LOCATING AND REDUCING TRANSLATION DIFFICULTY

by

Behrang Mohit

Bachelor of Computer Science, Carnegie Mellon University, 2000

Masters of Information Management and Systems, University of California at Berkeley, 2003

Masters of Intelligent Systems, University of Pittsburgh, 2006

Submitted to the Graduate Faculty of
the Intelligent Systems Program in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy in Intelligent Systems

University of Pittsburgh

2010

UNIVERSITY OF PITTSBURGH

INTELLIGENT SYSTEMS PROGRAM

This dissertation was presented

by

Behrang Mohit

It was defended on

December 3rd 2009

and approved by

Rebecca Hwa, Associate Professor of Computer Science, University of Pittsburgh

Janyce Wiebe, Professor of Computer Science, University of Pittsburgh

Daqing He, Assistant Professor of Information Science., University of Pittsburgh

Alon Lavie, Associate Professor of Language Technologies, Carnegie Mellon University

Dissertation Director: Rebecca Hwa, Associate Professor of Computer Science, University of Pittsburgh

LOCATING AND REDUCING TRANSLATION DIFFICULTY

Behrang Mohit, PhD

University of Pittsburgh, 2010

Abstract

The challenge of translation varies from one sentence to another, or even between phrases of a sentence. We investigate whether variations in difficulty can be located automatically for Statistical Machine Translation (SMT). Furthermore, we hypothesize that customization of a SMT system based on difficulty information, improves the translation quality.

We assume a binary categorization for phrases: easy vs. difficult. Our focus is on the Difficult to Translate Phrases (DTPs). Our experiments show that for a sentence, improving the translation of the DTP improves the translation of the surrounding non-difficult phrases too. To locate the most difficult phrase of each sentence, we use machine learning and construct a difficulty classifier. To improve the translation of DTPs, we introduce customization methods for three components of the SMT system: I. language model; II. translation model; III. decoding weights. With each method, we construct a new component that is dedicated for the translation of difficult phrases. Our experiments on Arabic-to-English translation show that DTP-specific system customization is mostly successful.

Overall, we demonstrate that translation difficulty is an important source of information for machine translation and can be used to enhance its performance.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

PREFACE

As this work comes to a close, there is a sense of joy and nostalgia as I reflect on the roller coaster ride of research progress. This is the moment of gratitude for those dear ones who helped me through those ups and downs.

I start with my advisor Dr. Rebecca Hwa who gets the primary credit for the supervision of this work. Thanks Rebecca for teaching me an excellent standard of academic work and life-style. Your support, specially in those frequent moments of failure were both motivating and tranquilizing.

I also extend my gratitude to members of my thesis committee Dr. Janyce Wiebe, Dr. Daqing He and Dr. Alon Lavie for providing valuable comments and suggestions during my PhD studies. I also thank my fellow ISP and CS students, staff, faculty and members of the NLP group at University of Pittsburgh and the MT group at Carnegie Mellon University.

The emotional support of this work was on the shoulder of my incredible family and circle of friends; I value and appreciate your love and support.

Throughout this work, there were moments that success was beyond imagination and only the magic of human connection and patient intellectual support brought the required persistence. I am indebted to two people for such support: My dear friend, Nilu and my excellent colleague, Frank Liberato.

I have been lucky to grow up by parents and my brother who always encouraged me to learn and experience. I hope I can maintain such life-style and help others to achieve it.

1.0 INTRODUCTION

Translation difficulty is a well known concept in the human translation community (Campbell, 1999). We investigate the application of this idea to Machine Translation (MT). MT is an intelligent system which translates a source-language input (e.g., text) to a target-language output. Information about translation difficulty enables us to adapt MT systems for translating more difficult parts of the inputs.

The translation difficulty notion has similarities for humans and computers: Most challenges exist in areas in which knowledge is sparse or ambiguous. A phrase that is difficult to translate for one human translator might be easy to translate for another human or an MT system that has the requisite knowledge.

Knowledge comes in different forms for each MT paradigm: For a rule-based system, knowledge is a collection of rules at different linguistic levels. We work with a Statistical Machine Translation (SMT) system in which knowledge as statistical models is automatically built from large volumes of parallel and monolingual corpora using machine learning techniques. There is an association between translation difficulty and the richness of a system's knowledge. We use this association to address translation difficulties by improving the usage of system's knowledge. We mainly focus on the translation of source language phrases which are sub-sentences with no syntactic constraint.

1.1 THESIS STATEMENT

Phrase translation difficulty can be quantified and automatically estimated, and MT quality can be improved by finding customized solutions for the most difficult to translate phrases.

1.2 CONTRIBUTIONS

Our research contributions are:

I. A method for the automatic estimation of translation difficulty. Our difficulty classifier highlights the most difficult phrases of a sentence with a promising 74.7% accuracy (Mohit and Hwa, 2007).

II. An empirical study on the impact of the difficult-to-translate phrases on MT and reasons that make phrases difficult.

III. Customization methods that improve the translation quality of difficult phrases. After finding the most difficult part of a sentence, we adapt two models within an SMT system for translating the difficult phrase (Mohit et al., 2009). We also adapt the extent to which these models influence the SMT process (Mohit et al., 2010). All of these adaptation are done based on the characteristics of the difficult phrase. Through these adaptations, we gain a range of translation quality improvements at the difficult phrase and also at the sentence level.

2.0 AN OVERVIEW OF THE THESIS

Our research focus is on improving Machine Translation (MT) quality for phrases that are difficult to translate. For an MT system, translation difficulty varies from one sentence to another, or even within phrases of a sentence. This variation in difficulty motivates us to locate and analyze MT difficulties at the sub-sentence (phrase) level. We hypothesize that for a given MT system, Difficult-to-Translate-Phrases (DTPs) can be detected and that difficulty can be reduced by customizing the system's knowledge. We conduct our studies on Arabic-to-English translation, using the framework of statistical MT.

This research consists of two phases: The first phase is to automatically locate the translation difficulty. We take a machine learning approach to find the most difficult phrase of each translation sentence. The second phase is reducing the translation difficulty. This includes customizing several components of the MT system for DTPs. A modified MT system which is tailored based on the characteristics of DTPs is expected to perform better. On these phrases, we decompose the translation task: the DTP is translated by the customized system, while the rest of the sentence is translated by the baseline system.

2.1 WHAT IS TRANSLATION DIFFICULTY?

Our goal is to locate phrases that are difficult to translate for an MT system. Our definition of translation difficulty is based on two premises: (I) Difficulty is system dependent. A phrase might be difficult to translate for one system and easy for another system. (II) Difficulty is indicated by poor translation quality.

As an example, we consider the sentences in Table 1. It presents the output of two MT

systems and the human reference translation and shows the difficulty variation of phrases for different MT systems. To represent the word order variations across two languages, we index the words used in the translations with the associated source-language words. For example abdullah_5 is the translation of the fifth word in the associated Arabic sentence and $\text{keynote}_{8,9,10}$ is the translation of the words 8 through 10 in the Arabic sentence.

For the MT-1 system, the starting source-language phrase has a syntactic structure that makes it difficult to translate. The Arabic word order is Verb-Subject-Object while in English the order is Subject-Verb-Object. Lack of this knowledge results in a poor word ordering in the MT-1 output. In contrast, the closing phrase is easy to translate for the MT-1 system because the system has the translation phrase in its lexicon. Therefore, it can easily translate such complex noun phrase.

For the MT-2 system, different phrases are easy and difficult to translate. The system has the knowledge of Subject-Verb-Object word movement and faces no difficulty in translating the early part of the sentence. However, due to noise and sparseness in its dictionary, it faces difficulty in the translation of the closing phrases.

Human:$\text{king}_{3,4}$ abdullah_5 will_2 deliver_2 the_6 $\text{keynote}_{8,9,10}$ address_7 in the_{12} conference_{13} at emirates_{16} $\text{center}_{14,15}$ of_{17} $\text{strategic}_{20,21}$ $\text{studies}_{18,19}$.
MT-1: **and will talk king abdullah the keynote address** in a conference in *emirates centers specialized in strategic studies* .
MT-2: and *king abdullah will deliver* **the speech AfttAH conference center in studying strategic UAE.**

Table 1: Sample **Difficult** and *Easy* phrases

There are a wide range of reasons that cause translation difficulty. Many of these reasons relate to the way that the underlying MT system works. Since we model difficulty based on the translation quality, we consider those problems which are reflected in system's translations. Linguistic challenges such as word order or semantic differences are only reflected in our modeling if they influence the translation quality.

2.2 ARCHITECTURE

Figure 1 presents the general architecture of this research. The center pieces of our work are the DTP classifier and the DTP handler. First, the controller reads in a source-language sentence. It interacts with the difficulty classifier and finds the most difficult phrase of the sentence (called the focus phrase). Then the focus phrase is passed to the DTP handler. The handler constructs a special resource for the translation of the focus phrase.

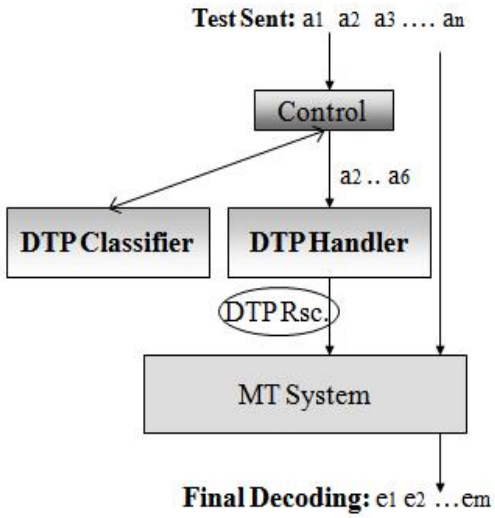

Figure 1: Our translation framework: Difficulty classifier and handler locate and modify the MT system for DTPs.

The special resource varies. It can be the human translation for DTP, or it can be a component (e.g., language model) to be used for DTP translation. In order to use the special resource, we modify the MT system. With the modified system, most parts of the sentence use the baseline translation resources while the focus phrase (e.g., DTP) uses the special resource.

2.3 LEARNING TRANSLATION DIFFICULTY

The first phase of our work is automating the prediction of translation difficulty. For us phrases are either easy or difficult to translate. We train a machine learning classifier which reads in a source-language phrase and decides if the phrase is easy or difficult for the system. To train the difficulty classifier, we need the following resources:

I. gold standard data

II. a classification model

III. a set of features.

The gold standard data is a set of source-language phrases that have easy or difficult labels. To label a source-language phrase as easy or difficult, we translate the phrase and use its translation quality. Phrases whose translation quality is above or below a certain threshold are labeled easy or difficult.

For binary classification, we use Support Vector Machines (SVM). SVMs have been reported robust classifiers dealing with large feature space. Since our difficulty modeling is system-dependent, we particularly incorporate knowledge (features) from the underlying MT system into the difficulty classifier. Additionally, we use source-language features which bring deeper linguistic knowledge into our modeling and classification.

DTPs play a critical role in translation of their sentence. Our experiments show that reducing the translation problems of a DTP, simplifies some problems for the rest of the sentence. These results motivate us to focus on problems related to DTPs and ways that they can be addressed.

2.4 SYSTEM CUSTOMIZATION FOR DTPS

As shown in Figure 1, after DTP classifier finds the most difficult phrase of a sentence, the DTP handler modifies (customizes) the MT system to improve DTP's translation. The system customization aims at the characteristics of the DTP and creates resources specifically

for the translation of DTPs. These resources are expected to make up for the missing or noisy knowledge of the baseline MT system. In this research, the baseline MT system is an instance in the Statistical Machine Translation (SMT) family: it is a Phrase-Based SMT (PB-SMT) system. We choose SMT for two reasons:

I. The statistical nature of the system makes it easier to be used for our statistical difficulty classifier. Moreover, system-based probabilistic features can be extracted easily for SMT's components.

II. SMT has a modular architecture which makes system customization easier to implement and trace.

For a Phrase-based SMT system, knowledge comes from two statistical models: (I)translation model (TM), which is a probabilistic dictionary of bilingual words or phrases; (II) language model (LM), which holds the statistical knowledge about generation of the target-language text. A SMT decoder searches these two knowledge sources to find the best translation (decoding) of an input. During the search, the decoder uses a set of weights to decide different models' influences on the translation. We conduct our experiments on a Phrase-based SMT (PB-SMT) decoder. In addition to the common SMT models (TM and LM), a PB-SMT system benefits from additional models (eg. phrase-reordering). We will discuss about SMT and PB-SMT systems in more details in Chapter 3.

Figure 2 illustrates the interactions between the SMT system and our framework:

We modify the underlying SMT system for the translation of DTPs. This customization includes the following components: (I) language model; (II) translation model and, (III) decoding weights. For customization, the handler constructs a special resource (e.g., language model). The SMT decoder uses the customized resource for the translation of the focus phrase and uses the baseline models for the translation of the rest of the sentence.

In the following chapters, we discuss these customizations and evaluate their utility by two types of focus phrases:

I. We use gold standard DTPs and easy phrases. (Chapters 5, 6 and 7)

II. To test the customization within a complete translation pipeline, we use the DTPs that the classifier finds. (Chapter 8)

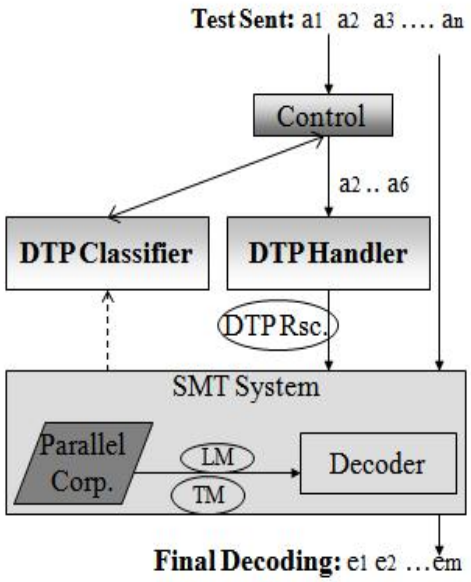

Figure 2: Our translation pipeline:The SMT system within our framework.

2.5 ADAPTATION OF THE LANGUAGE MODEL

A substantial part of translation difficulty relates to the noise and sparseness of the language model. We learned about this class of difficulty after an empirical and also a manual study of DTPs. For example, we observed that the SMT system uses back-off LM parameters for DTPs is significantly more frequent than non-DTP phrases. One way to address this problem is to use more data for model training. However this solution does not work for all target languages and systems that are constrained on training data size and memory capacities. An alternative approach is to adapt the language model based on the characteristics of the translation task. The model adaptation can be applied at different level: Corpus level, sentence level and finally, DTP level. We choose to adapt at the phrase-level because we aim to improve the translation of difficult phrases.

We adapt the language models at the phrase level. For a given source-language sentence, we use our difficulty classifier to find the most difficult phrase and train a language model adapted for translation of the highlighted DTP (one phrase per sentence). We use DTP's

words to find the relevant subset of the training data and construct an adapted language model with the new training subset.

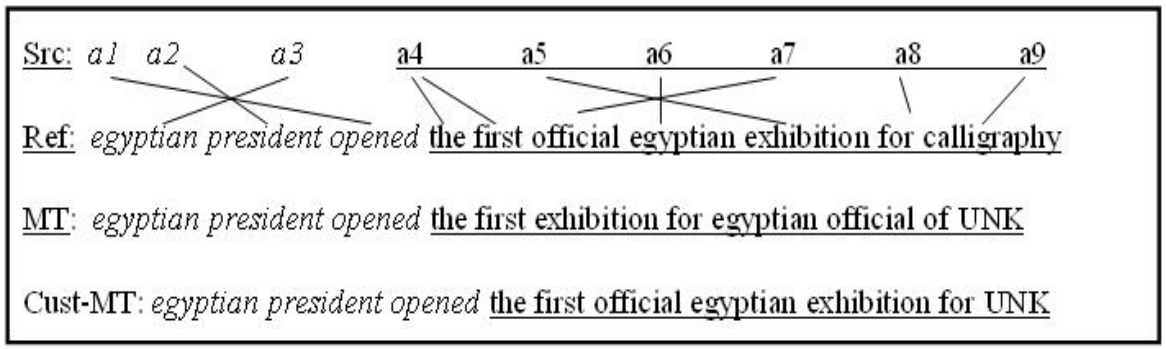

Figure 3: Example of *Easy (italic)* and Difficult (underlined) to Translate Phrases

The sentence in Figure 3 is an example of using the adapted language model. For comparing the effects of model adaptation on phrases with various levels of difficulty, we adapt the language models for two of the phrases. These model adaptation are applied separately for the easy (italic) and difficult (underlined) phrases, but are presented together. The English word official has two senses in English: *official* as an adjective, meaning formal (e.g., official meeting) or *official* as a noun, meaning a rank or position (e.g., Egyptian official). Since these two senses have different source-language (Arabic) equivalents, model adaptation based on relevant source-language sentences narrows the training data to the relevant sense. As we see in the example model adaptation does not result in translation improvements for all phrases and problems like unknown words stay intact. For an easy phrase where the baseline language model has the sufficient knowledge, model adaptation might deteriorate the translation quality. In the provided example, the decoder over-generates a longer phrase (president of egyptian) instead of using a learned phrase (egyptian president).

In Chapter 5 we present two methods for adapting the language model based on the source-language DTPs: (I) adapting language model's training data; and (II) adapting the actual probabilistic language model. We find strong quality improvements for translation of DTPs. Also, by using an oracle framework we learn that there is a large room to improve language models. Our adaptation methods are limited to fixing translation problems in a

short distance context. Other MT problems such as data sparseness (e.g., unknown words) or long distance word movements are beyond the scope of the present work and in some cases, capabilities of PB-SMT.

Our search for relevant training data is on the source-language side of the parallel corpus. We use two frameworks to locate and rank relevant parallel sentences: (I) string matching (II) using Information Retrieval (IR). The major difference of the two frameworks is the way that each weighs the relevant sentences in the adapted model. The adapted model is constructed from the target-language side of the relevant parallel sentences.

We evaluate the utility of LM adaptation in two ways: We first compare the translation quality of systems that use the baseline models against systems that use the adapted models. We also conduct a comparison of the models independent of MT. This comparison objectively estimates the likelihood that the baseline and adapted models generate gold standard phrases like the DTPs' reference translations. Both our evaluations show strong performance by the adapted language models.

2.6 ADAPTATION OF THE TRANSLATION MODEL

In SMT decoding, there is a tight interaction between the language and translation models. Our empirical analysis of DTPs' translations shows that the Translation Model (TM) is generally sparser and noisier for DTPs (e.g., number of unknown words in DTPs and Non-DTPs). We also perform experiments to learn about the decent room for improvement of the translation model, given our fixed training data. These experiments motivate us to construct alterative TMs which are adapted for the translation of DTPs.

Depending on the SMT system's architecture, translation model adaptation can take place at various steps of model training. For example, translation modeling for a phrase-based SMT system involves steps (e.g. word alignment, phrase extraction) that each one has its own parameter estimation. We compare the adaptation at various steps and settle with an adaptation at the phrase extraction step.

Our adapted phrase tables use phrase extraction heuristics which are different than the

baseline model. With an intuition that noisy knowledge is better than no knowledge, these heuristics improve the coverage (recall) of the phrase table with a small reduction of precision. Moreover they reduce number of unknown words and increasing the phrase length for DTP translation. These efforts result in the construction of a new translation model (phrase table) that for the most part performs superiorly to the baseline translation.

The above expansion of the TM is mostly effective for the translation of DTPs but not for all phrases. Our translation framework which isolates the DTP from the rest of the sentence, allows us to try a larger yet noisier model without influencing the translation of the rest of the sentence.

2.7 ADAPTATION OF THE DECODING WEIGHTS

A SMT decoder uses a set of weights to balance the influence of different knowledge resources on the final translation score of a sentence. These decoding weights are usually decided during the system training and are used for the translation of all phrases and sentences. We study the utility of adapting these decoding weights based on the translation task. We limit our focus to varying one decoding weight: the language model. We observe that usage of different LM weights for different parts of a sentence improves the translation quality. For example, DTPs share common characteristics that require them to use LM weights different from Non-DTP parts of the sentence.

Our first approach for adapting the LM weight is to find a DTP-specific weight that is used for all DTPs in the test corpus. The DTP-specific weight is automatically estimated by tuning the SMT system with a set of DTPs. We observe that such weight reduces some of the language generation problems that are common among DTPs.

Our second weight adaptation approach is finding the LM weight for the translation of individual DTPs. This approach uses a machine learning framework that takes the characteristics of individual DTPs into account. Initially we use an oracle to compile gold standard LM weights for a group of DTPs. This oracle discretely tries different LM weights and uses the reference translations to rank the weights based on their subsequent MT equality. We

use this gold standard data to train a supervised learning algorithm that ranks different LM weights. This ranking highlights the best weight for a DTP which is expected to produce an improved translation.

Both of the above methods share the intuition that different segments of the sentence require different levels of influence from SMT's components (e.g., LM weights). In both approaches, we achieve significant improvements over the baseline of using constant weights.

2.8 START-TO-FINISH AND SCALABILITY EXPERIMENTS

In the start-to-finish experiment, we incorporate our customization methods into a complete SMT pipeline. Following the framework of Figure 2, the interaction between the controller and the difficulty classifier results in finding the most difficult phrase for each sentence. Furthermore, the DTP handler constructs the special resource and the sentence get translated with both the baseline and customized configurations.

We also test the scalability of our entire framework by applying it to an SMT system that is trained on larger volumes of data. For this system, we compile a new set of easy and difficult to translate phrases. Moreover, we construct a new difficulty classifier that uses features from the new system. In these experiments, we would like to test the utility of our customization and also our entire framework with an SMT system whose models are less sparse. Furthermore, we validate wether our complete framework can be used within a standard MT evaluation.

In Chapter 8 we discuss the details of these experiments.

2.9 A REVIEW OF FINDINGS

This thesis explores the notion of translation difficulty and the ways that difficulty information can be used to enhance translation quality.

Our major research findings are:

I. Translation difficulty can be modeled and automatically predicted with decent accuracy.

II. Improving the translation quality of DTPs, boosts the translation quality of the neighboring phrases too.

III. Isolation of DTPs from the rest of the sentence, creates flexibility for applying different types of system customizations.

IV. Selective SMT customizations for DTPs, improve their translation quality significantly. We provided three successful methods for adaptation of the language model, translation model and decoding weights.

3.0 SMT AND RELATED RESOURCES AND METHODOLOGIES

This chapter provides an overview of the Statistical Machine Translation (SMT) as well as the evaluation of Machine Translation. We focus on Phrase-based SMT (PB-SMT), because the baseline MT system in our framework is an instance of it.

3.1 PHRASE-BASED STATISTICAL MACHINE TRANSLATION

Phrase-based SMT (PB-SMT) systems have been successful in many recent MT evaluations. PB-SMT models the translation task based on a probabilistic association of phrases of the source and the target-languages. phrases usually do not hold syntactic or semantic constraints; They are simply a sequence of words that have contiguous translations. The advantages of the Phrase-based translation to other SMT variants such as word-based or syntax-based is related to two premises: (I) Usage of phrases as the translation units improves the fluency of the translation. (II). The phrase definition in PB-SMT is free of any linguistic constraint, which increases the recall of the model and consequently the translation.

As shown in the following mathematical formulation, the SMT's task is to find the best target-sentence ($\mathbf{e}$) for the source-language sentence ($\mathbf{f}$).

$$e_{best} = \text{argmax}_e p(\text{e}|\text{f}) = \text{argmax}_e p(\text{f}|\text{e})\ p(\text{e}) \tag{3.1}$$

The Bayes rule helps to decompose the search into the translation and language models. In the following we discuss the PB-SMT's translation model and also its usage of the language model.

3.1.1 Translation Model

For PB-SMT, the translation model is decomposed to:

$$p(\bar{f}_1^I|\bar{e}_1^I) = \prod_{i=1}^{I} \phi(\bar{f}_i, \bar{e}_i) \tag{3.2}$$

Here, the source language sentence is broken into I phrases. The ϕ sign represents a vector of feature functions betweens the source and target-language phrases. The search for the best translation is to find the best phrase combination. Thus, PB-SMT's translation model is a probabilistic dictionary of parallel phrases. The dictionary entries are source-language phrases with their human translations and a set of probabilistic features (parameters).

There is a set of commonly-used features for representing parallel phrases. Phrase and lexical translation probabilities are two features that most PB-SMT systems use. The phrase translation probability, provides the co-occurrence ratio of the source and target phrases in the parallel corpus. The lexical translation probability gives an average word-to-word translation probability for the phrase. Phrase and lexical translation features are computed both for the source-to-target and target-to source directions. This bi-directional computation is informative for both filtering noisy phrases and also dealing with translation ambiguities that might exists for both source and target languages.

Parallel phrases are not usually available for most language pairs. PB-SMT uses a set of statistical and heuristic methods to estimate parallel phrases from the sentence aligned corpora. These statistical methods first find the word alignments between the source and target-language sentences. Then they heuristically extract contiguous word aligned chunks as the parallel phrases. Probabilistic feature values for each of the parallel phrases are estimated from different resources such as the parallel corpus.

3.1.2 Language Model

The language model is used for handling the fluency of the target-language text. It is a statistical model which estimates the probability of generating a target-language sentence. An N-gram language model approximates the generation probability of a word sequence by using shorter context of N words. Parameters of the model are the conditional probabilities

like: $p(w_n|w_1, w_2...w_{n-1})$ which estimates the generation probability of a word (w_n), given a context of $n-1$ words. For example, the trigram language model uses a context of two words to estimate the probability of a new word.

Due to its simplicity and strength, the N-gram language model has been widely used in SMT systems. The language model in PB-SMT is usually an standard tri or four-gram models. The model is usually trained on the target-language side of the parallel corpus. This makes the training domain of the translation and language models consistent. Adding more training data is expected to improve the richness of the language model. However, there are empirical evidence that shows addition of the out-of-domain data to the model might bias the model and consequently deteriorates the MT quality.

3.1.3 PB-SMT Decoding

Besides the above two models which are common to all SMT systems, some PB-SMT implementations, use other parameters and models (e.g., distortion) for influencing word or phrase movements, translation length, etc. Using the above components, the decoder's task is find the source-to-target phrase combination which minimizes the translation cost (maximizes the translation probability). The influence of the above models in the decoding is decided by a set of decoding weights. For example, language model probability or the source-to-target lexical translation probability influence the final translation score, based on their decoding weights. The decoding weights are generally are decided before the translation task and are constant for every test instance. There are machine learning procedures like the Minimum Error Training (MERT) that tune those weights.

3.1.4 MT Evaluation

The quality of translation is estimated by comparing system's output with a set of human reference translations. Assuming that the multiple references have diverse translations of the source-sentence, the metric can use partial matches with different reference translations. The usage of multiple reference translations helps the evaluation metric to credit the MT system for its alternative expressions of a concept.

There are many automatic evaluation metrics with a diverse set of criteria. For example, the BLEU score (Papineni et al., 2002), uses N-gram matching to estimate the precision of the translation. Using a variable-sized window, BLEU collects the number of N-grams in the translation output that match any of the reference translations. For example, for bigrams, it first collects all the bigrams of the translation output. Then for each bigram, it searches for matches among the reference translations and finally computes the ratio of the matched bigrams to the total number of them. The final BLEU score is an aggregation of the ratios for different lengths of N-grams. Longer N-gram matches with the references, gain stronger BLEU credits.

Other scores take different quality aspects such as the translation's recall, semantic and syntactic matching into account. The choice of MT evaluation metric is an open research debate in the research community. There are shortcomings for each metric and also in general for the automatic evaluation. While automatic evaluators are usually useful for tracing the progress of a system, their usages to compare different systems have shown inconsistencies.

3.2 USAGE OF MACHINE LEARNING IN PB-SMT

Machine learning is a computational framework for automating intelligent tasks (e.g., translation). SMT can be seen as a machine learning system which models translation as a probabilistic process. Machine learning systems have three major features: (I) task; (II) performance measure; and (III) training experience (Mitchell, 1997). For PB-SMT, the task is finding a fluent sequence of translation phrases for the source-language sentence. The performance measures are adequacy and fluency of the translation which are usually estimated by automatic metrics. The training experience is the parallel data that is a corpus of source-language sentences with their associated human translations in the target-language.

With the abstraction of PB-SMT's complicated pipeline, we can look at it as any other supervised learner. It is trained on source-language sentences as the input and target-language sentences as the output. Furthermore, a SMT system is tested in a similar fashion. The generated text is compared against gold standard reference translations with precision

and recall based metrics. The training of the PB-SMT systems includes the training of the translation and language models. In the following we discuss the usage two major learning components in the PB-SMT training.

3.2.1 Learning the Translation Model

The translation modeling in PB-SMT is the construction of the probabilistic phrase dictionary. The training data is the sentences aligned corpus. In order to extract phrases and other interesting features, word alignments between the source and target-language sentences are needed. Unsupervised learning algorithms like the Expectation-Maximization (EM) (Dempster et al., 1977) are used to word-align the corpus. In the EM framework, the algorithm starts with a simple word alignment model (e.g., random alignments). The model includes different parameters like the word-to-word translation, word-movements (distortion), etc. Through an iterative procedure, the algorithm calculates the expected parameter values, given the underlying alignments. Moreover, it chooses a new set of parameters and alignments which maximizes the observed data (parallel sentences). This procedure continues either for a fix number of iterations or until the model passes a certain convergence threshold.

Phrase extraction and parameterizations are done based on the word alignments. A set of heuristics are used to expand the word alignments and extract contiguous phrases. The phrase extraction provides a set of parallel phrases with the word alignments within the phrases.

Word alignment is the heart of translation modeling. Practically, the statistical modeling of the translation task takes place at the word alignment step. In contrast, the post-alignment steps such as phrase extraction and scoring are mostly deterministic. Therefore, the evaluation and optimization of the translation model is usually performed at the word alignment step. The performance measure for the alignment task is the the Alignment Error Rate (AER). The error rate is usually computed using a set of manually aligned sentences. The metric is mainly used to compare different alignment systems, but not within the PB-SMT training. Empirical evidence shows that large decrements of AER, results in quality im-

provements of the subsequent translations. Usually no labeled data is used for training word alignment. Therefore, the training experience for word alignment is hidden underneath the parallel corpus. The unsupervised EM framework, uses the data to iteratively estimate the training experience for the model.

3.2.2 Learning the Language Model

Training the N-gram language model only requires text samples in the target-language. The trainer uses count ratios to estimate the maximum likelihood estimate of different N-grams. For example, the trigram parameter estimation is done by collecting the counts of different the trigram and the bigram context.

$$p(w_3|w_1, w_2) = \frac{count(w_1 w_2, w_3)}{\sum_w count(w_1 w_2, w)} \tag{3.3}$$

Additional statistical estimations are used for estimating the parameters for the unseen words and phrases. The N-gram model allocates parts of its probability mass for the unseen N-grams. This allocation (called smoothing), prevents model deficiencies when we use the LM in translation. For each potential generation decision, the language model is queried for the generation probability of different N-grams. If the N-gram does not exist in the model, then a reduced order N-grams (e.g., a bigram for an unseen trigram) is used to estimate the generation probability of the unseen N-gram.

The performance of a language model is usually measured in comparison with other language models. Two language models can be compared by information theory metrics such as perplexity. Perplexity measures how much a language model fits a given text. An unseen large set of sentences is used to compute the perplexity of different models. Two LMs can also be compared in the framework of ranking different variations of the same text. In Section 5.8, we will discuss about the gold-in-sand method which is an MT-independent way of comparing language models.

3.3 IMPLEMENTATION OF PB-SMT IN OUR FRAMEWORK

Several open source and freeware PB-SMT solutions have been developed by MT researchers. In this project we benefit from two PB-SMT systems: Pharaoh (Koehn, 2004a) and Phramer (Olteanu et al., 2006). These packages have almost an identical design and model. The only difference is that Pharaoh that has a freeware C++ implementation, is stronger in computational performance. However, Phramer has an open source Java implementation that enables us to apply our decoding modifications.

For the construction of the phrase table, we use the pipeline provided by the Pharaoh training tool set. The pipeline includes word alignment, phrase extraction and phrase scoring. For construction of the trigram language model, We use the SRI language modeling package (Stolcke, 2002).

The PB-SMT decoder such as Phramer use a group of seven decoding parameters. These decoding parameters are weights assigned to different pieces of decoding information. For example, there is parameter for setting a weight for language model influence in the decoding. Generally, the Minimum Error Rate (MERT) procedure Och (2003) is applied to tune these parameters. MERT uses a small development parallel corpus to perform the tuning. We tune our baseline system with a development set of 200 sentences with the MERT framework[1].

Our experiments are performed on Arabic-to-English translations. As a highly inflected language, Arabic requires certain types of preprocessing to reduce the vocabulary size, which results in a reduction of data sparseness.

3.3.1 Modifying the Phramer decoder

Our upcoming model adaptation experiments involve using alternative models for the translation of different phrases of a sentence. We modify a few of Phramer classes to use two different models and parameter sets. The modified decoder reads in boundary limits for each of the models. At the time of cost calculations, the decoder uses the associated model for the given boundary. For example in case of language model adaptation, while picking a certain

[1]MERT uses 6 iterations to converge.

hypothesis expansion, if the chosen phrase translation falls into the boundary of a DTP, then the adapted language model which is built for DTP translation is used for computing the translation cost. We allow the decoder to use a window of one source word to the right and left to switch between the two models. This helps the decoder to continue decoding with phrases with one to three words at the DTP boundaries.

3.3.2 Preprocessing Steps

As a common practice that helps non-Arabic readers, the Arabic text is converted from the Arabic alphabets to a romanized form. This preprocessing also helps working with the Arabic data in the most basic text (ASCII) environments. In order to reduce the vocabulary size and the ambiguity, we tokenize the Arabic text with an off the shelf Arabic tokenizer. We also perform a standard basic English tokenization on the English side of the corpus. Due to technical limits of various components of the system, we remove all of the long sentence (more than 99 words) from both training and test corpora.

3.3.3 Parallel Corpora

We use the following two parallel corpora to train two PB-SMT systems:
I. Small parallel corpus for training the SMT system (LDC-Small): We use a corpus of 1 million words of Arabic-English news text from the Linguistic Data Consortium (LDC)[2]. We refer to this corpus as LDC-Small.
II. Medium parallel corpus (LDC-MED): In order to investigate the scalability of some of our experiments, we cumulatively use a medium size parallel corpus[3] to train and experiment with a larger SMT system (Chapter 8. We refer to this corpus as LDC-MED.

For the evaluation of the SMT systems and also for our work on translation difficulty, we use the following test sets of parallel corpora:
III. Multi-Translation parallel corpus (NIST-1) for classifier training and MT tests: We use the NIST 2002 Arabic-English test set (1037 sentences). We refer to this corpus as the

[2]The corpora can be obtained from the Linguistic Data Consortium under catalog ID LDC2004T17, LDC2004T18.
[3]LDC2004T17, LDC2004T18, LDC2004E13, LDC2004E72, LDC2005E46

NIST-1. This corpus comes with ten reference translations which enabled us to get multiple phrase translations for each Arabic phrase. We use 700 sentences from this corpus to extract training phrases for the classifier. The rest of the corpus (337 sentences) is used to extract test phrases, for evaluating the classifier. In experiments where we only work with the gold standard DTPs, we use a larger set of sentences and DTPs.

IV. Multi-Translation parallel Corpus (NIST-2) for Start-to-Finish Experiment: In Chapter 8, we conduct a set of experiments using our complete SMT pipeline using a held out test set. Those experiments use the NIST 2003 Arabic-English test set (661 Sentences) which comes with 4 reference translations. We refer to this corpus as the NIST-2 test set.

3.3.4 Mono-lingual Corpora

The Language Model component of the SMT system should be trained on target-language (English) text. We use the English side of the parallel corpus to train the baseline language model. As part of our language model adaptation work in Chapter 5, we construct a language model which is trained on a large volume of monolingual text. We construct this large language model by using a 200 million words subset of the English GIGA word corpus (Graff, 2005).

3.3.5 Evaluation Metrics

Our primary tool of automatic MT evaluation is the BLEU score. We use BLEU in two ways:

I. MT quality evaluation: a standard procedure that most MT research apply.

II. Phrase difficulty estimation: (to be discussed in Section 4.1.3).

Also in a few of our experiments, we obtain a second opinion from two other evaluation metrics:

I. METEOR (Metric for Evaluation of Translation with Explicit Reordering), (Lavie and Agarwal, 2007)

II. TER(Translation Edit Rate), (Snover et al., 2006)

3.3.5.1 Statistical Significance Testing: We need to follow consistent criteria to distinguish between system's significant and insignificant improvement. Ideally one would perform a null hypothesis testing on the test data. However, in most of our experimental framework such test is not practical. The bootstrap sampling framework (Koehn, 2004b) which is used to perform hypothesis testing involves selection of translating different subsets of the test set. However, our experiments which are performed on a 10 reference parallel corpus, use small test sets with less than 300 test instances. That makes the evaluation folds in the range of 30 sentences which is not a reliable size. Instead, we use an older SMT tradition to differentiate between results: We take all BLEU score changes bellow 0.5 insignificant and only pay attention to those above the 0.5 threshold.

4.0 DIFFICULT TO TRANSLATE PHRASE (DTP)

In this chapter, we focus on identifying Difficult-to-Translate Phrases (DTPs) within a source sentence and determining their impact on the translation process.

We investigate four questions:

I. How should we formalize the notion of difficulty as a measurable quantity?

II. What are the possible causes of translation difficulty?

III. To what level of accuracy can we automatically identify DTPs?

IV. To what extent do DTPs affect translation quality of the entire sentence?

We model difficulty as a system-dependent notion and estimate it by translation quality of the system. We present an automatic procedure to label difficult and easy to translate phrases. We manually examine a group of phrases to categorize the reasons that cause translation difficulty. We construct a translation difficulty classifier that reads in a phrase and labels it as easy or difficult to translate for a given system. We empirically examine the significance of DTPs and learn that DTPs deteriorate translation quality beyond DTPs' boundaries.

We use an automatic translation evaluator (BLEU score) to estimate translation difficulty. We also conduct experiments with other evaluators (Meteor and TER) to test if there is any metric effect in our difficulty estimation.

4.1 DEFINING DTPS

A Difficult-to-Translate Phrase (DTP) is a phrase that is translated poorly by a particular MT system (named S). The poorness of translation quality is judged by automatic translation

metrics such as BLEU (Papineni et al., 2002), in comparison with other phrases that S translates. Therefore, a DTP has a lower BLEU score than the majority of the other phrases that S translates.

4.1.1 What is a Translation Phrase?

As a first step towards defining and finding DTPs, we explore different options to settle on a phrase definition. Different MT paradigms look at a phrase in different ways:

From a syntax-based perspective, a phrase is a syntactic entity that is usually defined as a parse tree constituent. For example, in the tree-to-tree model, a source-language phrase is a node on the source-language parse tree which might have certain types of node alignment with a constituent on the target tree.

From a phrase-based (PB-SMT) perspective, a source-language phrase is a contiguous sequence of words that are aligned with a contiguous sequence of target-language words. These word alignments follow certain types of heuristics that are aimed for preserving the contiguity of the translation.

Our definition of a phrase has a close association with the PB-SMT view. A source-language phrase is seen as part of a longer sentence and has to have a contiguous translation. We formally define this contiguity as follows: Given a source-language sentence $f_1 f_2 ... f_n$, its translation $e_1 e_2 ... e_n$, a source-language phrase $f_g ... f_h$ and its translation $e_i ... e_j$; A contiguously translated source phrase $f_g ... f_h$ is one in which all the words between g and f are aligned with target words between i and j.

Figure 4 shows examples of phrases that pass and fail our contiguous translation constraint.

4.1.2 Compiling a corpus of parallel phrases

For our phrase difficulty research, we need a corpus of parallel phrases. Our aim is to have four reference translations for each phrase. For constructing the phrase corpus, we use the NIST 2002 Arabic-English test set (1037 sentences). We refer to this corpus as the NIST-1. This corpus comes with ten human translations which enabled us to get multiple phrase

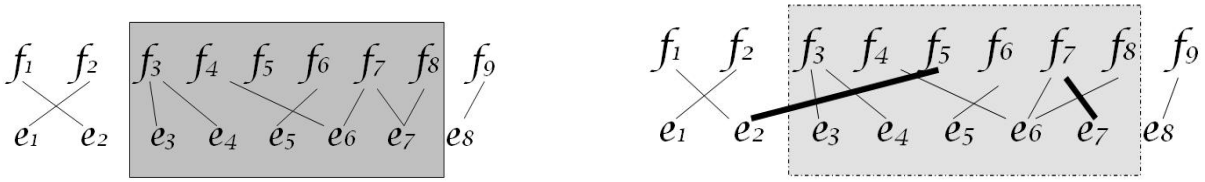

Figure 4: Examples of contiguous (L) and non-contiguous (R) phrase translation (problematic alignments are highlighted.)

translations for each Arabic phrase.

The phrase extraction procedure starts with word aligning the corpus for the source sentences and all the 10 reference translation. In order to gain a decent word alignment quality, we merge the NIST-1 corpus with a larger parallel corpus. The larger corpus helps us to increase word alignment quality. We use the GIZA++ (Och and Ney, 2003) software to align words of the merged corpora. We then use phrase-extraction tools to extract phrases from the word aligned corpora. At the end, we keep only phrases which are extracted from the NIST-1 corpus.

The phrases are extracted from 10 reference translation of the NIST-1 corpus. That means that we run GIZA++ on for each of the 10 reference translations (with the help of the extended corpus) and extract phrases from each of the 10 word-alignment corpora. We limit the phrase extraction only to the part of the NIST-1 part of the word-aligned corpus. Finally, We use the source side of the extracted phrases to match phrases with multiple (four) translations. This way we form a corpus of parallel phrases with four reference translations. These phrases that hold between 5 to 15 words, take about 32% of the word counts of their associated sentence corpus.

In total, we extract 3615 parallel phrases with 4 reference translations from the NIST-1 corpus. These phrases are later labeled as easy or difficult (Section 4.1.3) and are used to train and test the difficulty classifier (Section 4.3)

4.1.3 Automatic labeling of DTPs

We use translation quality of a phrase to decide if it is easy or difficult to translate. Each phrase is translated by the Phramer Phrase-Based SMT decoder (details in Chapter 3). Our decoder modifications (Section 3.3.1) allow us to separate the translation of the focus phrase from the translation of the rest of the sentence. This means that the when we translate the focus phrase, the preceding context from the earlier parts of the sentence is used, but the focus phrase is translated separate from the sentence. This isolated translation allows us to trace the boundaries of the translated DTP and evaluate it. Moreover, it helps us in our upcoming experiments where we use alternative models for isolated DTP translation and evaluation.

We would like to label such phrases as *easy* or *difficult* to translate. We use a held out parallel corpus (the fixed corpus) [1] to label each phrase:

1. We compute the translation quality (BLEU score) of the fixed corpus.
2. In order to label the phrase *phr*, we add its translation to the fixed corpus and recompute the translation quality.
3. If the BLEU score is improved beyond a certain threshold, the added phrase is labeled as easy. If the BLEU score is deteriorated, the phrase is labeled as difficult.
4. We remove the phrase from the fixed corpus and continue the process for labeling another phrase.

The intuition behind the above procedure is simple: A phrase that boosts the translation quality of a fixed corpus has a high translation quality and is easy for the MT system to translate. Similar intuition extends to the difficult phrases. The BLEU score variations for short phrases is large. Since BLEU uses geometric mean of different N-grams, there is a high chance of getting zero BLEU score for phrases with zero bigram matching. Moreover, the phrase length can vary the range of the phrase-level BLEU score a lot and setting a threshold for choosing easy and difficult phrase based on phrase-level BLEU score can become challenging. As a result we use the above round-robin framework of using a fixed

[1] Usage of a held out corpus makes the labeling independent of other labeled phrases.

corpus to estimate the translation quality and difficulty of a phrase.

The BLEU score change threshold is 0.01, meaning a phrase should impact the fixed corpus's BLEU score by at least 0.01 to be labeled as easy or difficult. Out of the 3615 parallel phrases, 3304 phrases are labeled as easy or difficult (the rest are neutral and are filtered out). The distribution of difficult-easy labeling is 56-44%. For labeling we use the first three references to compute the BLEU scores. We keep the last reference for future evaluations. This separation reduces the bias of our labeling on our further experiments.

The above labeling procedure helps us to create a gold standard corpus of difficult and easy to translate phrases. In the following sections, we use such labeled corpora to train and test a phrase difficulty classifier.

4.2 WHAT CAUSES TRANSLATION DIFFICULTY?

We manually examine 80 difficult and 80 easy (automatically labeled) phrases to learn the reasons behind translation difficulty. Our aim was to find problems that are DTP-specific. For most phrases, there are various interdependent reasons that make a phrase difficult to translate. Table 2 presents the most frequent reasons.

Some of the difficulty reasons are directly related to the size of the training data (e.g., unknown words). However some of the reasons are related to the shortcomings of the underlying translation and language models. For the above 80 phrases we manually trace the decodings and the associated translation and language model parameters. We observed that issues related to lexical ambiguity and short distance word movements (e.g., head modifier orders) can be addressed if the training data is used more intelligently.

4.3 DTP CLASSIFIER

Given a phrase in the source-language, the DTP classifier extracts a set of features and predicts whether it is difficult or not. Table 3 presents an overview of this component. In

Problem	**Frequency**
Unknown Source Language Word	30
Lexical Ambiguity	29
Articles/Punctuation/Numbers Deletion/Insertion	17
Cross Lingual Subject Verb Object Order Differences	16
Head-Modifier Ordering for long genitive phrases (*official egyptian exhibition*)	13
Arabic Noun Adjective Order (vs. English Adj. Noun order)	12
LM under-generation (*president of egypt* vs. *egyptian president*)	10
Evaluation Metric Problem (Fine translation not matching reference translations)	10
Word form error (plural, gerund, etc.)	8
Translation Divergence (concept expression differences across two languages)	6

Table 2: Most Frequent reasons behind phrase difficulty

Section 4.3.2, we will discuss the classification features.

For binary classification of phrases, we use Support Vector Machine (SVM) (Joachims, 1998). Due to strong classification results (Meyer et al., 2003) , SVMs have been used for many classification problems in computational linguistics. In addition to classification, we need to find the most difficult phrase for each sentence, where we care about the severity of the translation difficulty. For doing so, we benefit from the classification score as a measure of distance from the hyperplane barrier between the two classes.

Task	DTP Classifier
Input	A Phrase with baseline translation and SMT system's information
Output	Difficulty Label: Easy, Difficult

Table 3: An overview of the DTP classifier

4.3.1 Difficulty Classifier for the PB-SMT system

Our difficulty classifier is constructed for a certain PB-SMT system and uses system's features. These features allow the classifier to estimate the challenge that the system faces in translation of a given phrase. For example, the classifier uses a feature like the number of DTP words which are unknown to the PB-SMT system. For computing such feature, the classifier looks into the system's phrase-table and count number of missing DTP words.

Therefore, components such as the translation model or the language model are used in two ways:

(I) A component for PB-SMT system

(II) A feature source for the difficulty classifier which provides information about the PB-SMT system.

4.3.2 DTP Classification Features

We use 29 features for the difficulty classification. These features are collected from the system's models, syntactic structure and the baseline translation of the DTP.

Some of our phrase-level features are computed as an average of the feature values of the individual words. The following first four features use some probabilities that are collected from the parallel corpus and word alignments. For the syntactic features, we consider both the DTP and its contextual structure (e.g., type of the parent tree node). To collect syntactic features, we need to perform POS tagging and constituency parsing on the Arabic text. We use Diab's Arabic POS tagger (Diab et al., 2004) and Bikel's multilingual parser (Bikel, 2004). We use the Arabic Tree Bank (ATB) to train all the Arabic processing tools, including the POS tagger and the parser [2].

Our classification features are:

(f1) **Average probability of word alignment crossings**: word alignment crossings are indicative of word order differences and more generally the structural difference across two languages. We collect word alignment crossing statistics from the training corpus to esti-

[2]Our evaluation of these two tools shows an acceptable performance. (74% parsing accuracy and 92% POS tagging accuracy tested on a 230 sentences subset of the ATB).

mate the crossing probability for each word in a new source phrase. For example, the Arabic word *rhl* has 67% probability of alignment crossing (word movement across English). These probabilities are then averaged into one value for the entire phrase.

(f2) **Average probability of translation ambiguity**: words that have multiple equally-likely translations contribute to translation ambiguity. For example, a word that has four different translations (with similar frequencies) tends to be more ambiguous than a word that has one dominant translation. We collect statistics about the lexical translational ambiguities from the training corpus and use them to predict the ambiguity of each word in a new source phrase. The score for the phrase is the average of the scores for the individual words.

(f3) **Average probability of POS tag changes**: Change of a word's POS tagging is an indication of deep structural differences between the source phrase and the target phrase. Using the POS tagging information for both sides of the training corpus, we learn the probability that each source word's POS gets changed after the translation. To overcome data sparseness, we only look at the collapsed POS tags on both sides of the corpus. The phrase's score is the average of the individual word probabilities.

(f4) **Average probability of null alignments**: In many cases null alignments of the source words are indicative of the weakness of information about the word. This feature is similar to average ambiguity probability. The difference is that we use the probability of null alignments instead of lexical probabilities.

(f5-f9) **Normalized number of unknown words, content words, numbers, punctuations**: For each of these features we normalize the count (e.g.: unknown words) with the length of the phrase. The normalization of the features helps the classifier to not have length preference for the phrases.

(f10) **Number of proper nouns**: Named entities and proper nouns tend to create translation difficulty, due to diversity of spellings and also domain differences. We use the number of proper nouns to estimate the occurrence of the named entities in the phrase.

(f11) **Depth of the subtree**: This feature is used as a measure of syntactic complexity of the phrase. For example, continuous right branching of the parse tree which adds to the depth of the subtree can be indicative of a complex or ambiguous structure that might be

difficult to translate.

(f12) **Constituency type of the phrase**: We observe that the different types of constituents have varied effects on the translations of the phrase. For example, prepositional phrases tend to belong to difficult phrases.

(f13) **Constituency type of the parent phrase**

(f14) **Constituency types of the children nodes of the phrase**: We form a set from the children nodes of the phrase (on the parse tree).

(f15) **Length of the phrase**: The feature is based on the number of words in the phrase.

(f16) **Proportional length of the phrase**: The proportion of the length of the phrase to the length of the sentence. As this proportion grows, the contextual effect on the translation of the phrase becomes less.

(f17) **Distance from the start of the sentence**: Phrases that are further away from the start of the sentence tend to not be translated as well due to compounding translational errors.

(f18) **Distance from a learned translation phrase**: The feature measures how many words before the current phrase, a long phrase table entry (3 or more words) have been used in the decoding. Since usage of long learned phrases (in the phrase table) tends to be more accurate than word by word translations, the feature is an estimation of the contextual errors surrounding the current phrase.

(f19-f21) **Source language N-gram coverage**: Using a source-language model that is trained on the source side of the parallel data, the feature estimates the presence of unigrams (f19), bigrams (f20) and trigrams (f21) in the training data. The feature value is the average of the binary presence of the N-grams. For example, for a four-word-phrase (which has two trigrams), if one of the trigrams is present in the source-language model, then the feature value is 0.5.

(f22-f24) **Source language N-gram probability**: Similar to the previous set of language model features, with a difference that instead of a binary presence test, we use the actual N-gram probabilities and average them.

(f25-f29) **Target-language N-gram probability**: These features are similar to the previous six features, with a difference that they're computed using the phrase translation along

with a target-language model.

4.3.3 Evaluating Classifier

The distribution of the difficult vs. easy phrases stands between 50% to 57% (difficult being the majority class). This range stands as the baseline performance for our classifier. The gold standard phrases are split into three groups: 2013 instances are used as training data for the classifier; 100 instances are used for development (e.g., parameter tuning); and 200 instances are used as test instances. In order to optimize the accuracy of classification, we used a development set for feature engineering and trying various SVM kernels and associated parameters. We tested the SVM classifier with the polynomial-kernel under 10 folds cross validations. Classification accuracy stands at 71.5%. Table 4 presents the confusion matrix for the classifier. There is a stronger desire to classify phrases as difficult and we observe that the dominant error is when we classify an easy phrase as difficult.

Gold/Classified	**Diff**	**Easy**
Diff	65.8%	23.8%
Easy	33.2%	77.2%

Table 4: The confusion matrix for the performance of the DTP classifier

For feature engineering we conduct an all-but-one heuristic to test the contribution of each individual classification feature. We observe that the syntactic and the language model features are the most contributing classification features. Table 5 presents the top and bottom five contributing features.

Most contributing features	Least contributing features
f2:Translation ambiguity	f1:Alignment crossing
f11:Subtree depth	f10:Number of NNPs
f12:Const. Type of Phr	f8:Number of numbers
f25-29:Target lang. N-gram coverage	f9:Number of puncs
f22-24:Source lang. N-gram coverage	f4:Prob. of null alignments

Table 5: The top and bottom contributing classification features

4.4 THE SIGNIFICANCE OF DTPS

4.4.1 Using human translation

We hypothesize that DTPs have an important role in the translation of a sentences. This role is not only limited to the boundaries of the DTPs but also to the non-DTP segments of the sentence. In order to validate our hypothesis, we set up an experiment where we use gold standard translation for one of the phrases in the sentence. The phrases are syntactically meaningful, i.e., are nodes of the source-language parse tree. We use a corpus of 484 sentences in which half of them have a DTP highlighted and the other half have an easy phrase highlighted. In four scenarios we examine the external impact of using the perfect translation for phrases with various levels of difficulty. In each replacement scenario, one group of phrases are replaced:

Group 1: 242 sentences in which the DTP is highlighted, get the gold standard translation for the DTP part. This is a simulation of using the perfect difficulty classifier.

Group 2: 242 sentences in which the easy phrase is highlighted, get the gold standard translation for the easy phrase. This simulates using the worst difficulty classifier.

Group 3: A random selection of sentences get gold standard replacement for one highlighted phrase. This simulates using a random classifier.

Group 4: Our difficulty classifier is used to highlight the most difficult phrase within each

sentence and gold standard translation is used for the highlighted phrase.

Table 6 presents a comparison of the four translation replacement scenarios against the baseline. While it is unsurprising that the inclusion of human translations increases the overall BLEU score, this comparison shows the boost is sharper when more DTPs are translated. This is consistent with our conjecture that pre-translating difficult phrases may be helpful.

Replacement Group	**BLEU**
Baseline (No Replacement)	24.00
Random Phrs	35.1
Easy Phrs	33.7
Difficult Phrs	39.6
Classifier's choice	37.0

Table 6: Comparison of the effect of gold replacement on translation of the sentence.

A more interesting question is whether the human translations provide any benefit once we factor out their direct contributions to the increase in BLEU scores. To answer this, we compute the BLEU scores for the outputs again, this time filtering out the 484 identified phrases from the evaluation. In other words, in this experiment we focus on the part of the sentence that is not labeled.

Table 7 presents a comparison of the four translation replacement scenarios against the baseline. The largest gain (2.4 BLEU) occurs when all and only the DTPs were translated. In contrast, replacing phrases from Group II did not improve the BLEU score very much. These results suggest that better handling of DTPs will have a positive effect on the overall MT process. We also note that using our SVM-trained classifier to identify the DTPs, the constrained MT system's outputs obtained a BLEU score that is nearly as high as if a perfect classifier was used.

Replacement Group	BLEU
Baseline (No Replacement)	23.0
Random Phrases	24.5
Easy Phrases	23.9
Difficult Phrases	25.4
Classifier's choice	25.1

Table 7: Comparison of the effect of gold replacement on translation of the rest of the sentence

4.5 DECOMPOSING THE TRANSLATION PROBLEM

The above experiments support the claim that DTPs have an external influence on the translation quality of a sentence. In other words, solving the translation problem of the DTP of a sentence helps solving the translation problem of the non-DTP parts. Since we are able to automatically locate DTPs with a high accuracy (74.7%), we can divert the translation problem into a DTP problem and find solutions that take advantage of the characteristics of the DTPs. In the following chapters we continue our focus on DTPs and customize the PB-SMT system for reducing DTPs's problems.

4.5.1 Modifications of the PB-SMT decoder for focus phrases

Our system customizations (to be discussed in Chapters 5, 6 and 7) construct new resources (e.g., language model) for the translation of DTPs. We modify the PB-SMT decoder to decompose the decoding to two parts: (I) focus phrases; and (II) non-focus phrases. Our modification of the Phramer decoder includes passing the following new command line arguments at the run-time:

I. start and the end boundaries of the focus phrase (e.g., DTP) for the decoder.

II. customized model path for the focus phrase (e.g. language model).

III. customized weighs for the focus phrase.

The decoder uses the boundary indexes to decide which model and weight to use for the decoding. We use the following heuristic to decide at the boundary points:
I. The decoding of the focus phrase is done contiguously. This means that all phrase movements for the focus phrase should be kept within the boundaries of it. This allows us to extract the translation of the focus phrase as one contiguous chunk and evaluate it.
II. At each boundary point (e.g., start of the DTP), the decoder is allowed to extend using the previous model (e.g., baseline LM) up to two words. For example, while decoding around the starting point of the DTP, if a phrase-table entry is which covers two words before the DTP and two of DTP words, we allow the decoder to use the baseline model for its translation. However, if there is a phrase which covers more than two DTP words, we skip using that phrase. The same heuristic is used for ending of the DTP.

4.5.2 Evaluation of focus phrases

Since the system customizations are aimed at the translation, we evaluate the outcome at the focus-phrase-level. Our modification makes the decoder to translate the focus phrase contiguously. This allows us to simply extract the focus phrase from the sentence and evaluate it in isolation against its reference translation. In addition to this phrase-level evaluation, we evaluate the effects of the customization within the longer context of the sentence.

4.6 DIFFICULTY ANALYSIS AT THE SENTENCE LEVEL

In addition to our phrase-level difficulty analysis, we try some of the ideas for a sentence-level analysis and also system customization. The advantage of the sentence-level work is its simplicity in which there is no need for isolating focus phrases and modifying the decoder. However, applying our classification framework to sentences does not show a strong accuracy. Moreover, it requires more training data (multi-reference parallel corpora) than the phrase-

level classifier which gets multiple phrases from each parallel sentence. For example, in our phrase labeling, the NIST-1 corpus (1037 sentences), provided 3615 labeled phrases. The same corpus provides 891 instances for sentence-level labeling.

Also we observe more granularity among phrases than sentences. The variation of translation quality between easy and difficult sentences is not as dramatic as phrases. Table 8 compares the translation quality of the difficult and easy phrases vs difficult and easy sentences. Part of the sharp variation among phrases, relates to the evaluation. Phrases are shorter and small changes results in large score movements. These sharp variations are helpful for difficulty classification

Level	**Easy**	Difficult
Phrase	16.96	41.81
Sentence	25.17	33.21

Table 8: Comparison of easy and difficult phrases vs. sentences

4.6.1 Sentence-level Classifier

In order to train the sentence-level classifier, we first use our labeling procedure to compile a set of easy and difficult sentences. To maximize the size of the training data we use both the NIST-1 and NIST-2 corpora for labeling and later evaluating the classifier. We label a total of 1447 sentences. The easy vs. difficult sentence distribution is 46-54%. We use three-folds cross validation to train and test the difficulty classifier. Classifier's accuracy is at 65%. Comparing with the accuracy of phrase classification (74%), we have a significant accuracy drop.

We analyze the correlation between the classification features and the class values. We learn that for sentence-level classification, the discriminative power of our features reduces and deteriorates the accuracy of the classifier. Many of the features which are average counts over the long sentence, are not informative in the long context. Also the low classification accuracy relates to the smoother variation of easy vs difficult sentences. As we showed

above, the split between the easy and difficult sentences is more subtle than the equivalent split for phrases. This increases the chance of setting incorrect boundaries and consequently classification errors.

4.6.2 Sentence-Level Evaluation

In the following chapters, we introduce several system customization aimed at improving the translation of DTPs. It is important to clarify that there are two types of sentence level evaluations involved for those customizations:
I. Sentence-level evaluation in which the system customization is done for focus phrases: While the system customization is only applied for the focus phrase, we evaluate the customization methods for both the focus phrases and also the sentences that the focus phrases belongs to. The sentence level evaluation provides the scale of change that the modification brings to the entire corpus (in contrast to only focus phrases).
II. Sentence-level evaluation in which the system customization is done for the entire sentence.

4.7 DIFFICULTY LABELING WITH ALTERNATIVE MT METRICS

So far, for modeling translation difficulty we relied on the BLEU metric, which is the defacto MT evaluation metric. We would like to learn if the translation metric influences our difficulty modeling. In this section, we use two alternative MT evaluation metrics along with the BLEU score to investigate the metric's role.

Here, we investigate the following questions:
I. What is the agreement rate among different metrics for easy-difficult labeling?
II. What are the reasons for disagreement among metrics?

We first compare the evaluation frameworks of the BLEU and the two alternative metrics. We then investigate the above questions by:

I. Conducting an agreement study for the difficulty labeling with different evaluation metrics.
II. Conducting a quantitative analysis of disagreements between different metrics.

4.7.1 BLEU vs. METEOR and TER metrics

There are many MT evaluation metrics with different criteria, strength and weaknesses. Primarily we use the BLEU score (Papineni et al., 2002) which is a geometric mean of N-gram matches between the hypothesis and a set of reference translations. Using a variable-sized window, BLEU collects the number of N-grams in the translation output that match any of the reference translations. Longer N-gram matches with the references, gain stronger BLEU credits. BLEU's strength is the role of N-gram matching for estimating the translation precision. Translations which have many matches with the reference translations are expected to be fluent and accurate. Moreover, the geometric averaging (mean) and the brevity penalty penalize short matches and short decodings. However, there are well known shortcomings with the BLEU evaluation (Callison-Burch et al., 2006). For example, BLEU only considers exact word matching and does not take any syntactic or semantic information into account.

Here to study the agreement on difficulty labeling, we experiment with the following two metrics, which are based on word level matching (similar to BLEU):

METEOR: The Metric for Evaluation of Translation with Explicit Reordering (METEOR) (Lavie and Agarwal, 2007) measures MT quality based on both precision and recall. Moreover METEOR uses morphology (stemming) and lexical semantics in its evaluation . It provide credit to translation outputs which have the right word sense or roots, but not an exact match with the reference.

TER: Translation Edit Rate (TER) (Snover et al., 2006) measures translation quality based on the number of steps which it takes to edit an MT output and reach the reference translation. Three edit distance units of measurements are:(I) insertion; (II) deletion; and (III) substitution.

4.7.2 Agreement among metrics

In order to compare the utility of different metrics, we compare how they agree with each other in labeling phrases as easy or difficult. We first use our section 4.1.3 framework to re-label our set of 3615 parallel phrases which are translated by the baseline system. For each metric, we set the labeling thresholds based on the granularity of the metric's values.

Table 9 present the distribution of the easy vs. difficult phrases for the three metrics. Comparing with TER and METEOR, BLEU has a stronger tendency to label phrases as difficult. This mainly relates to BLEU's exact match feature that penalizes little mistakes heavily. We will discuss these cross metric difference in the following section.

Metric	**Easy**	**Diff**
BLEU	1379 (43%)	1834 (57%)
METEOR	1762 (50%)	1773 (50%)
TER	1465 (49%)	1557 (51%)

Table 9: Easy vs. Difficult label distribution for different evaluation metrics

After our multi-metric labeling, we use the agreement rate and also Cohen's Kappa metric (Carletta, 1996) to estimate the labeling agreement among these metrics. Table 10 presents the cross-metric agreements.

Metric	**Agreement**	**Kappa**
BLEU-METEOR	82.3%	64.5
METEOR-TER	85.6%	71.4
TER-BLEU	87.5%	74.9

Table 10: Labeling agreement among different metrics

4.7.3 Where do metrics disagree?

Table 10 shows that there is a moderate to strong agreement among the three metrics. However, 15 to 21 percent of phrases get different labels as we switch the metric. The

disagreements mostly are between METEOR and the two other metrics and are related to the feature differences between metrics. Here, we look into reasons behind the labeling differences. We ignore neutral labeling and focus on strong labeling disagreements. Those are phrases that are labeled easy by one metric and difficult by another one.

4.7.3.1 Disagreements of BLEU and METEOR

Majority of phrases which are labeled as difficult by the BLEU or TER and easy by the METEOR relate to lexical translation issues. An important feature of METEOR is the semantic matching which is performed by stemming and usage of the Wordnet ontology. This feature provides credit to translations which hold the right word sense. In contrast, BLEU and TER are restricted to exact word matching and penalize slight word form differences. There exist a group of phrases which have semi-correct lexical translations which do not match the reference translation. These phrases receive partial credit from METEOR and no credit from the BLEU and as a result have different difficulty labels.

Another characteristics' difference between METEOR and BLEU is their emphasis on precision and recall. This difference causes an infrequent set of labeling disagreements. BLEU favors fluent N-grams independent of translation adequacy. In contrast METEOR (with the default parameter set) has a strong emphasis on the translation adequacy. Missing a minor reference word in the a fluent hypothesis can make a phrase difficult (evaluated by METEOR) and easy (evaluated by BLEU).

4.7.3.2 Disagreements of BLEU and TER

The most frequent disagreements between BLEU and TER metrics relates to skip N-grams. For the BLEU score an extra token (e.g., punctuation) in hypothesis can break a matching 4-gram to a matching bigram which reduces the score. In contrast, TER has the insert and delete operations that simplifies skip N-gram matching and provides partial credits. Also TER has a lighter penalty for wrong word choices. The substitute operation which counts as one step, can remove a wrong and insert a correct word in one step. Comparing with BLEU's heavy penalty of mismatching, this causes a difference in difficulty labeling.

4.7.3.3 Disagreements of METEOR and TER

Some of the disagreement between METEOR and TER are similar to the disagreement of BLEU and METEOR. For example the lexical semantic matching of METEOR which provides partial credit to synonyms and word stems is missing with TER too. TER is capable of estimating translation adequacy indirectly. The metric estimates the edit distance from the hypothesis to the reference translation. Since the goal is the reference translation, all of its words would be counted in the estimation.

Each evaluation metric catches a subset of translation problems. For example, BLEU's N-gram matching is powerful in catching (dis)fluencies of the translation. Furthermore, METEOR is powerful in catching (in)adequacies of translation. The preference of one metric vs. the other one in difficulty labeling depends on the ultimate usage of the difficulty information. Our ultimate goal is to improve the BLEU score (i.e. translation quality) of the underlying PB-SMT system. Our choice of BLEU as the evaluation metric relates to its usage as the de-facto MT evaluation metric. Therefore, we optimize our difficulty reduction effort based on the BLEU score. A shortcoming of our approach which is reflected in the above agreement study, is that some translation problems (difficulties) which are exacerbated by BLEU evaluation get extra attention. Among them the most common problematic DTP labelings relate to BLEU's exact matching criteria. And finally, some problems which are missed by BLEU stay untouched.

The diversity of evaluation criteria for different metrics can be used in a hybrid form of difficulty classification. Consequently system customization can also be performed using the labeling tags of different metrics. We discuss this idea as a future work in Chapter 10.

4.8 SUMMARY

We modeled translation difficulty for phrases. Our modeling of phrase difficulty was binary: Phrases are easy or difficult to translate for a system. We introduced a framework to use the translation quality of a phrase to label it as easy or difficult. This framework allows us to compile a corpus of labeled phrases and train a binary difficulty classifier with a promising

accuracy.

Our experiments show that DTPs, have a significant influence on the translation of their neighboring phrases of the sentence. Therefore, we invest on improving DTPs' translation for the aim of improving MT in general.

Our difficulty labeling framework is dependent on translation quality metrics. We conduct a set of experiments to find the agreement of the evaluation metrics. As expected we find disagreements between easy or difficult labeling in areas where metrics' evaluation strategies disagree. We believe that an extended work on difficulty modeling should incorporate information from different MT evaluation metrics.

5.0 LANGUAGE MODEL ADAPTATION FOR DTPS

An important MT challenge is the generation of fluent text in the target-language. Statistical language models are used for handling the fluency by estimating the probability of generating a text. In Section 4.2, we observed that issues related to language modeling (e.g., short distance word movements) can cause translation difficulty. In this chapter we investigate the following three questions about language modeling:

I. How does language model cause translation difficulty? Is there a common pattern among DTPs with respect to the language model's influence?

II. What is an estimated upper-bound for language model adaptation?

III. What are effective methods for adapting a model based on the source-language text?

We empirically learn that the language model tends to be sparser and noisier for translation of DTPs. Following our general framework (Section 2.4), after finding the most difficult phrase in a sentence, the DTP-handler adapts the language model. Then, the decoder uses the adapted model for the translation of the DTP and the baseline model for the rest of the sentence.

We present a set of techniques for finding the relevant parts of the training data and adapting the language models with such relevant data. We perform a cross-language search to find the relevant parts of the target-language sentences. For our search we use two methods: (I) string matching; and (II) Information Retrieval. Also, we use two methods for model adaptation: (I) modifying the training data; and (II) modifying the parameters of the language model.

We conduct experiments using an oracle to estimate upper bounds of model adaptation. We observe that language model adaptation has strong potentials; The upper bounds perform significantly better than the baseline and also language models which are trained on large

volumes of data. Furthermore, we evaluate our adaptation methods in two ways: (I) the influence of adaptation on translation quality; and (II) An MT-independent comparison of the baseline and the adapted models. Both of these evaluations show strong improvements for model adaptation.

When we construct adapted language model for focus phrases (e.g., DTPs), we do MT evaluations both at the phrase level and also, for the sentence that the phrase belongs too. Furthermore, in a separate experiment we examine the idea of adapting the language model for the entire sentence (in contrast with only for DTPs).

5.1 TRANSLATION DIFFICULTY AND MODEL COVERAGE

In order to get a better picture of why difficult phrases result in poor translations, we compare the language model's coverage of the reference translations. For doing so, we count the percentage of the N-grams ($0 < N < 4$) that are present in the language model. Table 11 shows the coverage of our baseline language model for a set of difficult and easy to translate phrases. In order to compile this table, we use the reference translation focus phrases. Generally there is a coverage gap between the easy and difficult phrases. This shows that the LM is more sparse when it generates translation similar to DTPs' references. The gap between the easy and difficult coverage grows as we move to longer N-grams. Consequently this widens the fluency gap between the translation of the two groups.

Phrase	**1-gram**	**2-gram**	**3-gram**
DTP	86%	59%	18%
Easy	92%	76%	37%

Table 11: Comparison of Language Model Coverage for Unique N-grams.

We suspect that such gap is not only about the sparseness of the model, but also for the parameters of the existing N-grams in the model. Moreover, we suspect that the likelihood of generating translations close to DTP's references is less than the generation of easy

phrases. Some of these problems arise from N-grams which are incorrect or irrelevant with respect to domain of the text. With construction of one adapted LM for each DTP, we bias the LM towards the features of the DTP. We aim to decrease the likelihood of generating those problematic N-grams and increase the chance of generating relevant N-grams. In the following section we characterize a few generation problems that our adaptation is expected to reduce.

5.2 WHERE DOES MODIFIED LANGUAGE MODELING HELP?

By language model adaptation, we aim to improve translation quality in the following three ways:

I. Disambiguating target-language words: Target-language word ambiguity can be reduced if there are distinct source-language words. For example, the word *rice* is ambiguous in English, but it has two distinct Arabic translations for its named entity and eatable senses. An adapted language model trained in the right text domain can filter in or out the generation of a phrases like *in egyptian rice* and *rice in egypt*.

II. Short Distance word movements: Language model can also help the decoder to decide about short distance word movements. In some cases, exact translation of a phrases might be available in the parallel corpus. Due to reasons such as infrequency of the phrase, the word alignment or the phrase extractor might miss the phrase. However, the target side of the phrase is available in the training data. Therefore, a modified language model that strengthens the influence of such phrase will increase the generation of the right word order. For example, the ordering of adjectives and nouns are reversed in Arabic and English. Generation of a phrase like *dead sea region* depends on the N-gram parameters associated with that phrase[1]. A language model adapted for generation of the above phrase is likely to have a high trigram probability for the actual trigram or has high probability for the two bigrams: *dead sea* and *sea region* and lower probabilities for bigrams such as region dead.

III. Lexical Translation: There are source-language words which have lexical translation

[1]Here we are assuming that the phrase table is only providing the word-to-word translations

Task	Finding Relevant Training Data
Input	Translation Task (DTP), baseline language model, baseline training data
Output	The relevant subset of the training data

Table 12: An overview of methods for finding relevant data

ambiguities or errors. These ambiguities might relate to different linguistic reasons like morphology. For example, the morphological tokenization of the Arabic word *lhm*, maps two different Arabic words to one romanized form and causes translation ambiguity. Another example is when the phrase table holds wrong translations of a word. An adapted language model that triggers and improves the influence of context words, can reduce these ambiguities and errors.

5.3 OVERALL METHODOLOGY

We construct an adapted language model good for the translation of one focus phrase (e.g., DTP). We face two problems in model adaptation:

I. Finding the relevant data which is used to construct the adapted model

II. Adapting the baseline language model based on the relevant data.

Here we introduce different methods for each of the above two problems. Tables 12 and 13 present an overview of these groups of methods. We use features of the focus phrase to find relevant training sentences. Our search for relevant sentences takes place on the parallel corpus. We first use the methods in Section 5.5 to find the relevant source-language sentences. We then, use the association between the source and target-language sentences to find the relevant target-language sentences. We call this group the *seed* sentences. We then, use the methods in Section 5.6 to adapt the language model using the seed sentences along with their relevancy weights.

Task	Adapting the language model
Input	Baseline LM, Baseline Training Data, Relevant Subset of Training data
Output	Adapted Language Model

Table 13: An overview of the LM adaptation

5.3.1 Usage of larger language models

A traditional approach towards reducing language model's sparseness and noise is to train the model with larger corpus. This addresses the data sparseness problems, but not problems like target language disambiguation. Moreover, large volumes of training data may not always be available. Here, we compare our adapted language models with the language models that are trained on larger corpora. We cumulatively construct larger language models using up to 200 millions words of target-language text. These language models are not in the scale of the current state of the art ultra-large models (Brants et al., 2007). However, by modifying the size of the parallel corpora along with the larger language models (Chapter 8), we aim to simulate different data size scenarios.

5.4 ESTIMATING UPPER BOUNDS

To get a realistic picture about the potential of our constant resources (e.g., phrase table) and the expectations that we have from the LM adaptation, we first develop an upper bound estimate. These upper bounds estimate the potentials that the current training data and language modeling holds. We present two methods to gauge the impact of language modeling in the larger context of the decoder. Both of these methods find an improved combination of the training data to construct the language model.

5.4.1 An aggressive upper bound

Given a constant phrase table, what is the closest possible decoding to the reference translation? To obtain such an upper-bound, we train the language model with one reference translation sentence. Since our purpose is to overfit the model as much as possible to the reference translation, we do not perform smoothing. As a result, this ultra-overfited model is capable of generating only sentences very close to the reference translation. However, shortcomings of other translation resources such as unknown words or distortion errors are inherited when we use this language model. This upper bound tells us how much any N-gram language model, regardless of the training data, can be expected to improve translation quality. Since the phrase table and associated decoding methods are considered to be constant, we want to estimate how much we can accomplish without changing them.

5.4.2 A realistic upper bound

Unlike the aggressive upper-bound scenario, we train the language model on the target side of the parallel corpus. We are interested to learn what is the best language model we can build from that corpus. We still assume we have access to the reference translation, but we no longer include it directly in the training data. Instead, we assume we have a mechanism to choose the relevant parts of the target-language corpus to train a new language model. Given such a language model and a fixed phrase table, how close are the decodings of such system to the reference translation? This upper bound gives us an estimate of the potential that the current language model training data holds.

```
adapted-training-set=empty
for each N-gram (ng) in the reference translation: do
  if N-gram holds a content word: then
    for each training sentence (s) do
      if ng is in s then
        for size(ng) do
          Add s to adapted-training-set
        end for
      end if
    end for
  end if
end for
```

Figure 5: Oracle algorithm for training an upper bound LM

In order to train this upper bound, we follow the steps in Figure 5. The procedure, looks for all training sentences which have at least one content word match with the reference translations of the focus phrase. Such target-language sentences are relevant to the DTP translation. We let those sentences which have longer match, influence the adapted model stronger than others. For doing that, we repeat sentences in the training data based on their matching length. We used a development set to try various rates of repetition. As a result of this experiment, we chose the square of the longest matching length as the repetition factor. That means that sentences with maximum of a tri-gram match are repeated nine times[2].

It is worth mentioning that there are methods which may outperform our practical upper bound. For example, if we somehow tested a language model built from every subset of the target side of the corpus, we would obviously be guaranteed to do no worse than any other scheme using the same training data. However, we are interested in constructing a realistic upper bound that is computationally inexpensive and close to the spirit of the heuristics

[2]such sentence might hold more than one matching N-gram. We count the longest one.

used for model adaptation.

In addition to difficult phrases, we estimate the upper bound for easy phrases too. The performance of the easy phrase upper bound model tests how close is the baseline language model to an ideal language model.

5.4.3 Upper bound experiments

Our adaptation experiments are running the PB-SMT system on a group of 551 difficult and 453 easy phrases. Both groups are gold standard phrases. That means that the reference translation to label these phrases as easy and difficult. Such data is a simulation of the best and worst performances of the difficulty classifier. After constructing the oracle language model, our modified decoder uses the oracle model to translate the focus phrase contiguously and uses the baseline model for the rest of the sentence. In order to evaluate the focus phrase, we extract its translation from the sentence and compare against a phrase translation. We also evaluate at the sentence level.

Table 14 presents our upper bound estimation of DTPs' model adaptation for the phrase and sentence level evaluations. We have also conducted the same experiment using the larger language model (cumulatively trained on 200 million words). As explained in Section 5.4, there are two upper bound language models: the realistic upper bound and the aggressive upper bound. Using the realistic upper bound language models, difficult phrases get sharper improvements both at the phrase and sentence level. This large gap for the difficult phrases is indicative of the large potentials that the baseline training data and the idea of language model adaptation holds.

Table 15 presents a similar comparison of the two upper bounds and the larger language model for easy phrases. In contrast to DTPs, easy phrases gain a modest improvement from the upper bound language model. These results indicate how close the baseline language model is to our (approximately) ideal language model for translation of easy phrases. The parameter's of the baseline model are even better than a much larger language model for a fluent translation of easy phrases.

The large quality gap between the easy and difficult to translate phrases holds even when

Upper Bound	Phrase Level	Sentence Level
Baseline	16.96	21.63
Large LM	21.17	23.76
Realistic Upper Bound	26.83	25.98
Aggressive Upper Bound	54.23	35.92

Table 14: Upper bounds for LM adaptation of Difficult phrases compared with using a larger LM; BLEU Evaluations at the phrase and sentence levels

we use the aggressive upper bound language models. The large gap represents the limits of influence for language models (especially for the difficult phrases). Due to numerous interdependent reasons, the language model can not solely be responsible for resolving all the difficulties of the translation, and the translation quality of the difficult phrases stand well below the easy phrases. In addition to measuring the translation quality of phrases, we are interested to see how much phrase level adaptation influences the translation quality of the sentence. The right columns for tables 14 and 15 show the results of the same upper bound study, with evaluation at the sentence level. In this experiment, the language model adaptation is still taking place at the phrase level; however we evaluate the entire sentence decodings. Since the adaptation is taking place on a subset of the sentence, the score changes are smoothed as compared with the phrase level evaluation. However, we still observe significant changes with the same pattern that we see for phrases.

5.5 FINDING RELEVANT DATA

From this point we focus on our framework of LM adaptation. For model adaptation, we first need to find the subset of source-language sentences which are relevant to our translation task (focus phrase). We use the relevancy degree to add and reduce the weights of different parts of the training data. In the following we discuss two methods for locating and weighing

Upper Bound	Phrase Level	Sentence Level
Baseline	41.81	27.75
Large LM	39.26	26.93
Realistic Upper Bound	42.91	28.12
Aggressive Upper Bound	73.17	37.19

Table 15: Upper bounds for LM adaptation of DTPs compared with a larger LM; BLEU at the phrase and sentence levels

these seed sentences.

5.5.1 String Matching

We focus on the source-language content words of the translation phrase. We call these source-language terms, *seed* words. From the parallel corpus we extract sentences that hold at least one of these seed words. These are seed source-sentences which are relevant to our translation task. The match length between the focus phrases and the seed sentence decides the relevancy degree of the seed sentence. We consider the match length to be between 1 and 7 words. Matching phrases of length three or more words are allowed to include function words.

5.5.2 Using Information Retrieval

One problem with the usage of string matching for relevancy is that all terms receive similar importance. However, when we search for sentences related to the phrase *president khatami goes*, the term *khatami* is a stronger discriminative word to find relevant sentences than the term *goes*. We use the relevancy as an important weight factor in our model adaptation. Therefore, it is beneficial to improve the way that we rank relevant sentences.

In order to reduce such problems, we borrow techniques from Information Retrieval (IR). Some IR algorithms take the discriminative importance of different query words into account.

In the following, we will introduce the IR framework used to find, rank and weigh relevant sentences. We also describe the LM adaptation experiments that we conduct using the new ranked sentences.

Information Retrieval

Here, we employ Information Retrieval to search for sentences which are relevant to the translation task (focus phrase). The Lemur IR toolkit (Allan et al., 2003) is an open source platform for text searching. Lemur uses language modeling to construct index models of documents in the collection. Furthermore, it uses the language models of documents to compute the similarity score between the query and each document.

The search is performed on the source-language text. Documents are individual source-language sentences in the source side of the parallel corpus. Queries are the focus phrases (DTPs). Therefore, our IR deployment starts with indexing the source-language sentences, assuming each one is a document. At this phase we have a search engine for searching our source-language sentences. We then find the relevant sentences for each one of the DTPs. The target side of these relevant sentences is chosen for training an adapted language model.

Stemming in IR

Stemming is the process of converting words to their root form. For example, stemming the English terms *goes, went, gone* returns the root form of *go*. It has been shown that stemming improves the accuracy of information retrieval (Kantrowitz et al., 2000). For a morphologically rich language such as Arabic, stemming is expected to be more effective and increase the search recall strongly. However, the success of stemming in IR is dependent on the accuracy of the stemmer.

The Lemur toolkit has built-in stemming components for English and Arabic with medium-level accuracy. We construct a stemmed version of the search engine and perform the search with stemming. Due to the accuracy of the Arabic stemmer, we limit our usage of the stemmed index to only queries in which the search returns are infrequent (below 1000 returns).

Relevancy Weighing

The advantage of using IR is to benefit from its ranking which differentiate words based on frequency. During the training of the adapted language models, we group the IR ranked sentences into three or four groups (depending on the number of search returns):

I. Top 100 ranked sentences

II. 100th-300th ranked sentences

III. 300th-1000th ranked sentences

IV. Above 1000th ranked sentences

The above grouping categorizes sentences based on their degree of relevancy. When we construct adapted language models, we use different training weights for each of one these groups.

5.6 MODEL ADAPTATION METHODS

We now focus on the problem of language model adaptation. We are given the entire target-language corpus along with the sentences which are relevant to our translation task. Our goal is to bias the language model training towards N-grams relevant to the translation task. In the following we present two methods for constructing an adapted language model with the relevant sentences.

5.6.1 Adaptation Method 1: Changing the training data

In the first method, we use the relevant sentence set to train a new language model. This new mode only represents the probability space associated with relevant sentences and the irrelevant sentences do not have any presence in it. Each sentence carries a relevancy weight. We use repetition to set the influence of each sentence on the adapted model. Based on an empirical test on a development set, the number of repetitions is chosen as the squared factor of the relevancy weight. For example, a seed sentence whose source side has a trigram

match with the focus phrase, is repeated 9 times in the LM training.

It might be argued that exclusion of the non-relevant sentences and words prevents the SMT system from generating certain types of lexical translations. To answer such concerns, we remind that the relevant source-language sentences hold all occurrences of the source-language content words. As a result, all the lexical translations existing in the phrase table are present in the relevant sentences and the consequent language model. However, certain N-grams which hold only functions-word might be excluded from the language model which might influence the fluency of the translations. Our second adaptation method reduces such concern by including all N-grams in the adapted language model.

5.6.2 Adaptation Method 2: Modifying the Model Parameters

In our second approach, we modify the LM by redistributing the probability mass among the relevant and irrelevant N-grams. With this type of model modification, we aim to reduce the generation probability of irrelevant N-Grams while we maintain their presence in the model. This is in contrast to our first approach in which we exclude a large portion of the model's N-grams.

In order to redistribute the probability mass, we work with an intermediate form of the N-gram model. This is the counts of N-grams that is later used to estimate smoothing factors and model parameters. The advantage of working with this intermediate step is that we can rely on the state of the art methods of parameter smoothing.

Our aim is to build a model in which relevant N-grams have a stronger influence. The early step of the process is similar to our first method: We first search the source side of the parallel corpus to locate sentences that are relevant to our translation task. We then use the cross-lingual link of the parallel corpus sentences to choose the target-language sentences. From these relevant target-language sentences, we extract a set of relevant N-grams. Any N-gram that is not in this set but is present in the LM training corpus is an irrelevant N-gram. The only exception are N-grams that are all function words. These N-grams are domain independent and should not be affected by the adaptation. Since our aim in adaptation is to improve the lexical translation, we leave the model to use the baseline distribution for these

N-grams. After compiling the list of relevant and irrelevant N-grams, we modify the model as follows:

I. Reduce the count of irrelevant N-Grams and all higher order N-Grams which can generate an irrelevant N-Gram (e.g., A trigram which has an irrelevant bigram suffix). We keep track of all reduced counts and distribute them among the relevant N-grams.

II. Increase the count of relevant N-grams using mass collected from irrelevant N-grams.

Figure 6 shows a pseudo code representation of the above procedure. For constructing the LM, we use the Witten-Bell smoothing algorithm. The smoothing algorithm uses N-grams with low counts (e.g., singletons) to estimate and allocate the probability mass for unseen N-grams. This uniformly reduces the probability mass of the irrelevant N-grams, which is to our advantage. The default count value for irrelevant N-grams is 1. However for a small subset of N-grams (e.g., all function words) we use heuristics to re-estimate the count values.

The count mass that we collect from the irrelevant N-grams is distributed among relevant N-grams based on their degree of relevance. Similar to method-1, the matching words between the translation task (DTP) and the source-language sentence decide the degree of relevance for that sentence. As a result, all relevant N-grams from the pair (parallel) sentences receive the same relevance degree and later receive the probability mass.

```
for each N-gram (ng) in IrrelevantSet: do
  count(ng) -= mass
  MassHash (Prefix(ng)) += mass
  for each (N+1)gram (ng1) in RelevantSet do
    if suffix(ng1) == ng then
      count(ng1) = count(ng1) - mass
      MassHash(Prefix(ng1)) += mass
    end if
  end for
end for
for each N-gram (ng) in RelevantSet: do
  mass = MassHash(prefix(ng))
  if mass > 0 then
    COUNT(ng) += mass/relScore
  end if
end for
```

Figure 6: Pseudo-code for the adaptation method-2

After performing the above count modifications, we build a language model from these counts. We use the Good-Turing smoothing which allocates a parts of the probability mass for the unseen N-grams. This probability mass is taken from the singleton N-grams which are the irrelevant ones. We should note that our model modification does not introduce deficiencies to the model. The resulting model uses the same modeling and smoothing framework that the baseline model does. Moreover, the model holds the entire baseline model's vocabulary.

An important difference between the two approaches is the way we separate the relevant and irrelevant data. The first approach takes a radical approach and excludes all the irrelevant sentences and N-grams. The second uses the count changes to separate the two

groups. Although the second method has the advantage of having a larger and more realistic vocabulary, it requires the computational cost of tracing and allocating the mass count for the relevant N-grams.

5.7 EXPERIMENTS

The experimental setup is similar to our upper bound experiments (Section 5.4.3). The input is the parallel corpus and the output is the set of target language sentences which are relevant to the translation tasks. The experiments are the translation of a group of 551 gold-standard difficult phrases. After constructing the adapted language model, our decoder uses the adapted model to translate the focus phrase contiguously and uses the baseline model for the rest of the sentence. In order to evaluate the focus phrase, we extract its translation from the sentence and compare against a phrase translation. We also evaluate at the sentence level.

5.7.1 Comparison of two methods of finding relevant data

Here we compare the string matching and the Information Retrieval for finding relevant data and constructing the adapted language model. We construct the adapted model for the gold standard DTPs (551 phrases). The language model is constructed by the first adaptation method whose performance is superior.

Table 16 presents the results of the experiments. For this evaluation, we use the Meteor metric along with the standard BLEU. Both metrics show strong improvements for our language model adaptation framework. However, we do not observe a significant difference between the two search methods. The BLEU score of the two methods are almost similar. The METEOR score shows a small improvement for the IR method. In the following we will discuss about the reasons for different BLEU and METEOR evaluations.

The most common difference between rankings shows up for DTPs with named entities and other infrequent content words. The IR framework give extra importance to infrequent

Customization	**BLEU**	METEOR
Baseline	16.96	48.02
String Matching	21.12	53.94
Information Retrieval	20.78	54.36

Table 16: BLEU and METEOR Comparison of Usage of IR vs. String matching for LM adaptation for DTPs

terms. In IR, named entities show strong discriminative power and most of the top ranked returns are the ones with the names. In certain cases, the high rank of named entities helps the disambiguation problems (e.g. egyptian rice vs. rice in egypt).

Another advantage of using IR is its usage of stemming. In some cases, stemming helps us to collect training instances that have the proper source-language root form which doesn't match the exact DTP words. These are relevant sentences and help us to collect relevant N-grams on the target side. However, the quality of our stemming (incorporated in the IR software) is not good enough and results in selecting erroneous source language sentences.

There is a Meteor score gain when we use the IR to locate relevant sentences. We observe that this gain is achieved by a improvement in the recall and a smaller decrement of precision. This small gain is attributed to the usage of stemming in our IR work. Stemming on the source side allows some of the alternative forms of the seed words be used as relevant terms and sentences. Thus, the target side of the corpus includes some variations of the seed words. METEOR's semantic analysis provides credit for these partially right words in the decoding.

Problems with the usage of Information Retrieval

A major advantage of using the string matching is the strong credit that the method provides to longer matches. For example, matching a tri-gram is more important than matching three unigrams. The IR framework that we used, does not discriminate the variation of the

matching length strongly. However, for the PB-SMT and evaluation scores such as BLEU contiguity and having longer matches between the MT output and reference counts strongly.

A second major disadvantage of our IR employment is that IR ranks beyond the top 100 sentences are meaningless and our relevancy estimation of sentences based on their IR ranks is not accurate. The inaccurate weights that we take from the ranking and use to allocate probability mass to the relevant N-grams harm the model adaptation.

And finally our usage of source language sentences as a substitute for search engines' documents make the search index sparse. Due to the shortness of the document unit (sentences), each document have very few discriminative terms. This causes the IR formulation to use parameter smoothing extensively for many of the index terms and as a result very little observed data is used in probability estimations of the search.

To conclude, we observe that both the IR and the string matching methods have advantages and disadvantages which make them equivalent for our adaptation purpose. This equivalency is also reflected in the evaluation of the two methods (Table 16).

5.7.2 Comparison of two adaptation methods

Here, we apply our two language model adaptation methods to difficult phrases and evaluate the translation quality both at phrase and sentence levels. Table 17 presents a comparison of these methods (applied to the difficult phrases) along with the baseline translation quality both at the phrase and sentence levels.

Adaptation	**Phrase Level**	**Sentence Level**
Baseline	16.96	21.63
Upper bound	26.83	25.98
Adaptation-1	21.12	23.47
Adaptation-2	18.67	22.39

Table 17: Comparison of LM adaptation Methods for difficult phrases; BLEU evaluations at phrase and sentence levels

Both adaptation methods significantly improve the translation quality. Improvements at the sentence level are modest because model adaptation is performed only on one DTP per sentence that is about 32% of the corpus. Despite its simplicity, the first adaptation method performs better than the second method. The second method which is based on modification of the model's counts has a tendency to generate function-words N-grams. This relates to the different methods of modifying the parameters for function-words N-grams. Unlike the first method, the second approach keeps the counts of the function-words N-grams unchanged. However, since the rest of the model is strongly modified, the adapted model gets a bias towards generation of these frequent N-grams.

5.7.3 Comparison of model adaptation vs. model expansion

We presented our model adaptation as an alternative for the usage of larger language models (5.3.1). In Figure 7, we compare various sizes of language models with our first adaptation method(for difficult phrases). We observe that our method is competitive with the usage of the large language model. Moreover, the realistic upper bound that uses the baseline training data is well above the largest tested language model, which encourages further exploration of this idea.

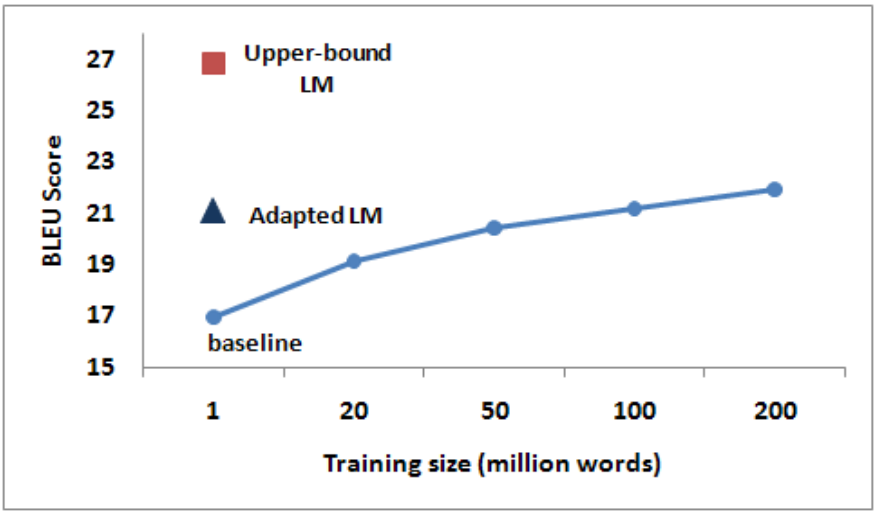

Figure 7: A comparison of using language model adaptation vs. expanding the model

5.7.4 Model adaptation for easy phrases

We are also interested to learn about the effectiveness of the adaptation on easy phrases. We apply our stronger adaptation and search methods to translate a group of 453 gold-standard easy phrases. Table 18 presents the results of the model adaptation for easy phrases. We observe that the adaptation deteriorates the translation quality of easy phrases. Our adaptation brings new model biases that are not helpful for the translation of easy phrases. For example, many of the word movements and disambiguation problems that we are targeting in our adaptation are not applicable for easy phrases. Moreover, modification of the model, specially for function-word N-grams, works as a disadvantage for the translation of easy phrases. We should also note that the upper bound performance of the LM adaptation (using the oracle), does not show a very strong improvement. This indicates that the best modification of the baseline language model does not change its performance.

Customization	**Phrase Level**	**Sentence Level**
Baseline	41.81	27.75
Upper bound	42.91	28.12
Adaptation-1	41.77	27.64

Table 18: Comparison of LM adaptation for easy phrases

5.7.5 Discussion on various combination of methods

The set of solutions offered above were aimed at solving the following problems:
I. Finding those parts of the language model which are relevant to the translation task
II. Adapting the language model for the parts that are found in the previous step.

For each of the problems we explored two solutions and we experimentally compared their performances. Moreover, we experimented with the pair of the best solutions.

For finding the relevant parts of the training data and the language model, we compared two solutions of using a string matching method and the Information Retrieval. Our experiments showed that a simple form of IR which considers each source language sentence as a

document and the DTP as a query does not provide us enough information about the level of relevancy of each sentence. As a result the performance of the IR experiments did not surpass the performance of the the string matching method. Based on this results, we continued our experiments with the string matching method which has a simpler implementation.

For adapting the language model, we explored two methods: (1) Modifying the training data and training the model similar to the baseline models. (2) Modifying different parts of the language model parameter which are relevant to the translation tasks. For both of these methods we use the above string matching method to find relevant parts of the training data and then adapt the model.

We tested the three most promising combination of methods (our of 4 possible combination). The method combination that we did not experiment was the usage of information retrieval to find relevant training data and modifying the baseline language model counts. The pair-wise comparison of methods shows the lowest performance for this method. From what we learned about our IR experiments, the estimating the relevancy of IR returns is not meaningful beyond the top 100 returns. This results in noisy weight values which will be inherited by the modified count values.

5.8 AN MT-INDEPENDENT COMPARISON OF LMS

So far we have used the end result translation quality (BLEU Score) to validate the language model improvements. However, we are interested in comparing the quality of the two language models, independent of the translation task.

Traditionally, information theory metrics such as model perplexity are used to compare two models. These comparisons are usually conducted at the corpora level. In our framework, the models are constructed for generation of a short phrase and there is not any large corpus involved. Therefore, such comparisons are not applicable to our short phrase scenario.

As an alternative evaluation, we use the gold-in-sand framework of (Zhang, 2008) to objectively compare language models independent of the translation task. Table 19 presents an overview of this method. For such comparisons the procedure uses a re-ranking task as

Task	Gold-in-sand comparison of two LMs
Input	Two LMs, corpus of DTPs
Output	The ranking of the two LMs

Table 19: An overview of the gold-in-sand procedure

following:

I. Using the baseline model, translate a corpus of DTPs and generate best-N translation for each hypothesis

II. For each DTP, add r reference translations to the n-best list (total of $N+r$ instances).

III. Using one LM at a time, use LM scoring to re-rank the step II list.

IV. Record the ranking of the reference translations.

In our experiments, we work with the best-200 hypothesis for each DTP and used four reference translations *(N=200, r=4)*. For the t-th DTP, there are four reference translations: $r_t^{(1)}, r_t^{(2)}, r_t^{(3)}, r_t^{(4)}$. These references are re-ranked by each LM as: $R(r_t^{(1)}), R(r_t^{(2)}), R(r_t^{(3)}), R(r_t^{(4)})$. We experiment with three language models: Baseline LM, Adapted LMs and Upper Bound (oracle) LMs.

We use the following quantities for our LM comparison:

I. Average ranking of the highest ranked reference translation:

$$\frac{1}{T}\sum_{t=1}^{T} min(R(r_t^{(1)}), R(r_t^{(2)}), R(r_t^{(3)}), R(r_t^{(4)})) \tag{5.1}$$

II. Average ranking of the all reference translations:

$$\frac{1}{T}\sum_{t=1}^{T} \frac{1}{r}(R(r_t^{(1)} + R(r_t^{(2)}) + R(r_t^{(3)}) + R(r_t^{(4)})) \tag{5.2}$$

The gold-in-sand experiment compares the potential two LMs have for generating fluent sentences like the reference translations. A language model that has higher average ranks of references is closer to an ideal model. Table 20 presents the results of the gold-in-sand experiments. There is an improvement in both rank averages as we switch from the baseline model

to the adapted models and later, the upper bound (oracle) models. The result is consistent with the performance that we observe in SMT experiments. However, the improvements from the adapted models to the upper bound models do not show a similar sharp boost.

LM	**Avg. Highest Refs**	**Avg. All Refs**
Baseline	81.9	125.4
Upper bound	43.8	100.3
Adaptation-1	56.1	89.1
Adaptation-2	54.1	88.7

Table 20: The *Gold-in-sand* experiment: Comparing the likelihood of generating reference translations by different LMs.

5.9 LANGUAGE MODEL ADAPTATION FOR SENTENCE TRANSLATION

Our focus is difficulty analysis and model adaptation at the phrase level. A natural extension of this idea would be to adapt the model for sentence level translation. The major difference between phrase and sentence level adaptation is the larger size of querying and searching for relevant sentences which results in finding more relevant sentences and constructing larger adapted language models.

Similar to our phrase level work, we experiment with the held out multi-translation NIST-2[3] corpus. Sentences in this corpus are labeled with the gold standard difficulty labels. The procedure to label sentences as easy or difficult is similar to the phrase labeling. From the 661 sentences in the corpus, 319 are labeled as difficult and 237 are labeled as easy.

We construct the adapted language model and evaluate the translation quality in three scenarios:

(I) Adapting the LM for all Sentences

[3]corpora are explained in Section 3

(II) Adapting the LM only for difficult sentences

(III) Adapting the LM only for easy sentences

.

We use the string matching method to find the relevant sentences. Here we deal with long sentences and most of the test sentence have a match with the source language training sentences. Since the matching length is usually long for most of those sentences, we find the repetition based on squared match length meaningless and use a linear weighting based on the match length. That means that a sentence with a four-gram match will be repeated four times in the adapted training data. Finally, we translate each sentence with its associated adapted LM. Table 21 presents the results of these adaptation experiments.

LM	**Baseline**	**Adaptation**
All Sentences	18.09	18.32
Diff. Sentences	13.71	14.09
Easy Sentences	27.19	26.02

Table 21: LM Adaptation at the Sentence Level

For both the general and DTPs experiments, we observe an insignificant improvement; for the easy sentences we observe a significant deterioration of the translation quality. As we examine each sentence's relevant training data and the language models, we observe that, on average, each adapted model is consuming 78% of the training data with almost similar relevancy weights for most of the sentences. For long sentences (more than 50 words), the adapted model uses almost the entire training data. As a result, the discrimination power of the adaptation algorithm is reduced. The resulting adapted model is close to the baseline model. The major difference between the two models will be the repetition of sentences. This modification only influences the translation of the easy sentences which benefit from being tuned with the baseline SMT system.

5.10 SUMMARY

We introduce a novel framework for adapting the language model for DTPs; we construct an adapted language model for the translation of each individual focus phrase. Through adaptation, we target a few common problems among DTPs: (a) target-language lexical disambiguation; (b) word ordering of compound phrases. We introduce two methods to locate the relevant parts of the data: String matching; and Information Retrieval. We use the relevant part of the data to construct adapted language models. We also introduce two methods for language model adaptation: Training new models; and modifying the model parameters. In different combinations of these search and adaptation methods, we achieve significant improvements for the translation of DTPs. We also evaluate our adapted model in an MT independent framework. The experiment shows that the adapted models are better than the baseline model for the generation of the gold standard translations.

6.0 TRANSLATION MODEL ADAPTATION FOR DTPS

The Translation Model (TM) is a major component of SMT that provides the source to target-language associations. For PB-SMT, the translation model is a bilingual dictionary of phrases with different types of translation probabilities (parameters). These parameters, along with other probabilities such as the language model and distortion score, influence the final score (cost) of each decoding hypothesis.

Here, we would like to address the following questions:

I. How does the translation model influence translation difficulty? What are the missing information of a translation model and where does the information get lost in the translation model training?

II. How can one adapt the translation model for a PB-SMT system?

III. How does word alignment quality and quantity influence translation difficulty?

IV. What kinds of information is needed to improve the phrase table and subsequently translation quality of DTPs?

To answer these questions, we explore the following experiments:

I. We estimate TM's coverage for the translation of DTPs and easy phrases. We also compare the strength of the phrases which are used for the translation of each group. These estimates point us to some TM features that influence the translation difficulty of phrases. For example, we learn that phrases which have unbalanced length or too many unaligned words are a source of noise and their exclusion can enrich the translation model.

II. We compare model adaptation at different stages of TM construction and settle on adaptation at the phrase extraction and scoring step.

III. We estimate an upper bound TM which is a simulation of the best model from the parallel corpus. We analyze the difference between the baseline and the upper bound TMs.

IV. We introduce heuristics from the word alignment and shallow semantics to balance between the precision and the recall of the TM. Moreover, we modify the baseline TM by increasing the strength (accuracy) of its phrase entries.

As in previous chapters, we modify parts of the SMT pipeline to use alternative translation models for translating *focus* phrases (e.g., DTPs). The evaluations are performed both at focus phrase and sentence levels.

Through our analysis of the baseline TM, we observe that there is a coverage gap between the phrases needed for the translation of DTPs and easy phrases. Moreover, we learn that those translation phrases used for the translation of DTPs do not have frequent word alignments among their words. The heuristics we introduce to modify the phrase extraction are aimed at narrowing the coverage gap for phrases holding more word alignments (stronger phrases). The resulting phrase table which is used only for the translation of DTPs, improves their translation quality significantly. The improvements are achieved mainly from the addition of longer phrases (3-5 source words), which affect translation fluency.

Due to the technical richness of the topic, we first offer a brief overview of TM construction in PB-SMT. Then, we focus on the influence of TM on Translation Difficulty.

6.1 A REVIEW OF TRANSLATION MODEL IN PB-SMT

In PB-SMT, the translation model is a probabilistic dictionary of phrases. The phrase dictionary is constructed using a pipeline of statistical and heuristic-based methods. In the following we present an overview of this pipeline and the characteristics of the TM within PB-SMT. Some earlier steps are common among most SMT systems and later steps are specific to PB-SMT.

6.1.1 Word Alignment

The training data for a SMT system is a parallel corpus which is a set of source and target-language texts, aligned at the sentence level. The first step of SMT training is to align the

source and target-language sentences at the word level.

Statistical word alignment algorithms estimate the probability of aligning a source and target word by several probabilistic parameters, such as co-occurrences, word movements, etc. The statistical estimation is done through an iterative procedure. First, words are aligned randomly. In each iteration, parameters are recomputed based on the current alignments. Furthermore, a new set of probabilistic alignments that maximize the likelihood of the data (parallel corpus) are estimated. The iterative procedure usually stops after a preset number of iterations. In addition to the obtained word alignments, a probabilistic word dictionary can be constructed from the final parameters of the model.[1]

6.1.2 Word Alignment Combination

Since the computational cost of the above word alignment procedure is expensive (trying every possible alignment), alignment algorithms make certain assumptions about the source and target-language words. For example, they might assume only source-language words can be aligned to multiple target words and not vice versa. Therefore, if we switch the order of source and target languages, we obtain different word alignments.

Both of these word alignment set are useful. A common practice in SMT research is to heuristically combine these two sets of word alignments and make a final set. Three commonly used heuristics are:

Intersection: The combination word alignment set includes only word alignments that exist in both alignment sets. The combination heuristic maximizes the precision of the word alignments.

Union: The combination word alignment set includes all word alignments from both directions. This heuristic maximizes the recall of the word alignments.

Grow Diagonal: This heuristic expands word alignment for the purpose of aligning neighboring words which form contiguous chunks. The process starts with the Intersection alignment set, which are the most reliable alignments. New alignments (from the Union set)

[1] A set of modeling algorithms (introduced at IBM) implement the above word alignment procedure. The open source GIZA++ software is a suite of different IBM word alignment algorithms and is widely used in the SMT community. The term *GIZA++ alignments* is currently accepted as a mention to the IBM's statistical word alignment algorithms.

are added to the final list only if both the source and target words are not aligned to any other word. In other words, we expand the alignment set only when there is a gap in the contiguity of aligned phrases. This selective expansion improves the alignment recall with small sacrifice to precision.

6.1.3 Phrase Extraction and Scoring

After word-aligning the sentence, source and target phrases that are contiguously word-aligned can be extracted as parallel phrases. We should note that the phrase extraction algorithm searches for all parallel and contiguous chunks on both sides of the text to extract. Each additional word alignment can work as a new constraint which reduces the number of extracted phrases. Therefore, when we have less word alignments (Intersection: high precision alignments), we extract more phrases and vice vera.

The extracted phrase dictionary is probabilistic, because it is built based on word alignments that are gained through a probabilistic procedure. At the final stage, we need to estimate the translation probability of the extracted phrases. The Phramer and Pharaoh PB-SMT decoders use four probabilistic parameters for phrase translations:

I. Source-to-Target Lexical Translation Probability: Using the word-to-word translation probabilities, we compute the average probability of translating the source phrase to the target phrase.

II. Source-to-Target Phrase Translation Probability: Using the phrase level alignment $(\bar{e}, \bar{f})$, we compute the relative frequency by counting the number of time that the source-language phrase translates to the target-language phrase:

$\phi(\bar{e}, \bar{f}) = \frac{count(\bar{e},\bar{f})}{\sum_{\bar{f}_i} count(\bar{e},\bar{f}_i)}$

III. Target-to-Source Lexical Translation Probability: Similar to parameter I, but computed in reverse order.

IV. Target-to-Source Phrase Translation Probability: Similar to parameter II, but computed in reverse order.

6.2 HOW DOES TM INFLUENCE TRANSLATION DIFFICULTY?

In Chapter 4, we discussed some reasons for translation difficulty. Most of the problems are directly or indirectly attributed to the translation model's noise and sparseness. In the following subsections, we focus on three types of TM problems frequently related to DTPs, and in further sections we explore a few solutions towards reducing them.

6.2.1 TM's Coverage for DTPs and Easy Phrases

As a first step towards understanding the TM's influence on DTPs, we compare the TM's coverage for the translation of DTPs and Easy focus phrases. To estimate coverage, we start from coverage on the parallel corpus, which is the source of TM training. We extract the source and target-language N-grams ($1 < n < 7$) from the corpus of focus phrases. We then query the parallel corpus to find the coverage percentage of different sizes of N-grams (e.g., bigram). We then query the phrase table to see what percentage of the DTPs and easy phrases are covered in the translation model. To do so, for each N-gram which belongs to the reference translation of the DTP (or easy phrase), we search for phrase table entries. We constrain this search to only phrases whose source side is useful for the translation of the DTP (or easy phrase). In other words, we perform a phrase table search for reference translation N-grams, even as the search is constrained by the source-language side of the phrase too. In the following sections we refer to this method of querying as Source-Constrained-by-Target (TCS).

Table 23 presents the result of the above experiments for DTPs and Easy phrases while we query the source (SRC) and target (TGT) sides of the parallel corpus and the Phrase Table (PT) for different N-gram lengths. We observe a wide coverage gap across all sizes of N-grams. The coverage gap hints that the shortage of TM's coverage (sparseness) is a strong feature for translation difficulty. The gap expands for longer N-grams which influences the fluency and quality of translation. Here, we focus on a comparison of 2-5 grams which are generally more frequent and influential on the translation quality.

A vertical comparison shows the level of information loss that exists in the phrase extrac-

tion process. For example, while 39% of DTPs' trigrams are present in the parallel corpus, only 11% of them get into the phrase table. A comparison of DTPs and easy phrases shows that this information loss is stronger for DTPs. The loss might relate to several factors such as low frequency of the lexemes, long distance word movements, etc. These factors reduce the recall and precision of word alignments and consequently the phrase extractions.

Lang	Phr Type	1-gram	2-gram	3-gram	4-gram	5-gram	6-gram	7-gram
SRC	**DTP**	98%	66%	39%	19%	8.4%	5.3%	3.5%
	Easy	98%	82%	57%	35%	20%	12%	7.5%
TGT	**DTP**	95%	69%	35%	14%	6.4%	3.3%	2.1%
	Easy	97.7%	83.4%	53.7%	30.9%	15%	7.3%	4.0%
PT	**DTP**	74%	33%	11%	3.1%	1.1%	0.4%	0.001%
	Easy	90%	55%	24%	11%	4%	1.7%	0.7%

Table 22: Comparison of the coverage of the Phrase-table's N-grams from DTPs and Easy Phrases

An intuitive way of reducing this gap is to increase the coverage of the phrase table for the DTPs' translation. The absence of certain phrases in the TM does not necessarily mean that the parallel corpus does not hold the phrase. For example, Table presents an example of TM sparseness while the relevant data is available. The baseline grow-diagonal does not include the short phrase *gyrhArd $rwdr*, but longer phrases which have the two words are in the model, but not useful for our translation. Using the alternative (Intersection) heuristic, the TM adaptation collects shorter phrases which might be noisy, but with reduced unknown words.

As explained in Section 6.1, the construction of the phrase table is a heuristic-based procedure based on statistical word alignments. The procedure has the strategy of maximizing the precision of phrase extraction. For DTPs, alternative phrase extraction strategies that increase recall might improve the TM coverage and translation quality. We explore this approach in Section 6.5

Given the parallel corpus, the SRC and TGT rows present an *upper bound* coverage

Baseline: ... ends his talks at gyrhArd $rwdr ...
Ref.:... conclusion of his talks with gerhard schroeder ...

Table 23: An example of a missing phrase while the lexeme is present in the parallel corpus

for the N-grams of DTPs and easy phrases. These are the highest coverage point that any alternative alignment and phrase extraction mechanism can reach with our parallel corpus. In Section 6.3, we will discuss the upper bound of the TM in more detail.

6.2.2 Lexical Ambiguity for DTPs and Easy Phrases

Lexical translation ambiguity is a common problem in MT that is frequently observed among DTPs. The problem occurs when there is more than one translation for a source or target-language word (or phrase). It is expected that context words and the language model help the translation model in choosing the right translations. However, when models are noisy and sparse, model cooperation becomes loose.

The Context's role in reducing the translation ambiguity problem is limited. The ambiguity problem is exacerbated when multiple translations of words have close meanings and belong to close senses. Then, it becomes difficult to use the context to disambiguate and choose the correct translation. For example, the Arabic word *Omr* has several multiple translations: *year, age, life*. An alternative diacritical reading of the word is a named entity that is transliterated as *Omar*. Disambiguating between the named entity sense and other senses is probably achievable via context words. However, the distinction between close meaning such as years and age is more challenging for the language model.

To reduce this ambiguity problem, we attempt to adapt the TM by recalculating the phrase translation probabilities from relevant parts of the parallel corpus. In Section 6.4 we describe this unsuccessful approach in more details.

6.2.3 Phrase Strength for DTPs and Easy Phrases

We have observed that TM's coverage is lower for the translation of DTPs. We suspect that the TM problem is not solely related to the coverage of the model, but also the strength of the phrases that are present in the model. We define phrase strength based on an analogy of phrases and bridges. A strong bridge (phrase) is the one with frequent and reliable pillars (alignments).

We would like to compare the length and the strength of the phrases that are used for the translation of DTPs, and easy phrases. We perform an experiment on the length and the strength of the phrase table's entries that are used for the baseline translation of DTPs and easy phrases. We use heuristics to estimate the phrases' strength. Using a bridge analogy, a phrase which has strong and frequent piers (alignment), is more accurate and stronger. We observe that the average phrase length in the baseline decoding of DTPs is 1.51 words. For easy phrases, the average is 2.2 words. Furthermore, we look into the strength of long phrases that are used for decoding. Specifically, we compare a few criteria for all phrases with 3 or more source-language words.

Here is the list of criteria used to determine phrase strength:

I. The ratio of phrases with three or more words to all phrases

II. The ratio of aligned content words in the source and target sides of the phrase

III. The proportion of word alignments to the length of the phrase

IV. The ratio of content words aligned to content words

V. The ratio of intersection alignments to total number of alignments

Table 24 presents a comparison of these criteria for the phrases that are used in the baseline decodings of the DTPs and easy phrases.

Phr Type/Qual Crit.	**Avg. Phr Len**	**I**	**II**	**III**	**IV**	**V**
DTP	1.5	9%	81%	62%	68%	53%
Easy	2.2	27%	94%	79%	85%	77%

Table 24: Phrase strength for the translation of DTPs and easy phrases

From these comparisons, we observe that the data gap between DTPs and easy phrases is not only related to the model's coverage but also to the strength of the phrase table entries. For example, the fourth criterion shows that the phrases used for DTP translation have a lot of unaligned content words which are indications of a phrase's noise. Moreover, we observe that for DTPs, phrases rely more on alignments that come from the grow-diagonal heuristics rather than automatic GIZA alignments[2].

Improvements of the phrase table and the TM should be towards increasing both the precision and recall. In our work on expanding the phrase table in section 6.6, we focus on increasing the phrase extraction quality based on the above criteria.

6.3 ESTIMATION OF AN UPPER BOUND TM

Before modifying the baseline TM, we face a fundamental question: How much more can one improve TM? We investigate this question by branching it into the following queries:
I. Given an automatically word aligned parallel corpus, what is the best possible TM that can be built from the corpus? We consider the model to be the *upper bound* TM for the given corpus.
II. What percentage of N-grams extracted from DTPs and Easy phrases, exist in the upper bound phrase table?
III. How does the upper bound translation model influence the translation quality?

In the following we present a few experiments to estimate the coverage of an upper bound phrase table. We also estimate the translation quality of an upper bound phrase table.

6.3.1 Estimation of Coverage of the Upper Bound TM

We would like to estimate an upper bound for the recall of a phrase table that is constructed from our fixed parallel corpus. Similar to Section 6.2.1, we focus on the coverage of N-grams that are extracted from a corpus of focus phrases (e.g., DTPs).

[2]We heuristically give extra value to GIZA alignments that are estimated from observed data probabilities (compared with grow-diagonal alignments that are completely heuristic based).

We query two resources for estimating the coverage:

I. Parallel Corpus: We estimate the coverage independent of the word alignments and phrase extraction heuristics (Similar to Table 23).

II. Intersection Phrase Table: This way, we estimate the coverage from the phrase table with the largest number of phrases (high recall). This estimate takes the strength of the automatic word alignment and phrase extraction into consideration and is a more realistic estimate.

TM	**1-gram**	**2-gram**	**3-gram**	**4-gram**	**5-gram**	**6-gram**	**7-gram**
Baseline	83%	53%	26%	10%	4.3%	2.1%	0.8%
Intersect	90%	63%	35%	16%	6.2%	3.5%	2.1%
Corpus	98%	66%	39%	19%	8.6%	5.3%	3.5%

Table 25: Comparison of the coverage of the source-language N-grams

In Section 6.2.1, we computed the parallel corpus's coverage for N-grams of DTPs and east phrases. Here we compute the coverage of the baseline and the intersection phrase tables in a similar way. We query the two tables with three types of phrases: source-language, target-language and target-language constrained by the source.

Tables 25, 26 and 27 present the upper bound estimates of the coverage for DTPs' N-grams using the baseline and the intersection phrase tables and also the parallel corpus. When we query the two phrase tables, the TCS is our best upper bound estimate, because it only counts those N-grams that practically have a chance to be included in the decoding of the DTP.

Going through the above three coverage tables, we observe that the coverage deteriorates at two steps:

I. Word alignments (From coverage of the corpus to the coverage of the intersection table).

II. Phrase Extraction (From intersection table to the baseline table)

For example, we observe that over 10% of unigrams are not in the baseline phrase table,

TM	1-gram	2-gram	3-gram	4-gram	5-gram	6-gram	7-gram
Baseline	84%	45%	26%	4.5%	1.4%	0.5%	0.1%
Intersect	91%	64%	28%	9.8%	5.0%	1.5%	0.4%
Corpus	95%	69%	35%	14%	6.4%	3.3%	2.1%

Table 26: Comparison of the coverage of the target-language N-grams

TM	1-gram	2-gram	3-gram	4-gram	5-gram	6-gram	7-gram
Baseline	74%	33%	11%	3.1%	1.1%	0.4%	0.001%
Intersect	88%	58%	24%	8%	3%	0.8%	0.3%

Table 27: Comparison of the coverage of the TCS N-grams

while they exist in the corpus or the intersection table. Alternative approaches should take advantage of these sparse parts of the model by relaxing the alignment and extraction procedures. We will examine such approaches in Section 6.5.

6.3.2 Estimation of Translation Quality of the Upper Bound (Oracle) TM

Here, we estimate the upper bound (oracle) TM for the translation of a development set of DTPs and Easy phrases. Similar to the previous sections, our focus is on the comparison of DTPs and easy phrases. The development sets consist of 451 DTPs and 353 easy phrases which are translated in the context of their associated sentences.

Our aim is to direct the decoder to translate each focus phrase as close as possible to its reference translation. To do so, we have to construct individual phrase table for each DTP. Each table holds only source-language words and phrases that match the translation focus phrase. The target side of these phrases should also match the reference translation of the focus phrase. However, There are source-language words and phrases that have no matching

translation with the reference. We keep all translations for those entries.

After constructing individual phrase tables, we translate focus phrases in the context of their sentences and perform a phrase and sentence-level evaluation. Table 28 present a comparison of translation quality when we use an upper bound table for the translation of DTPs and Easy phrases.

Phr Type	**Phr Baseline**	**Phr Upper Bound**	**Sent Baseline**	**Sent Upper Bound**
DTP	17.81	19.91	22.87	23.72
Easy	43.67	45.28	29.51	30.13

Table 28: Comparing the translation quality using the upper bound model

The above upper bound estimation is a loose estimation. In practice, we can enforce all the reference matching phrases into the final decoding. Even when we exclude the alternative translation of a source-language phrase, its alternative translations can be generated by using shorter phrases. However, the results provide enough estimate for the improvements that we can expect from TM adaptation.

We observe that the information in the baseline translation model is being used close-to-perfect and there is a small room for improvement if we are going to limit the TM adaptation to the baseline model. This motivates us to look into alternative path of adaptation through changing the word alignments and phrase extraction methods.

6.4 ON PRACTICALITY OF TM ADAPTATION THROUGH WORD ALIGNMENT AND PHRASE SCORING

Here, we discuss two unsuccessful ideas for the TM adaptation.

It can be argued that the real TM adaptation should take place at the word alignment step in which the most crucial lexical translation parameters are estimated. As a first step of examining the idea of TM adaptation, we analyze the feasibility of word alignment adaptation.

The widely used GIZA++ word alignment systems use unsupervised (or semi-supervised) learning frameworks. These systems iteratively use several alignment algorithms to estimate the word-to-word alignments and consequently the lexical translation probabilities. The procedure is computationally intensive and with the current computational power, each run takes a few hours for a small parallel corpus. Thus, a per-phrase TM adaptation is not a feasible and practical solution (even for a proof of concept).

In addition to the above computation cost, there is a second problem with TM adaptation through the world alignment. In the LM adaptation (Chapter 5), the model was being constructed from simple counts of N-grams. Filtering irrelevant sentences only reduces the counts of some N-grams and does not interfere with the general modeling. However, the translation modeling is more complex with a set of interconnected parameters. In the TM, the context that influences the parameter estimation can be as long as the length of the sentence. For example, the distortion parameters in the IBM model 3,4 and 5 are estimated by collecting word movements within the entire sentence. Due to this strong context influence, filtering the irrelevant sentences, influences the parameter estimation beyond the words of the irrelevant sentences. It can even make the model incapable of aligning function words. Therefore, the resulting phrase table will be too sparse and noisy.

6.4.1 TM Adaptation by Narrowing the Phrase Extraction

In Section 6.2.2, we discussed translation ambiguity as one common TM problem. We also explained how context can be used to reduce ambiguity problems. We would like to know whether the adaptation of TM through phrase extraction and scoring reduces the translation ambiguity problem. Here we follow an adaptation framework that is similar to our earlier work on the LM adaptation. We construct phrase tables that are good only for the translation of an individual DTP.

The adaptation takes place when we score the phrases. First, for the phrase extraction we use the same baseline word alignments that are generated for the entire parallel corpus. For the phrase scoring, we focus on those sentences that have N-gram matches with our translation task (focus phrases). This adaptation procedure is similar to the LM adaptation

(Section 5.6). We weigh each parallel sentence, based on the total length of N-gram matches that it has with the focus phrase. The weighing is done by repetition and a new small parallel corpus is formed for the phrase extraction and scoring. This weighing does not have any effect on lexical translation probabilities because those probabilities are estimated from the word aligned corpus. However, the corpus adaptation influences the phrase translation probabilities.

The premise of this filtering and repetition is to give extra influence to those phrase translations that share more context with the source-language phrase. We should note that unlike the LM adaptation, the filtering of non-relevant sentences does not have a radical influence on the model. The reason is that most of the filtered sentences have no words in common with the translation task, and their extracted phrases would not be used.

After constructing a small adapted phrase table for each DTP, we translate DTPs with the adapted model. After evaluation, we observe that there is not any significant change in the translation quality for either the DTPs or the easy phrases. These results are contrary to our similar approach to LM adaptation. In the LM adaptation, filtering the irrelevant sentences sharpened the model parameters and added to the discriminative power of the model. However in the TM, the filtering has an insignificant effect on the model and the model modification is mainly related to the extra weighing that we provide to the more relevant sentences.

Tracing through the translations, we observe that translation disambiguation occurs infrequently. Although ambiguity is a common problem among DTPs, our context words can not reduce the problem effectively. In many cases the match between the training sentence and the translation task share one content word which is not sufficient context for disambiguation. As a result many of the decodings are similar to the baseline.

Another problem with this type of TM adaptation is that estimation of the phrase translation probabilities will be limited to a subset of the phrase table that matches the source-language translation task. This is not a problem when we estimate the source-to-target probabilities, because all instances of the relevant source-language phrases are present in the subset. However, for estimating the target-to-source probabilities, we are bound to a subset that gives us noisier probability estimate. As a result we have seen instances of

getting noisier DTP translation after adapting the model.

6.5 TM ADAPTATION BY MODIFYING WORD ALIGNMENTS' RECALL

In previous sections, we learned that the coverage of the baseline TM is low for DTPs. We also observed that on average the decoder can not use long phrases for the translation of DTPs. Here, we hypothesize that employing phrase tables with higher recall improves the translation quality of DTPs.

The baseline TM uses the grow-diagonal heuristic for finalizing the word alignments. In this section, we modify the TM and use two other heuristics to vary the precision and recall of the word alignment and phrase table. The two alternative heuristics are the *Intersection* and the *Union* alignments.

The Intersection alignment is meant to maximize the quality of word alignment. For each intersection alignment, both source-to-target and target-to-source alignments should agree. Therefore, the quantity of word alignments is smaller while the quality of alignments are stronger (low recall and high precision). In contrast, the union alignment relies on either one of the two alignments and has a larger number of word alignments with lower quality.

The phrase extraction heuristics, starts from one word alignment with a moving window on the source and target languages. As the phrase gets expanded, new parallel phrases are recorded in the phrase table. The phrase expansion continues until it reaches an alignment that constrains further expansion. The phrase extraction restarts at the next word alignments. For this heuristic, each word alignment is a constraint that limits phrase extraction. Therefore, the intersection heuristic that generates few word alignments, results in a very large phrase table. In contrast, the union word alignment that has more word alignments, extracts less parallel phrases.

In an experiment, we used the TMs constructed from Union and Intersection alignments to translate a group of DTPs and easy phrases. Table 29 presents the results of this experiment.

We observe that translation quality of both types of focus phrases deteriorates when we

Phrase	Baseline	Intersect	Union
DTP	17.81	16.57	16.09
Easy	43.67	41.36	40.18

Table 29: MT evaluation for various alignment quality

use either one of the alternative alignments. The results are expected for the union phrase table that is a smaller table with lower recall. However, the results for the Intersection phrase table contradict our earlier hypothesis.

Our manual trace of DTPs' decodings show that due to the shortage of alignment constraints, the Intersection table is a very noisy phrase table. Over 50% of phrases are noisy and imbalanced. Imbalanced phrases are the ones that the ratio of source and target-language phrases length are too small or too big (e.g., two source-language words being mapped to seven target-language words). Tracing through the decoding shows that the model's noise offsets the gains of the model's higher recall and the final result is inferior to the baseline.

The Intersection table results shows that blind increment of the TM's recall with high sacrifice of the precision deteriorates the final results. In the following section, we will use additional knowledge to increase the recall in a more reserved procedure.

6.6 INTELLIGENT INCREMENT OF PHRASE TABLE'S RECALL

In the previous section, we observed that aggressive recall increase through loose heuristics, introduces new types of TM noise that in practice reduces the translation quality. In this section, we focus on increasing TM's recall while we avoid sacrificing model's precision. Our work is focused on phrase extraction. We do not modify phrase extraction's heuristics. However we constrain the phrase extraction by applying a sets of shallow linguistic constraints. These constraints help us to remove noisy phrases from the baseline TM and add new and

Task	Expanding the Phrase table's recall
Input	Baseline Phrase table, Intersection phrase table, corpus' word alignments
Output	A new phrase table

Table 30: An overview of expanding the phrase table recall

more accurate phrases to the model.

Here we introduce a framework to intelligently increase the coverage of the phrase table. Table 30 presents an overview of this method. In Section 6.2.3 we introduced a few criteria for measuring the quality of a phrase table entry. In this section, we extend and implement those criteria to filter phrases and improve the accuracy of the phrase table.

I. Short phrase constraint: All phrases with one or two source-language words and one target-language word are accepted.

II. Long phrase constraint: All phrases with six or more source-language and five or more target-language words are accepted.

III. Medium Phrases: Phrases which do not fit in the above short or long phrase definitions are considered medium. For medium phrases, we use the bridge analogy (Section 6.2.3) and the number quality of word alignments to estimate the strength of the phrase. To do so, we consider three factors:

a. The ratio of the number of word alignments to the number all words in the source and target sides of the phrase. With this ratio we b. The ratio of unaligned content words. We use a self-compiled set of function words to distinguish between the content and functions. We than calculate the ratio of the unaligned content words to the total number of content words. We do this calculation both for the source and target language side of the phrase.

c. The ratio of word alignments that come from the grow-diagonal heuristic.

We use two sets of soft and hard thresholds to define two sets of filtering criteria. The difference between these two set of constraints is the intensity of the filtering that they apply; i.e. soft constraint are more generous in keeping phrases in the model.

We start from the baseline TM that is constructed by the grow-diagonal heuristic. Since these phrases are meant to be reliable, we apply the soft constraints to only filter the most suspicious group of phrases. We then focus on the Intersection phrase table and look for new and reliable phrases. Since the Intersection table has more noisy phrases, we apply our hard thresholds to filter those phrases. After extracting phrases from both baseline and the Intersection table, we recompute the phrase translation probabilities with the new phrase set and assign the four translation parameters to each phrase.

We decide the value of soft and hard thresholds with a brute force-like mechanism. We start from the average values of the baseline and intersection tables. We then intuitively try different values till reaching a decent quality improvement in translation and coverage of the phrase table. We use a development set to test the resulting table's translation quality and coverage. At the end of this procedure we settle with a new phrase table that has a *combination* of subsets of the baseline and the Intersection table.

In the above combination method, we include a large number of phrases from the Intersection table. This is motivated form the vast number of phrases that we observe being eliminated because of the grow-diagonal heuristic. The heuristic expands word alignment combination with an aim of expanding the phrase length[3].

6.6.1 Experiments on DTPs

We first evaluate the coverage of the combination phrase table with a group of 451 unseen DTPs. Similar to experiments in Section 6.2.1, we evaluate the coverage in three ways: Source-language, Target-Language, Target constrained by the source (TCS). Tables 31, 32 and 33 present the results of this coverage study in comparison with the baseline and the Intersection table's coverage.

Finally we evaluate the translation quality using the new combination table. Table 34 compares the translation quality of 451 DTPs when we use the baseline, Intersection and our new combination phrase tables. We gain a decent quality improvement as we increase the precision and recall of the TM in our combination table. This one BLEU point improvement

[3]The newly added intersection alignments act as new phrase extraction constraints which limit the extraction of many short phrases.

Phr Table	1-gram	2-gram	3-gram	4-gram	5-gram	6-gram	7-gram
Baseline	83%	53%	26%	10%	4.3%	2.1%	0.8%
UB:Intersect	90%	63%	35%	16%	6.2%	3.5%	2.1%
UB:Corpus	98%	66%	39%	19%	8.6%	5.3%	3.5%
Comb	89.5%	59.5%	29.5%	12.9%	5.0%	2.5%	1.3%

Table 31: Comparison of the coverage of the source-language N-grams

is almost half-way between our baseline and the upper bound estimation that we had in Section 6.3.2. In these small number of experiments, we observe an almost linear relationship between the translation quality and the TM's recall.

Table 35 is an instance of the TM adaptation where intelligent combination of the baseline and intersection models help. The baseline grow-diagonal does not include the short phrase *tqryb wjhAt:bring views closer*, because the the grow-diagonal heuristic adds alignments which make this phrase inconsistent with the phrase extraction algorithm. Using the intersection heuristic, there are less alignments (only the word wjhAt is aligned to views). Our phrase extraction method which looks for balanced phrases with content words accepts this phrase as correct and includes it in the translation model.

Phr Table	1-gram	2-gram	3-gram	4-gram	5-gram	6-gram	7-gram
Baseline	84%	45%	26%	4.5%	1.4%	0.5%	0.1%
UB:Intersect	91%	64%	28%	9.8%	5.0%	1.5%	0.4%
UB:Corpus	95%	69%	35%	14%	6.4%	3.3%	2.1%
Comb	86%	53%	29%	6.9%	2.2%	0.7%	0.2%

Table 32: Comparison of the coverage of the target-language N-grams

Phr Table	1-gram	2-gram	3-gram	4-gram	5-gram	6-gram	7-gram
Baseline	74%	33%	11%	3.1%	1.1%	0.4%	0.001%
UB:Intersect	88%	58%	24%	8%	3%	0.8%	0.3%
Comb	76%	43%	14%	5.7%	1.5%	0.55%	0.15%

Table 33: Comparison of the coverage of the TCS N-grams

Phrase	Baseline	Intersect	Comb	UB
DTP	17.81	16.57	18.94	19.91

Table 34: MT evaluation for the Combination Phrase table

6.6.2 Sentence Level Translation

Previously we used the Intersection and our combination phrase tables as alternative TMs which have increased recall. We gained decent improvements in translation quality for DTPs when we used a set of constraint to increase the TM's recall. We would like to know how such TMs perform if we use them for sentence level translation.

We use a development set of sentences to tune the SMT engine with the Intersection and Combination phrase tables. We then use the re-tuned system to translate the LDC-1 corpus. Table 36 compares the translation quality of these two experiments against the

Baseline: ... paper on these issues of tqryb view ...
Ref.: ... a paper concerning these issues to bring their points of views closer ...
TM-adapt.: ... paper on these issues to bring views closer ...

Table 35: A sample improvement from TM adaptation

baseline. As expected the intersection table's performance is inferior to the baseline. Our combination phrase table shows an insignificant improvement over the baseline (based on the 0.5 significance criteria). We observe that our strategies for increasing the model's recall and precision is mainly successful for DTPs and on average, it does not get extended to other parts of the corpus.

Phrase	**Baseline**	**Intersect**	**Comb**
DTP	23.02	22.17	23.40

Table 36: Sentence Level MT evaluation for the Combination Phrase table

6.7 DISCUSSION

In the following we compare our LM and TM adaptation and the reasons behind their degree of success. Moreover, we discuss the shortcoming of PB-SMT and the requirements of TM modification beyond the scope of our adaptation work.

6.7.1 A comparison between LM and TM Adaptation

Both of the studied adaptation frameworks have shown some degree of success in improving the translation quality of the DTPs. However, the improvements from the TM adaptation is not as significant as the LM adaptation. There is a fundamental difference between our LM and TM adaptations. For the LM adaptation, we change the training data and construct a new model from scratch. In other words both the parameter variables and values are redefined. In all of the experimented TM adaptation methods, we modified the parameter variables, but we could not change the parameter values. This relates to the limits that we face in the long process of TM construction. Parameters of the translation models are estimates through a pipeline of statistical components. Unlike LM adaptation, our TM adaptation does not strongly change the baseline model.

An important premise of the PB-SMT is the usage of long parallel phrases that improve translation's fluency. Our coverage studies show that despite our efforts to increase the phrase table recall, a decent portion of long phrases that exist in the training data are missing. For example, in Tables 25, 26 and 27 we observe that for tri or four grams, more than half of the phrases that can potentially be useful for the translation are missing in the baseline grow-diagonal phrase table. We miss these phrases because of the problems related to the word alignment and phrase extraction heuristics. As a solution to such problems, we can use several levels of back-off strategies for the word alignment and phrase extraction when it comes to sparseness. Usage of alternative word alignment heuristics (e.g., intersection) and also our new phrase extraction constraints (Section 6.5 and 6.6) were steps towards this approach. However instead of applying those approaches to all DTPs, it might be better to apply them based on features of the DTP. For example, we can include Intersection table phrases for words and phrases which are absent in the baseline phrase table. A future work on this subject involves modifying some of the automatic word alignment constraints. For example, while processing some of the parallel sentences, we can add alignment constraints to make sure certain words that are needed in the translation task get aligned.

6.7.2 What needs to be done?

In PB-SMT, TM lacks any syntactic information and only relies on the surface lexical information for modeling. This lack of syntactic constraint during the word alignment and phrase extraction causes problems that are frequent for DTPs. Based on the grow-diagonal heuristic, phrases grow as long as there is no alignment that breaks the contiguously of the phrase. This syntactically unreserved growth, results in extraction of phrases that have random syntactic structure. A simple example is Arabic phrases that end with a preposition or the *al* determiner. The determiner which syntactically should be adjacent to its modifier word, loses the syntactic connection. Furthermore, phrases can move into different places while there is no connection between the determiner and its head word. As a result, this simple yet frequent phrase boundary problem causes the generation of many syntactically and semantically problematic segments.

Inclusion of syntactic information in the PB-SMT has been approached in different ways (Chiang, 2005), (Habash, 2006), (Hassan et al., 2008). Some of these studies (Habash, 2006) have shown that syntactic constraints improve certain types of PB-SMT's errors. However these new constraints, lower the TM's recall and robustness and in practice the lose offsets the gains and no significant improvement is gained. We suspect that a selective approach towards the usage of these syntactic constraints (e.g., only for DTPs) provide better results.

6.8 SUMMARY

In this Chapter we explored the idea of Translation Model (TM) adaptation in different directions. We first learned that the baseline TM is more sparse for phrases and words related to DTPs (comparing with easy phrases). We also learned the gap is not only quantitative, but also qualitative. This coverage gap is initiated by the sparseness in the parallel corpus and is widened through different steps of TM constructs (e.g., word alignment).

We estimated some upper bounds for the rooms that we have to improve the TM coverage and its subsequent translation quality. We then take a few steps to increase both recall and the precision of the phrase table construction. Our new phrase table *combines* the baseline (grow-diagonal) heuristic with a selective subset of the intersection heuristic. We achieve a half way improvement between our baseline and our estimated upper bounds both in terms of model's coverage and the subsequent translation quality.

7.0 ADAPTATION OF DECODING PARAMETERS

SMT engines have a set of decoding weights that decides the influence of the underlying components (models). A common practice is to use a set of fixed weights for translation of all sentences and phrases in a test set. In this chapter we explore the idea of adapting these weights for translation of individual sentences and phrases. Specifically, we focus on adapting the Language Model (LM) weight and answer the following questions:

I. Does modifying the LM-weight specifically for DTP translations enhance the translation quality?

II. Do DTPs share a common characteristic with respect to the LM's influence on their decoding?

III. Does adapting the LM weight for individual phrases or sentences enhance translation quality? What learning framework can be used to adapt the LM weight?

To answer these questions, we explore the following:

I. Usage of an adapted LM weight for the translation of all DTPs.

II. Estimation of a gold standard LM weight for individual phrases and sentences. Using the reference translations, we vary the LM weight to reach the weight that causes the best translation quality

III. Usage of LM weights that are automatically predicted by machine learning for the translation of individual phrases and sentences.

Similar to previous chapters, we modify parts of the SMT system (the LM weight) for translating specific phrases (e.g., DTPs). The phrase is translated by the modified LM weight, while the rest of its associated sentence is translated with the baseline LM weight. Moreover, the evaluation is also done both at the phrase and sentence levels.

Our experiments show modifying the language model weight for an individual or a set of DTPs has a positive effect on their translation quality. Also we construct a learning framework for predicting the LM weight. When we apply the predicted weights, we achieve promising improvements in the translation quality of DTPs. However, for easy phrases and regular sentences, we fail to achieve similar improvements.

Similar to previous chapters, the easy and difficult phrase labels are gold standard. Hence, we used reference translations to decide the difficulty level of these phrases. In Chapter 8, we reevaluate our weight learning experiments in a start-to-finish SMT pipeline. There, our difficulty classifier finds one DTP of the translation sentence, and LM weight modification will be applied to the translation of that DTP.

7.1 DECODING WEIGHTS IN PB-SMT

SMT decoders use a set of features to compute a decoding score for each translation hypothesis. These features are collected from different resources such as the translation and language models of the system, linguistic properties of the source-language text, etc. The decoding weights decide the influence of each feature. For example, for a PB-SMT decoder like Phramer, there are seven major decoding features with seven associated decoding weights (Section 3.1). The final decoding score is decided by a log-linear interpolation of these features with their weights. An example of such interpolation is the following formulation of PB-SMT decoding that is in the Phramer. There are four major probabilistic components:

I. Language Model (LM)

II. Translation Model (TM): As explained in Section 6.1, TM uses a vector of four parameters for each of the parallel phrases.

III. Distortion (d): Incorporates the information for word and phrase movements from the source to target-language.

IV. Word Penalty (WP): A parameter to force the decoder to generate more words.

Considering the four parameters of the TM, we totally have seven parameters which require

seven decoding weights.

$$e_{best} = argmax_e \prod_{i=1}^{n} \phi(\bar{f}_i|\bar{e}_i)^{\lambda_\phi} d(start_i - end_{i-1} - 1)^{\lambda_d} \prod_{i=1}^{|e|} p_{LM}(e_i|e_1...e_{i-1})^{\lambda_{LM}} |e|^{\lambda_{WP}} \quad (7.1)$$

In the above equation the (λ)s are the decoding weights for the LM, TM, WP and d. The translation model feature function(ϕ) holds four model parameters, and each parameter gets an entry in the λ_ϕ weight vector.

Generally, one set of feature weights is used for translation of all test sentences. Moreover, decoders use the same weights for translation of all parts (phrases) of the test sentences. The weight values are real numbers between zero and one that are initialized and optimized using a set of development parallel corpus.

7.1.1 Minimum Error Rate Training

The Minimum Error Rate Training (MERT) framework (Och, 2003) is a procedure to optimize the decoding weights. MERT iteratively tries a large collection of decoding weights on a development set. The weight set that maximizes the evaluation (e.g. BLEU) score is chosen as optimal. Since running the decoder is an expensive operation, the search to find the best decoding weights is performed for a set of n-best hypotheses. The decoder generates a set of feature values for each hypothesis. Then, various feature weights are explored on the n-best hypothesis and those that improve the translation quality are kept. After finding a set of optimized weights, the decoder runs again with the new optimized weights to collect new sets of hypotheses. In a new iteration, the search for optimized weights continues with the old and new sets of hypotheses. This process continues iteratively until reaching a weight set that no new hypothesis is generated for the entire development set. Assuming that the process does not sink into a local maximum, it stops when it reaches the best possible translation quality for the development set.

7.1.2 Extending the Tuning

Currently, SMT tuning takes place at the level of a development set of sentences. This means that the final tuned weights are averaged across different parts of the sentences of the development corpus. However, we believe this corpus-level tuning can be extended.

Each sentence has different characteristics that requires different influence from SMT components. For example, in PB-SMT, for sentences that the phrase table is sparse (many unknown words), the language model can glue the broken translated parts. Therefore, its stronger influence (weight) is helpful. In contrast, for sentences that the language model itself is sparse and noisy, a reduced LM influence might improve the sentence translation. These differences exist even within different parts of a sentence. In PB-SMT, the long distance dependencies between phrases are weak. Practically a sentence translation can be broken into a few long phrase translations. Similar to the sentences, each phrase might require its own decoding parameters. For the above reasons, we extend the system tuning beyond the development set.

In the following sections we explore two extensions:

a. Employing different decoding weights for different parts of sentences. (Section 7.2)

b. Employing different decoding weights for individual sentences or phrases. (Section 7.3)

7.2 TUNING THE DECODER FOR DTPS

We have been working on the hypothesis that dedicates a special decoding setup for DTPs (section 4.4). In Chapters 5 and 6, we investigated this hypothesis by modifying the language and translation models of the system. In this chapter, we continue by modifying the decoding weight for the baseline language model. In our framework, the decoder uses two different LM weights for the translation of the DTP and the rest of a sentence.

We start with the baseline set of decoding weights. These weights come from running MERT on a development set of sentences (explained as the baseline system in Chapter 3).

To find the LM weight for DTP translation, we switch to a development set of 100 DTPs. Starting with the baseline weight set, we rerun a modified implementation of the MERT. In this implementation, only the LM weight changes and other decoding features (e.g., lexical translation) are left intact.

We also modified the Phramer decoder to use different LM weights for different parts of the sentence. This modification allows us to use the new DTP-specific weight for translation of DTPs, while the rest of the sentence uses the baseline LM weight.

Table 37 compares the effect of weight tuning on the chosen difficult segments. We see that the LM weight is decreased from the baseline value of 0.37 to the optimized value of 0.28. The new optimized weight results in an increment of about 3 BLEU score for the development set. A smaller (1.5 score) improvement holds when we use the learned LM weight for the unseen test set of 301 difficult segments. This suggest that the influence of the baseline LM should be decreased for the difficult segments' translations.

	Wt. Val	**Dev**	**Test**
Baseline	0.37	19.93	15.96
Diff. Seg.	0.28	22.81	17.44

Table 37: Comparison of the usage of baseline and difficult-segment specific LM weights (BLEU evaluation at the *segment* level)

We next consider the effect of a segment level weight adaptation over the entire sentence. Table 38 compares three cases: translating the entire test set using the default weights, replacing the LM weight with the weight adapted specifically for the difficult segments for the entire sentence, and using a combination of the adapted weight on the difficult segments and the default weight on the rest of the sentence. As expected, there is little change in performance when applying the adapted weight indiscriminately over the entire test set. On the other hand, when we tailor the LM weight for only the difficult segment, there is a small gain in the overall BLEU score. These results indicate that i) "translation difficulty" is a useful proxy for identifying problematic spots for the language model; and ii) adapting weights at a sub-sentential level is a promising strategy.

	Wt. Val	**Test**
Baseline	0.37	22.56
Diff-Specific	0.28	22.69
Combined	0.28/0.37	23.17

Table 38: Comparison of the usage of baseline and difficult segment-specific LM weights (BLEU evaluation at the *sentence* level)

The above results show that DTPs as a set require less influence from the language model. This relates to the sparseness of the baseline translation and language models which prefer shorter translation of DTPs. Table 39 presents an example of the frequent short translation of DTPs and the effects of using a smaller LM weight which results in longer phrase generation.

Ref:... last may , king abdallah had met with the head ...
Baseline:... last may , met king abdallah president ...
New LM-Wt:... last may , had met king abdallah with the president ...

Table 39: A sample of under generation problem of DTPs

In the following sections, we will analyze these results in more details. Moreover we will confirm if these characteristics holds for the majority of individual DTPs in the test set. Finally, we would like to extend this weight adaptation to individual phrases (in contrast with a group of phrases).

In the following experiments, we refer to the above DTP-specific weight setting as the *DTP-Specific Baseline.*

7.3 MODIFYING INDIVIDUAL LM WEIGHTS

Our experiment in the previous section showed that modifying the LM weight for translation of different parts of a sentence can be rewarding. We also observed that DTPs as a group, prefer smaller LM influence (weight). However, we would like to know if this weight change applies to the majority of individual DTPs. In order to answer this question, we need to have an estimation of the gold standard LM weight for each phrase.

7.3.1 Estimation of Gold Standard LM Weights For Different Phrase Types

Given the baseline PB-SMT system, we would like to estimate the best LM decoding weight for translating an individual phrase. We use an oracle setup as following: For each phrase, we modify the LM weight and record its effect on the translation quality of the phrase. Our weight modification is discrete (+/-0.01) and we try 20 increment and decrements. We use the BLEU score (BLEU-3, BLEU-2 as back-off) for the evaluation. For each instance, we locate the LM weight which results in the best translation quality. We choose this LM weight as a gold standard weight for that specific phrase.

For weights that result in the same BLEU score, we chose the weight that is the closest to the baseline weight. In Section 7.4 we will use these gold standard weights as the training data for a learner that predicts the LM decoding weight.

We conduct this oracle study on two groups:

I. A set of 301 DTPs

II. A set of 253 Easy phrases

We would like to study the distribution of the gold standard weights and see if each group of text (e.g., DTPs) share any weight characteristics. Table 40 presents our experimental results. For each set, we record the percentage of the gold standard weight changes with respect to the baseline weight. We also obtain an oracle BLEU score for each set which gives us an estimate of the upper-bound for LM weight learning.

For a majority of phrases, the modified weights do not cause any change in translation quality (from the baseline) or the oracle simply prefers the baseline weight. When weight

LM Group	Oracle Weight Change (-/=/+) %	Baseline	Oracle
DTPS	36/49/15 %	15.96	21.93
Easy Phrs	11/72/16 %	46.09	47.23

Table 40: Comparison of the usage of baseline and oracle LM weights for DTPs

changes for DTPs, the oracle prefers smaller LM weights as the gold standard. This is equivalent to reduction of LM influence on the decoding. In contrast, the change distribution for the easy phrases is almost even and there less tendency towards changing the weight at all. This means that the baseline weight usually provides the best possible translation for easy phrases. These results are also reflected in the BLEU score changes. For DTPs, the improvements by the oracle weights are sharper than the easy phrases or the neutral sentences. We suspect that DTPs share a problem that weight reduction reduces the problem significantly. In order to confirm this claim, we manually examine the data.

7.3.2 Observations From the Effects of Weight Modification

We manually examined a group of 70 phrases to see the effects of weight changing. We did not observe frequent modification of content words after the weight adaptation. This is different from our language model adaptation (Chapter 5), where we gained translation improvements from better content word choices.

For DTPs' weight modification, majority of translation improvements are related to the under-generation problem. A common problem among DTPs' decoding is their short length. Due to sparseness of the translation and language models, DTPs are usually translated word-to-word. Some of the source language content words have phrase translations along with articles, auxiliary verbs and punctuations. In many cases, the decoder is hesitant to generate those longer phrase translations because the generation cost of those longer translations (estimated by the language model) are too high for the decoder.

In the case of complex verbs such as the past participle or the passive form, problems such as morphology and word alignment errors cause sparseness in the phrase table. Therefore, it is upon the language model to generate the auxiliary verbs. Reduction of the LM weight, reduces the cost of target-language generation in the decoding. This eases the generation of longer hypothesis with more function words, specially the auxiliary verbs. In Table 41 we present an example of the under-generation problem that is reduced with a smaller LM weight. Here the phrase table holds both translation of the source word (*visited* and *had visited*). After the generation of the unknown word (nykswn), the decoder chooses the path of shorter translation of the phrase. This happens because the sparse language model has a very low probability of the bigram translation (*had visited*), even-though its phrase translation parameters are not much different than the unigram translation

Ref: richard nixon had visited syria , which is still ...
Baseline: richard nykswn visited syria , which still ...
New LM-Wt: richard nykswn *had* visited syria , which *is* still ...

Table 41: A sample of under generation problem of DTPs

For easy phrases the oracle weight changes are less frequent than the DTPs. This infrequency gives us small amount of gold standard data. Also for easy phrases and also some of the neutral sentences, we do not observe a dominant pattern in the hypothesis improvements. Most improvements relate to punctuation and also infrequent cases of the over-generation problem.

The over-generation problem is common when the decoder generates extra content words. This problem is reduced when we increase the LM-weight which shortens the generation length. The following Table 42 demonstrates an example of the problem. Here the word *joint* is an over-generated term that is not needed. The new LM weight which is above the baseline weight, omits the extra generation.

The over-generation problem is infrequent among the DTPs, because for DTPs the translation is done by short phrases with one content word. Moreover, the DTP source words do not have long phrase-table entries with multiple content words.

Ref: kharazi said in a press conference with ...
Baseline: kharazi said in a *joint* press conference he held with ...
New LM-Wt: kharazi said in a press conference he held with ...

Table 42: A sample of the over generation problem

Task	Learning the decoding weight
Input	LM, Baseline LM weight, Translation of the phrase (DTP), Sys Info
Output	The new LM weight

Table 43: An overview of the decoding weight learner

7.4 LEARNING THE LM WEIGHT

In the previous section, the oracle scores showed that modifying the LM weight for individual DTPs is rewarding. Also we observed that for non-DTPs, the variations of LM weights is not frequent. Moreover, for most easy phrases, it is not possible to find a pattern of change in the decodings. Therefore, we focus our weight learning effort on the DTPs.

In order to train a learner we need labeled data. The oracle study in Section 7.3.1 provides us a set of DTPs with an estimation of their gold standard LM weights. We train a learning function that maps these phrases to their gold standard weight value. Table 43 presents an overview of this learner.

In the following we explore two learning frameworks: First, we considered a regression learner that maps a DTP to its LM weight value. For this approach we assume there is a direct relationship between a DTPs and the decoding weight and the weight value can be predicted directly from the features of DTP. It bypasses the translation quality as an important factor for distinguishing between different hypothesis. We will discuss this approach in Section 7.4.1.

In our second approach, we map the DTP to the decoding weight through the translation quality of the DTP. The decoding quality is caused by a certain LM weight. In this approach the input to the learning function consists of the DTP and the decoding weight, and the output is a representation of the translation quality. Furthermore, we search for the weight value which maximizes the output (the translation quality). We will discuss this approach in Section 7.4.2

7.4.1 Direct prediction of the LM weight

We explore modeling the relationship between a DTP (along its baseline translation) and a new LM weight. The assumption is that there is a function that maps a feature representation of the DTP to a real value LM weight. Regression learning provides a framework for estimating such function.

For a proof of concept, we actually train a Support Vector Regression model for learning the weight prediction function. We ported our difficulty classification features to construct the regression model. Features were extracted from the source-language DTP, its baseline decoding and the underlying SMT system. The gold standard data for training comes from the oracle weights that we compiled in Section 7.3.1.

We tested the regression model on a set of unseen DTPs and used the predicted weights to translate the DTPs. The resulting translation quality was not significantly different from using a DTP-specific weight (Section 7.2). Moreover, we observed that the predicted weights are uniformly close to the DTP-specific weight. In other words, the regression learner is only learning to predict weight values close to the average of the training weights.

We trace the failure to the way we have framed the learning problem. Moreover, the problem relates to our abstraction of the intermediate information that makes one decoding weight better than the other. The LM decoding weight that we used as the target value does not hold any information about its final MT effect. Conceptually, there is no direct relationship between a DTP and its decoding weight. What makes them related is the translation quality which makes one weight better than the other ones. Therefore, the reward function should include MT quality.

We also observe that the change intensity that the gold standard weight causes is also missing in our modeling. For example. the gold standard weight might cause significant quality boost for some phrases and very small or no improvement for other phrases (compared with the baseline weight)

An Alternative path is to learn and predict the actual translation quality (e.g., BLEU score) along with the gold standard weight. In order to prefer one LM weight over another one, we need to compare the final SMT effect of the weight. Hence we need to learn a function similar to the work of Albrecht and Hwa (2007) that maps a hypothesis to its translation quality. In the following section we train a new learner that chooses a LM weight by comparing different weights and their subsequent translations. This comparison is performed by a ranking model. Usage of ranking helps us to abstract away from the evaluation score (BLEU) and represent the translation quality in terms of quality ranks. However in theory there is no major advantage between using the regression and ranking. As we will present in Section 7.4.2.1, ranking models are a simple transformation of classification and regression models.

7.4.2 Ranking the LM weights

Our ranking model, ranks a set of hypothesis that are generated from different LM decoding weights. Two aspects of our ranking make it different from the ranking that takes place inside a statistical decoder:

I. In order to find the best decoding weight, we have to use new system and linguistic features. Injecting new features into a phrase-based decoder like Phramer or Moses is not a trivial task. Besides implementation complications, new decoding features expand the search space and slow down the decoding significantly. Similar to other SMT reranking frameworks (Och et al., 2004), a richer feature space can be used in a post-decoding reranking component. However, instead of reranking the hypothesis from the best-n hypothesis of the baseline system, we work with a set of 40 hypothesis that are generated by different LM weights.

II. Varying the LM weight adds new hypothesis that might not be available on the top-n

hypothesis list. For example, we used the baseline configuration of the decoder to generate the best-100 hypothesis for a DTP set (200 phrases). We observed that for 72 DTPS (36%), the hypothesis generated by the oracle-weight, does not exist in the best-100 list. In other words for *at least* 36% of DTPs, the weight modification, provides us decodings that are not present in the best-100 list.

We convert the oracle training to a ranked data set. In the gold standard data, the hypothesis that are generated by different training weights are sorted based on their translation quality. A feature representation of this sorted list is one training instance for our ranking model. Weights that generate the same hypothesis are merged into one item on the ranked list.

At the test time, all 40 variation of the LM weights are used to generate the hypothesis set. Finally, the ranking model will be used to rank the set and chose the best LM weight. There are phrases that weight modification does not change the baseline decoding. For them the baseline LM weight is assumed to be the best weight. Therefore, they are neither included in the training nor the testing of the ranking model.

7.4.2.1 The Ranking Model Given a featured representation of a set of instances (DTPs), we would like to rank them based on their translation quality. Support Vector Machine (SVM) has been widely used for ranking in Information Retrieval. The Ranking SVM algorithms maps the ranking task into a classification task for pairs of instances. The simplified procedure for such mapping is as following:

For two a pair of ranked instances that are ranked as (x1,x2), there is ranking function f such that: $f(x_1) > f(x_1)$.

The function can be a linear function of $f : (w, x)$ where w is a set of weights. Therefore, the ranked relation between (x_1, x_2), can be written as:

$(w, x1 - x2) > 0$

From the x_1, and x_2 and their associated y_1 and y_2 ranks, we can form new instances as following:

$(x_1 - x_2, z)$, where:

$$z = \begin{cases} +1 & if(y_1 > y_2) \\ -1 & if(y_2 > y_1) \end{cases}$$

Now we can construct a binary SVM classifier which learns and predicts the z value.

7.4.2.2 Training the ranking model Usage of features from the baseline translation and the source-language phrase do not provide enough information for modifying the decoding weight. Moreover, we require features that help the learner judging the translation quality for different decodings. Here we use a combination of some of our difficulty features and also features from Specia et al. (2009) and Albrecht and Hwa (2007).

Our ranking model uses the following group of features (totally 17 features):

I. N-gram matching with the baseline LM

II. N-gram probabilities from the baseline LM

III. target to source-language lexical ambiguity

IV. Average word movement from source to target-language

V. The ratio of punctuation and digit in the source and target phrases

VI. The average BLEU score: A BLEU evaluation of the hypothesis, using the other competing hypothesis as the pseudo references

VII. Bigram and trigram POS tags: In order to find patterns of compound nouns and verbs, we use a few POS tag patterns.

We use the 301 DTPs with their gold standard weights as the training data and 50 DTPs to tune the ranking model. Since we are training a ranking model, we need to use all variations of a DTP's translation. Thus, we use translation quality to rank all decodings of a DTP (by different LM weights).

We use an SVM-light implementation of the SVM-Rank algorithm. We choose the polynomial kernel and use a set of 50 instances to tune its single kernel parameter.

At test time, we translate each unseen DTP with different LM weights and construct a feature representation of each decoding. The trained ranker takes all such decoding as one instance and ranks them. The ranking algorithm outputs a score for each entry of a ranking task. We simply sort those entry scores and pick the top entry as the best decoding weight.

Finally, we translate the unseen set of 150 sentences, using the weights that the ranking model has picked.

7.4.2.3 Experimental Results Here we use 301 DTPs to train and 150 DTPs to test the ranking model and later evaluate the resulting translations of the 150 DTPs. We conduct a three-folds cross validation between these 301 and 150 DTPs and compute the final BLEU score as the average of the three folds. Table 44 presents the results of the average BLEU score along with the baseline and the DTP-specific tests on the same 450 DTPs.

The BLEU score improvements over the baseline and also the DTP-specific baseline are significant. An examination of results show the systematic improvements mainly relate to the under-generation problem (Section 7.3.2).

LM weight used	**Phr Eval**	**Sent Eval**
Baseline	15.02	23.04
DTP-Specific	16.06	23.30
Ranking	17.30	24.06

Table 44: LM weight ranking for DTPs

We should note that for about 40% of test DTPs, the decoding does not change when we modify the LM weight. However those phrases are still present in our evaluation. Using the gold standard rankings, we observe that for 19% of the ranked DTPs, the ranker chooses the gold standard rank. Moreover for 68% of DTPs, the ranker chooses weights that result in translation improvements against the baseline.

The following Table 45 presents a sample of translation improvement as a result of learning a new LM weight. Here weight learning targets the lexical translation problem of DTPS. This error relates to the sparseness of the language model which does not hold some of the required N-grams for the word *mediation*. The phrase table has a high probability for the right translation of the Arabic word *wsATp* to the English word mediation. However, the sparse language model which does not have the context parameters for the *mediation* translation, forces the decoder to choose the noisy translation (through). As the ranker

reduces the weight of the language model, the decoder gets a chance to ignore the LM and use the right translation. Moreover, the under-generation problem of the auxiliary very (is doing) is also solved by the new reduced LM weight.

Ref.: is doing mediation between the government in khartoum and the rebels in the south
Baseline: does " through between the government in khartoum and rebel in the south
New LM Wt.: is doing" mediation between the government in khartoum and rebel in the south

Table 45: A sample improvement of the lexical translation problem with the modified LM weight

7.5 EXPLORING A CUMULATIVE ADAPTATION

In Chapter 5, we introduced a series of methods to adapt the language model for each individual DTP. For each DTP, we collected the relevant subset of the training data and build a new language model which was used for translation. In our experiments we used the same baseline decoding weight for all the adapted language models.

In this chapter we explored the idea of using decoding weights different than the baseline weight. One might consider applying this weight variation to the adaptation of the language model. Here as a basic experiment, we would like to know if the baseline language model weight is the best choice for the adapted LMs. And if that is not the case, what is the characteristics of a better language model weight.

In this experiment we use a group of 378 DTPs and construct the adapted language model with the adaptation method-1 (Section 5.6.1). The decoding weight for the baseline system (which was used to extract DTPs) is 0.35. We then vary the decoding weight in small steps of 0.05 to study the effects of different weight values on the performance of the model adaptation. Figure 8 shows the variation of the MT quality as we modify the decoding weight for the adapted model. The red dash line highlight the position of the baseline LM weight (0.35). As we decrease the weight, the translation quality drops. In

contrast increasing the weight (up-to a certain level has a positive effect on the translation quality). This result is different than the case of group weight learning that we studied earlier in this chapter (Section 7.2). Earlier we observed that the system prefers to decrease the weight of the baseline language model for the translation of DTPs. In contrast, here the system prefers a stronger influence from the adapted language models. This experiment is a reminder of the *gold-in-sand* experiment in section 5.8 in which, we observed that the new adapted language model gets a higher ranking than the baseline language model. Here those ranks are substituted with stronger decoding weights.

The individualized weight learning framework that we introduced in Section 7.4.2 can not be applied to our LM adaptation framework. The weight learning framework relies on one language model to construct training data and during the training and testing queries the central language model for computing different feature functions. When we construct individual language models for each DTP, such learning becomes meaningless. As a result the combination of the LM adaptation and decoding weight adaptation (at least with our introduced methods) is bound to learning one LM weight for all adapted models.

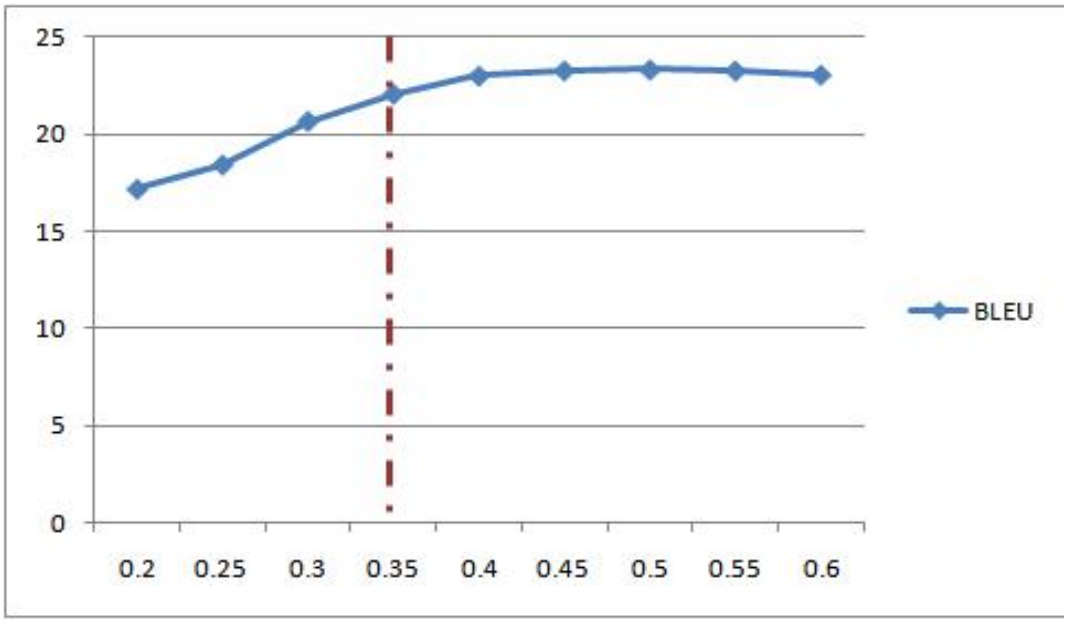

Figure 8: Effects of weight change on LM adaptation

7.6 WEIGHT ADAPTATION FOR SENTENCES

So far, we explored the idea of weight adaptation in a selective fashion. We selectively chose the DTP part of each sentence and modified its LM weight. As an extension of the idea, we would like to know how that plays out for sentence level translation. Also, we examine how our re-ranking framework performs for sentence level weight learning.

We follow the same setup as Section 7.3.1 to estimate an oracle LM weight for a set of sentences. These sentences are used both to estimate an upper bound for the weight learning and also as the gold standard data for weight prediction. When we switch from phrases to sentences, weight modification has a stronger influence on the resulting decoding and translation quality. Table 46 presents the results of oracle weight learning for a group of 400 sentences. We observe that more than 70% of sentences' decoding get modified with the new decoding weight.

Group	**Oracle Wt. Change** (-/=/+) %	**Baseline**	**Oracle**
400 Sentences	39/28/33 %	17.86	20.93

Table 46: Comparison of the usage of baseline and DTP-Specific LM weights

However this modification is almost uniform in terms of increment or decrement. Using the oracle weights has more than 3 BLEU score improvements at the sentence level evaluation. This is a significant improvement which is an estimate upper bound for the weight learning.

Weight Learning for Sentences

In order to learn and predict the LM weight, we port our SVM-Ranking framework to sentence level decoding. In our earlier experiments (Section 7.3.1), we compiled the oracle LM weights for the translation of 400 sentences. We split that set into training and test sets (300/100) and train the SVM weight ranker for sentences. The same feature sets and training and tuning procedures as the DTP experiments get applied.

Table 47 presents the results of the sentence level experiment. We observe there are

decent improvements in the translation quality. However, these improvements are modest. We observe that sentence level ranking does not have the same significant improvement effect as the DTP ranking had. These results are validated when we evaluate the ranking accuracy with the gold standard weights. We observe that more than 50% of the new weights deteriorate the translation quality.

As the translation text becomes longer, it becomes more difficult to find a pattern of change associated with the weight modifications. For long sentences, several parts of the sentences are affected by the new weight and each part has its own characteristics. Also there are more cases of random small changes that influence the evaluation scores. All these makes the modeling and learning of the weights more difficult at the sentence level.

In Chapter 8, we compare these results against a start-to-finish experiment that weight adaptation is applied selectively to the DTPs.

LM weight used	**Sent Eval**
Baseline	18.13
Ranking	18.67

Table 47: LM weight ranking at the sentence level

We performed a manual examination of feature values for our training data. For doing so, we substituted the BLEU score of each instance as the class value and performed a correlation analysis of each feature. We observe that unlike the earlier DTP ranker, the correlations between our features and the class value is very low. We also observe that when it comes to the translation of long sentences, most of our effective features loose their discriminative power. For example, the variation of the language model probability features at the sentence level does not show any relationship with the translation quality. We conclude that for porting the weight learner to the sentence level translation, new feature analysis and different ways of normalization are needed.

7.7 SUMMARY

We explored two modifications in the decoding:
I. Employing different decoding weights for different parts of sentences.
II. Employing different decoding weights for individual sentences or phrases.

Our focus was the Language Model (LM) weight. While working on (II), we explored the adaptation of LM weight for different types of phrases. In summary, we learned that:
I. Difficult to translate phrases share problems (e.g., compound verb translation) that can be reduced by modification of the LM weight. In contrast, we could not find a common pattern of problems in easy phrases that can be reduced by weight modification.
II. LM Weights can be adapted for each DTP by machine learning. We trained a learner that its weight predictions, improves the translation quality of DTPs.

We first used MERT to tune the PB-SMT system with a development set that was all DTPs. We learned that DTPs (as a group) prefer a lower LM influence for decoding. Furthermore, we conducted an oracle study to find the best LM weight for the translation of individual DTPs. We observed that majority of individual DTPs choose LM weights that are below the baseline weights of the system. This indicates that the tendency for a less LM influence is a common characteristic among DTPs. Also manual examination of a subset of DTPs showed that weight decrease boosts the generation power of decoder. This reduces the problem of translating compound verbs and nouns.

We successfully trained a ranking model that predicts a new LM weight for DTPs. However, our application of the same ranking model to the translation of long sentences was not as successful. We suspect that for modeling the sentence level weight prediction, we need to expand the feature extraction both quantitatively and qualitatively.

8.0 EXTENDED EXPERIMENTS

In previous chapters we experimented with gold standard focus phrases (e.g., DTPs). We evaluated several types of system customization on these phrases and also in the context of their associated sentences. In this chapter we extend our experiments to answer the following three questions:

I. Do our proposed difficulty detection and reduction scale up for a larger SMT system?

II. Does our framework of DTP decomposition and the consequent system customizations, enhance the translation quality?

To answer the first question, we use a larger parallel corpus (LDC-2: 50 million words) to train a medium PB-SMT system. Since our difficulty detection and reduction framework is system-dependent, we need to reconstruct the framework for this new system. Therefore, we compile a new set of easy and difficult phrases and train a new difficulty classifier for the medium system. With this new system and difficulty framework, we re-apply two sets of system customization experiments: (I) Language Model Adaptation (II) Adaptation of the decoding weights.

Most of our experimental results on the medium system are consistent with the ones on the small system (Chapters 5 and 7). However, the scale of quality enhancements are smaller for the medium system. This relates to the fact that the baseline medium system's models are less sparse, and less noisy and practically translation difficulty is less frequent.

To answer the second question, we conduct a start-to-finish experiment with our entire architecture. We use the difficulty classifier as part of the PB-SMT pipeline to find the most difficult phrase of each translation sentence. We then apply the system customization to

the highlighted phrase as the rest of the sentence gets translated by the baseline system. Specifically we apply two of our most successful system customizations: (I) Adaptation of Language Model (II) Adaptation of decoding weight.

In our SMT pipeline, the difficulty classifier introduces its own noise into the customization of the system. For example, for a subset of sentences, an easy phrase gets highlighted for customization. Some of our customization methods (e.g., LM adaptation) are ineffective or even deteriorate the translation quality for easy phrases. As a result the classification noise offsets parts of the enhancements.

8.1 SCALING UP THE FRAMEWORK

All of our experiments in previous chapters were conducted on a PB-SMT system which is trained on a small amount of parallel corpus (1 million words). We would like to know how much our framework which is based on the decomposition of the DTPs and customization of the MT system, scales up. We also would like to know if we achieve similar types of translation enhancements (if any) when we apply our methods on the medium system.

8.1.1 The medium PB-SMT system

We use a parallel corpus[1] of 50 million words to train the medium PB-SMT engine. We follow the same preprocessing and training and tuning steps of the small system. The same development set is used by the Minimum Error Training (MERT) to tune the system. Since we use a larger corpus to train the medium system, its models are expected to be less sparse. However, translation difficulty of a phrase is relative with respect to other phrases in a sentence and corpus. Therefore, we still have a set of phrases that are more difficult to translate than the others for the medium system.

[1]This is the LDC-MED corpus, introduced in Section 3.3.3

8.1.2 Difficulty labeling for the medium system

For easy-difficult phrase labeling, we start from the same 3615 phrases we used for the small system. Using the medium system, we receive different translations for these phrases, which changes their quality and, consequently, their labeling. Since we choose the most difficult and easy phrases for each sentence, the boundary of these phrases can also be different than what we had for the small system. The medium system's labeling of 3615 phrases has 81% agreement with the small system. However, the choice of the sentences' most difficult and easy phrases has a lower agreement rate of 59%. Table 48 presents a comparison of the difficulty labels of the two systems. We observe that both systems chose almost the same number of phrases for one of the two classes. However, a number (about 20%) of phrases that are difficult for the small system, are not difficult for the medium system. The ratio shows that the cause of difficulty is beyond data sparseness. This is consistent with our analysis of the difficulty reasons in Chapter 4.

Exp.	**Small Sys**	**Medium Sys**
Labeled Phrases	3215 (88%)	3161 (87%)
Difficult Labels	1834 (57%)	1456 (46%)

Table 48: LM Adaptation on the Small and Medium Systems

8.1.3 System Customization

The major difference between the language model adaptation for the small and medium system is the scale of the data. The adapted language models for the medium system hold over 5 million words which are larger than the baseline language model of the small system. Therefore, concerns such as the sparseness of the model for all function word N-grams (Section 5.6), are reduced here.

Both of our customizations use algorithms with an $O(n)$ computational cost where n is the size of the training data. Therefore, the computational cost increases linearly as we switch from the small to medium systems.

8.1.4 Experiments

For testing the LM adaptation on the medium system, we experimented with a corpus of 544 sentences that their most difficult phrases are highlighted. Table 49 presents the adaptation results both by the small system and medium systems. The improvements are smaller for the medium system. The baseline language model for the medium system is richer and generally language model contributes less to the difficulty problems than the small systems.

Exp	Small Sys.	Medium Sys.
Baseline	16.96	18.58
LM Adaptation	21.12	22.16

Table 49: LM Adaptation on the Small and Medium Systems

The Figure 9 presents a comparison of LM adaptation (method-1) with language models trained on larger data for the small and medium systems. An important difference is medium system's slow pace of improvement from the larger data. This relates to the richness of the baseline model for the medium system which is not affect much by the new data expansions.

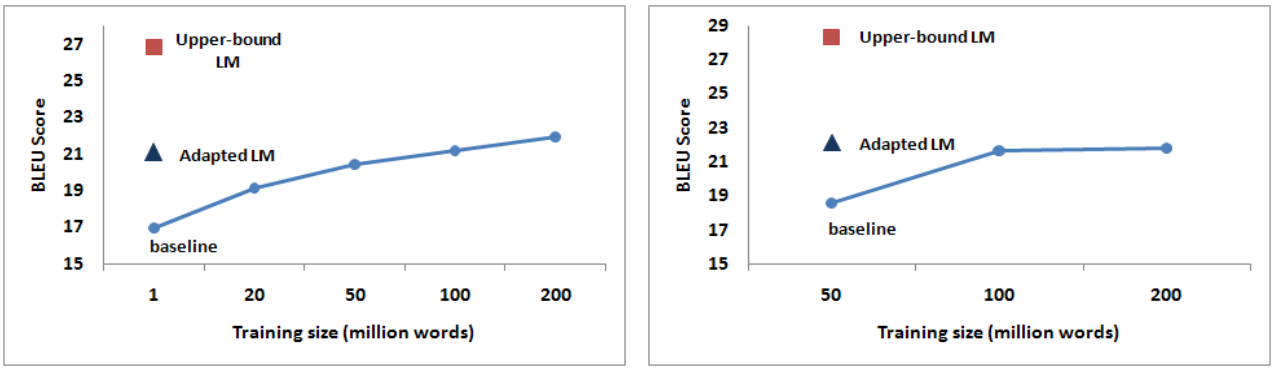

Figure 9: A comparison of model adaptation and training data expansion for systems that are trained on Small (Left) and Medium (Right) size parallel corpora

The evaluation of the adaptation of decoding weights is different than the LM adaptation. A large subset of DTPs are used for training our LM weight predictor that uses Support

Vector Rankings. (similar to Section 7.4.2). Similar to the small system, the baseline LM is used again for the new decoding weights. For evaluation of the LM weight adaptation, we use the remaining 150 DTPs along with their associated sentences. Table 50 presents a comparison of LM weight adaptation for the small and medium systems.

Exp.	**Small Sys.**	**Medium Sys.**
Baseline	15.02	20.13
Adaptation of LM Wt.	17.81	22.65

Table 50: Adaptation of LM weight on the Small and Medium Systems

Here both systems have almost same level of improvement. Moreover, the gold standard weights and also the predicted weights follow similar patterns of increments and decrements. This indicates the adaptation of the decoding weight is an effective method across systems with different sizes of training data.

8.2 START-TO-FINISH EXPERIMENTS

So far we have been using gold standard DTPs and easy phrase in our decomposition architecture (Figure 10). In this chapter we use the difficulty classifier to locate the most difficult phrase of each translation sentence. Moreover, the DTP handler conducts system's customization for the highlighted DTP. Finally, the sentence is translated with the customized system (for the DTP) and the baseline system for the non-DTP parts.

For the translation of each sentence, we apply the following procedure:

I. Compile the set of all source-language phrases. To reduce the scale and keep the procedure similar to our gold standard labeling, we only consider phrases that have 5 to 15 words and have a contiguous baseline translation.

II. Extract classification features for all phrases and their translations.

III. Classify all phrases of a sentence and use the classifier's score to choose the most difficult phrase.

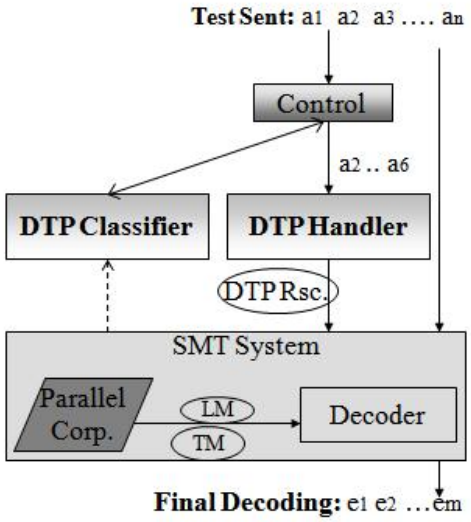

Figure 10: Start-to-finish: classifier finds the DTP, and handler modifies the SMT system

IV. Customize the SMT system for the most difficult phrase.

V. Translate the sentence by using the customized system for the difficult phrase and the baseline setup for the rest of the sentence.

8.2.1 Difficulty Classifier

We conduct the start-to-finish experiments on both small and medium systems. Each system has its own difficulty classifier. For each sentence the controller extracts an average group of 43 phrases. It queries the classifier for difficulty classification of these phrases and based on the classification score chooses the most difficult phrase of the sentence.

8.2.2 Experiments

We use two of our system customization methods in these experiments:

I. LM Adaptation

II. Adaptation of the LM Weight

For the LM adaptation, we use the string-matching method to locate the relevant sen-

tences and then, customize the model by our first adaptation method. This combination of methods had shown a better performance in our earlier experiments (Chapter 5). We conduct our experiments on the unseen NIST-2 test set (661 sentences). The evaluation is done at the sentence level. Table 51 present the results of the start-to-finish experiment.

Exp	**Small Sys**	**Medium Sys**
Baseline	18.09	22.51
LM Adaptation	19.06	**23.55**
Adaptation of LM Wt	**19.51**	23.32

Table 51: Adaptation of LM weight on the Small and Medium Systems

The improvements from the baseline are similar for both systems. For the small system, the adaptation of the LM weights is showing a stronger performance. This is related to the classifier's noise. For LM adaptation, classifier's false positives results in adaptation of easy phrases. We have seen that LM adaptation for easy phrases harms the translation quality (Section 5.7.4). Such problem does not happen in decoding weight adaptation. Even if we chose the wrong phrase as DTP, the weight adaptation is not harmful. Therefore, in a similar set up of the two adaptations, the classification noise hurts the LM adaptation more.

9.0 RELATED WORK

In this chapter we review some of the major works related to this dissertation.

Translation difficulty has been studied in the human translation community where translation challenge matters for translators' training. Hale and Campbell (2002) present an empirical study on finding most frequent syntactic and lexical patterns that cause translation difficulty. Their study is conducted on English as the source and 3 other languages (Arabic, Spanish and Vietnamese) as the target languages. In addition to the bilingual aspects of translation difficulty, they examine the universality of the difficulty concept. We consider some of the iconic bilingual differences (e.g., reverse noun adjective ordering of Arabic-English). However, our general view of the translation difficulty is a system-based view. We address difficulty problems by customizing the translation of difficult parts.

Translation difficulty has also been studied by the SMT community as a framework for semi-automatic detection of SMT errors. Uchimoto et al. (2005) translate the text in two directions and highlight the segments with poor double translation as the difficult to translate segments. Kirchhoff et al. (2007) present a set of criteria in the translation document (e.g. genre, percentage of named entities, etc.) and compare the final effects of their variations on translation quality. The detection of translation problems are done by a combination of automatic and manual methods. In contrast, our definition of translation difficulty is closely associated with the BLEU score and is independent of any linguistic criteria.

We now look into different areas of SMT to review works which have some overlap and relevancy to our study. We review three bodies of work: (i.) Automatic prediction of translation quality (ii.) MT system adaptation (iii.) Other relevant SMT developments.

9.1 AUTOMATIC PREDICTION OF TRANSLATION QUALITY

Our work on translation difficulty lies in a larger body of works that we refer to as *Automatic prediction of translation quality*. Parallel to the development of automatic MT evaluation, researchers have been studying the problem of statistical modeling of various aspects of translation quality. A major resource which influences such modelings is the presence of reference translations in prediction of translation quality.

9.1.1 Confidence Estimation

One set of research has been done under a constraint of not having access to reference translations. This is the Confidence Estimation for Machine Translation which is simply system's judgements on its own performance. The confidence measure is a score for N-grams (substrings of the hypothesis) which are generated by a MT system. Confidence estimation is performed at the word level (unigram outputs)(Blatz et al., 2003) or phrase level (Zens and Ney, 2006). The measure is based on feature values extracted from the underlying SMT system and also the training data of the SMT system. One of the most important confidence estimation features is the word posterior probability. This feature which can be computed from translation models shows how confident is the system about outputting a certain word at a certain position of the sentence. Mathematical normalization is used to constrain the score within a certain boundary.

Confidence measures have been used as part of the Computer Assisted Translation (CAT) systems where human's translation input gets integrated with MT (Ueffing and Ney (2005). Also confidence measure can improve the performance of the SMT system if they are used as new features in reranking frameworks (Zens and Ney, 2006).

9.1.2 Prediction of Human Judgements on MT

As part of the annual NIST evaluations, human translators are employed to judge the output of different MT systems. The results of these human judgements has become a useful resource for modeling translation quality. These are discreet valued judgements of MT's *fluency* and

adequacy and are generally used as a the gold-standard data for human evaluation of MT. This dataset has been used a way of statistical modeling of translation quality (measured by human judgements).

A successful example of such modeling is the work of Albrecht and Hwa (2008). They model the human judgements by regression learning and train support vector regression models. For this regression learner the MT decodings are the input and an aggregated fluency and adequacy scores are the output. In the human judgement data, there is little information about the underlying MT systems. As a result, the features are mainly limited to the linguistic processing of the target language translation hypothesis. This study also highlights some of the biases that exist in human judgement scores. An interesting result from the prediction of the human judgements is that high judgement score do not necessarily mean high translation quality.

9.1.3 Learning the Automatic Evaluation Scores

There has been some efforts on modeling the translation quality by a reference-free prediction of evaluation scores at different levels (sentence, document, etc.). Specia et al. (2009) use regression learning to predict a quality score for MT output sentences. Their work includes a comprehensive examination of features for such modeling.

The recent work of Soricut and Echihabi (2010), estimates the translation quality at the document level. Motivated by commercial needs, this reference-free quality estimation provides a quality ranking of different translated documents. With some overlaps with our difficulty framework, they use regression learning and use the BLEU score of documents as the gold-standard. An interesting consistent result across different experimental setup is that direct prediction of the BLEU score is not accurate while the score rankings have high accuracies.

There are different levels of overlap between our work and the works that we discussed in the above. The common denominator of all these works is the prediction of translation quality without having access to the reference translations. The major elements that distinguishes our work from the above works is: (1) Phrase-level emphasis: Our hypothesis is the variation

of translation difficulty in different segments of the sentence. Thus, we model difficulty at the phrase level. Our goal in detection of translation difficulty is not evaluation of the system. In contrast, we use it as a mean to find the right way of system modification.

9.2 MODEL ADAPTATION

Model adaptation has been applied extensively into different computational linguistics tasks including SMT. An important step in model adaptation is the method of finding the relevant subset of the training data. In SMT a range of simple similarity-based methods to sophisticated search methods have been tried. Brown (2008) uses simple similarity metrics to choose the relevant subset of training documents. Tam et al. (2007) use Information Retrieval (IR) metrics of Latent Semantic Analysis to weigh different model parameters. Also, Hildebrand et al. (2005) use IR to find the relevant subset of training documents for adapting the translation model.

In our work, the search for the relevant training data was at the sentence level. We constructed adapted models for individual DTPs and similarity was defined at a fine grained sentence level. This data selection based on the translation difficulty information has some overlap with the work of Mandal et al. (2008). They conduct an active-learning based data selection for those sentences that SMT systems disagree in their translation (an indication of translation difficulty). However, their data selection and also weighing methods do not take the characteristics of the translation sentence into account.

We now shift our focus on the previous works on the actual adaptation methods for the two major SMT models:

9.2.1 Language Model Adaptation

Language Model Adaptation has been studied in the speech recognition and speech translation communities . In many of those studies, relevancy is enforced at the document level by choosing in-domain documents. For example, to adapt an Automatic Speech Recognizer

(ASR) to the sport domain, only sport related documents are used for model training (Kim and Khudanpur, 2004; Tam et al., 2007). The adaptation is applied through modification of model parameters through different means. Tam et al. (2007) compute a cross-lingual posterior factor, using Latent Semantic Analysis and use the factor to weigh different language model parameters. Koehn and Schroeder (2007) use an interpolation of different language models (in different domains) to apply domain adaptation. Snover et al. (2008) build a bias language model for a set of translation sentences (a document) and uses an interpolation of the baseline and the bias models. Finally, Zhang (2008) presents different language modeling techniques based on syntactic and structural information of the text. In order to compare the performance of the alternative language models, he presents the novel *gold-in-sand* model ranking procedure that we also borrowed in the evaluation of our adapted language models.

The major difference between the previous adaptation attempts and our work was the framework of one adapted model per translation task. This framework radically changes the parameter space of the baseline language model. Also our usage of the adapted language model is novel: we selectively use the adapted model only for the DTPs.

9.2.2 Translation Model Adaptation

Comparing with other types of model adaptation, translation model adaptation is an under-explored area. This is related to the complex nature of translation modeling in SMT which requires a computationally expensive pipeline of statistical methods.

Hildebrand et al. (2005) perform a one-model-per-test-set adaptation of the translation model. They employs information retrieval to find the relevant parts of the parallel corpus and build an adapted translation model from those relevant sentences. The adaptation takes place once for the entire test set. Snover et al. (2008) expand the translation model using new noisy (comparable) corpora. The new extracted phrases from this noisy data are considered as bias rules with small uniform translation probabilities. Special decoding features highlight the difference between the bias and regular entries of the translation model.

Civera and Juan (2007) adapt the SMT system using a mixture translation model. Their mixture model is an extension of the HMM word alignment model, aiming at increasing the

contextual information for the alignment of the polysemic words. A major challenge in such work is the assessment of the new adapted model within a Phrase-based SMT system. Similar to many other modification of the word alignment models, the new model do not necessarily results in major modification of the phrase table and consequently the resulting decodings.

The novelty of our approach to the translation modeling adaptation is the step and also the heuristics that we use to apply model adaptation. We do not work on modification of the underlying data and instead choose a step that strongly influences the resulting translation model: The phrase extraction procedure.

A major difference between our and the previous model adaptation methods is the selectiveness of our model adaptation based on the difficulty information. We locate the areas that general model is noisy and apply the adaptation only to those areas. These selected difficult to translate phrases are the parts that can potentially benefit from model adaptation. We empirically show that the non-DTP areas do not benefit from the model customization and in some cases translation quality is even deteriorated.

9.3 OTHER RELEVANT SMT WORK

In addition to the above major components, this dissertation has partial overlaps with a few other areas of SMT research:

9.3.1 System Combination and Modification

The idea of decomposing the translation sentence and reattaching phrasal decodings has been also studied in the Multi Engine MT (MEMT) community. Among the large body of MEMT work, Mellebeek et al. (2006) choose syntactically meaningful segments to decompose the sentences. In our approach we do not consider any syntactic constraint. However, we constrain the usage of different system configurations based on the translation difficulty of the phrase.

Another relevant area of work is the training data subsampling. Johnson et al. (2007) use the chi-squared test to validate the accuracy of the phrase table entries. They are able to reduce the size of the phrase table to 10% of its original size without a major loss of translation quality. It is not clear what percentage of the original training data is required to construct the reduced translation model. However their work confirms that training data can be used more efficiently. Ueffing et al. (2007) also apply transduction learning to bootstrap new training sentences for SMT. New source-language sentences are translated via a baseline SMT system. Confidence estimation and model parameters are used for deciding to keep the sentence (and its decodings) in the new round of training. Our work follows a similar idea for altering the training data, but instead of adding additional data, we filter out and reweigh training data based on the relevancy to the translation task.

In our framework we improve translation of a special phrase (DTP) in the context of a sentence translation. This framework overlaps with the work of Koehn and Knight (2003) in which, they present a frame work of isolated translation of noun phrases and re-combining the phrase translation with the rest of the sentence. In contrast, our phrase translation takes place in the context of the entire sentence but with different language model. We still benefit from the full sentence context which reduces the translation error.

An important component of our work was the usage of rank learning to decide the right decoding weights for DTPs. Our work is relevant to the work of Duh (2008) on the usage of ranking in MT evaluation. Although we use similar learning frameworks, we pursue different goals from such ranking models.

One premise of our proposed customization was to reduce the translation ambiguity by tuning the language model to the proper domain. Our work has a slight overlap with Word Sense Disambiguation (WSD) in the context of MT (Carpuat and Wu, 2007). However we do not use any target-language WSD technique and our disambiguation gains are simply based on projection of word senses across languages.

10.0 CONCLUSION AND FUTURE WORK

In this project, we enhanced the translation quality of two MT system by focusing on input phrases which are difficult to translate. We provided a framework for detecting such phrases and further modified the systems to reduce some of their common problems.

We achieved translation enhancements in locating and reducing the following types of translation problems:

I. Word ordering of compound phrases: We achieved improvements by adapting the language model to translation task. This allowed us to improve the influence of the infrequent yet relevant N-grams in the model. It also helped us to disambiguate certain target-language words.

II. Under-generation problems: A common problem among difficult phrases is the under generation of function words such as auxiliary verbs, punctuation, etc. By adapting the language model weight, we reduced this problem.

III. Translation Model Sparseness: We observed the translation model sparseness is effectively larger for difficult phrases. The heuristics behind translation model construction We reduce this sparseness by an intelligent increase of the translation model's recall, which minimizes the sacrifice of the precision too.

IV. Improving the translation of Non-difficult phrases: We empirically showed improving the translation of difficult phrases significantly improves the translation quality of their neighboring non-difficult phrases.

In addition to the above experimental enhancements, we implemented the following SMT resources:

I. A framework for difficulty labeling of phrases and sentences.

II. Customization of the Phramer PB-SMT decoder to accept alternative resources and

weights for translation of certain (focus) phrases.

Our work contributes to the enhancement of the machine translation technology. Moreover, we extend the application of machine learning to estimating the difficulty of language and the quality of language generation.

10.1 APPLICATION

In this section we discuss the potential applications of this thesis beyond the immediate research community. We discuss its application within the Machine Translation and Human Language Technology industries and also its broader impact. We have shown that our framework of automatic detection of translation problems (difficulties) and customizing the MT system based on those information improve translation quality in a different settings. Here we tested the idea on a PB-SMT system, but most of our methods are applicable to other MT paradigms, not to mention SMT system. An important application of our framework is within a Computer Aided Translation (CAT) system where MT and human translations are employed concurrently (Elliott, 2006). These systems are expected to optimize the usage of human labor in translation by passing only those translation tasks which are difficult to translate for the MT system. Our framework can be used in two ways in such systems: (I) Locating the difficult to translate segments. (II) Iterative modification and adaptation of the baseline MT system to improve its quality and reduce the translation difficulty. Moreover, the difficulty information can be used not only to optimize human input for translation, but also to optimize the creation of new training data for the MT system.

Many components and ideas from this work can be ported into other tasks in human language technologies. Application systems such as Question Answering, Information Extraction, Speech Processing can benefit from automatic detection of difficult subtasks and the consequent system adaptations. More broader, the subject of difficulty in performing intelligent tasks in an open research questions in several disciplines such as language acquisition, Psycholinguistics.(Mitchell et al., 2010). On the system side of the research we rely on intuition and human-related features to construct our systems. Such relationship can be-

come bidirectional, i.e. the intelligent system's features for dealing with language difficulty be used to learn about human and cognitive aspects of language difficulty.

10.2 FUTURE WORK

System customization for human language technologies is an area of interest. As we showed in this thesis, the degree of difficulty is a useful feature for customization methods. We plan to continue our exploration of this area in the following directions:

10.2.1 Going Beyond PB-SMT

In this thesis, we studied translation difficulty in the framework of a Phrase-based SMT system. Many of our classification features and customization methods were related (but not limited) to the architecture of this system. As an extension, we would like to go beyond PB-SMT and look into other variants of SMT such as Syntax-based systems (e.g., Hiero). Such an extension might require a different framework of modeling difficulty. For example, usage of BLEU score or N-gram features might not be as effective as they were for PB-SMT. To reduce such concern, difficulty labeling should be done with an integrated usage of other evaluation metrics.

10.2.2 Noise Reduction in Labeling

The evaluation of metrics' features influence difficulty labeling. Some MT outputs with decent quality fail to meet the evaluation criteria of BLEU and get a low score. We touched on some of these problems in Section 4.7 when we used alternative metrics (e.g., METEOR) to label phrases. We observed that some MT outputs with decent semantic matching were labeled easy while the exact matching BLEU had labeled them as difficult. However, this problem is beyond word form and semantics. We observe that problems such as punctuation mismatch strongly influence the difficulty labeling. An automatic review procedure which reduces these data points in our DTP labeling is expected to boost our classifier and its

subsequent usages for system customization. In such a procedure, we need to introduce few heuristics to modify the evaluation criteria which also influence our labeling.

10.2.3 Going Beyond BLEU

In addition to the above problems with labeling noise, we need to incorporate different types of criteria into difficulty labeling. The BLEU metric evaluates fluency based on the length of matching n-grams. There are wide range of metrics which use other criteria such as matching the semantics, syntax, etc. We believe an extension of our difficulty analysis should use a voting mechanism to take a variety of these metrics into account.

10.2.4 Extended Adaptation of decoding Weights

In this thesis, we obtained strong improvements in translation quality by adapting the language model's decoding weight. This approach can be extended to learning and predicting more than one decoding weight. An extension requires a more sophisticated oracle for finding the gold standard multi-variate vector of weights. Furthermore, it requires a multi-dimensional search for the optimal weight vector.

10.2.5 Hybrid Model Adaptation

Our customization methods can be applied concurrently. Both LM adaptation and the weight adaptation are independent of the translation model. Hypothetically a hybrid joint customization (TM+LM adaptation, TM+weight adaptation) should apply preserve the positive aspects of both adaptations.

BIBLIOGRAPHY

Joshua Albrecht and Rebecca Hwa. Regression for sentence-level mt evaluation with pseudo references. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 296–303, Prague, Czech Republic, June 2007. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/P/P07/P07-0038.

Joshua Albrecht and Rebecca Hwa. Regression for machine translation evaluation at the sentence level. *Machine Translation*, 22(1-2), 2008.

J. Allan, J. Callan, K. Collins-Thompson, F. Feng, D. Fisher, J. Lafferty, L. Larkey, T. N. Truong, P. Ogilvie, L. Si, T. Strohman, H. Turtle, and C. Zhai. The lemur toolkit for language modeling and information retrieval. Technical report, Carnegie Mellon University and University of Massachusetts-Amherst, 2003. URL http://www.lemurproject.org/.

Daniel M. Bikel. A distributional analysis of a lexicalized statistical parsing model. In Dekang Lin and Dekai Wu, editors, *Proceedings of EMNLP 2004*, pages 182–189, Barcelona, Spain, July 2004. Association for Computational Linguistics.

John Blatz, Erin Fitzgerald, George Foster, Simona Gan drabur, Cyril Goutte, Alex Kulesza, Alberto Sanchis, and Nicola Ueffing. Confidence estimation for machine translation. Technical report, Center for Language and Speech Processing, Johns Hopkins University, Baltimore, Summer Workshop 2003.

Thorsten Brants, Ashok C. Popat, Peng Xu, Franz J. Och, and Jeffrey Dean. Large language models in machine translation. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 858–867, 2007. URL http://www.aclweb.org/anthology/D/D07/D07-1090.

Ralf D. Brown. Exploiting document-level context for data-driven machine translation. In *Proceedings of the Eighth Conference of the Association for Machine Translation in the Americas (AMTA-08)*, Waikiki, USA, 2008.

Chris Callison-Burch, Miles Osborne, and Philipp Koehn. Re-evaluation the role of bleu in machine translation research. In *Proceedings of the EACL 2006, 11st Conference of the European Chapter of the Association for Computational Linguistics, April 3-7, 2006, Trento, Italy*, 2006. URL http://acl.ldc.upenn.edu/E/E06/E06-1032.pdf.

Stuart Campbell. A cognitive approach to source text difficulty in translation. *Target*, 11 (1):33–63, 1999.

J. Carletta. Asseing agreement on classification tasks: The kappa statistic. *Computational Linguistics*, 22(2):249–254, 1996.

Marine Carpuat and Dekai Wu. Improving statistical machine translation using word sense disambiguation. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 61–72, 2007. URL http://www.aclweb.org/anthology/D/D07/D07-1007.

David Chiang. A hierarchical phrase-based model for statistical machine transla tion. In *Proceedings of the 43rd Annual Meeting of the Association for Com putational Linguistics (ACL'05)*, pages 263–270, Ann Arbor, Michigan, June 2005. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/P/P05/P05-1033.

Jorge Civera and Alfons Juan. Domain adaptation in statistical machine translation with mixture modelling. In *Proceedings of the Second Workshop on Statistical Machine Translation*, Prague, Czech Republic, June 2007. Association for Computational Linguistics.

A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of Royal Statistical Society*, 39(1):1–38, 1977. URL http://dx.doi.org/10.2307/2984875.

Mona Diab, Kadri Hacioglu, and Daniel Jurafsky. Automatic tagging of arabic text: From raw text to base phrase chunks. In Daniel Marcu Susan Dumais and Salim Roukos, editors, *HLT-NAACL 2004: Short Papers*, pages 149–152, Boston, Massachusetts, USA, May 2 - May 7 2004. Association for Computational Linguistics.

Kevin Duh. Ranking vs. regression in machine translation evaluation. In *Proceedings of the Third Workshop on Statistical Machine Translation*, pages 191–194, Columbus, Ohio, June 2008. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/W/W08/W08-0331.

Macklovitch Elliott. Transtype2: The last word. In *The fifth international conference on Language Resources and Evaluation (LREC-06)*, Genoa, Italy, 2006.

David Graff. *English Gigaword Corpus*. Linguistic Data Consortium (LDC), Philadelphia, PA, 2005.

Nizar Habash. Arabic preprocessing schemes for statistical machine translation. In *Proceedings of the Human Language Technology Conference of the NAACL, Companion Volume: Short Papers*, pages 49–52. Online]. Available, 2006.

Sandra Hale and Stuart Campbell. The interaction between text difficulty and translation accuracy. *Babel*, 48(1):14–33, 2002.

Hany Hassan, Khalil SIMA'AN, and Andy Way. Syntactically lexicalized phrase-based smt. *IEEE Transactions on Audio, S peech and Language Processing*, 16(7):1260–1273, 2008.

Almut Silja Hildebrand, Matthias Eck, Stephan Vogel, and Alex Waibel. Adaptation of the translation model for statistical machine translation based on information retrieval. In *Proceedings of the Conference of the European Association of Machine Translation (EAMT-05)*, Budapest, Hungary, 2005.

Thorsten Joachims. Making large-scale svm learning practical. In Bernhard Schölkopf, Chris Burges, and Alex Smola, editors, *Advances in Kernel Methods – Support Vector Learning*, pages 169–185. MIT Press, 1998.

Howard Johnson, Joel Martin, George Foster, and Roland Kuhn. Improving translation quality by discarding most of the phrasetable. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 967–975, 2007. URL http://www.aclweb.org/anthology/D/D07/D07-1103.

Mark Kantrowitz, Mohit Behrang, and Mittal Vibhu. Stemming and its effects on tfidf ranking. In *Proceedings of the ACM Conference of the Special Interest Group on Information Retrieval (SIG-IR*, 2000.

W. Kim and S. Khudanpur. Cross lingual latent semantic analysis for language model. In Erhard Hinrichs and Dan Roth, editors, *Proceedings of the International Conferences on Acoustic Speech and Signal Processing, ICASSP 2004*, 2004.

Katrin Kirchhoff, Owen Rambow, Nizar Habash, and Mona Diab. Semi-automatic error analysis for large-scale statistical machine translation. In *Proceedings of MT Summit 2007*, 2007.

Philip Koehn. Pharaoh: A beam search decoder for phrase-based statistical machine translation models. Technical report, USC Information Sciences Institute, Marina Del Rey, USA, 2004a.

Philipp Koehn. Statistical significance tests for machine translation evaluation. In Dekang Lin and Dekai Wu, editors, *Proceedings of EMNLP 2004*, pages 388–395, Barcelona, Spain, July 2004b. Association for Computational Linguistics.

Philipp Koehn and Kevin Knight. Feature-rich statistical translation of noun phrases. In Erhard Hinrichs and Dan Roth, editors, *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 311–318, 2003. URL http://www.aclweb.org/anthology/P03-1040.pdf.

Philipp Koehn and Josh Schroeder. Experiments in domain adaptation for statistical machine translation. In *Proceedings of the Second Workshop on Statistical Machine Translation*, Prague, Czech Republic, June 2007. Association for Computational Linguistics.

Alon Lavie and Abhaya Agarwal. METEOR: An automatic metric for MT evaluation with high levels of correlation with human judgments. In *Proceedings of the Second Workshop on Statistical Machine Translation*, pages 228–231, Prague, Czech Republic, June 2007. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/W/W07/W07-0734.

Arindam Mandal, Dimitra Vergyri, Wen Wang, Jing Zheng, Andreas Stolcke, Gokhan Tur, Dilek Hakkani-Tr, and Necip Fazil Ayan. Efficient data selection for machine translation. In *Proceedings of the Second IEEE/ACL Spoken Language Technology Workshop*, 2008.

Bart Mellebeek, Karolina Owczarzak, Josef Van Genabith, and Andy Way. Multi-engine machine translation by recursive sentence decomposition. In *Proceedings of the Seventh Conference of the Association for Machine Translation in the Americas (AMTA-06)*, Boston, USA, 2006.

David Meyer, Friedrich Leisch, and Hornik Kurt. The support vector machine under test. *Neurocomputing*, 55(1-2):169–186, September 2003.

Jeff Mitchell, Mirella Lapata, Vera Demberg, and Frank Keller. Syntactic and semantic factors in processing difficulty: An integrated measure. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 196–206, Uppsala, Sweden, July 2010. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/P10-1021.

Tom Mitchell. *Machine Learning*. Computer Science. McGraw-Hill, 1997.

Behrang Mohit and Rebecca Hwa. Localization of difficult-to-translate phrases. In *Proceedings of the Second Workshop on Statistical Machine Translation*, pages 248–255, Prague, Czech Republic, June 2007. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/W/W07/W07-0737.

Behrang Mohit, Frank Liberato, and Rebecca Hwa. Language model adaptation for difficult-to-translate phrases. In *Proceedings of the Conference of the European Association of Machine Translation (EAMT-09)*, Barcelona, Spain, 2009.

Behrang Mohit, Rebecca Hwa, and Alon Lavie. Using variable decoding weight for language model in statistical machine translation. In *Proceedings of the Ninth Conference of the Association for Machine Translation in the Americas (AMTA-10)*, Denver, USA, 2010.

Franz Josef Och. Minimum error rate training in statistical machine translation. In Erhard Hinrichs and Dan Roth, editors, *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 160–167, 2003. URL http://www.aclweb.org/anthology/P03-1021.pdf.

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

Franz Josef Och, Daniel Gildea, Sanjeev Khudanpur, Anoop Sarkar, Kenji Yamada, Alex Fraser, Shankar Kumar, Libin Shen, David Smith, Katherine Eng, Viren Jain, Zhen Jin, and Dragomir Radev. A smorgasbord of features for statistical machine translation. In Daniel Marcu Susan Dumais and Salim Roukos, editors, *HLT-NAACL 2004: Main Proceedings*, pages 161–168, Boston, Massachusetts, USA, May 2 - May 7 2004. Association for Computational Linguistics.

Marian Olteanu, Chris Davis, Ionut Volosen, and Dan Moldovan. Phramer - an open source statistical phrase-based translator. In *Proceedings on the Workshop on Statistical Machine Translation*, pages 146–149, New York City, June 2006. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/W/W06/W06-1521.

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

Mathew Snover, Bonnie Dorr, Richard Schwartz, Linnea Micciulla, and John Makhoul. A study of translation edit rate with targeted human annotation. In *Proceedings of the Seventh Conference of the Association for Machine Translation in the Americas(AMTA-06)*, Boston, USA, 2006.

Mathew Snover, Bonnie Dorr, and Richard Schwartz. Language and translation model adaptation using comparable corpora. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing(EMNLP-2010)*, Waikiki, USA, 2008.

Radu Soricut and Abdessamad Echihabi. Trustrank: Inducing trust in automatic translations via ranking. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 612–621, Uppsala, Sweden, July 2010. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/P10-1063.

Lucia Specia, Marco Turchi, Nicola Cancedda, Marc Dymetman, and Nello Cristianini. Estimating the sentence-level quality of machine translation systems. In *Proceedings of the Conference of the European Association of Machine Translation (EAMT-09)*, Barcelona, Spain, 2009.

A. Stolcke. Srilm – an extensible language modeling toolkit. In *Proc. Intl. Conf. on Spoken Language Processing, 2002.*, 2002. URL citeseer.ist.psu.edu/stolcke02srilm.html.

Yik-Cheung Tam, Ian Lane, and Tania Schultz. Bilingual-lsa based lm adaptation for spoken language translation. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, Prague, Czech Republic, June 2007. Association for Computational Linguistics.

Kiyotaka Uchimoto, Naoko Hayashida, Toru Ishida, and Hitoshi Isahara. Automatic rating of machine translatability. In *Proceedings of the tenth Machine Translation Summit (MT-Summit X)*, Phuket, Thailand, 2005.

Nicola Ueffing and Hemann Ney. Application of word-level confidence measure in interactive statistical machine translation. In *10th Conference of the European Association of Machine Translation (EAMT-2005)*, Budapest, 2005.

Nicola Ueffing, Gholamreza Haffari, and Anoop Sarkar. Transductive learning for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 25–32, Prague, Czech Republic, June 2007. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/P/P07/P07-0004.

Richard Zens and Hermann Ney. N-gram posterior probabilities for statistical machine translation. In *Proceedings on the Workshop on Statistical Machine Translation*, pages 72–77, New York City, June 2006. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/W/W06/W06-1510.

Ying Zhang. *Structured Language Model for Statistical Machine Translation*. PhD thesis, Carnegie Mellon University, Pittsburgh, PA, June 2008.