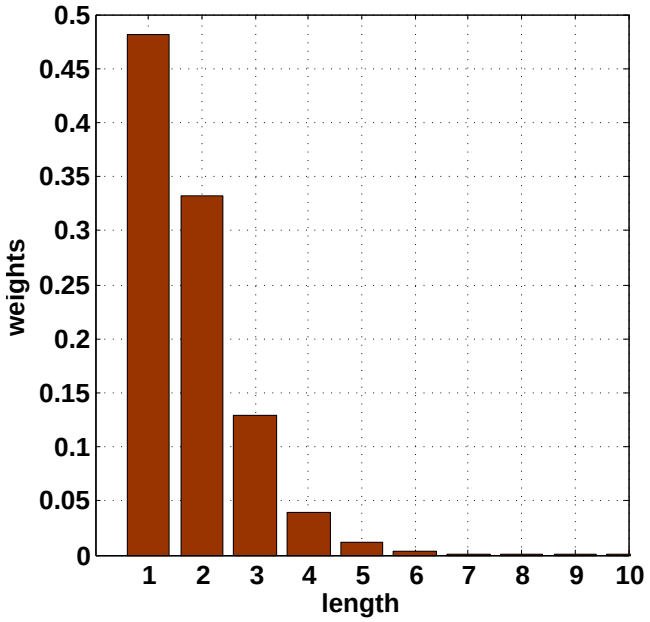

Figure 6.5: Weights for the n-gram matches.

original MT system trained on the larger data set or with a human labeler to decide which translations to store on the device.

In the absence of Human labelers: defining leniency

Since none of the Machine Translation systems today are 100% accurate, some leniency is required while matching the segment-pairs to the MT translations. We define leniency by the amount the black box MT system diverges from the true translations. For this we used a development set of 200 sentence-pairs and translated the source side ($s \rightarrow t$) and the target side ($s \leftarrow t$) of the language-pair under consideration using the black box MT system. We find the quality score by linearly combining all the n-gram matches between the translations and the references as follows,

$$th = \sum_{n=1,...N} w_n * \frac{2 * \#co - occurring\ n - grams}{\#n - grams\ in\ the\ ref + \#n - grams\ in\ the\ system\ output} \tag{6.20}$$

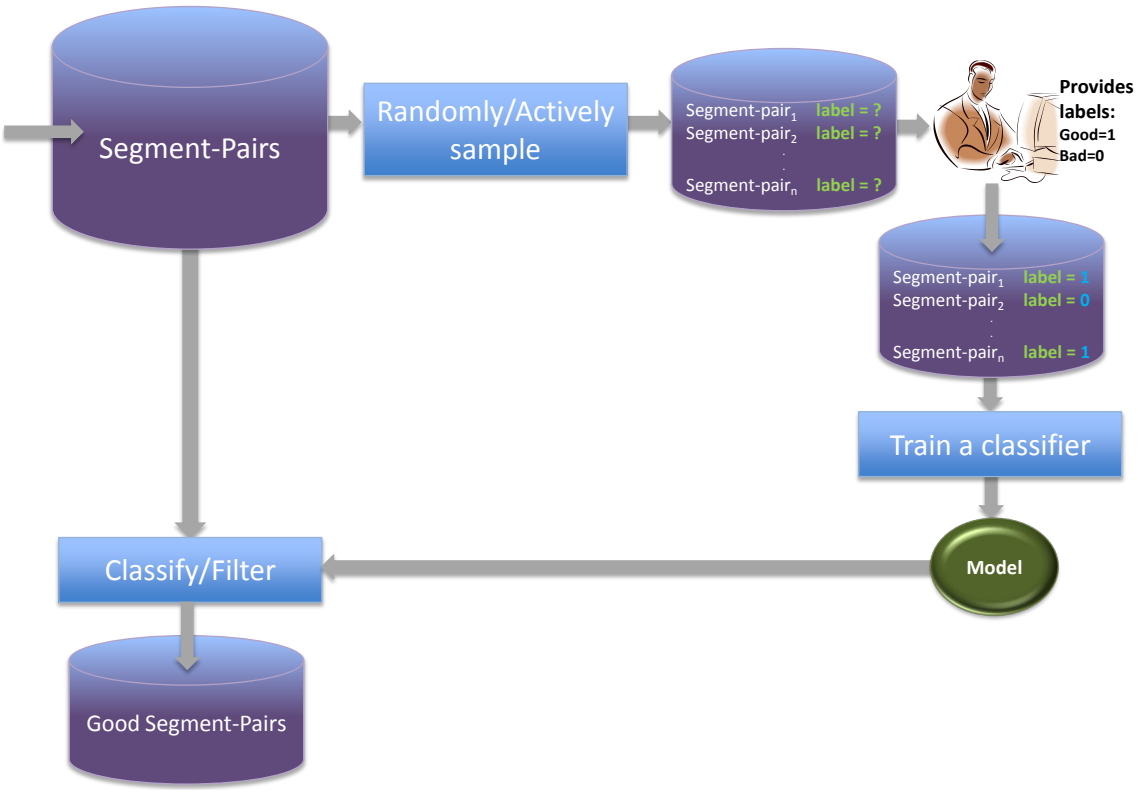

Figure 6.6: Filtering as a Classification Task

We computed the distribution of the length of the segment-pairs (Figure 6.5) that were extracted previously and used the probabilities as weights ($w_1, ...w_N$) for the quality score. The main idea behind weighing each n-gram score differently is as follows. If most of the segment-pairs have only two words on their target side, then we want the black box MT system to be more accurate at obtaining bi-gram translations with respect to the reference translations and so a higher weight should be given to the bi-gram matches between the translations and the references to penalize/reward the black-box MT system for bi-grams more than any other higher order n-grams. Hence, the weight sets the importance of a particular n. This in turn has a larger effect on the overall EBMT system since there are a large number of bi-grams in the extracted segment-pairs. The quality score can then be used as a threshold (separate thresholds for translating source to target $th_{s \rightarrow t}$ and target to source $th_{s \leftarrow t}$) to decide if a segment-pair should receive a label of 0 or 1.

For example, in our experiments with the 30k Eng-Chi data set(see Section 3.4 for details on the data set), $th_{s \rightarrow t}$ ($s \rightarrow t$: while comparing references and translations in Chinese) was found to be 0.714. This implies that for a segment-pair to obtain a label of 1, it is enough if 71.4% of the target words of the segment-pair match with that of the black box MT system. Similarly, $th_{s \leftarrow t}$ ($t \rightarrow s$: while comparing references and translations in English) was found to be 0.769. The threshold value for $th_{s \leftarrow t}$ is higher and is not surprising because the black box MT system performed much better when translating to English due to the usage of larger language models for English.

We will now proceed to explain the features that were used for Classification.

Features for Classification

We finally extract a set of features based on alignment scores, length of segment-pairs and source-target labels that are good indicators of the 'goodness' of a segment-pair. Each segment-pair is represented as a feature vector containing the following 9 features. Say a consistent segment-pair contains source chunks: $schk_j....schk_{j+h}$ and target chunks: $tchk_i....tchk_{i+w}$

Feature1: computed as the average of chunk alignment scores (χ defined in Eqn. (6.18)) of the segment-pair

$$\text{Feature1} = \frac{\sum_{x=i}^{i+w} \sum_{y=j}^{j+h} \chi x,y}{(h+1) * (w+1)}$$

Feature2 and **Feature3**: fraction of chunk alignments within the segment-pair

$$\text{Feature2} = \frac{\sum_{g=i}^{i+w} sgn\Big[1_{h+1}^{T} sgn(\chi_{j:j+h,g})\Big]}{w+1}$$

where 1_{h+1}^{T} is a row vector of ones of size $h+1$, $\chi_{j:j+h,g}$ is a column vector corresponding to rows j to $j+h$ and column g of χ.

For example, if we are computing $Feature2$ for the consistent segment-pair: `[NP(5 men) VBN(coordinated) PP(for 20 years)]` ↔ `[NP(二十 年 来) PU(,) NP(五 个 男人) NP(合作)]` that has 3 source chunks and 4 target chunks, so the matrix under consideration has 3 rows (i.e., $h+1$=3) and 4 columns (i.e., $w+1$=4) as follows:

$$1_3^T = \begin{bmatrix} 1 & 1 & 1 \end{bmatrix}$$

and,

$$sgn(\chi_{1:3,1:4}) = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \end{bmatrix}$$

So,

$$1_3^T * sgn(\chi_{1:3,1:4}) = \begin{bmatrix} 1 & 0 & 1 & 1 \end{bmatrix}$$

and,

$$sgn\Big[1_3^T * sgn(\chi_{1:3,1:4})\Big] = \begin{bmatrix} 1 & 0 & 1 & 1 \end{bmatrix}$$

Summing the columns ($i : i+w$) and dividing by $w+1$ gives $Feature2$= $\frac{3}{4}$. This score implies that: of the 4 columns, 3 columns have one or more alignment points. Similarly,

$$\text{Feature3} = \frac{\sum_{g=j}^{j+h} sgn\Big[sgn(\chi_{g,i:i+w}) 1_{w+1}\Big]}{h+1}$$

where, 1_{w+1} is a column vector of ones of size $w+1$, $\chi_{g,i:i+w}$ is a row vector corresponding to columns i to $i+w$ and row g of χ.

Feature4: Number of words in the source-half of the segment-pair.

$$\text{Feature4} = S$$

Feature5: Number of words in the target-half of the segment-pair.

$$\text{Feature5} = T$$

Feature6: Number of chunks in the target half of the segment-pair

$$\text{Feature6} = w + 1$$

Feature7: Number of chunks in the source-half of the segment-pair

$$\text{Feature7} = h + 1$$

Feature8 and **Feature9**: Since syntactic labels for the source and target chunks are available, we could compute the probability of observing the source-chunk label sequence and the target-chunk label sequence. Maximum likelihood estimates for these probabilities are obtained from a labeled corpus.

$$\text{Feature8} = \frac{0.5 * P(label_{schk_j} .... label_{schk_{j+h}})}{P(label_{schk_j}) * P(label_{schk_{j+1}}) * ..... * P(label_{schk_{j+h}})} + \frac{0.5 * P(label_{tchk_i} .... label_{tchk_{i+w}})}{P(label_{tchk_i}) * P(label_{tchk_{i+1}}) * ..... * P(label_{tchk_{i+w}})}$$

$$\text{Feature9} = 0.5 * [P(label_{schk_j}, ...label_{schk_{j+h}} \mid label_{tchk_i}, ...label_{tchk_{i+w}}) + P(label_{tchk_i}, ...label_{tchk_{i+w}} \mid label_{schk_j}, ...label_{schk_{j+h}})]$$

Once these features have been extracted for all the segment-pairs, they are normalized to have mean of 0 and variance of 1. The length bias in **Feature8** is removed by normalizing the scores separately based on the length of the segment-pairs.

We used Support Vector Machines to train and classify the segment-pairs. For training the classifier, 2000 segment-pairs were picked randomly and were labeled 1 if the fraction

of matches of the target side of the segment-pair and the translation of the black box MT was greater than $th_{s \rightarrow t}$ or if the fraction of matches of the source side of the segment-pair and the translation of the black box MT (when translating the target to its source) was greater than $th_{s \leftarrow t}$. The classifier gave an accuracy of 83% with leave-one-out cross-validation.

6.3.5 Clustering: Based on chunk label sequences or syntactic labels

Now that a manageable number of segment-pairs are extracted, clustering is performed using the chunk label sequences of the segment-pairs. Segment-pairs are clustered based on their source-target chunk label sequences. The segment-pairs corresponding to `[5 men]` and `[this work]` from Figure 6.3 are clustered under the `<NP>` class as their source label sequences are the same (i.e., `NP`) and their target label sequences are the same (i.e., `NP`). It is not necessary for both the source and the target segments to have the same sequence of labels. In Figure 6.3, the source segment `[for 20 years]` has a single chunk label PP aligned to a target segment `[二十 年 来]` with a single chunk label NP, under such label-mismatch conditions, the segment-pair is clustered under `<Ci>` whose members all have a single PP chunk label on the source side and a single NP label on the target side. Consider, `[people protested for 6 months]` aligned to `[六 个 月 来 , 人们 一直 抗议]`. It has the source chunk label sequence `[NP VBN PP]` and target chunk label sequence `[NP PU NP NP]`. This matches the source and target label sequence of `[5 men coordinated for 20 years]` aligned to `[二十 年 来 , 五 个 男人 合作]`, hence these two segment-pairs will be put under the same class `<Cj>`.

6.4 Results

The experiments in this section use the Eng-Chi, Eng-Fre and Eng-Hai data sets described in Section 3.4. We had the Stanford parsed data [Levy and Manning, 2003] for both Chinese and English and so we obtained chunks and phrase labels from these parse trees using a set of rules. For Eng-Fre training data, chunking was performed using Schmid [1994] on English and French independently.

The most important rules used in chunking Eng-Chi data are as follows. A subtree was made a chunk if it included only words and POS tags. For example, if there is a subtree such as `[NP (NN propaganda) (NN drive)]`, the subtree that qualifies to be a chunk is `[NP propaganda drive]` and not the unary rules containing the POS

Language-Pair	Training data size	Baseline	G-EBMT
Eng-Chi	15k 30k 200k	0.1076 0.1245 0.1785	0.1169 0.1310 0.1815
Eng-Fre	30k 100k	0.1577 0.1723	0.1667 0.1778

Table 6.1: Comparison of translation scores of the Baseline system and G-EBMT system with Phrase-Generalization. Statistically significant improvements with $p < 0.0001$.

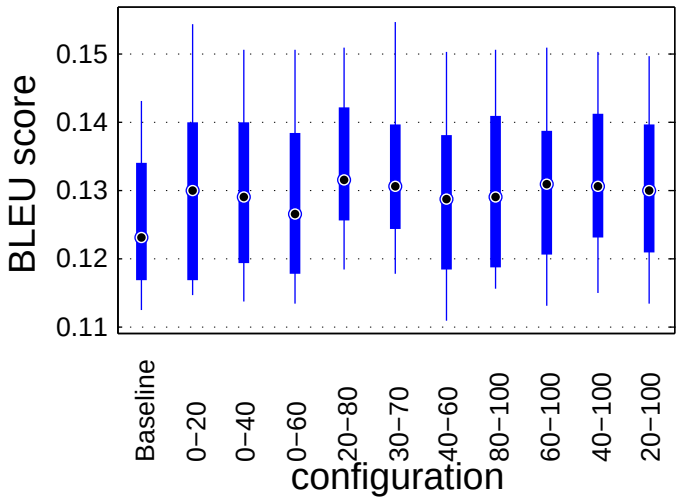

Figure 6.7: BLUE scores with segment-pairs filtered at various percentile intervals of segment-pair frequencies.

tag and the word, i.e., [NN propaganda] and [NN drive] are not eligible chunks. The trees were flattened based on subtrees closer to the leaves, making sure that subtrees within complex embeddings are flattened correctly. When a PP contains a single NP, the NP was not separated. If a PP has more than one phrase, then the preposition is made one chunk and the other phrases are flattened as separate chunks. Verbs and conjunctions were separated as chunks with their POS tag as their chunk label.

6.4.1 Template-based vs. Baseline EBMT

We used the phrase-based EBMT system with no templates as our baseline system. It is known that increasing the amount of training data in a generalized-EBMT system eventually leads to saturation in performance [Brown, 2000], where all template-generation methods perform about as well as the phrasal-EBMT baseline with no templates. The same is true with phrases that appear frequently. In order to find the right percentile interval where the template-based system provides the highest improvement, the segment-pairs from Section 6.3.3 were first sorted in ascending order based on their frequency of occurrence in the training data. For a particular percentile interval, say 20%-80%, we clustered segment-pairs that belong to the percentile interval only and created templates with the resulting clusters. Figure 6.7 shows the effect of various percentile intervals on 30k Eng-Chi where the templates were obtained from syntactically clustered segment-pairs. In this case, segment-pairs from the 20% to 80% (mid-frequency) region produce better scores as the interquartile range (extension of the box) in the box plot of $20 - 80$ (in Figure 6.7) is smaller when compared to the other percentile intervals and also its upper and lower quartiles are higher than other box plots for other percentile intervals. With all data sets, higher improvements were seen with segment-pairs from the mid-frequency region.

The overall translation score obtained with the baseline system and the template-based system for both the language-pairs are shown in Table 6.1. Improvements were seen on all the subfiles and were found to be statistically significant ($p < 0.0001$).

6.4.2 Two levels of generalization

The segment extraction model collected a large number of segments with single chunk labels. Many of the segments contained just NP or PP as their only chunk labels. To make segment-pairs containing NP or PP chunk labels in their sequence of labels to be interchangeable with any cluster member of NP or PP, we performed another level of

generalization resulting in a two level hierarchical model. For example, the segment-pair that contains `[5 men coordinated]` was converted to [<NP> coordinated] and the translation of `[5 men]` in the target phrase was also replaced by <NP>.

The performance of this two level generalization was tested on 200k Eng-Chi. The performance was better on only 9 sub files when compared to the system with 1 level of generalization and was not statistically significant.

6.4.3 Further Analysis

In this section, we will further analyze the output of the translation model and the resultant translations from the decoder to understand the benefits of phrase-clustering as was done in Chapter 5. The analyses in this section are only performed on the 30k Eng-Chi training corpus with segment-pairs from the best percentile interval (20% to 80%).

Coverage

The coverage analysis that was performed in Sect. 5.6.6 to see how many source phrasal matches could be obtained with respect to the test set is also performed here with phrase-generalization. Figure 6.8 shows the number of matching $n-$grams in the test set- with and without generalization. The plot indicates a small increase in the number of source phrasal matches with generalization when compared to the number of source phrasal matches without generalization. Since segment-pairs from the mid-frequency region of 20%-80% percentile are used, many of the word-pairs are discarded as they belong to the higher percentile levels (as words appear more frequently than phrases of length greater than 1). Also, the total number of matches for the test sentences (from the corpus) obtained with phrase-generalization are much lower than the number of matches obtained with word-generalization (in Chapter 5) as word matches are more likely to occur in the test set than phrases (of length > 1).

Percentage of words generalized

As was done in Section 5.6.6, we generalized segment-pairs obtained from the clusters of each of the percentile intervals. Since we were using segment-pairs to generalize the corpus, if a source phrase of v words was generalized, then the count for the number of words generalized was incremented by v. Figure 6.9 shows the percentage of words generalized for the 30k Eng-Chi training data set. The same behavior as was seen in

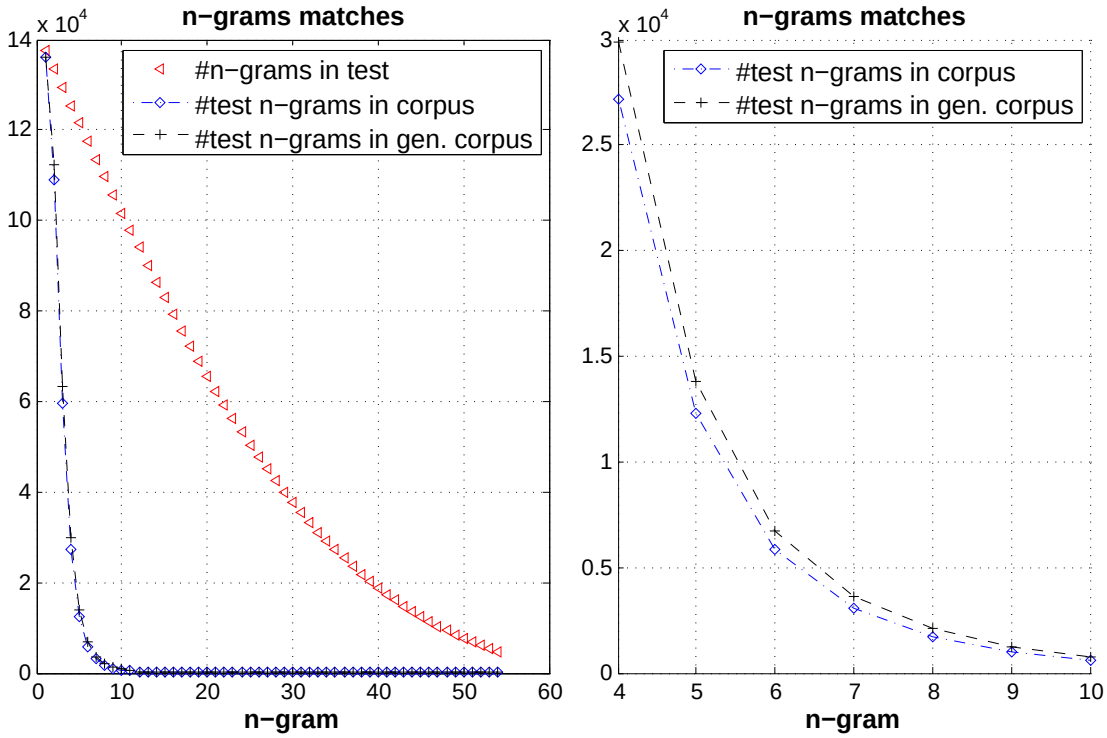

Figure 6.8: Left hand side plot: Number of n-grams (i) in the test set (ii) matches between the test set and source side of 30k Eng-Chi (iii) matches between the generalized test set and generalized source side of 30k Eng-Chi. The right-hand side figure shows a closer look of the same plot.

Section 5.6.6 with various frequency regions was also seen here. Again we notice that although fewer words were generalized with the 20-80 percentile interval when compared to the higher frequency intervals, the translation quality score obtained with this interval was found to be the best. Also, the 20-80 percentile interval had about $18k$ sentence-pairs (out of 30k) containing at least one phrase generalized. The higher percentile intervals (80-100, 60-100, 40-100, 20-100) had almost all the sentence-pairs with at least one phrase generalized.

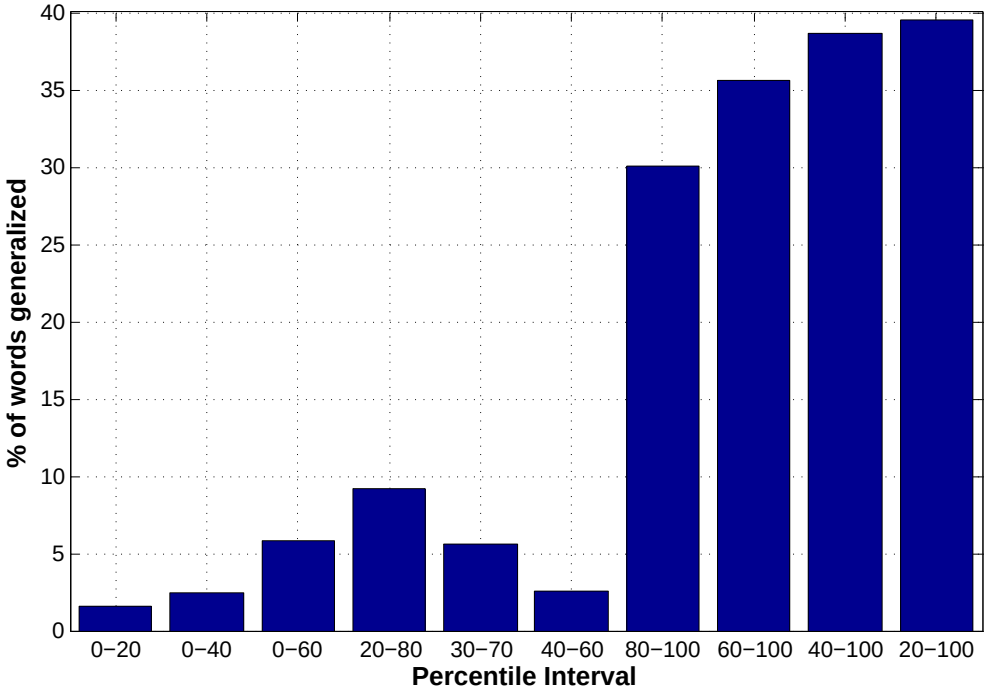

Figure 6.9: % of words generalized (with respect to the training corpus) with segment-pairs from each of the percentile intervals with the 30k Eng-Chi training data set.

Output Analysis: translations and target phrases obtained from the translation model

An example of a phrase-pair (or candidate translation pair) from the output of the G-EBMT's TM that was generated due to generalization is as follows:
<CL8>$_1$ of the chinese <CL8>$_2$ $\leftrightarrow$ 中国 <CL8>$_2$ <CL8>$_1$
The subscripts are used to disambiguate source-target correspondences when similar labels are present in the candidate translation pair. After putting back the values of the class labels:
vice premier of the chinese state council $\leftrightarrow$ 中国 国务院 副总理

The phrasal G-EBMT system was able to generalize 1712 test sentences out of the 4000 test sentences. The plot in Figure 6.10 shows the number of lexical (no generalizations) and new generalized phrase-pairs (whose target halves with class labels replaced by their corresponding values, were not present in the lexical phrase-pairs) with respect to

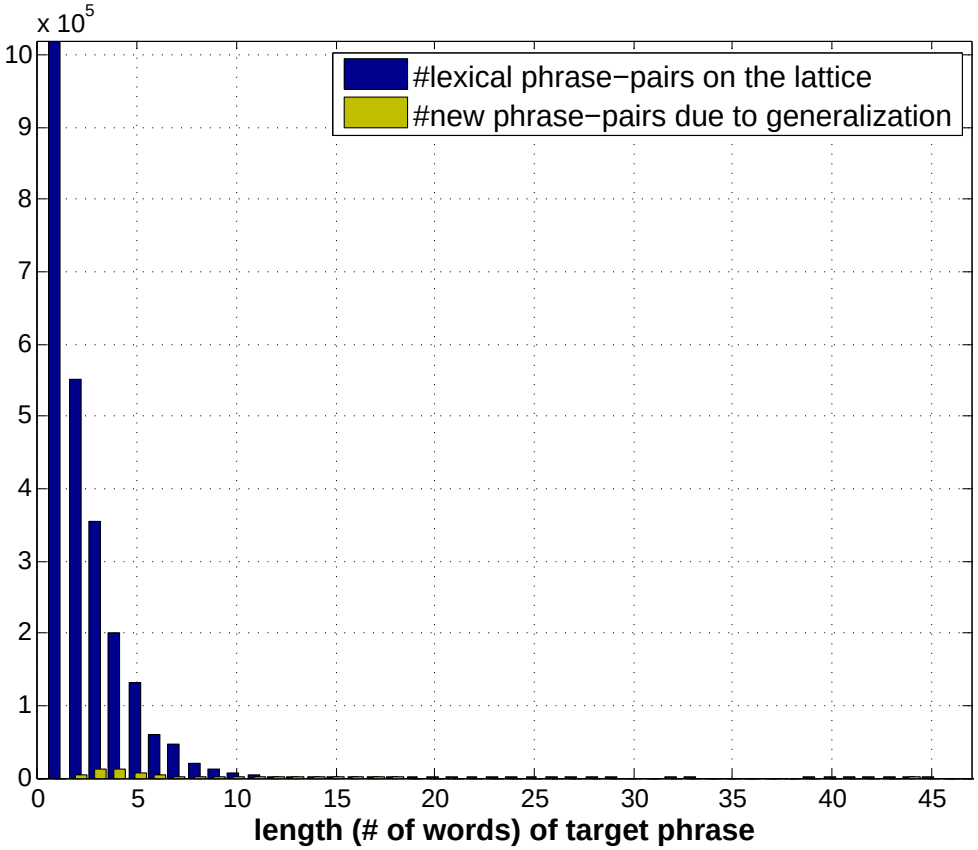

Figure 6.10: Number of phrase-pairs with increasing values of the length of the target halves (i) from lexical phrase-pairs only (ii) new phrase-pairs solely due to generalization. Max-Alternative=25.

the length of the target phrases present in the output of the G-EBMT's TM. From the best path information of the decoder- of the 1712 sentences, translations of 1040 test sentences contained partial translations that were generated due to generalization. The 'Maximum Alternatives' was set to 25.

We increased the 'Maximum Alternatives' to 200 to see if more new generalized phrase-pairs (whose target halves were not present in the lexical phrase-pairs) could be extracted and also to check whether generalization was really needed to generate new target fragments. For example, if a target phrase is generated by a lexical phrase-pair and also generated by the generalized phrase-pair (after replacing the values of the class la-

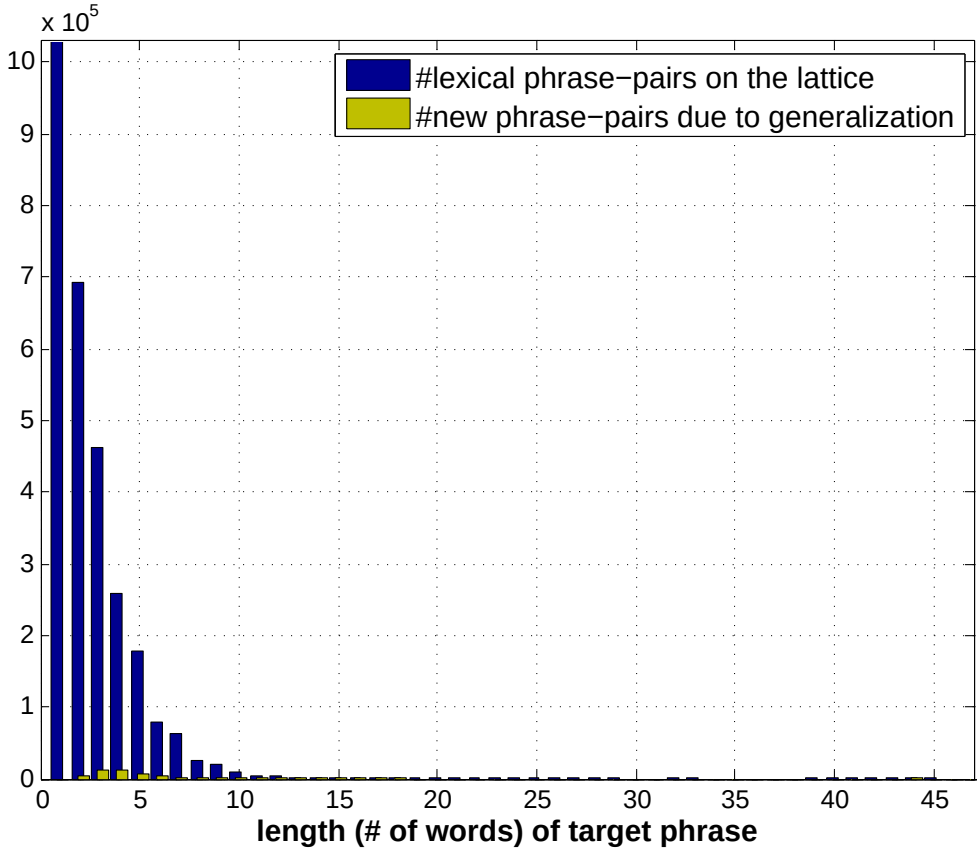

Figure 6.11: Number of phrase-pairs with increasing values of the length of the target halves (i) from lexical phrase-pairs (ii) new phrase-pairs due to generalization. Max-Alternative=200.

bels), then the case could be made that generalization is not helping. The plot in Figure 6.11 shows the number of lexical phrase-pairs and new phrase-pairs generated due to generalization with respect to the length of the target $n-$grams. A closer look at the same plot is given in Figure 6.12.

Our ultimate goal is to check whether the new candidate translations appear in the reference translations. Figure 6.13 shows number of target phrases that also appear in the reference translations. This can be treated as a lower bound on the number of grammatical candidate translations that the G-EBMT system is able to extract from the TM and shows that phrase-generalization is useful. The plot in 6.14 clearly indicates the increase

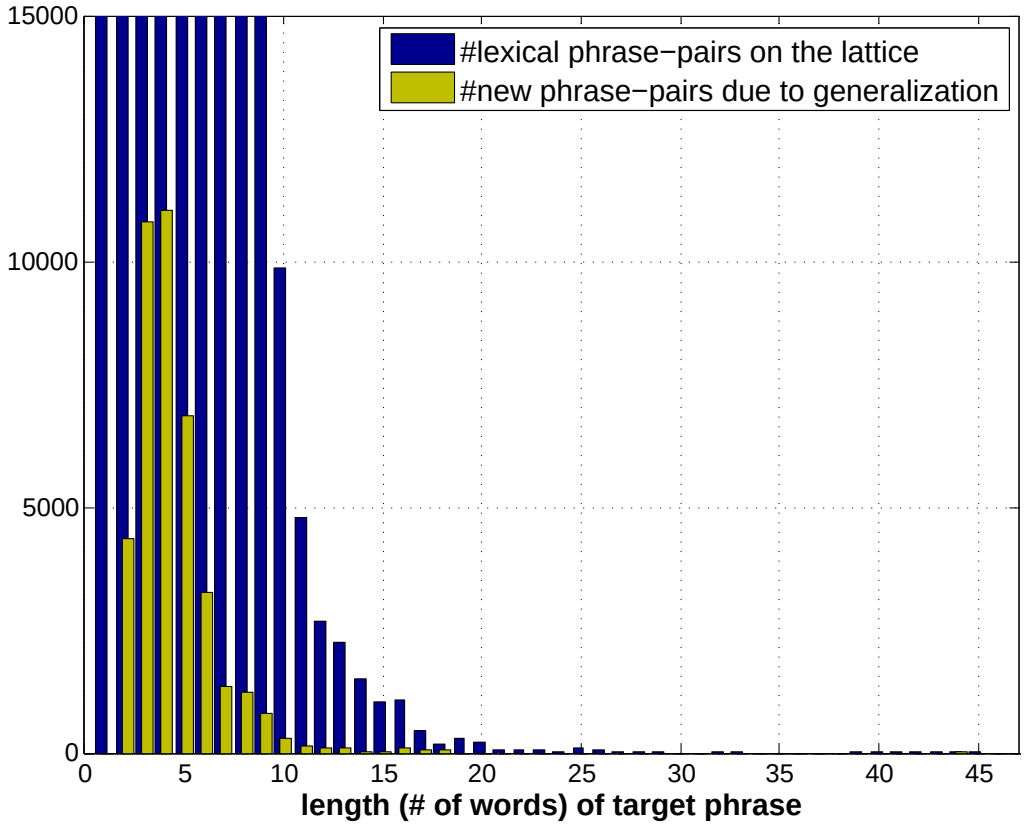

Figure 6.12: Closer look (same as Figure 6.11): Number of phrase-pairs with increasing values of the length of the target halves (i) from lexical phrase-pairs (ii) new phrase-pairs due to generalization. Max-Alternative=200.

in the number of useful (present in the reference translations) target phrasal matches when templates are applied.

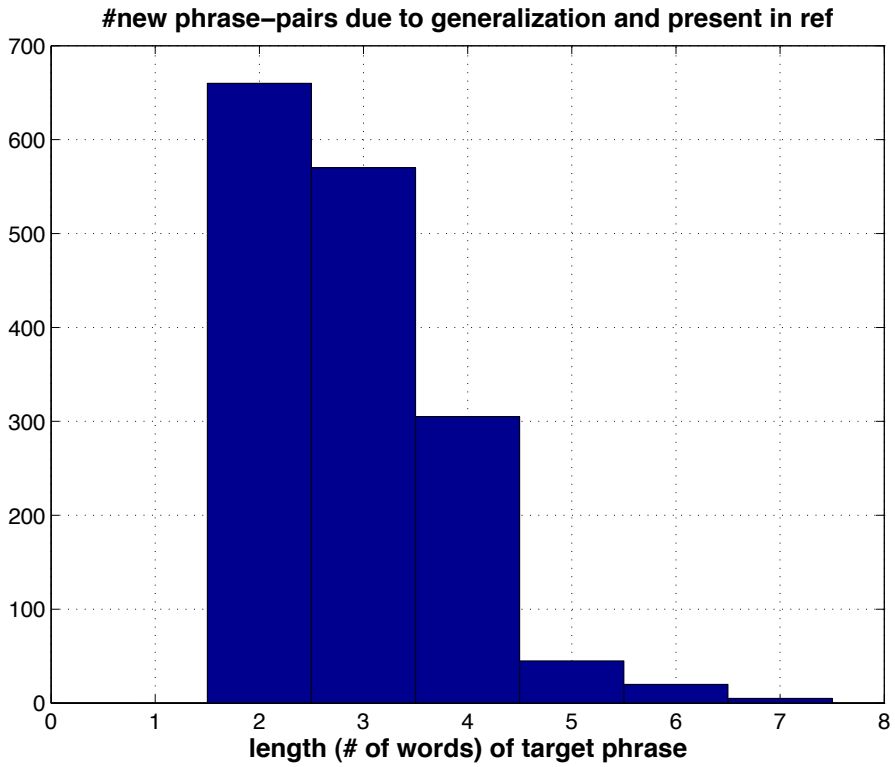

Figure 6.13: number of new partial translations solely due to generalization and present in the reference translations. Maximum-Alternatives=200.

Sample Clusters

A few sample clusters extracted by our method are given below:
Cluster1
tech zone chinese president jiang zemin ↔ 中国 国家主席 江泽民
the tibet autonomous region government ↔ 西藏 自治区 政府
chinese president jiang zemin ↔ 中国 国家主席 江泽民
the taiwan affairs office ↔ 台湾 事务 办公室
greek president stephanopoulos ↔ 希腊 总统 斯特凡诺普洛斯
the korean peninsula issue ↔ 朝鲜 半岛 问题
indian prime minister vajpayee ↔ 印度 总理 瓦杰帕伊
israeli prime minister sharon ↔ 以色列 总理 沙龙

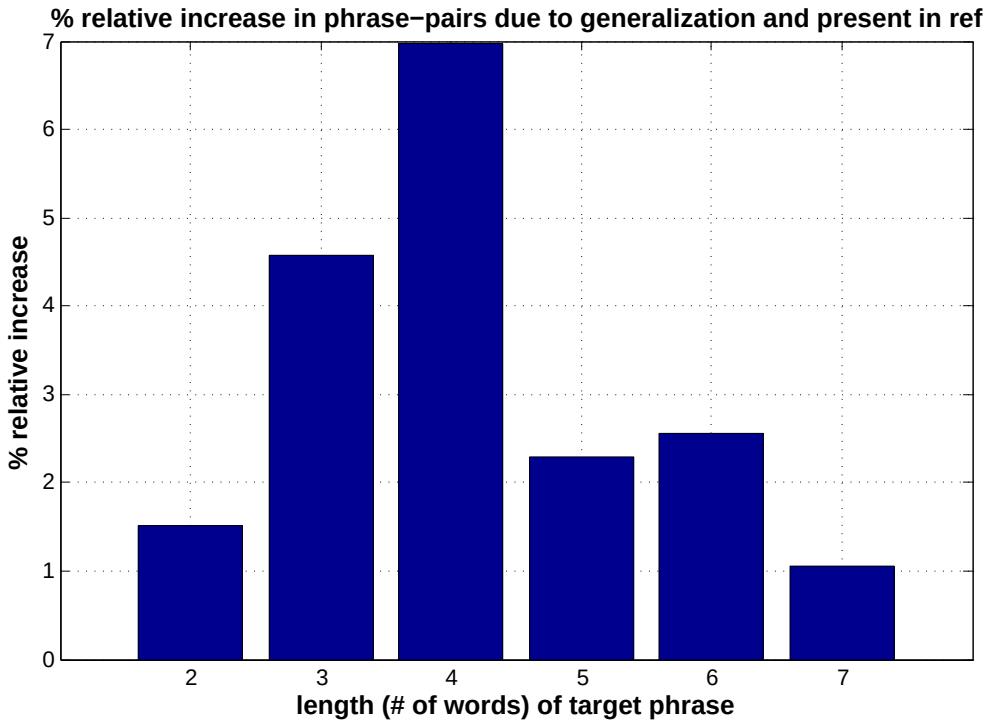

Figure 6.14: % Relative improvement in additional new (not found in the lexical phrase-pairs) partial translations solely due to generalization and present in the reference translations. Max-Alternatives=200.

social security system reform ↔ 社会 保障 体系 改革
the korean peninsula nuclear issue ↔ 朝鲜 半岛 核 问题
russian president vladimir putin ↔ 俄罗斯 总统 普京

Cluster2
per capital net income ↔ 人均 纯 收入
by relevant departments ↔ 有关 部门
through profound changes ↔ 深刻 变化
of relevant departments ↔ 有关 部门
of rapid development ↔ 快速 发展
of joint operations ↔ 联合 作战
with new changes ↔ 新 变化
on multilateral trade ↔ 多边 贸易

on low incomes ↔ 低 收入
through fair competition ↔ 公平 竞争
of friendly cooperation ↔ 友好 合作
with highly pathogenic avian influenza ↔ 高致病性 禽流感

Sample translations

A few sample translations produced by the baseline with no templates and the generalized system are given below. The highlighted text (in green) in the output translation of the G-EBMT system, indicates that the target phrase was generated by a source phrase that contained generalized tokens. The actual translation of its corresponding source phrase in the test sentence is also highlighted in the reference translation.

Test sentence:**beijing , 31 august (xinhua)** – presidential decree number 57 of the people 's republic of china (prc) :
Reference: 中华 人民 共和国 主席令 （ 第五十七 号 ） 新华社 北京 8 月 3 1 日 电 中华 人民 共和国 主席令
Baseline: 北京 1 月 3 1 日 新华社 8 月 电 （ 记者 张晓松 、 贺劲松 ） 总统 国务院令 批 57 的 ， 中华 人民 共和国 政府 ：
G-EBMT: 北京 8 月 3 1 日 电 （ 新华社 总统 国务院令 批 57 我 国 的 宪法 规定 了 中华 人民 共和国 政府 ：

Test sentence:**chairman hu pointed out that in the new century** and new phase , major changes have occurred in our army 's international and domestic situation and in the historic tasks that it shoulders .
Reference: 胡主席 指出 ， 新 世纪 新 阶段 ， 我军 所 处 的 国际 国内 环境 和 肩负 的 历史 使命 发生 了 重大 变化 。
Baseline: 主席 指出 ， 在 新 世纪 新 阶段 ， 发生 了 重大 变化 ， 我们 的 国际 国内 形势 ， 历史 任务 ， 肩负 。
G-EBMT: 胡主席 指出 ， 新 世纪 新 阶段 ， 发生 了 重大 变化 ， 我们 的 国际 国内 形势 ， 历史 任务 ， 肩负 。

Test sentence:with respect to the second generation of central leading collective **with comrade deng xiaoping as the core** , the achievements of comrade chen yun were even more unforgettable .
Reference: 对于 以 邓小平 同志 为 核心 的 中央 第二 代 领导 集体 来 说 ， 陈云 同志 更 是 功不可没 。
Baseline: 方面 ， 第二 代 中央 领导 集体 的 邓小平 同志 为 核心 ， 陈云 同志 的

更 足 忘怀 。
G-EBMT: 对于 第二 代 中央 领导 集体 的 邓小平 同志 为 核心 的 成果 , 陈云 同志 更 足 忘怀 。

To summarize, this chapter investigated another template-based approach that clustered segment-pairs as well as word-pairs based on their syntactic structure. We also explained a method to filter out unreliable segment-pairs using a set of feature scores. The overall G-EBMT system gave statistically significant improvements over the baseline EBMT system in translation quality ($p < 0.0001$) on all language-pairs and data sets.

Chapter 7

Templates in the Translation Model: using semantically related phrase-pairs

The previous chapter investigated a template-based approach where the members (segment-pairs) of the equivalence classes were grouped based on structural similarity. In this chapter, the segment-pairs obtained in Section 6.3.5 are clustered based on semantic similarity.

Just as in the previous chapter, the search space for finding semantically-related phrase-pairs (or segment-pairs) is also large. So we use the segment-pairs extracted in Chapter 6 (Section 6.3.3) but cluster them using a semantic-similarity metric which makes use of contextual information between pairs of segment-pairs rather than using their structural knowledge. This chapter can thus be treated as an extension of the previous chapter.

Features for computing similarity between segment-pairs are modeled as vector space models (VSMs) [Turney and Pantel, 2010]. Similar to word-context matrices, we construct segment*pair*-context matrices. The word-context matrix uses the *Distributional Hypothesis* [Harris, 1954] which indicates that words that occur in similar contexts tend to have similar meanings. [Callison-Burch et al., 2006] uses the distributional hypothesis to extract paraphrases for Phrase-based SMT.

In our case, the segment*pair*-context matrix can be treated as a loose version of pair-pattern matrix [Lin and Pantel, 2001] for segment-pairs. In the segment*pair*-context matrix, the context is treated as a bag of words while in the pair-pattern matrix each segment-pair can be treated as an entity that defines the relationship between the left and right context. Positive Point-wise Mutual Information (PPMI) is used to convert the segment*pair*-context matrix into a PPMI matrix. Bullinaria and Levy [2007] showed that PPMI outperforms many approaches for measuring semantic similarity. Once these clusters are

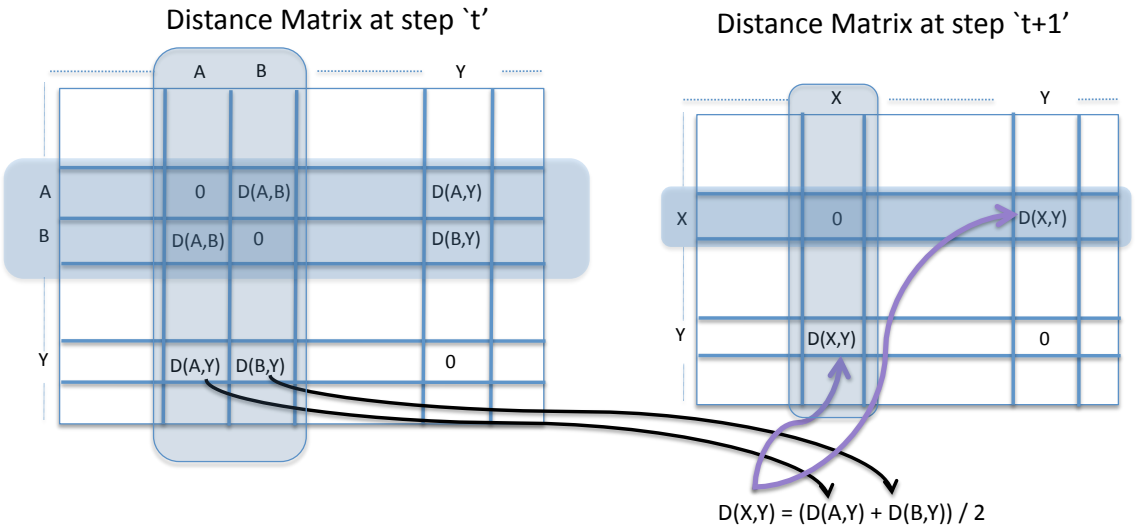

Figure 7.1: Updating distances while clustering segment-pairs. Cluster X is created by combining clusters, A and B. The distance between X and another cluster, Y, is updated as shown.

obtained, we proceed to the template induction.

7.1 Clustering based on semantic-relatedness of segment-pairs

In order to cluster segment-pairs, a pair-wise Adjacency matrix (Distance matrix=1-Adjacency matrix) is constructed with the i^{th} row and the j^{th} column corresponding to the similarity score between $segment\text{-}pair_i$ and $segment\text{-}pair_j$.

Any clustering algorithm that works on affinity (or Adjacency/Distance) matrices can then be used to cluster the segment-pairs. The Spectral Clustering algorithm that was described in Chapter 5 is a very good option. We have seen improvements with this algorithm while creating word-generalized templates. However, the algorithm is computationally expensive when made to cluster segment-pairs as the number of segment-pairs that can be extracted out of a corpus is substantially larger than the number of words in the same corpus. In our experiments we had about $30,000$ to $100,000$ segment-pairs to be clustered. So we choose a cheaper and faster algorithm called the hierarchical weighted-single-linkage

clustering. We do this knowing that using single-linkage clustering could lead to degradation in performance but we make this choice because of computational reasons. Another reason for adopting this approach is that hierarchical clustering also provides a principled way to determine the number of clusters [Goutte et al., 1998].

Weighted-single linkage is an agglomerative clustering algorithm where the clustering process begins with each data point in its own unique cluster. Pairs of clusters are then merged at each step of the algorithm. We use a weighted-average approach to decide which pairs of clusters to be combined. Weighted-average linkage uses a recursive definition for the distance between two clusters. So, if cluster X (Figure 7.1) was created by combining clusters A and B, the distance between X and another cluster Y is defined as the average of the distance between A and Y and the distance between B and Y:

$$d(X, Y) = \frac{d(A, Y) + d(B, Y)}{2} \tag{7.1}$$

To compute the pair-wise Adjacency matrix (Adj_{combi}) for clustering, an Adjacency matrix based on contextual scores ($Adj_{context}$) and an Adjacency matrix based on word token matches (Adj_{wm}) between pairs of segment-pairs is first obtained.

$Adj_{context}$: Adjacency matrix based on contextual scores

Since a segment-pair appears multiple times in a parallel corpus, a list of all words (along with their frequency of co-occurrence with the segment-pair) appearing within a window of two words prior to (left context) and two words following (right context) the source and target sides of the segment-pairs is first obtained. Hence, a segment***pair***-context matrix with segment-pairs as the rows and context words as the columns is obtained. Positive point-wise mutual information (PPMI) [Bullinaria and Levy, 2007] is then calculated from the frequency counts.

If X represents a segment***pair***-context frequency matrix with r rows and c columns. $X_{i,:}$ represents all elements in row i, $X_{:,j}$ represents all elements in column j. $X_{i,j}$ represents the number of times segment-pair sp_i appears with the context-word $cword_j$. The elements of the PPMI matrix are calculated as follows:

$$p_{i,j} = \frac{X_{i,j}}{\sum_{i=1,...r} \sum_{j=1,...c} X_{i,j}}$$

$$p_{i,*} = \frac{\sum_{j=1,..c} X_{i,j}}{\sum_{i=1,...r} \sum_{j=1,...c} X_{i,j}}$$

$$p_{*,j} = \frac{\sum_{i=1,..r} X_{i,j}}{\sum_{i=1,...r} \sum_{j=1,...c} X_{i,j}}$$

$$pmi_{i,j} = log\Big[\frac{p_{i,j}}{p_{i,*}\ p_{*,j}}\Big] \tag{7.2}$$

$$PPMI_{i,j} = \begin{cases} pmi_{i,j}, & \text{if } pmi_{i,j} > 0 \\ 0, & \text{otherwise} \end{cases} \tag{7.3}$$

where, $p_{i,j}$ is the estimated probability that the segment-pair sp_i occurs in the context $cword_j$, $p_{i,*}$ is the estimated probability of the segment-pair sp_i, $p_{*,j}$ is the estimated probability of the context word $cword_j$. The idea behind using PPMI is as follows. If $cword_j$ and SP_i are independent (co-occurred by random chance), then $pmi_{i,j}$ is zero as $p_{i,*}p_{*,j} = p_{i,j}$. If there is a strong semantic relation between SP_i and $cword_j$ then $p_{i,j} > (p_{i,*}p_{*,j})$ and $pmi_{i,j}$ will be greater than zero. If SP_i and $cword_j$ are unrelated, then $pmi_{i,j}$ will be less than zero. PPMI gives a high score only when there is a strong semantic relation between SP_i and $cword_j$ and gives a score of zero if the relation is uninformative.

Cosine similarity is then used to find similarity between all pairs of segment-pairs (or pairs of rows of the PPMI matrix) resulting in $Adj_{context}$. Hence, the i^{th} row and j^{th} column of the $Adj_{context}$ represents the contextual similarity between $segment\text{-}pair_i$ and $segment\text{-}pair_j$.

Adj_{wm}: Adjacency matrix based on word co-occurrences

The fraction of the number of source and target words in common between $segment\text{-}pair_i$ and $segment\text{-}pair_j$ is used to find $Adj_{wm(i,j)}$.

$$Adj_{wm(i,j)} = \frac{2 * \#co - occurring\ words(segment - pair_i, segment - pair_j)}{\#words\ in\ segment - pair_i + \#words\ in\ segment - pair_j} \quad (7.4)$$

To compute a combined similarity score between $segment\text{-}pair_i$ and $segment\text{-}pair_j$, $Adj_{context(i,j)}$ and $Adj_{wm(i,j)}$ are linearly combined.

$$Adj_{combi(i,j)} = c * Adj_{wm(i,j)} + (1 - c) * Adj_{context(i,j)} \quad (7.5)$$

Weights (c,1-c) are tuned with hill-climbing with the optimization function in Figure 7.6.

Adj_{combi} is then converted into a distance matrix $Dist_c(1)$ ($= 1 - Adj_{combi}$). Entries in $Dist_c(1)$ with a value of 1 are replaced by 1000 (indicating that the segment-pairs are infinitely far apart when similarity is 0). The clustering begins with each segment-pair as a separate cluster. Two closest clusters are merged iteratively until all the segment-pairs belong to one cluster. Clustering can be stopped when the algorithm tries to cluster two distant clusters. Figure 7.2 shows the average distance $Dist_c(t)$ between the two closest clusters that are merged at each step t with weight c ($0 \leq c \leq 1$) in Figure 7.5. For this example, clustering can be stopped at the 4019^{th} iteration (with *number of clusters=number of data points - 4019*) when a sudden change in average distance is observed. When the algorithm tries to combine clusters that are further apart (which happens after the algorithm has finished merging the closer clusters), the average distance jumps drastically.

$$\hat{c} = \arg\max_c [max(Dist_c(t+1) - Dist_c(t))] \quad (7.6)$$

7.2 Results

Since this chapter is an extension of the previous chapter, the segment-pairs were extracted from the same data sets that were used in the previous chapter (for more information on the data sets: Section 3.4). The test sets and scoring metric remain the same (see Section 6.4)

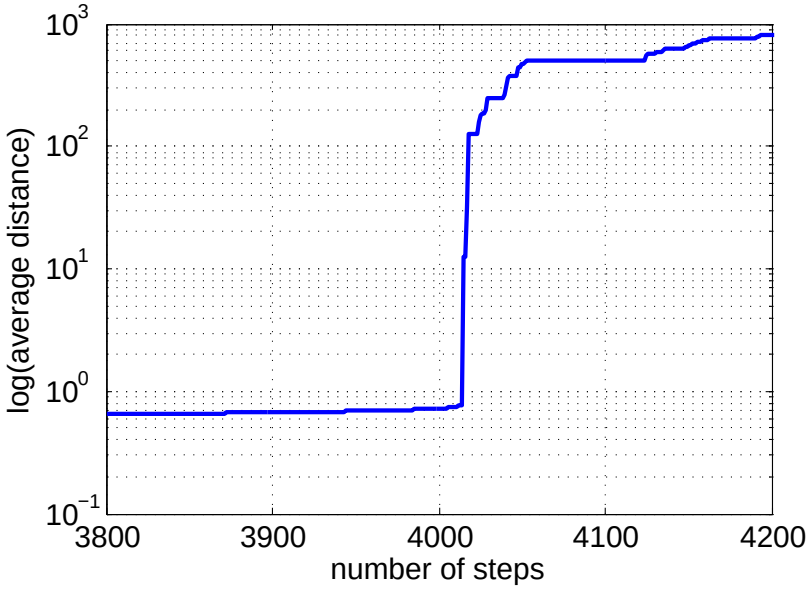

Figure 7.2: Average distance between clusters that are combined in each iteration.

7.2.1 Template-based vs. Baseline EBMT

We used the phrase-based EBMT system with no templates as our baseline system. As was done in the previous chapter: in order to find the right percentile interval where the template-based system provides the highest improvement, the segment-pairs obtained from Section 6.3.3 were first sorted in ascending order based on their frequency of occurrence in the training data. For a particular percentile interval, say 20%-80%, we clustered segment-pairs that belong to the percentile interval only and created templates with the resulting clusters. Again, higher improvements were seen with mid-frequency segment-pairs. Improvements were seen on all the subfiles and were found to be statistically significant ($p < 0.0001$).

7.2.2 Further Analysis

In this section, we will further analyze the output of the translation model and the resultant translations from the decoder. The analyses in this section are only performed on the 30k Eng-Chi training corpus.

Language-Pair	Training data size	Baseline	G-EBMT
Eng-Chi	15k	0.1076	0.1147
	30k	0.1245	0.1323
	200k	0.1785	0.1817
Eng-Fre	30k	0.1577	0.1718
	100k	0.1723	0.1811

Table 7.1: Comparison of translation scores of the Baseline system and G-EBMT system with Phrase-Generalization from syntactically related segment-pairs. Statistically significant improvements with $p < 0.0001$.

Coverage

The coverage analysis to see how many source phrasal matches could be obtained with respect to the test set that was performed in Sect. 5.6.6 was also performed here. Figure 7.3 shows the number of matching $n-$grams in the test set- with and without generalization. As observed in Chapter 6, only a small increase in the total number of source phrasal matches is seen with generalization when compared to the number of source phrasal matches without generalization.

Output Analysis: translations and target phrases obtained from the translation model

The G-EBMT system was able to generalize 2283 test sentences out of the 4000 test sentences. The plot in Figure 7.4 shows the number of lexical (no generalizations) and generalized phrase-pairs with respect to the length of the target phrases present in the output of the G-EBMT's TM. From the best path information of the decoder- of the 2283 sentences, translations of 1434 test sentences contained partial translations that were generated due to generalization. The maximum alternatives was 25.

We increased the maximum alternatives to 200 to see if more new generalized phrase-pairs (whose target halves were not present in the lexical phrase-pairs) could be extracted and also to check whether generalization was really needed to generate new target fragments. As mentioned in Chapter 6, if a target phrase is generated by a lexical phrase-pair

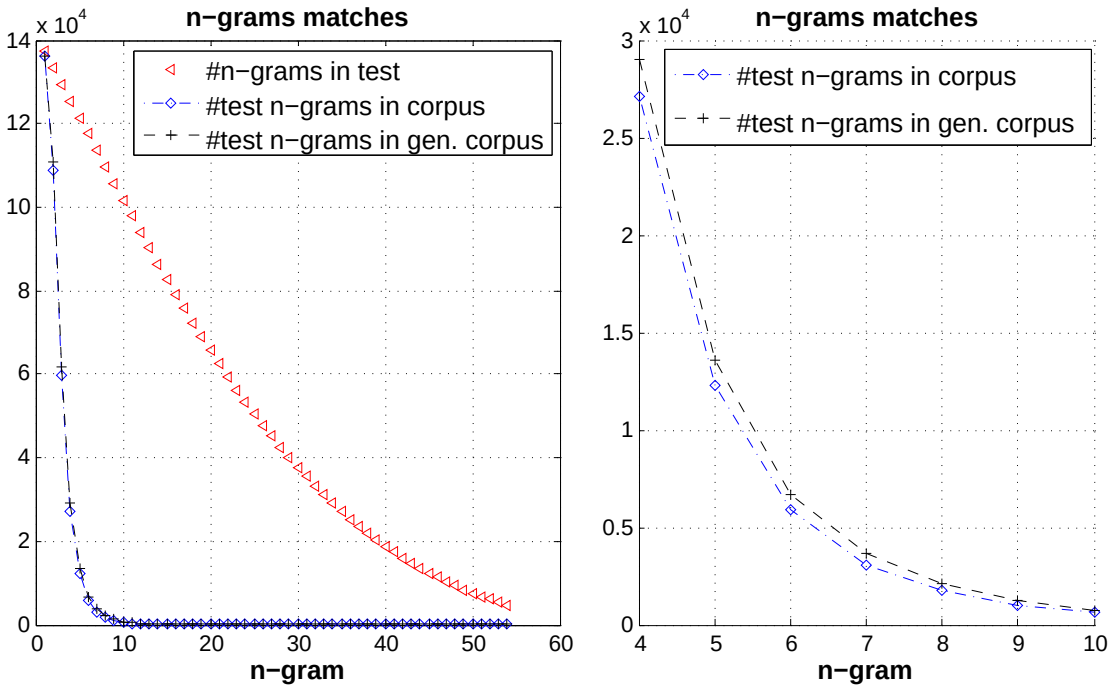

Figure 7.3: Left hand side plot: Number of n-grams (i) in the test set (ii) matches between the test set and source side of 30k Eng-Chi (iii) matches between the generalized test set and generalized source side of 30k Eng-Chi. The right-hand side figure shows a closer look of the same plot.

and also generated by the generalized phrase-pair (after replacing the values of the class labels), then the case could be made that generalization is not helping. The plot in Figure 7.5 shows the number of lexical phrase-pairs and new phrase-pairs generated due to generalization with respect to the length of the target $n-$grams. A closer look at the same plot is given in Figure 7.6.

Figure 7.7 shows that a large number of new target phrases also appear in the reference translations. As mentioned in Chapter 6, this can be treated as a lower bound on the number of grammatical phrase-pairs (or translation candidates) that the G-EBMT system is able to extract from the TM. The plot in 7.8 clearly indicates the relative increase in the number of useful (present in the reference translations) target phrasal matches. With semantically related segment-pairs for creating templates, as opposed to just syntactically related segment-pairs (Figure 6.14) and word-pairs (Figure 5.13), there is a substantial

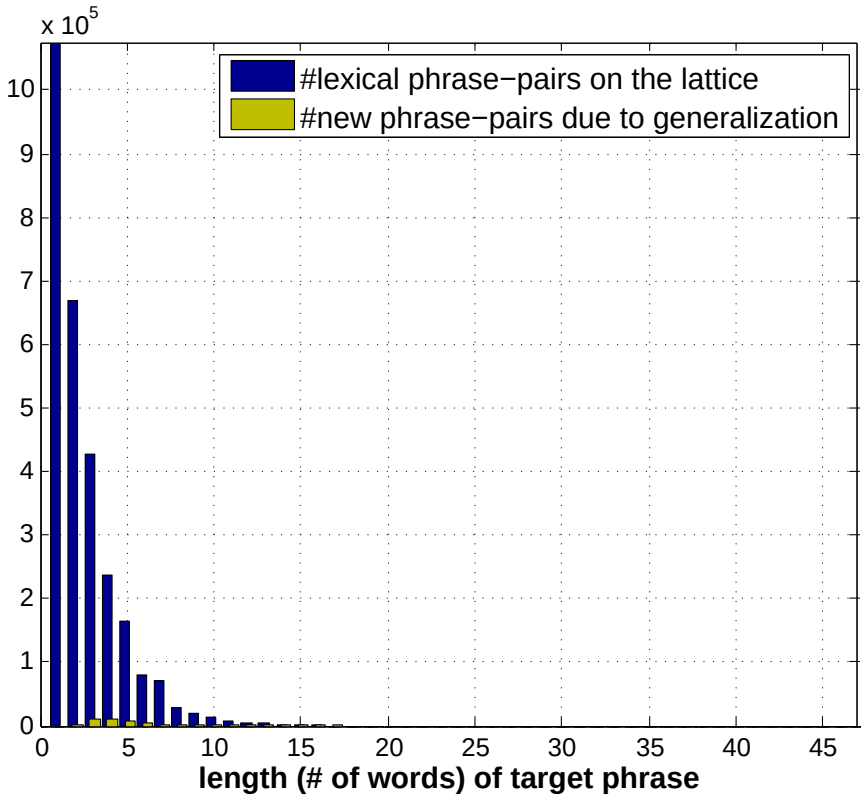

Figure 7.4: Number of phrase-pairs with increasing values of the length of the target halves (i) from lexical phrase-pairs only (ii) new phrase-pairs solely due to generalization. Max-Alternative=25.

increase in the number of longer target phrases.

To summarize, this chapter investigated another template-based approach that also clustered segment-pairs. The clustering was performed using contextual information and lexical word matches between segment-pairs. The algorithm was capable of automatically finding the number of clusters while clustering. The overall G-EBMT system gave statistically significant improvements over the baseline EBMT system in translation quality ($p < 0.0001$) on all language-pairs and data sets.

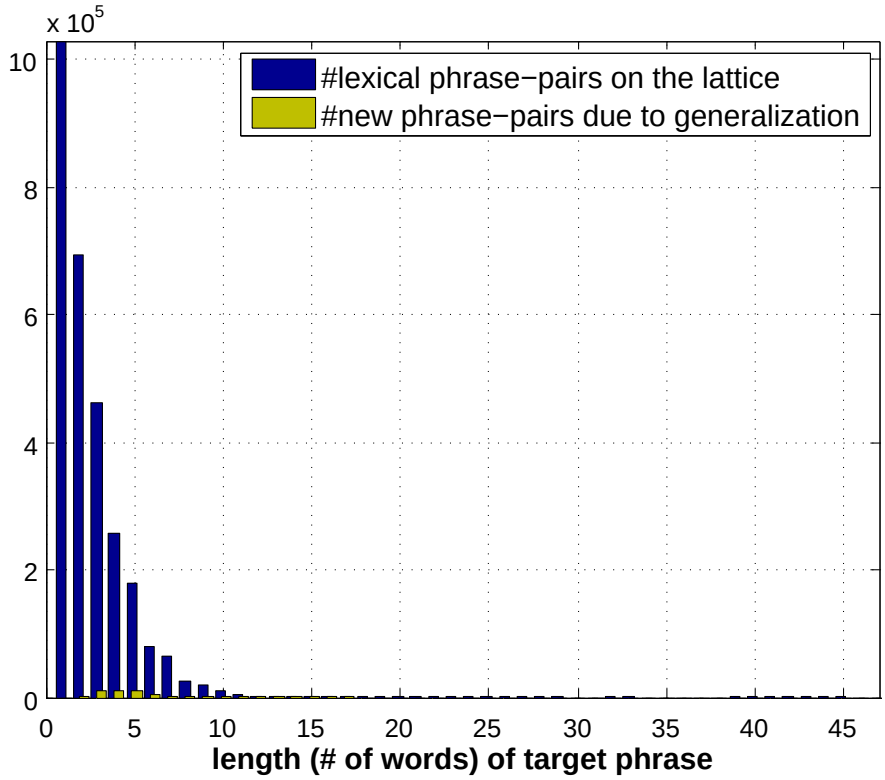

Figure 7.5: Number of phrase-pairs with increasing values of the length of the target halves (i) from lexical phrase-pairs (ii) new phrase-pairs due to generalization. Max-Alternative=200.

Sample Clusters

A few sample clusters extracted by our method are given below:
Cluster1
extremely happy ↔ 非常 高兴
happy ↔ 高兴
very glad ↔ 很 高兴
very glad ↔ 非常 高兴
very pleased ↔ 十分 高兴
very pleased ↔ 很 高兴
very pleased ↔ 非常 高兴

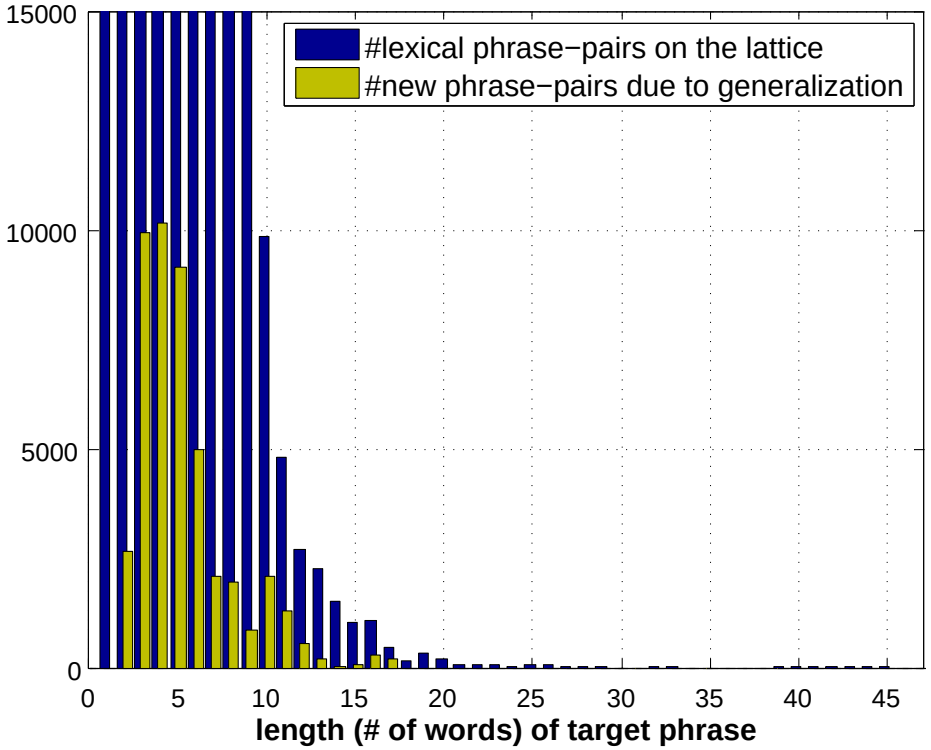

Figure 7.6: Closer look (same as Figure 7.5): Number of phrase-pairs with increasing values of the length of the target halves (i) from lexical phrase-pairs (ii) new phrase-pairs due to generalization. Max-Alternative=200.

Cluster2
balance policy ↔ 平衡 政策
ecological equilibrium ↔ 生态 平衡
strike a balance ↔ 平衡
ecological balance ↔ 平衡
ecological balance ↔ 生态 平衡
sunshine policy ↔ 阳光 政策

Cluster3
with yoshiro mori ↔ 与 森喜朗
spanish prime minister ↔ 首相

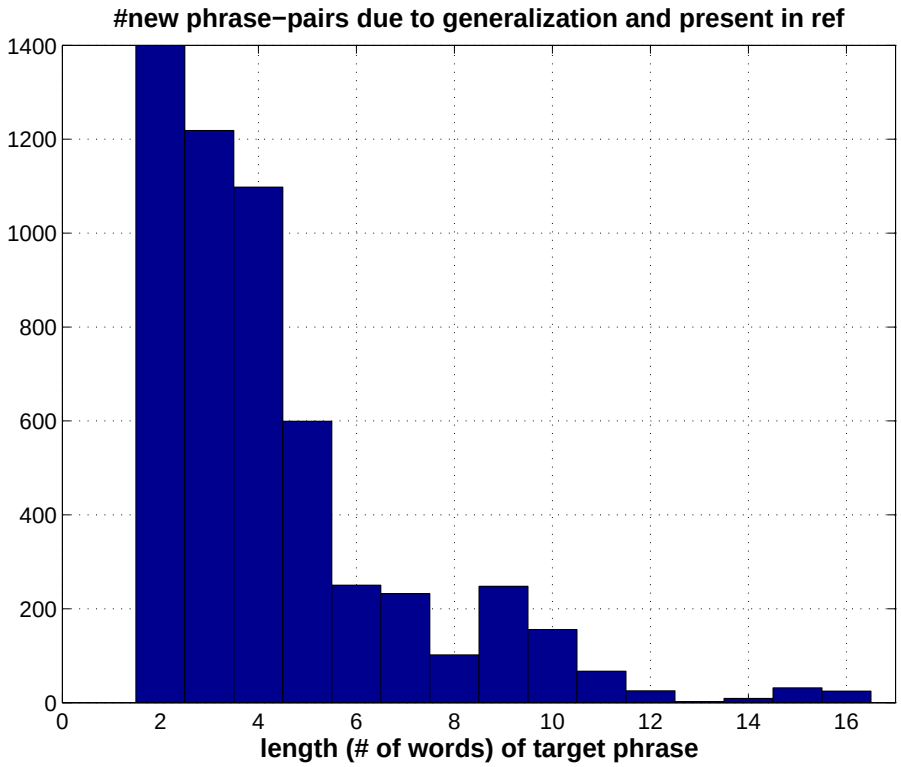

Figure 7.7: number of new partial translations solely due to generalization and present in the reference translations.

japanese prime minister keizo obuchi ↔ 首相 小渊惠三
prime minister tony blair ↔ 布莱尔 首相
prime minister tony blair ↔ 首相 布莱尔
tony blair ↔ 布莱尔
prime minister blair ↔ 布莱尔 首相
prime minister blair ↔ 首相 布莱尔
hun sen ↔ 洪森
prime minister yoshiro mori ↔ 森喜朗 首相
prime minister yoshiro mori ↔ 首相 森喜朗
the prime minister ↔ 首相
prime minister junichiro koizumi ↔ 小泉 首相
prime minister junichiro koizumi ↔ 首相 小泉纯一郎

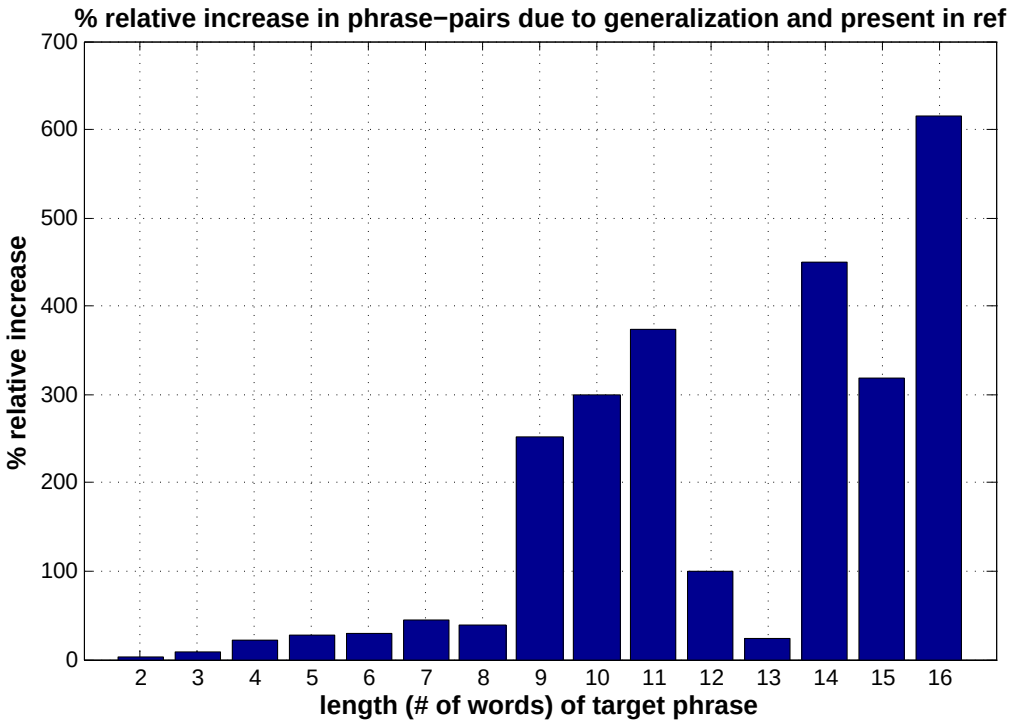

Figure 7.8: % Relative improvement in additional new (not found in the lexical phrase-pairs) partial translations solely due to generalization and present in the reference translations. Max-Alternatives=200.

prime minister koizumi ↔ 小泉 首相
prime minister koizumi ↔ 首相 小泉

Sample translations

A few sample translations produced by the baseline with no templates and the generalized system are given below. The highlighted text (in green) in the output translation of the G-EBMT system, indicates that the target phrase was generated by a source phrase that contained generalized tokens. The actual translation of its corresponding source phrase in the test sentence is also highlighted in the reference translation.

Test sentence:a cpc delegation led by zeng qinghong arrived in japan on 4 april for a good - will visit at the invitation of the **japanese government** and liberal democratic party .
Reference: 曾庆红 是 应 日本 政府 和 自民党 的 邀请 率 中共 代表团 于 四月 四日 抵 日 进行 友好 访问 的 。
Baseline: 共产党 领导 曾庆红 代表团 抵达 4 日 ， 日本 为 一 个 将 访问 ， 政府 的 邀请 ， 日本 自民党 干事长 。
G-EBMT: 一 曾庆红 是 应 韩国 政府 的 邀请 率 中共 代表团 于 4 月 日本 一 个 将 访问 的 邀请 ， 日本 政府 自民党 。

Test sentence:**beijing , 5 june (xinhua)** the npc (national people 's congress) environmental and resources protection committee today held in beijing a forum on the western regions ' development and ecological building to mark the " 5 june world environment day . "
Reference: 新华社 北京 6 月 5 日 电 （ 记者 沈路涛 ） 全 国 人大 环境 与 资源 保护 委员会 今天 在 京 举行 " 6 . 5 世界 环境日 " 西部 开发 生态 建设 座谈会 。
Baseline: 北京 7 月 5 日 电 （ 记者 6 月 ， 全 国 人大 的 《 日 资源 环境 今天 在 京 召开 座谈会 ， 尽快 改变 西部 地区 发展 ， 创造 和 生态 建设 ， 世界 5 月 环境 。
G-EBMT: 新华社 北京 6 月 5 日 电 ） 全 国 人大 环境 ： 资源 今天 在 京 召开 座谈会 ， 西部 地区 发展 和 生态 建设 ， 纪念 5 月 ， " 世界 环境 。

Test sentence:the model for implementing the basic policy **on the morning of 22 september** , the cpc central committee held a lecture on legal system focusing on western development and on providing legal protection to accelerate development in central and western regions .
Reference: 实行 基本 方略 的 榜样 今年 9 月 2 2 日 上午 ， 中共 中央 举办 法制 讲座 ， 内容 是 西部 开发 与 加快 中西部 发展 的 法治 保障 。
Baseline: 实施 典型 ， 基本 政策 ， 2 2 日 上午 9 月 ， 中共 中央 召开 的 一 次 讲座 上 法制 抓住 西部 大 开发 和 提供 法律 保护 ， 加快 中西部 地区 发展 。
G-EBMT: 实施 ， 种 基本 政策 ， 9 月 2 2 日 上午 ， 中共 中央 讲座 法制 重点 、 西部 大 开发 提供 ， 保护 ， 加快 中西部 地区 。

Chapter 8

Templates for Language Model

Data sparsity has remained a great challenge even in statistical language modeling and templates can be used here as well to provide better probability estimates. Translating into a minority language that does not have enough monolingual data results in sparse language models. Class-based language models were introduced to handle such challenges. These class-based language models make reasonable predictions for unseen histories by using the class information of all the words present in the histories. Hence, class-based models require all words present in the training data to be clustered.

When hand-made clusters are not available, automatic clustering tools can be used to obtain clusters. We generate a variant of the class-based LM, where only a small set of reliable words are clustered to create templates and call it the template-based language model as the word sequences of n-grams are converted into short reusable sequences containing either the word or its class label (but not both).

It should be noted that the template-based model is equivalent to a class-based model formed by placing each of the words that were not clustered in a unique class, leading to singleton clusters for unclustered words. Hence, the template-based language model can be considered as a specific case of the class-based language model. The aim of this thesis is to identify the unreliable words and not consider them for clustering. In this thesis, we only analyze language models created using equivalence classes of words only. This can be extended to handle phrases.

8.1 Motivation: Template-based models or Class-based models

We first motivate the use of template-based language models. Suppose the target language corpus contains the sentences, *S1* and *S2*.

S1: the school reopens on Monday
S2: the office is too far

<ORG> and *<WEEKDAY>* are example clusters available.

Example Clusters
<ORG>: school, company, office
<WEEKDAY>: Monday, Tuesday, Wednesday,...

Clustered words are first replaced by their labels in the target language corpus to obtain templates. In templates, *T1* and *T2*, *"school"* and *"office"* are replaced by *<ORG>* and *"Monday"* by *<WEEKDAY>*:

Templates
T1: the *<ORG>* reopens on *<WEEKDAY>*
T2: the *<ORG>* is too far

Reusable templates of the above form are used to build the target language model. The process involved in building these models is similar to that of building word-based language models except that now the conditional probability is based not just on words in the history but on class labels as well.

With a word-based language model (for simplicity, trained on the corpus containing only *S1* and *S2*), if a subsequence such as *"the office reopens"* was encountered during decoding, the model would return less reliable scores for *p(reopens|the office)* by backing off to the uni-gram score, *p(reopens)*. However, the template-based model makes use of the available data well by converting the subsequence, *"the office reopens"* to *"the <ORG> reopens"* and hence, a more reliable score i.e., *p(reopens|the <ORG>)* contributes to the score from the language model for this sequence.

8.2 Template-based language model Formulation

An n-gram template-based language model can be given by,

$$p(w_i|h) \approx xp(f_i|f_{i-1}, ..., f_{i-n+1}) \tag{8.1}$$

$$where f_j = \begin{cases} c(w_j), & \text{if } w_j^{th} \text{ class is present} \\ w_j, & \text{otherwise} \end{cases} \tag{8.2}$$

$$x = \begin{cases} p(w_i|c(w_i)), & \text{if } w_i^{th} \text{ class is present} \\ 1, & \text{otherwise} \end{cases} \tag{8.3}$$

The probability of the i^{th} word (w_i) given its history h is represented as the probability of feature f_i corresponding to w_i given its previous history of features. Each feature can represent a word, w_j or its class, $c(w_j)$ if w_j is clustered.

8.3 Incorporating Template-Based language models

The template language model builder takes in training data and a class file consisting of words with their corresponding equivalence classes. The model is built by replacing the words that occur in the class file by their class names. It should be noted that this model allows us to use only the reliable words to be replaced by their class names. The words and their class names are stored for future look ups for generalizing target fragments during decoding.

To incorporate the template based language model scores, words on the lattice are replaced by their equivalence classes and their n-gram probabilities are determined using the template-based language model. These scores are then interpolated with the probabilities obtained with the word-based model. An optimization technique like hill-climbing can be used to find the best λ on a tuning set. The best values for λ based on our experiments varied between 0.4 to 0.6.

$$\begin{aligned} p(w_i|h) &= \lambda[xp(f_i|f_{i-1}, ..., f_{i-n+1})] + \\ &\quad (1-\lambda)p(w_i|w_{i-1}, ..., w_{i-n+1}) \end{aligned}$$

Lang-Pair	data	Manual	SangAlgo	Mod Algo
Eng-Chi(LM)	30k	0.1290	0.1257	0.1300

Table 8.1: BLEU scores with templates created using manually selected N, SangAlgo [Sanguinetti et al., 2005] and the modified algorithm to automatically find N.

8.4 Results

The experiments in this section use the Eng-Chi, Eng-Fre and Eng-Hai training and test data sets described in Section 3.4. We used the same Spectral Clustering algorithm that was applied to generalize words in Chapter 5 to also cluster the words in the target language. Words clustered from the mid-frequency and high-frequency regions gave better performance (details on different regions are in Chapter 5). Since low-frequency words appear rarely in the training data, they do not appear in many contexts and hence their term vectors are not well-defined causing low quality clusters. This in turn effects the quality of the translations if these words are extracted (which would happen rarely if the test set belongs to the same domain as the training data) by the Translation model as translations of the source phrases in the test set.

8.4.1 Number of clusters (N) and removal of Incoherent members

The algorithm given in Section 5.5.2 removed unclassifiable points from the rows of the U matrix (containing eigenvectors with greatest eigenvalues stacked in columns) while determining the optimum N. This section (Table 8.1) compares the average BLEU scores obtained with language models created using manually selected N, SangAlgo [Sanguinetti et al., 2005] and our modified algorithm (Section 5.5.2) to automatically find N. The average scores obtained with Sanguinetti et al. [2005] are much lower than the scores obtained by the empirically found N (about 0.4 BLEU points on average) . As seen from the results, the modified algorithm is able to obtain scores close to the scores obtained by the empirically found N even in the language model.

Table 8.2 shows the importance of finding the right N. If a random N (or if not chosen carefully) was chosen while clustering the 30k Chinese data, the scores (BLEU score on average) could have gone as low as 0.1230 (a difference of 0.6 BLEU points from the best scores on average).

File	Man. worst	Man. best	Auto
tune	0.1280	0.1323	0.1328
test	0.1230	0.1290	0.1300

Table 8.2: Average BLEU scores on test and tune files with templates created using manually and automatically found N on 30k Eng-Chi.

	POS	Auto Clus
LM	0.1288	0.1300

Table 8.3: Average BLEU scores with templates created using POS and Automatic clusters on 30k Eng-Chi.

8.4.2 POS vs. Automatically found clusters

As mentioned in Section 5.6.1, POS tags are good candidates for equivalence classes and can be obtained with semi-supervised learning [Tseng et al., 2005] techniques with training data. However, for languages with limited data resources (like Haitian), obtaining POS tags may not be possible. To see if the automatically found clusters perform as well as POS, we created language models using POS tags and compared their performance with language models created using automatically found clusters on 30k Eng-Chi. The POS tags were obtained using Tseng et al. [2005]. For the comparison to be fair, we grouped only those words that were also used in the automatic clustering process. Target words with multiple POS tags were not considered. The BLEU scores on the test files were almost the same with both the models (average BLEU scores over the test files in Table 8.3). It can be concluded that automatically found clusters are good candidates for creating language models as well in sparse data conditions.

8.4.3 More Results: template-based language models with Eng-Chi, Eng-Fre and Eng-Hai

Table 8.4 shows the average BLEU scores obtained by using template-based language models and compares the scores obtained on a system that used a conventional word-based n-gram model.

Template-based language models continue to give better probability estimates and hence better translation quality even with larger training data sets and do not show the

Lang-Pair		lexical word-based LM	template-based LM
Eng-Chi	15k	0.1076	0.1098
Eng-Chi	30k	0.1245	0.1300
Eng-Chi	200k	0.1785	0.1822
Eng-Fre	30k	0.1577	0.1613
Eng-Fre	100k	0.1723	0.1764
Eng-Hai		0.2182	0.2370

Table 8.4: BLEU scores with templates applied in the language model (LM) for various data sets. Statistically significant improvements over the Baseline with $p < 0.0001$.

quick saturation as seen with templates in the translation model.

8.4.4 Perplexities

Often it is expensive to compute the error rates on the final outputs generated by natural language processors to evaluate language models. For example, in Speech Recognition, it is computationally expensive to find word error rates on the recognized output. Hence, computationally cheaper methods are used for evaluating language models, one of them being Perplexity.

Perplexity, an information theoretic approach based on entropy is used to evaluate language models [Bahl et al., 1990]. Lower the perplexity, the lower the number of bits (entropy) and less surprised we are about a test set [Clarkson and Rosenfeld, 1997]. However, there are issues with using perplexity to evaluate natural language systems ([Martin et al., 1997];[Iyer et al., 1997];[Chen et al., 1998]), where, lower perplexity has not shown improvement in performance. Hence, in this thesis we only report translation quality scores on the output of the EBMT system.

8.4.5 Analysis

Coverage

Here we try to get an idea of the number of target phrases for which the language model would back off to lower order n-gram probabilities assuming the EBMT system was able find translations exactly as the translations in the reference for every test sentence.

We generalized the target half of the 30k Eng-Chi training corpus and the reference file of the test data. We then found how many $n-$grams of the reference were present in the training corpus. Of course, in general there would be many partial candidate translations for different spans of the test sentence generated from the translation model, but for this analysis, we could assume the case that non-overlapping candidate translations were found and only one translation candidate was found for every fixed span of text in the test sentence. Figure 8.1 shows the number of matching $n-$grams in the reference set-with and without generalization. If we consider the $5-$grams in the plot as an example, there are about 12,000 ($1.78*10^4$ - $0.6*10^4$) $5-$grams for which the system backs off to $4-$gram probabilities. This gives us an intuitive reasoning for why the template-based language models help improve the translation quality by providing better probability estimates for higher order $n-$grams. It should also be noted that even with templates only a relatively small proportion of $5-$grams actually occur in the training data ($1.78*10^4$ out of $9.7*10^4 = 18.35\%$).

Interpolation Weights

Template-based language models in Table 8.4 showed statistically significant improvements in translation quality over standard word-based language models. Another factor that indicates the importance of the template-based language models are the interpolation weights for the template-based language model and word-based language model in Eqn. 8.4. Figure 8.2 shows the variation in translation scores of the interpolated model with various interpolation weights with the 30k Eng-Chi data set. With a positive weight of 0.6 for the template-based model, the interpolated model performs the best. However, when the template-based language model is used alone (with a weight of 1.0), the system performs poorly (even below the word-based language model). This is because the conditional probability ($p(f_i|f_{i-1}, ..., f_{i-n+1})$) in equation 8.1 receives high scores for overgeneralized histories ($f_{i-1}, ..., f_{i-n+1}$, especially when the overgeneralized sequence: $f_i f_{i-1}, ..., f_{i-n+1}$, appears many times in the generalized target data that is used to build the template-based language model) and lower scores for more specific histories causing degradation in translation quality.

To summarize, this chapter applied the unsupervised clustering algorithm that was used in Chapter 5 to create clusters to build template-based language models. Removing incoherent points from clusters helped in determining the number of clusters for better performance in terms of translation quality. Statistically significant improvements ($p <$0.0001) were found on all the data sets with templates over the baseline system with just the lexical word-based model.

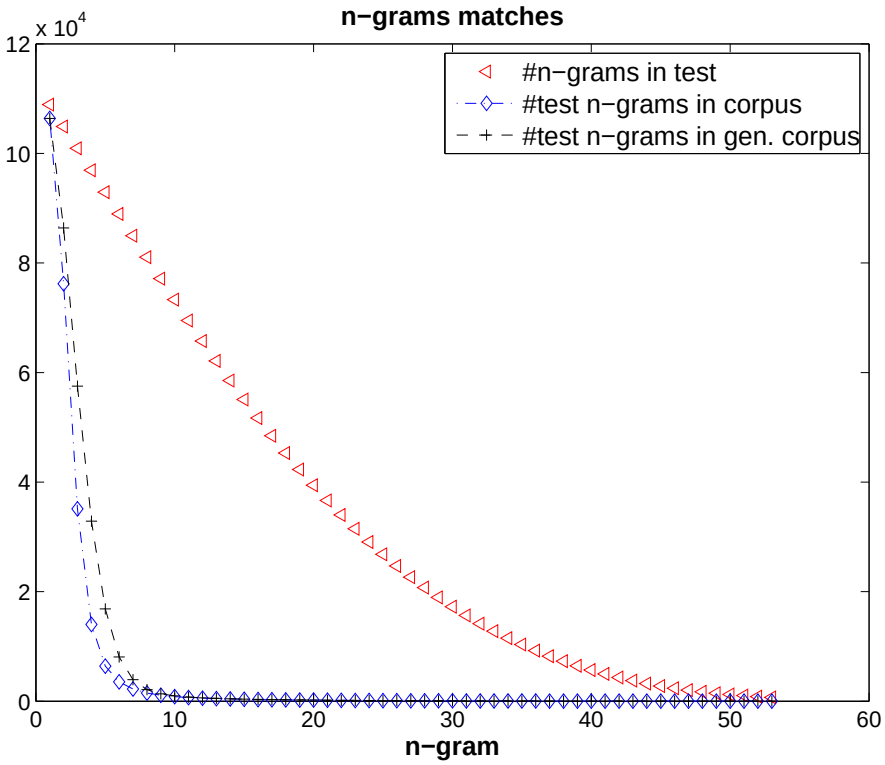

Figure 8.1: Number of n-grams (i) in the reference set (ii) matches between the reference set and target side of 30k Eng-Chi (iii) matches between the generalized reference set and generalized target side of 30k Eng-Chi.

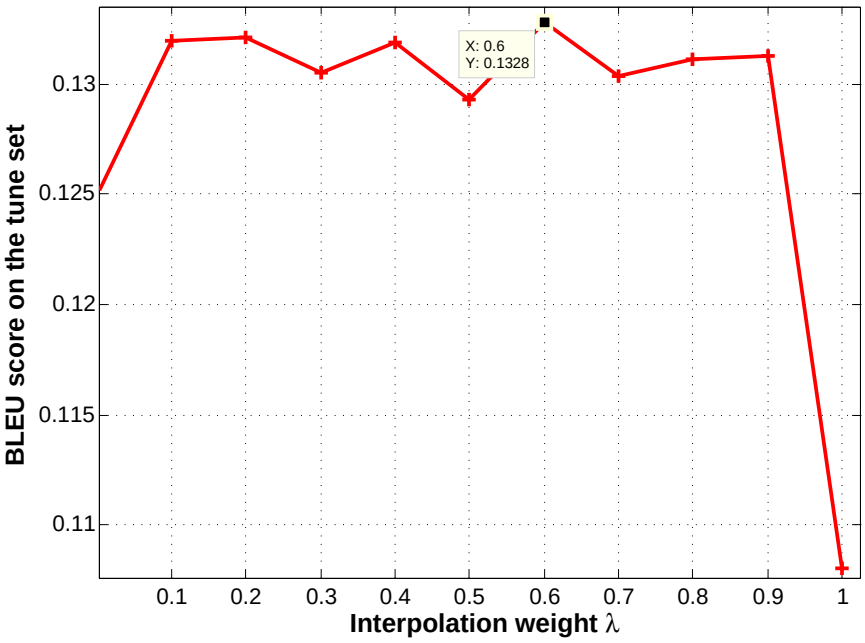

Figure 8.2: Variation in translation scores on the tune set with various interpolation weights (λ) with the 30k Eng-Chi data set.

Chapter 9

Putting It All Together (A Hybrid Model) and Looking into the Future

This thesis showed different ways of obtaining longer and more useful target phrases from the translation model. We also developed a way to obtain reliable language models to stitch the target phrasal candidates together resulting in better translations (or outputs). This was achieved by using templates- both in the translation and the language models. Handling out-of-vocabulary words and rare words was also treated as a generalization task that involves clustering possible candidate replacements for an out-of-vocabulary or rare word in one cluster. We also looked at different ways to generate templates by using different clustering techniques. Chapters 4 through 8 used contextual, syntactic and semantic information to guide the clustering algorithms.

The results obtained with all our techniques for using templates in the translation model, are summarized in the form of a plot (Figure 9.1). The plots clearly indicate the usefulness of word- as well as phrase-generalized templates over the Baseline (pure lexical EBMT), of which segment-pairs created by semantically clustering syntactically-coherent segment-pairs (from Chapter 7) performs the best. The best performance of the templates is seen with moderate amounts of data (for example, 30k in Eng-Chi). As expected, the improvement with templates over the baseline starts to diminish and saturate to the score obtained by the baseline as the training data is increased. Only three training data sets were used for Eng-Chi and only two data sets for Eng-Fre in this thesis. As future work, it would be interesting to see the performance with more training data sets between 30k and 200k for Eng-Chi and Eng-Fre.

Before we conclude this thesis, we would like to take our approach a step further by combining all the techniques in to a hybrid model in order to give a lower bound on the

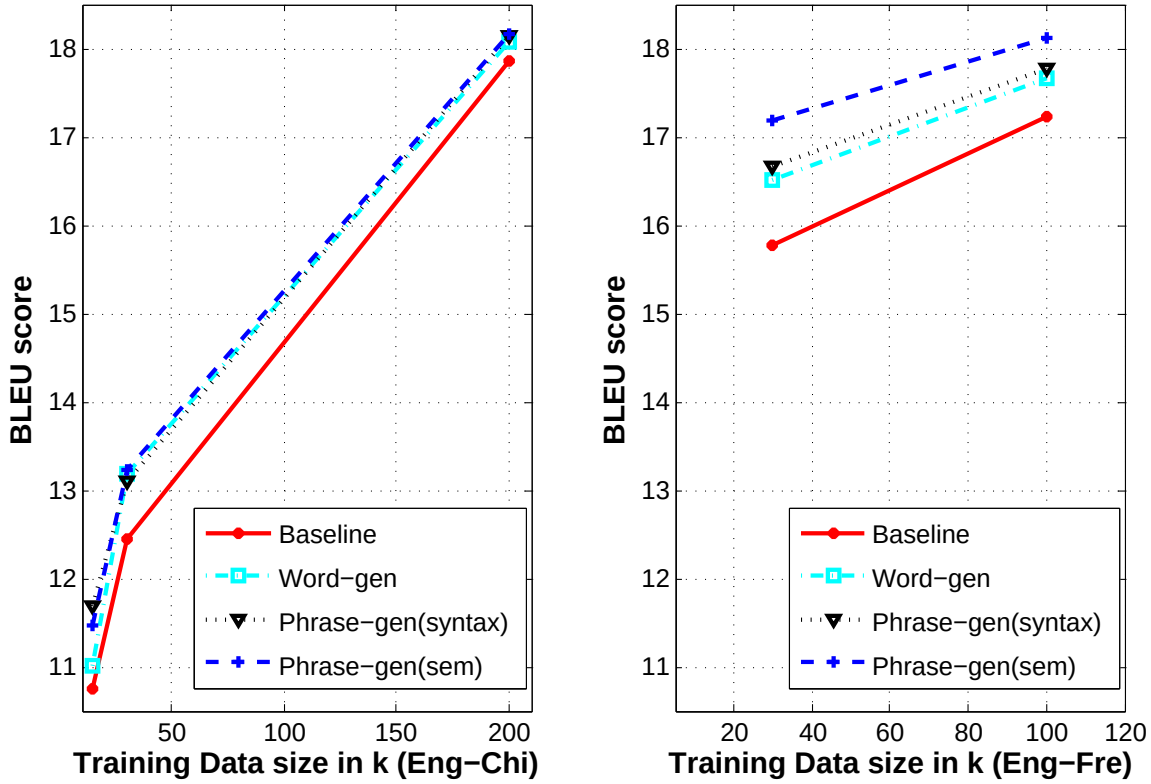

Figure 9.1: Results from Chapters 5 (Word-gen: word-generalized templates in TM), 6 (Syntax: syntactically clustered segment-pairs) and 7 (Sem: semantically clustered segment-pairs).

amount of improvement that can be achieved. This is only a lower bound since even better results may be obtained by joint tuning of parameters and better selection of segment-pairs.

This chapter also suggests possible future work and finally concludes this thesis.

9.1 Further Analysis

In this section, we analyze the effect of a combined model built with all the techniques suggested in this thesis to improve translation quality in data sparse conditions. This section also reports scores with two other commonly used evaluation metrics- NIST and TER.

9.1.1 Usage of templates in both the translation model and the language model

In order to show the effect of combining templates in the TM with template-based LMs, we performed the following experiment. We combined the templates obtained from clusters created by grouping semantically related phrase-pairs and the template-based language model. The combined model was then tested on the test set (details in Section 3.4) with the 30k Eng-Chi training data set. The results obtained on all the 10 test files are shown in Table 9.1. The overall score of the combined model was slightly better than the overall scores of both the individual models. The scores on the combined model were better or almost the same on 6 test files and worse on 4 files when compared to the scores with templates in the TM. However, the scores with the combined model was better than the scores obtained with just the template-based LM on all the test files.

9.1.2 Effect of Larger Language models

All the experiments in this thesis used language models built only from the training half of the data based on the assumption that the target language could also be sparse. There are cases where obtaining monolingual data is easier (like English) than obtaining bilingual data, such as while translating from Korean to English. Hence, we were interested in seeing the performance of templates in the translation model when the decoders used larger language models. For this, we built a lexical word-based language model with 200k target sentences and used 30k Eng-Chi training data in the translation model. As done in Chapters 5, 6 and 7, $word$-generalized and $phrase$-generalized templates were built from the 30k Eng-chi training data. Hence, the experiment remains the same as the experiments performed with 30k Eng-Chi in Table 5.9, 6.1 and 7.1, except that now we used a larger language model. Chinese was chosen as the target language as it was easier to compare these results with the results that we already had from the previous chapters.

Table 9.2 shows the effects of a larger language model on the baseline EBMT system and on the Generalized-EBMT systems with 30k Eng-Chi bilingual training data. With

Baseline	Phrase-gen Templates in translation model (TM)	Template-based language model (LM)	Templates in TM+LM
0.1349	0.1376	0.1375	0.1379
0.1428	0.1511	0.1490	0.1567
0.1239	0.1331	0.1328	0.1366
0.1341	0.1422	0.1374	0.1429
0.1232	0.1299	0.1262	0.1292
0.1237	0.1361	0.1339	0.1361
0.1102	0.1183	0.1175	0.1179
0.1161	0.1257	0.1207	0.1274
0.1191	0.1287	0.1276	0.1284
0.1144	0.1199	0.1174	0.1173

Table 9.1: Combined model. Column1: Baseline, Column2: Phrase-generalized templates in the translation model, Column3: Template-based language model, Column4: Phrase-generalized templates in the translation model and Template-based language model on 30k Eng-Chi.

a larger language model, the Baseline and the word-generalized EBMT systems perform better by about 0.5 BLEU points on the test set of 4000 sentences. Whereas, a much higher gain of 1.6 BLEU points is seen with the phrase-generalized systems. An intuitive reasoning for this is, a stronger language model helps weed out target candidate phrases with poor generalizations and at the same time provide higher scores to target phrases that have good generalizations. If our reasoning that, "a stronger language model helps more when the translation model is weak" is correct, we would expect to see lower benefits with a stronger translation model. This is indeed the case, the larger language model when used with a stronger translation model (of 200k bilingual training data), the improvements seen over the baseline (0.1817-0.1785) are not high as the improvements seen with a weaker (0.1442-0.1306) translation model (30k bilingual training data).

9.1.3 Other Scores: NIST, TER scores

The chapters on generalized templates only showed BLEU translation scores. As there are objections to using BLEU as an evaluation metric in the MT community and to also check the general applicability of our results, we used two more evaluation metrics- NIST and TER. Table 9.3 shows the translation scores obtained with the two evaluation metrics.

Lang-Pair		Baseline	Word-generalized TM (Chapter 5)	Phrase-generalized TM (Syntax-:Chapter 6)	Phrase-generalized TM (Semantics-based :Chapter 7)
Eng-Chi (Big LM)	30k	0.1306	0.1347	0.1421	0.1442
Eng-Chi (Small LM)	30k	0.1245	0.1319	0.1310	0.1323
Eng-Chi (Big LM)	200k	0.1785	0.1807	0.1815	0.1817

Table 9.2: Scores with templates in the TM and a larger LM for 30k Eng-Chi data set. Statistically significant improvements over the Baseline with $p < 0.0001$.

Language-Pair	Training Data Size	System	NIST	TER
Eng-Chi	30k	Baseline	4.8041	0.7813
		G-EBMT(syntax phrase gen.)	5.0293	0.7496
		G-EBMT(semantics phrase gen.)	5.0052	0.7429
Eng-Chi	200k	Baseline	5.5196	0.7192
		G-EBMT(syntax phrase gen.)	5.5745	0.6970
		G-EBMT(semantics phrase gen.)	5.5433	0.7018
Eng-Fre	30k	Baseline	4.0241	0.8240
		G-EBMT(syntax phrase gen.)	4.2767	0.7847
		G-EBMT(semantics phrase gen.)	4.2812	0.7734
Eng-Fre	100k	Baseline	4.2805	0.7787
		G-EBMT(syntax phrase gen.)	4.4913	0.7573
		G-EBMT(semantics phrase gen.)	4.5123	0.7416

Table 9.3: Quality scores for the Baseline EBMT and G-EBMT with phrase-generalized templates using the NIST and TER evaluation metrics. Statistically significant improvements over the Baselines($p < 0.0001$) as observed with the BLEU score.

9.1.4 Hybrid Model

We wanted to further check whether a combined model with OOV as well as rare word handling would perform better or worse than any other model. For this, we analyzed a Hybrid model that included the following three strategies to improve the translation scores on the 30k Eng-Chi data set:
(a) OOV and rare-word handling
(b) Phrase-generalized templates in the translation model
(c) Template-based language model

We used clusters generated by grouping semantically related phrase-pairs to create the phrase-generalized templates as it performed better than grouping the phrase-pairs based on their syntactic structure. We used the same test set (details in Section 3.4) to perform this experiment. OOV and rare word replacements were found and scored as done in Section 4.3. Out of the 4000 test sentences, 619 sentences contained at least one or more OOVs and 778 sentences contained one or more rare words. There were 2397 OOV or rare words in the entire test set and we were able to find replacements for only 353 words (14.73%).

The results obtained with the Baseline system (no OOV/rare word handling) and by handling OOV/rare words on all the ten test files are shown in Table 9.4. The configuration (parameters of the EBMT system) of the Baseline system was used to obtain the results for OOV and rare word handling (first column in Table 9.4). Scores were also obtained with the template-based language model and translation model. The parameters of the template-based language model were tuned on a tune set that did not handle any OOV/rare words and was tested on the ten test files with (sixth column in Table 9.4) and without OOV/rare word handling (fourth column in Table 9.4). Similarly, the parameters of the translation model with templates were tuned on the tune set that did not handle any OOV/rare words and was tested on the ten test files with (fifth column in Table 9.4) and without OOV/rare word handling (third column in Table 9.4).

Although handling OOV and rare words improves each of the systems overall, no improvement is seen on 3 test files. This is because there were no alternatives found for the OOV or rare words in the 3 files. Hence, it is difficult to see benefits from handling OOV or rare words in these experiments and we would expect to see more improvements when more OOV or rare words are handled (as was seen in Chapter 4) and with other languages like Chinese, Arabic or Indian languages on the source half.

A combination of all the chapters in this thesis (Column 6 in Table 9.4) gives a larger overall gain over the baseline EBMT system with no templates or OOV/rare word han-

Baseline	b Templates in TM	a OOV/Rare handling	c Templates-based LM	a + b	a + c	a + b + c
0.1349	0.1376	0.1375	0.1375	0.1379	0.1377	0.1379
0.1428	0.1511	0.1428	0.1490	0.1511	0.1490	0.1567
0.1239	0.1331	0.1244	0.1334	0.1365	0.1347	0.1372
0.1341	0.1422	0.1338	0.1374	0.1430	0.1374	0.1429
0.1232	0.1299	0.1248	0.1272	0.1322	0.1292	0.1319
0.1237	0.1361	0.1237	0.1339	0.1361	0.1339	0.1361
0.1102	0.1183	0.1125	0.1167	0.1181	0.1167	0.1183
0.1161	0.1257	0.1170	0.1217	0.1268	0.1223	0.1282
0.1191	0.1287	0.1191	0.1248	0.1287	0.1248	0.1284
0.1144	0.1199	0.1148	0.1184	0.1199	0.1186	0.1186

Table 9.4: Hybrid model: Comparison of translation scores of the Baseline system and the system handling OOV and rare words, templates in the translation model and language model on the ten test files. (a): OOV and rare-word handling. (b): Phrase-generalized templates in the translation model. (c): Template-based language model. (a+b): OOV and rare-word handling with templates in the translation model. (a+c): OOV and rare-word handling with the template-based language model. (a+b+c): OOV and rare-word handling with templates in the translation model and template-based language model.

dling. The scores on 3 test files on the combined hybrid model (Column 6) were almost the same or slightly below the scores of other models (Columns 2 to 6). Perhaps a better way of tuning the hybrid model would have helped us see more improvements. However, these results are encouraging in that the overall system can be improved further.

9.2 Future Work

This section suggests a few more improvements that can be performed to improve the system further.

9.2.1 Improvements to Chapter 4: OOV and rare-word handling

Larger Monolingual Corpus for handling OOV and rare words This thesis used a fairly large monolingual corpus to extract replacement candidates for OOV and rare words. Improvement in quality and increase in number of replacements can boost the performance of the system. This can be achieved by increasing this monolingual corpus further.

Design choices There were a few design choices made in this thesis: for a replacement to be chosen as a possible candidate for replacing OOV or rare words, we placed a constraint that the replacement should contain *at least one content word*. Also the context windows of the OOV/rare words were extended until the window contained *at least one content word*. There was also a *length restriction on the candidate replacements* to reduce the number of replacements extracted for a particular OOV/rare word. Only those words in the test data that appeared *less than three times in the training corpus were considered as rare words*. An analysis can be done to see if varying these choices alters the translation quality.

Features for scoring replacements The features that were used for scoring the replacements were only contextual and voting features. We could certainly have more features like: proportion of content words in the candidate replacements, part of speech tags of the context words and the replacement itself, etc.

Replacements for OOV/rare phrases In this thesis, replacements were found only for OOV/rare $words$ and no experiments were performed with OOV/rare phrases, especially, multi-word named entities (proper nouns, organizations, etc). The procedure adopted in this thesis to find replacements, can still be used to find replacements for OOV and rare phrases, but, the search procedure that is used to find the OOV and rare words in the test set needs to be extended to find phrases of length greater than 1.

Other language-pairs For OOV and rare-words handling, we used English as our source language- a language that has much fewer inflections and complexities compared to other languages (like Arabic, all Indian Languages, etc.). Our method of finding replacements

may show more benefit in such languages as the number of OOV and rare words will be much larger than what we saw for English on small training data sets.

9.2.2 Improvements to Chapters 5, 6 and 7: Templates in the translation model

Other newer Automatic Clustering algorithms We explored just a few standard well-known and powerful automatic clustering algorithms, it would be interesting to explore other unsupervised clustering algorithms to see if they are beneficial for natural language processing tasks, especially machine translation.

Filtering/Classification task The filtration step involved in discarding unreliable segment-pairs in Chapter 6 gave an accuracy of 83%. More distinguishable features can be added to perform a better classification task.

Chapter 6 in this thesis also used a simple leniency threshold computation. As future work, other ways of defining leniency can be adopted.

Segment-pairs for training the classifier in the filtration step of Chapter 6 were chosen $randomly$ from the list of all possible segment-pairs. Other selection strategies can be used to select these segment-pairs. This may increase the accuracy of the classifier which will in turn provide better segment-pairs for creating even better templates.

Non-contiguous phrase-pairs This thesis only looked at extracting contiguous phrases for clustering and template-induction. This can be extended to extracting non-contiguous phrase-pairs with gaps to enable the translation model fill in other phrase-pairs in order to generate even longer translation candidates. Of course, this requires changing the phrase-extraction procedure adopted by the translation model.

9.2.3 Improvements to Chapter 8: Template-based Language modeling

Improvements to the language model The template-based language model used in this thesis only used clusters that contained word-pairs. This can be extended to handle phrase-pairs.

9.2.4 Improvements to the Hybrid model

Better Tuning and selection of features Better ways of tuning the features, selection of features and clustering can definitely improve the system further. We will now discuss the applicability of our techniques to other language-pairs.

9.2.5 Applicability of our approaches to new language-pairs

Time required to extend our techniques to other language-pairs The techniques presented in this thesis take just a few hours to generate the templates and a few days to tune the parameters of the EBMT system. Hence, these approaches can be quickly extended to new language-pairs.

***Ideal* language-pairs:(Properties of languages)** Our approach to handle OOV and rare words in Chapter 4 is well suited for languages that have limited parallel training data but it does require a large monolingual corpus for the source language in order to find candidate replacements. Hence, all language-pairs that have small amounts of bilingual data (for example, while translating from English to any Indian languages) but large amounts of source monolingual data (like, English, French, Chinese, etc.) are suitable candidates for our approach. Languages that have rich morphology can gain sufficiently in translation quality by applying our technique.

Templates in this thesis were also suggested for language-pairs that have limited amounts of training data. As more and more data become available, the effectiveness of templates starts to diminish (as with the 200k Eng-Chi training data set in Table 5.9, 6.1, 7.1). To support our reasoning, we could consider a case where we have infinite amounts of training data for a particular language-pair where any new test sentence to be translated appears in the training data. Templates in such conditions are not useful anymore as the EBMT system will be able to find entire translations (or very long phrases with a large parallel corpus) even without the use of templates.

Word-Generalized templates in the Translation model can be used on any language that does not have enough data (hence, limited bilingual data) and/or on all those languages that completely lack knowledge sources (such as parsers). These templates can even be applied when large amounts of data are available but the effectiveness might be minute. However, if the languages have rich morphology (on the source language and/or the target language), we expect (not tested in this thesis) to see improvements even when large amounts of data are available.

Our phrase-generalized templates make the assumption that monolingual chunkers are available. Today robust monolingual chunkers can be easily trained with small amounts of annotated data. Hence, languages that have small amounts of annotated data (indicating linguistic-phrasal boundaries) and small amounts of bilingual data (to train the EBMT system) can benefit significantly from our approach. As mentioned with word-generalized templates, if the languages are rich in morphology, we expect to see improvements even with large amounts of bilingual data.

Although templates in the translation model may not be very useful with large amounts of data, templates in the Language model continued to show gains in translation quality even when data was increased (Table 8.4). So template-based language models can be used even when large amounts of target language data are available.

Although not verified in this thesis, languages with highly complex word-compounding may not benefit much from our approaches as the case may be that many (or almost all) words appear very few times in the bilingual corpus because of which the word-alignments may not be reliable. Since our clustering techniques require word-alignments for its members (segment-$pairs$ and word-$pairs$), the resultant members of the equivalences classes may not have accurate or good enough correspondences. Hence, pre-processing of the data with a compound-word-splitter will be required to see the benefits of our approaches in such cases.

9.3 Conclusion

Computers are the most useful tools ever invented to further human knowledge. They have always been perfect for "number crunching" problems; however their full power has not been unleashed because they are not good at certain tasks that humans perform with ease, such as translation tasks. Computers do not have the ability to handle language complexities that humans handle with ease. There is a great potential to combine these two aspects: the generalizability of human thinking and the speed and accuracy of machine thinking [Hutchins, 2001].

Currently there is a lot of interest in data-driven machine translation approaches as they can be trained automatically on any language-pair without substantial human involvement. Although Machine Translation systems are much faster and cheaper than human translators, these systems are far from perfect. This work focused on improving Example-Based Machine Translation (EBMT) systems by handling the available data at hand more efficiently and at the same time improving the components of the system to handle such language-pairs. It is hoped that these techniques could be applied in other MT approaches

or perhaps even other natural language processing tasks.

The use of templates dates back to the 1980's when statistical decoders were not being used in EBMT and templates provided a way to group partial target phrases to generate the target translations. With the advent of statistical decoders in EBMT, templates were completely ignored with the belief that they were not useful anymore. This thesis gave a complete survey of the templates used in the past and extended its usage in present EBMT systems that use statistical decoders. This thesis successfully showed that templates are still useful in EBMT as they can be used to obtain longer phrasal matches from the translation model to overcome the reordering constraints of present statistical decoders and to obtain better language model estimates. The powerfulness of templates was proved by showing statistically significant improvements in translation quality on all the data sets and on all the language-pairs used in this thesis. We presented novel approaches to find equivalence classes automatically that used fewer or no knowledge sources. No other work in Generalized-EBMT has ever found the ideal conditions under which templates provide maximum gain over a lexicalized EBMT system. Since this thesis studied the usage of templates in detail, it can form a very useful guide while developing machine translation systems for new language-pairs that lack data and rich knowledge sources.

This thesis focused on improving the MT system for low-resource languages, where, finding bilingual speakers is difficult and expensive to hire human translators. The experiments were all performed on an EBMT system but can be extended to other machine translation systems.

This thesis also suggested a very simple technique to handle out-of-vocabulary as well as rare words. All the approaches in the past have only looked at handling out-of-vocabulary words. This thesis found replacements for rare words as well and showed how this can also improve the translation scores. An understanding of how to combine the dual techniques of replacements and estimating parameters directly from data seems to be a fundamental Artificial Intelligence problem.

We also tried combining all the techniques suggested in this thesis to provide better translation quality in data sparse conditions and the improvements obtained were encouraging, though work is yet to be done on finding the best way to combine the different subsystems.

Bibliography

L. R. Bahl, F. Jelinek, and R. L. Mercer. Readings in speech recognition. chapter A maximum likelihood approach to continuous speech recognition, pages 308–319. Morgan Kaufmann Publishers Inc., 1990. 8.4.4

R. Barzilay and K. R. McKeown. Extracting paraphrases from a parallel corpus. In *Proceedings of the 39th Annual Meeting on Association for Computational Linguistics*, pages 50–57. Association for Computational Linguistics, 2001. 2.3.1, 2.3.2

S. Baskaran. Hindi pos tagging and chunking. In *Proceedings of the NLPAI Machine Learning Contest*, 2006. 6

A. L. Berger, S. A. Della Pietra, and V. J. Della Pietra. A maximum entropy approach to natural language processing. In *Computational Linguistics*, pages 39–72, 1996. 2.1.2

C. M. Bishop. Springer, 2006. 4.3.4

H. U. Block. *Example-Based Incremental Synchronous Interpretation.* In W. Wahlster (ed.). Vermobil: Foundations of Speech-to-Speech Translation, Springer, Heidelberg, 2000. 2.3.2, 2.3.2

P. F. Brown, J. Cocke, S. A. D. Pietra, V. J. D. Pietra, F. Jelinek, J. D. Lafferty, R. L. Mercer, and P. S. Roossin. A statistical approach to machine translation. *Computational Linguistics*, 16(2):79–85, 1990. 2.1.2

P. F. Brown, P. V. DeSouza, R. L. Mercer, V. J. D. Pietra, and J. C. Lai. Class-based n-gram models of natural language. *Computational Linguistics*, 18:467–479, 1992. 2.3.3

R. D. Brown. Automated dictionary extraction for "knowledge-free" example-based translation. In *In Proceedings of the Seventh International Conference on Theoretical and Methodological Issues in Machine Translation*, pages 111–118, 1997. 3.2, 3.2

R. D. Brown. Example-Based Machine Translation in the PANGLOSS System. In *Proceedings of The International Conference on Computational Linguistics*, pages 169–174, 1998. 2.1.2, 3.1

R. D. Brown. Automated Generalization of Translation Examples. In *Proceedings of The International Conference on Computational Linguistics*, pages 125–131, 2000. 2.3.2, 2.3.2, 5, 5.4.1, 6.4.1

R. D. Brown. Transfer-Rule Induction for Example-Based Translation. In *Proceedings of The Machine Translation Summit VIII Workshop on Example-Based Machine Translation*, pages 1–11, 2001. 2.3.2

R. D. Brown. A Modified BWT for highly scalable Example-based translation. In *Proceedings of The Association for Machine Translation in the Americas*, pages 27–36, 2004. 3.2

J. Bullinaria and J. Levy. Extracting semantic representations from word co-occurrence statistics: A computational study. In *Behavior Research Methods*, pages 510–526, 2007. 7, 7.1

C. Callison-burch. *Paraphrasing and Translation*. PhD thesis, School of Informatics, University of Edinburgh, 2007. 2.3.2

C. Callison-Burch, C. Bannard, and J. Schroeder. Scaling Phrase-based Statistical Machine Translation to larger Corpora and longer phrases. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 255–262, 2005. 2.1.2, 2.3.2, 5.1

C. Callison-Burch, P. Koehn, and M. Osborne. Improved Statistical Machine Translation Using Paraphrases. In *Proceedings of The Human Language Technologies: Conference of the North American Chapter of the Association for Computational Linguistics*, pages 17–24, 2006. 2.3.1, 7

J. Carbonell, S. Klein, D. Miller, M. Steinbaum, T. Grassiany, and J. Frey. Context-based machine translation. In *In Proceedings of the 7th Conference of the Association for Machine Translation in the Americas*, pages 8–12, 2006. 2.3.1

M. Carl. Inducing Translation Grammars from Bracketed Alignments. In *Proceedings of The Machine Translation Summit VIII Workshop on Example-Based Machine Translation*, pages 12–22, 2001. 2.3.2, 2.3.2

M. Carl, A. Way, and W. Daelemans. Recent advances in example-based machine translation. *Computational Linguistics*, 30, 2004. 2.3.2

S. Chen, D. Beeferman, and R. Rosenfeld. Evaluation Metrics for Language Models. In *In Proceedings DARPA Broadcast News Transcription and Understanding Workshop*, 1998. 8.4.4

D. Chiang. A hierarchical phrase-based model for statistical machine translation. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 263–270, 2005. 2.2

I. Cicekli and H. A. Güvenir. Learning translation rules from a bilingual corpus. In *In Proceedings of the Second International Conference on New Methods in Language Processing (NeMLaP-2), Kemal Oflazer and Harold Somers (Eds.)*, pages 90–97, 1996. 2.3.2

P. Clarkson and R. Rosenfeld. Statistical language modeling using the cmu-cambridge toolkit. In *In Proceedings of Eurospeech*, 1997. 8.4.4

Carnegie Mellon University CMU. Public release of haitian-creole language data, 2010. 3.4

A. Dalal, K. Nagaraj, U. Sawant, and S. Shelke. Hindi part-of-speech tagging and chunking: A maximum entropy approach. In *Proceedings of the NLPAI Machine Learning Contest*, 2006. 6

G. Doddington. Automatic evaluation of machine translation quality using n-gram co-occurrence statistics. In *Proceedings of the Second International Conference on Human Language Technology Research*, pages 138–145, 2002. 3.5

A. T. Freeman, S. L. Condon, and C. M. Ackerman. Cross linguistic name matching in english and arabic: a "one to many mapping" extension of the levenshtein edit distance algorithm. In *Proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, pages 471–478. Association for Computational Linguistics, 2006. 2.3.1

R. Gangadharaiah and N. Balakrishnan. Application of linguistic rules to generalized example based machine translation for indian languages. In *Proceedings of the First National Symposium on Modeling and Shallow Parsing of Indian Languages*, 2006. 2.3.2

R. Gangadharaiah, R. D. Brown, and J. G. Carbonell. Spectral clustering for example based machine translation. In *HLT-NAACL*, 2006. 2.3.2

R. Gangadharaiah, R. D. Brown, and J. G. Carbonell. Automatic determination of number of clusters for creating templates in example-based machine translation. In *Proceedings of The Conference of the European Association for Machine Translation*, 2010a. 2.3.2

R. Gangadharaiah, R. D. Brown, and J. G. Carbonell. Monolingual distributional profiles for word substitution in machine translation. In *The 23rd International Conference on Computational Linguistics*, 2010b. 4.1

R. Gangadharaiah, R. D. Brown, and J. G. Carbonell. Phrasal equivalence classes for generalized corpus-based machine translation. In Alexander Gelbukh, editor, *Computational Linguistics and Intelligent Text Processing*, volume 6609 of *Lecture Notes in Computer Science*, pages 13–28. Springer Berlin / Heidelberg, 2011. 6

C. Goutte, P. Toft, E. Rostrup, F. A. Nielsen, and Lars Kai Hansen. On Clustering fMRI Time Series. In *NeuroImage*, pages 298–310, 1998. 7.1

H. A. Güvenir and I. Cicekli. Learning translation templates from examples. In *Information Systems*, pages 353–363, 1998. 2.3.2, 2.3.2

N. Habash. Four techniques for online handling of out-of-vocabulary words in arabic-english statistical machine translation. In *Proceedings of Association for Computational Linguistics-08: HLT*, pages 57–60, 2008. 1, 1.2, 2.3.1

Z. Harris. Distributional structure. In *Word, 10(23): 146-162*, 1954. 2.3.1, 7

W. J. Hutchins. Machine translation over fifty years. *HISTOIRE, EPISTEMOLOGIE, LANGAGE, TOME XXII, FASC. 1 (2001)*, 23:7–31, 2001. 1, 9.3

W. J. Hutchins. Current commercial machine translation systems and computer-based translation tools: system types and their uses. *International Journal of Translation*, 17: 5–38, 2005. 1

R. Iyer, M. Ostendorf, and M. Meteer. Analyzing and predicting language model improvements. In *In Proceedingsof the IEEE Workshop on Automatic Speech Recognition and Understanding*, 1997. 8.4.4

F. Jelinek. *Statistical methods for speech recognition*. MIT Press, 1997. 2.3.3

H. Kaji, Y. Kida, and Y. Morimoto. Learning Translation Templates from Bilingual Text. In *Proceedings of The International Conference on Computational Linguistics*, pages 672–678, 1992. 2.3.2, 2.3.2

M. Kay. Machine translation. *Computational Linguistics*, 8:74–78, 1982. 1

J. D. Kim, R. D. Brown, and J. G. Carbonell. Chunk-based ebmt. In *Proceedings of The Conference of the European Association for Machine Translation*, 2010. 2.3.2, 6.3.2

K. Kirchhoff and M. Yang. Improved language modeling for statistical machine translation. In *Association for Computational Linguistics, Workshop on Building and Using Parallel Texts*, pages 125–128, 2005. 2.3.3

P. Koehn. Pharaoh: a Beam Search Decoder for Phrase-Based Statistical Machine Translation Models. In *Proceedings of The Association for Machine Translation in the Americas*, 2004. 2.1.2, 2.3.2

P. Koehn, H. Hoang, A. Birch, C. Callison-Burch, M. Federico, N. Bertoldi, B. Cowan, W. Shen, C. Moran, R. Zens, C. Dyer, O. Bojar, A. Constantin, and E. Herbst. Moses: Open Source Toolkit for Statistical Machine Translation. In *Annual Meeting of ACL, demonstration*, 2007. 6.3.3

J. Lafferty, A. McCallum, and F. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of The International Conference on Machine Learning*, pages 282–289, 2002. 6.3.2

A. Lavie. Stat-XFER: A General Search-Based Syntax-Driven Framework for Machine Translation. In *Proceedings of The Conference on Intelligent Text Processing and Computational Linguistics*, pages 362–375, 2008. 2.2

Linguistic Data Consortium LDC. Hansard corpus of parallel english and french. linguistic data consortium, 1997. 3.4, 4.4, 5.4

R. Levy and C. D. Manning. Is it harder to parse chinese, or the chinese treebank? In *Association for Computational Linguistics*, pages 439–446, 2003. 6.4

D. Lin and P. Pantel. Dirt - discovery of inference rules from text. In *Knowledge Discovery and Data Mining*, pages 323–328, 2001. 7

Y. Ma, N. Stroppa, and A. Way. Alignment-guided chunking. In *Proceedings of The Conference on Theoretical and Methodological Issues in Machine Translation*, 2007. 2.3.2, 6.3

D. Marcu and W. Wong. A phrase-based joint probability model for statistical machine translation. In *In the Proceedings of The Conference on Empirical Methods in Natural Language Processing*, 2002. 2.1.2

S. C. Martin, J. Liermann, and H. Ney. Adaptive topic-dependent language modeling using word-based varigrams. In *Proceedings of Eurospeech*, pages 1447–1450, 1997. 8.4.4

Y. Marton, C. Callison-burch, and P. Resnik. Improved statistical machine translation using monolingually-derived paraphrases. In *Proceeding of The Empirical Methods in Natural Language Processing*, pages 381–390, 2009. 1.2, 2.3.1, 4.3.1

K. McTait. Translation patterns, linguistic knowledge and complexity in ebmt. In *Proceedings of The Machine Translation Summit VIII Workshop on Example-Based Machine Translation*, pages 23–34, 2001. 2.3.2

Dragos Stefan Munteanu and Daniel Marcu. Improving machine translation performance by exploiting non-parallel corpora. *Computational Linguistics*, 31:477–504, 2005. 1, 1.1

A. Y. Ng, M. I. Jordan, and Y. Weiss. On spectral clustering: Analysis and an algorithm. In *Advances in Neural Information Processing Systems*, pages 849–856, 2001. 5, 5.2

F. J. Och and H. Ney. Discriminative training and maximum entropy models for statistical machine translation. In *In the Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 295–302, 2001. 2.1.2

F. J. Och and H. Ney. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51, 2003. 6.3.2

F. J. Och and H. Ney. The alignment template approach to statistical machine translation. *Computational Linguistics*, pages 417–449, December 2004. 2.2

K. Papineni, S. Roukos, and T. Ward. Maximum likelihood and discriminative training of direct translation models. In *In Proceedings of The International Conference on Acoustics, Speech and Signal Processing*, pages 189–192, 1998. 2.1.2

K. Papineni, S. Roukos, T. Ward, and W. Zhu. Bleu: a method for automatic evaluation of machine translation. In *Association for Computational Linguistics*, pages 311–318, 2002. 2.3.1, 3.5, 4.3.5

F. Peng, F. Feng, and A. McCallum. Chinese segmentation and new word detection using conditional random fields. In *Proceedings of the 20th international conference on Computational Linguistics*, COLING '04, 2004. 2.3.3

A. B. Phillips. Sub-phrasal matching and structural templates in example-based mt. In *Proceedings of The Conference on Theoretical and Methodological Issues in Machine Translation*, 2007. 2.3.2

A. B. Phillips. The cunei machine translation platform for wmt '10. In *Proceedings of the Joint Fifth Workshop on Statistical Machine Translation and MetricsMATR*, pages 149–154, 2010. 2.1.2

M. Popovic and H. Ney. Towards the use of word stems and suffixes for statistical machine translation. In *Proceedings of The International Conference on Language Resources and Evaluation*, 2004. 1.2, 2.3.1

M. J. D. Powell. An efficient method for finding the minimum of a function of several variables without calculating derivatives. 7:152–162, 1964. 4.3.5

C. Quirk, C. Brockett, and W. Dolan. Monolingual machine translation for paraphrase generation. In *In Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pages 142–149, 2004. 2.3.2

K. Rose. Deterministic annealing for clustering, compression, classification, regression, and related optimization problems. In *Proceedings of the IEEE*, pages 2210–2239, 1998. 4.3.5

G. Sanguinetti, J. Laidler, and N. D. Lawrence. Automatic determination of the number of clusters using spectral algorithms.in. In *IEEE Machine Learning for Signal Processing*, pages 28–30, 2005. (document), 5.5, 5.4, 5.5.1, 5.3, 5.4, 5.5.2, 8.1, 8.4.1

H. Schmid. Probabilistic part-of-speech tagging using decision trees. In *International Conference on New Methods in Language Processing*, pages 44–49, 1994. 6.4

M. Snover, B. Dorr, R. Schwartz, L. Micciulla, and J. Makhoul. A study of translation edit rate with targeted human annotation. In *Association for Machine Translation in the Americas*, 2006. 3.5

H. L. Somers, I. McLean, and D. Jones. Experiments in multilingual example-based generation. In *International Conference on the Cognitive Science of Natural Language Processing*, 1994. 2.3.2

F. C. Stanley and J. Goodman. An empirical study of smoothing techniques for language modeling. In *Proceedings of the 34th annual meeting on Association for Computational Linguistics*, pages 310–318, 1996. 2.3.3

H. Tseng, P. Chang, G. Andrew, D. Jurafsky, and C. Manning. A conditional random field word segmenter. In *Fourth SIGHAN Workshop on Chinese Language Processing*, 2005. 3.4, 5.6.1, 8.4.2

P. D. Turney and P. Pantel. From frequency to meaning: Vector space models of semantics. *Journal of Artificial Intelligence Research*, pages 141–188, 2010. 7

T. Veale and A. Way. Gaijin: A bootstrapping, template-driven approach to example-based mt. In *In International Conference, Recent Advances in Natural Language Processing*, pages 239–244, 1997. 2.3.2

D. Verma and M. Meila. A comparison of spectral clustering algorithms. Technical report, 2003. 5.2

D. Vilar, J. Peter, H. Ney, and L. F. Informatik. Can we translate letters? In *Proceedings of Association Computational Linguistics Workshop on SMT*, pages 33–39, 2007. 2.3.1

S. Vogel. Pesa phrase pair extraction as sentence splitting. In *Machine Translation Summit X*, 2005. 2.3.2, 6.3.2

S. Vogel, Y. Zhang, F. Huang, A. Tribble, A. Venugopal, B. Zhao, and A. Waibel. The cmu statistical translation system. In *In the Proceedings of Machine Translation Summit IX*, 2003. 2.1.2

Y. Y. Wang. *Grammar Inference and Statistical Machine Translation*. PhD thesis, Language Technologies Institute, Carnegie Mellon University, 1998. 2.2

F. Wilcoxon. *Individual comparisons by ranking methods*. 1945. 3.5

P. C. Woodland, S. E. Johnson, P. Jourlin, and K. S. Jones. Effects of out of vocabulary words in spoken document retrieval (poster session). In *Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 372–374, 2000. 1

K. Yamada and K. Knight. A syntax-based statistical translation model. In *Association for Computational Linguistics*, pages 523–530, 2001. 2.2

M. Yang and K. Kirchhoff. Phrase-based backoff models for machine translation of highly inflected languages. In *In Proceedings of the European Chapter of the ACL*, pages 41–48, 2006. 2.3.1

L. Zelnik-manor and P. Perona. Self-tuning spectral clustering. In *Advances in Neural Information Processing Systems 17*, pages 1601–1608, 2004. 5.2

R. Zens, F. J. Och, and H. Ney. Phrase-based statistical machine translation. In *KI Advances in Artificial Intelligence*, pages 18–32, 2002. 6.3.3

Y. Zhang and S. Vogel. An efficient phrase-to-phrase alignment model for arbitrarily long phrase and large corpora. In *Proceedings of The Conference of the European Association for Machine Translation*, 2005. 2.1.2, 2.3.2, 5.1