DATA ANALYSIS AND SELECTION FOR STATISTICAL MACHINE TRANSLATION

By

Sauleh Eetemadi

A DISSERTATION

Submitted to
Michigan State University
in partial fulfillment of the requirements
for the degree of

Electrical Engineering – Doctor of Philosophy

2016

ABSTRACT

DATA ANALYSIS AND SELECTION FOR STATISTICAL MACHINE TRANSLATION

By

Sauleh Eetemadi

Statistical Machine Translation has received significant attention from the academic community over the past decade which has led to significant improvements in machine translation quality. As a result it is widely adopted in the industry (Google, Microsoft, Twitter, Facebook, ...etc.) as well as the government (http://nist.gov). The biggest factor in this improvement has been the availability of ever increasing sources of training data as digital multilingual communication and information dissemination become ubiquitous. Relatively little research has been done on training data analysis and selection, despite training data being the main contributor of machine translation quality.

In this work, we first examine fundamental properties of translated and authored text. We introduce a new linguistically motivated feature (Part of Speech Tag Minimal Translation Units) that outperforms prior work in sentence level translation direction detection. Next, we develop a cross-domain data matrix that enables comparison between different features in the translation direction detection task. We extend our previously introduced feature for translation direction detection to use statistically trained brown clusters instead of part of speech tags. This new feature outperforms all prior work in all cross-domain data matrix combinations.

Data selection in machine translation is performed in different scenarios with different objectives including: reducing training resource consumption, domain adaptation, improving quality or reducing deployment size. We develop an efficient (computational complexity and memory consumption is linear in training data size) framework for training data selection and compression called Vocabulary Saturation Filter (VSF). In our experiments we show the machine translation system trained on data selected using VSF is comparable to prior data selection methods with quadratic computational complexity. However, VSF is sensitive to data order. Therefore we exper-

iment with different orderings of the data and compare the results.

Finally, we develop a highly scalable and flexible data selection framework where arbitrary sentence level features can be used for data selection. In addition, a variable threshold function can be used to incorporate any scoring function that is constant throughout the selection process. After introducing this framework, inspired by the features we introduced for detecting translation direction, we use joint models of source and target using Minimal Translation Units (MTU) in addition to source side context using brown clusters to compare various features and threshold functions within this framework. We run end-to-end experiments using data selected by various methods and compare the statistical translation models using various test sets and phrase table comparison metrics.

Copyright by
SAULEH EETEMADI
2016

To anyone who will read and use it!

ACKNOWLEDGEMENTS

I started my Ph.D. in 2006 shortly after I joined Microsoft Research as a software engineer. Indeed, my job at Microsoft Research was the only aspect of my life that did not change during these years. We were renting, then we bought a condo. I was overweight, I started the new sport of Brazilian Jujitsu, I lost weight and eventually earned a black belt. I went to pilgrimage. Then, I went again. I became a father. We travelled to many different countries many times. We left the condo, became a landlord and started renting again. We had our second daughter. We renovated our house. My wife went through life threatening surgeries and recovered. I lost my grandfather and grandmother and so on .... When I started, I never thought it would take me 10 years to defend my dissertation and graduate. While I always knew, I'm not going to give up, I never knew how much help I needed. First and foremost, my deepest gratitude is for the One who presented me with this challenge along with many other challenges in life, so I may become better.

I would like to thank my adviser, Dr. Radha. While he warned me of the difficulty of doing a Ph.D. and working a fulltime job at the same time, his intricate combination of patience, support and challenge was instrumental in helping me to finally overcome this difficulty. If it wasn't for the incredible help and support of my co-advisor Dr. Kristina Toutanova, I would not have been able to complete my Ph.D. I want to express my gratitude to her for mentoring me and providing me with supportive and critical feedback. I would also like to thank Dr. Rong Jin who served on my committee before leaving MSU. If it wasn't for his suggestion to have a co-advisor that can mentor me in the area of Statistical Machine Translation, I would not have had Kristina as my co-advisor. I also want to thank Dr. Esfahanian who agreed to replace Dr. Jin in my committee only few months before my defense. I'm thankful to my committee members Dr. Deller and Dr. Aviyente for their support and feedback as well.

I'm extremely thankful to my mangers and colleagues at the Machine Translation Group in Microsoft Research who were always supportive and encouraging towards the pursuit of my Ph.D.

My managers, Steve, Andreas, Federico, Mei-Yuh, Arul and Vishal were always flexible and supportive of my Ph.D. and for that, I'm very thankful. While I lacked the benefit of having labmates for many years, Jon Clark and Qin Gao filled this gap as they joined our group few years ago while they were Ph.D. students as well. We started having weekly meetings which was incredibly helpful in pushing us towards graduation and for this, I'm very thankful. I had many fruitful discussions with my colleague and mentor Dr. William Lewis who also shared my passion for the role of data in machine translation. His optimism, dedication and support was a positive force throughout my PhD. Since the pursuit of PhD was the other big thing in my life after my job, the subject had come up once or more with all colleagues I've had throughout the years. I cannot name them all here, but without exception, they were all encouraging and played a positive role in this endeavour and for that I'm very thankful.

While being supported at work and school by incredible people, I still needed time to work on my Ph.D. after fulfilling my responsibilities at work. My wife and soulmate, Hoda has been incredibly gracious and selfless throughout these years while she started and finished her masters degree in Architecture, started her Ph.D. in Construction Management, had two kids and defended her Ph.D. last year. Thanks!

At the age of 36, I have now officially been in school for over 30 years. From the day I went to kindergarten till this very day, my mom always encouraged me and believed in me. She got me into good schools, she stayed up late to help me with my homework, get good grades, stay in good schools, get into university and so on. The words cannot begin to describe how much she sacrificed for my education and success and for this I'm forever thankful. My dad always was and continues to be my role model. He blessed me with his wisdom, great advice, direction and support throughout the years and I cannot thank him enough for this. I have been away from my parents now for over a decade in pursuit of higher education and career. Although this pursuit has taken much longer than anticipated, they remained encouraging and supportive and for this I'm forever indebt to them.

I always benefited from discussions about career and higher education with my younger brother

Ameen, throughout his years at Microsoft as well as the two years since he started his Ph.D. at UC Davis. My parents in law have also blessed us with visiting us here in the States and helping us with our two daughters in addition to being great hosts for my wife and daughters during the summer months while I was not with them. I'm incredibly thankful and indebt to them. Many friends and families have supported me with their thoughts and prayers throughout the years. While I cannot name all of them here, I do thank them for their help and support.

Finally, my daughters Aula and Hannah have provided me with incredible joy, challenge and headache for the better half of my Ph.D. While I have spent many nights, weekends and summer months away from them working on my dissertation, I was always welcomed and accepted by them. If it wasn't for them, although this dissertation might have been completed earlier, but overall I am a better person because of them. Thank You!

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

With the digital age there is ever more natural language preserved in digital format. The availability of natural language increases with every news piece, new article, blog post, recorded conversation and virtually everything spoken or written by billions of people on the planet. It is only natural to develop algorithms and techniques enabling humans to consume, absorb and leverage this ever expanding source of data. For example, summarisation enables people to absorb more information with less effort. Sentiment detection builds on top of that and provides aggregate information on, for example, product reviews written by consumers. Machine Translation provides cost effective and seamless access to information that would have been otherwise inaccessible. Researchers develop increasingly more complex algorithms to perform these tasks with higher precision. Many of these algorithms are not inherently decomposable into pieces that can efficiently run in parallel. Also, not all researchers have access to large scale clusters to develop or run such decomposable algorithms. An example of this is deep neural networks. Although the original idea was developed in the 70s, it has only recently been used effectively and successfully in a range of natural language processing problems such as speech recognition and machine translation. This implies it is possible for a range of currently available algorithms that are not scalable, to only be used in several years or decades with improved computation power. This has motivated a new approach to fill this gab. The approach is to compact training data to a small enough size such that the algorithms can train on it in a reasonable amount of time. An example of such work in the field of machine learning is core-sets ([36]).

The focus of this thesis is the training data used in Machine Translation. Machine Translation has been around for several decades. The quality of Machine Translation however has improved significantly over the past several years with the development of statistical methods that can learn from existing translation data instead of having linguists develop translation rules for each language

pair. Machine Translation has received a second boost with the development of data collection methods that have harvested the digital resources such as the web to compose more and more training data. While the consensus is that more data improves machine translation the effort to collect more data has diminished for dominant languages such as English, Spanish and French since the computation capacity to efficiently train a machine translation system on such large data sets has been saturated. Take the language modeling task. The size of the indexed web is estimated to be 50 billion web pages [105]. Assuming an average words per page of 429.2 ([78]), this is approximately 21.4 tera words. The size of the deep web is estimated to be 500 times the indexed web ([14]). That brings the total estimated natural language available on the web close to 10.7 peta words. The largest language model ever reported to be built is by Google over 1 tera words [38]. That is 5% of the indexed web and 0.09% of the entire web. The second type of training data is parallel data. French-English has the largest publicly available parallel corpus with close to 40 million sentence pairs or 2.8 billion total words [21]. While there is a lot more parallel data available reported by the industry ([66]), there is little effort to collect such data since the training algorithms are already computationally saturated with existing hardware.

In this thesis, first we focus on a fundamental aspect of machine translation training data. Parallel natural language data is unique to machine translation. It is also the main and key type of training data for machine translation. In some recent work in machine translation using deep neural networks it is the only type of training data ([94]). As far as we are aware, parallel data has only been used for the purpose of directly or indirectly training a machine translation. The only other task that can also be performed using parallel data is Translationese detection. In a parallel corpus setting we call this Translation Direction Detection or TDD for short. The significance of this task is that it can be evaluated objectively, accurately and meaningfully. In comparison, the evaluation of a machine translation algorithm over parallel data is first of all computationally complex. Second, the evaluation is less meaningful as a sentence can be translated correctly in many different ways and for that reason, finally, the evaluation is not accurate as a completely valid translation can be evaluated as incorrect due to reference mismatch. TDD does not have

any of these issues. Thus it is a suitable task for exploration and accurate comparison of many different feature functions. In Chapter 4 we develop and compare the effectiveness of different features in selecting sentence pairs of a specific direction. Although these features are developed and evaluated in the context of TDD, we show in Chapter 5 that some of these features are also most effective in data selection.

The data selection literature in machine translation is reviewed in Chapter 3. We focus on the quality improvement and training resource limitation scenario from the literature using data compacting techniques to train on the most informative subset of the data when it is not practical to train on the entire data set.

In Chapter 5, we first introduce a linear but suboptimal search algorithm for optimizing a commonly used surrogate objective function (n-gram coverage) for data selection. We show the quality of the machine translation output produced by models trained on the data set chosen by our suboptimal algorithm is not statistically different from the output produced by models trained on the optimal subset. In other words, we are able to produce same quality machine translation output using a linear and suboptimal algorithm.

Finally, in Chapter 6 we introduce a flexible framework for data selection that is able to use any sentence pair level feature along with an arbitrary threshold function. We propose using more sophisticated features from Chapter 4 with the same linear search algorithm. These features use word alignment information combined with semantic features from brown clusters to capture joint lexical and semantic information. These features are able to capture information that has not been directly used in prior data selection methods. In addition, we further investigate the effects of data selection on vocabulary distribution and improve the Jensen–Shannon divergence between selected data and selection pool using an entropy based variable threshold function. Improving upon the previously introduced efficient search algorithm we are able to partition very large data sets (most prior work do not scale to such large data sets) into an ordered and tunable number of partitions[1] and out perform all prior work that can scale to such large data sets. We perform

[1]Partitions are in the order of more desirable data. The desired number sentence pairs can be

end to end statistical machine translation experiments comparing different feature sets and variable threshold functions. We use translation model statistics in addition to BLEU score to compare the effectiveness of various selection techniques.

selected by selecting partitions in order.

CHAPTER 2

BACKGROUND

Translation between different languages is nearly as old as spoken and written language itself. Due to the laborious task of translation, the interest in using computers for translation came about as soon as computers became available. **Warren Weaver**, co-author of the landmark work on communication, *The Mathematical Theory of Communication*, with **Claud Shannon** of Bell Laboratories, is widely recognized as the pioneer of Machine Translation. On March 4th, 1947, in his letter to Professor **Norbert Wiener** of Massachusetts Institute of Technology he wrote ([103]):

> When I look at an article in Russian, I say "This is really written in English, but it has been coded in some strange symbols. I will now proceed to decode." Have you ever thought about this? As a linguist and expert on computers, do you think it is worth thinking about?

Also, in his 1949 "Translation" memorandum in reference to **Shannon**'s work on *Theory of Communication*, he writes ([103]):

> Probably only Shannon himself, at this stage, can be a good judge of the possibilities in this direction; but as was expressed in W.W.'s original letter to Wiener, it is very tempting to say that a book written in Chinese is simply a book written in English which was coded into the "Chinese code." If we have useful methods for solving almost any cryptographic problem, may it not be that with proper interpretation we already have useful methods for translation?

Inspired by the work of **Warner**, in their seminal work , "*The Mathematics of Statistical Machine Translatoion: Parameter Estimation*" ([23]), Brown et al. introduced the concept of using Shannon's *Noisy Channel Model* ([89]) for the purpose of Statistical Machine Translation (Figure 2.1). A noisy channel framework for machine translation works similar to how **Warner** de-

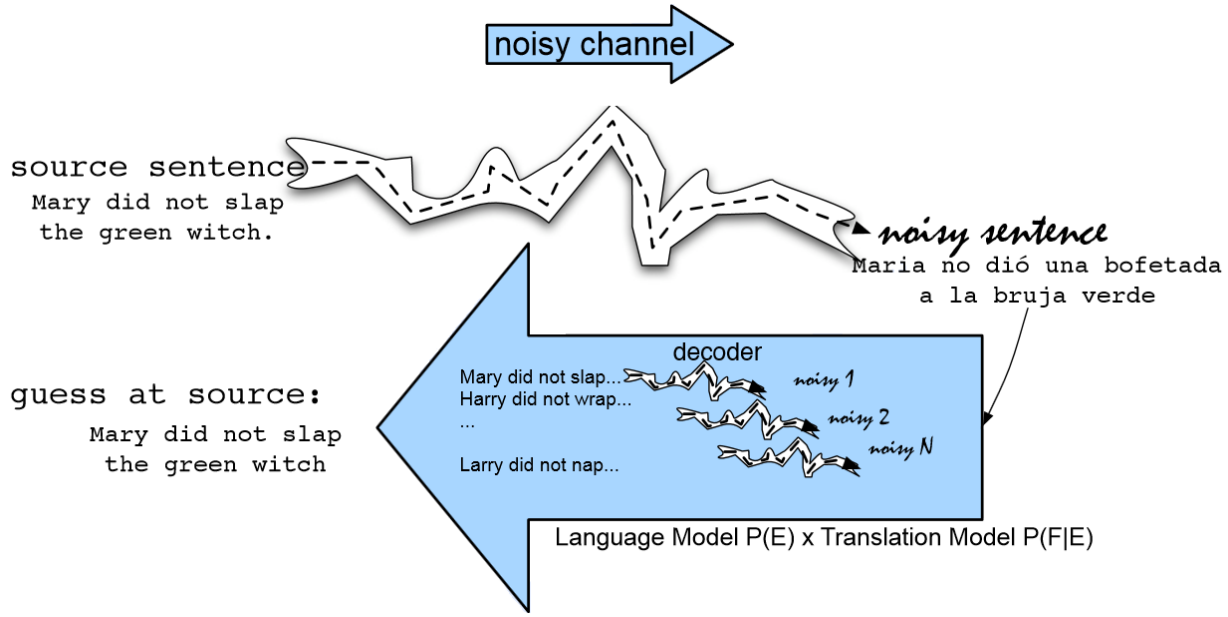


Figure 2.1 Noisy Channel Model for Statistical Machine Translation [50]

scribed in his memorandum: For example, an English sentence goes through a "noisy" channel and comes out as French. Now given the observed "noisy" sentence (which is now in French), what is the most likely English sentence at the source of the noisy channel? Recovering the original English requires reasoning about 1) How the English language is written ("source modeling") and 2) How English is turned into French ("channel modeling") ([55]).

Formally, when translating a French word sequence, f, to English, we assume a French native speaker had a sequence of English words, e, in mind when producing the French word sequence. The process of translating f into English is to find the English word sequence $\hat{e}$ that maximizes $P(e|f)$. Using Bayes' rule, we can write:

$$P(e|f) = \frac{P(f|e) \times P(e)}{P(f)} \tag{2.1}$$

Since the denominator in Equation 2.1 is constant for all values of e, the maximization problem for finding $\hat{e}$ can be formulated as Equation 2.2, also known as the **Fundamental Equation of Machine Translation** ([23]).

$$\hat{e} = \arg\max_{e} P(f|e) \times P(e) \tag{2.2}$$

This equation presents the three fundamental computational problems of statistical machine translation:

1. Estimating the *language model probability*, $P(e)$.
2. Estimating the *translation model probability*, $P(f|e)$.
3. Finding the word sequence e that maximizes Equation 2.2 (Also called decoding).

2.1 Estimating Language Model Probability

A language model is formally defined as a function that takes a word sequence in a given language as input and returns the probability that the word sequence was produced by a native speaker of that language ([57]).

Language Model:

$$P_{LM}(e_1, e_2, ..., e_n) \tag{2.3}$$

For Example[1]:

$$P_{LM}(\text{passed the exam}) > P_{LM}(\text{failed the exam})$$

.

N-Gram language models are the leading method for language modeling. This method is based on the likelihood that individual words follow each other. To make the language modeling problem tractable a **Markov Chain** with a fixed memory. The memory length used in the Markov Chain is called the **language model order**[2]. A trigram[3] language model assumes a Markov chain of length

[1] See Figure 2.2 for more details on this example ([42])

[2] By definition, the order of a language model is two plus the memory length of the assumed Markov chain. That is, a bigram language model assumes a memoryless Markov chain.

[3] A language model with order 3 is also called a trigram language model.

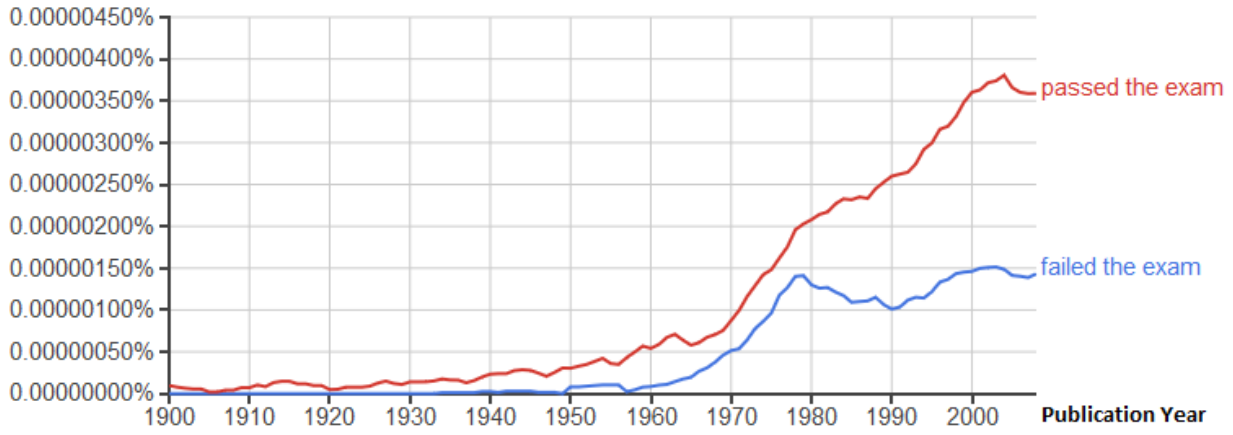


Figure 2.2 Probability of "passed the exam" and "failed the exam" word sequences according to Language Models trained on published books every year since 1900 with a smoothing factor of three ([42]).

two. That is, the probability of the next word in a sequence of words is only dependent on its previous two words.

$$\begin{aligned} P_{LM}(e_1, e_2, ..., e_n) &= \prod_{i=1}^{n} P_{LM}(e_i | e_1, ..., e_{i-1}) \\ &= \prod_{i=1}^{n} P_{LM}(e_i | e_{i-m-1}, ..., e_{i-1}) \qquad (\textit{Markov Chain with memory } m) \end{aligned} \tag{2.4}$$

A n-gram language model is estimated using a large corpus in the desired language by gathering n-gram counts of various lengths up to the language model order and calculating sufficient statistics for the language model. In this thesis, we regularly build and use language modeling tools using off-the-shelf tools such as KenLM ([46]) and SRILM ([93]). Since our contributions in this thesis are not directly related to language models, a general understanding of language models and how they are used as described above is sufficient for understanding this thesis. For more information see [57].

2.2 Estimating Translation Model Probability

There are several different modeling techniques developed for estimating translation model probabilities. Word based models ([23]), phrase based models ([58]) and hierarchical phrase based models ([29]) are some of the most commonly used models. A prerequisite for all of these modeling techniques is word alignment.

2.2.1 Word Alignment

Given a sentence pair that are translations of each other, the word alignment task is to align each word in one sentence to its translation in the other sentence. This is formally defined using an alignment function, a : $i \rightarrow j$ which maps the word in position i in one sentence to another word in position j in the other sentence where these two words are translations of each other. One of the main contributions of Brown et al. was modeling this alignment function and developing an Expectation Maximization algorithm to estimate it ([23]). In its simplest form the alignment function can be modeled using IBM Model 1 ([23]) in Equation 2.5 where e is the English or target sentence with length l_e, f is the French or source sentence with length l_f, ε is the defined as $P(l_e|l_f)$ and a is the alignment function.

$$P(a|e,f) = \frac{P(e,a|f)}{P(e,f)} \qquad \textit{chain rule} \tag{2.5}$$

$$P(e,a|f) = \frac{\varepsilon}{(l_f+1)^{l_e}} \prod_{j=1}^{l_e} P(e_j|f_{a(j)}) \tag{2.6}$$

After replacement and simplification ([57]) we can calculate $P(a|e,f)$ as below.

$$P(a|e,f) = \prod_{j=1}^{l_e} \frac{P(e_j|f_{a(j)}}{\sum_{i=0}^{l_f} P(e_j|f_i)} \tag{2.7}$$

IBM Model 1 only takes into account lexical information for alignment. More complex models have been developed (IBM Models 2, 3, 4 and 5) to consider the position of source and target words

in the sentence, *fertility*[4] and *distortion*[5]. For the purpose of this thesis understanding the concept of word alignment is essential, but estimation methods and further details about alignment models are not necessary.

Given a word-aligned parallel corpus[6] the probability of each target word(s) given a source word can be estimated by gathering sufficient counts from the corpus and applying the chain rule (For more details see [57] and [55]).

2.2.2 Phrase Table

Word based translation models do not allow for the statistical models to directly learn the translation of phrases longer than length one. Phrase based models have been developed to address this deficiency. Different phrase extraction techniques have been developed for this purpose. The phrase extraction process takes a word-aligned parallel corpus as input and outputs a translation table with entries corresponding to source phrase, target phrase and the probability of the target phrase given the source phrase. For more details see [57] and [58].

2.3 Machine Translation Evaluation

Automatic evaluation methods for machine translation output has been one of the key contributing factors to the significant improvement of machine translation quality over the past decade. Amongst different evaluation metrics such as BLEU[7] ([83]) and METEOR ([12]), BLEU is currently the most popular automatic evaluation metric ([57]). In this thesis we make extensive use of BLEU for comparing the quality of the translation system trained on different training data, en-

[4]The *fertility* of a word in the context of machine translation is defined as the number of words on the target generated by (or aligned to) a single word in the source sentence.

[5]In the context of machine translation *distortion* is defined as the relative position of word in the target sentence compared to the position of its corresponding word in the source.

[6]A parallel corpus is simply a collection of sentence pairs in two specific languages where each sentence is paired with its corresponding translation sentence.

[7]BLEU: **Bi**-**L**ingual **E**valuation **U**nderstudy

abling us to have an objective comparison between various data selection techniques in the absence of a human translation evaluator.

Similar to other evaluation techniques, BLEU makes use of reference translations. BLEU is a precision based metric. That is, it takes into account the number n-grams in the translation that occur in any of the reference translations and ignores any *n*-gram in the reference translations that do not appear in the produced translation. The downside of this approach is that a translation output can get a perfect score if it only generates a single matching unigram. This problem is addressed in BLEU by introducing a *brevity penalty* (Equation 2.8).

$$brevity\text{-}penalty = \min(1, \frac{output\text{-}length}{reference\text{-}length}) \tag{2.8}$$

The precision for each *n*-gram length is calculated by counting the total number of *n*-grams in the output that also exists in the reference divided by the total number of *n*-grams of that length that appear in reference translation (Equation 2.9).

$$precision_i = \frac{\sum_{ng_i \in Output} \sum_{ng_i \in Reference} 1}{\sum_{ng_i \in Reference}} \tag{2.9}$$

ng_i :n-gram of length i

A BLEU score can be calculated for an *n*-gram of up to a certain length (BLEU-n).

$$BLEU\text{-}n = brevity\text{-}penalty * e^{\sum_{i=1}^{n} \lambda_i \log precision_i} \tag{2.10}$$

The *n*-gram length of four and uniform lambda weights is a common configuration used for evaluation ([57]). This simplifies Equation 2.10 to Equation 2.11

$$BLEU\text{-}4 = \min(1, \frac{output\text{-}length}{reference\text{-}length}) \prod_{i=1}^{4} precision_i \tag{2.11}$$

For more details on multiple references and more detailed discussion on BLEU and other evaluation metrics see [57].

In this chapter, we covered a brief history of machine translation, key ideas and concepts essential for understanding this thesis. For a comprehensive and detailed discussion on various topics related to Statistical Machine Translation, see the book written with the same title by Koehn ([57]).

CHAPTER 3

LITERATURE REVIEW

3.1 Introduction

Training data for statistical machine translation has increased significantly over the past several years and is continuing to grow even further as digital multilingual communication and information dissemination become ubiquitous. However, the data is also coming in an increasingly wider spectrum of quality, from sentence level professional translations to crowd-sourced translations, to machine-generated outputs from free online translators. This is one of the reasons why data selection and cleaning has become a common step in the development of machine translation systems. At the same time, while training data for statistical machine translation is abundant in some language pairs, development of high quality machine translation systems for low-resource languages remains a challenge due to lack of training data. Using human translators to produce new training data has been a common solution to this problem ([6, 5, 24, 44]), yet the cost of translation has been a limiting factor. As a result, data selection techniques have been developed to select monolingual data that, if translated, can improve the quality of a machine translation system the most. While industry leaders in the field have the infrastructure and resources to build and iterate over ever growing data sets, this is not cost effective for small businesses and academics. And, even in the presence of abundant training data for statistical machine translation, it is still desirable to select a subset of the data that enables the best translation quality in a statistical machine translation system. Finally, offline mobile applications which generally have constrained memory limitations are another important application of data selection in machine translation.

As a result, a subfield of data cleaning and selection has formed in the field of machine translation, and some of the most practical methods for data cleaning and selection have appeared in system submission papers . This chapter attempts to exhaustively review the literature in machine

translation data selection and offer a comparative overview of the proposed methods in each application scenario.

All data selection and cleaning algorithms select a subset of some original dataset. They differ in three main characteristics:

1. **Application Scenario:** The application scenario for each method (e.g. resource reduction for high or low-resource languages, quality improvement in general or domain adaptation settings) defines the type of data that is being selected (parallel or monolingual), and the problem that the method aims to solve (e.g. maximize translation quality with minimal training time memory consumption).

2. **Scoring Functions:** Since it is infeasible to train a machine translation system on each subset of the data and evaluate the quality of the resulting system for each subset, researchers have developed scoring functions (features) defined on subsets of the data, which are expected to correlate with the usefulness of the data subset for training a high-quality machine translation system. Different application scenarios motivate different scoring functions.

3. **Selection Algorithms:** Given one or more scoring functions for evaluating a subset of data, we need to find the subset that maximizes this scoring function subject to some constraints. Depending on the properties of the scoring function, exact or approximate search algorithms have been developed.

In the rest of the chapter we first review the variations in the literature based on the main components listed above (Fig. 3.1). Next we provide a comparative overview of literature on how each work has combined the components above and how they compare with other research. This chapter is organized as follows: In Section 3.2 we break out and detail the application scenarios foundational to data selection algorithms, as reviewed in (1) above. In Section 3.4, we formalize the data selection problem as a constrained optimization problem and elaborate on different formulations of such problems corresponding to different application scenarios. An exhaustive list of

[1]Numbers in Fig. 3.1 indicate related section numbers in this chapter.

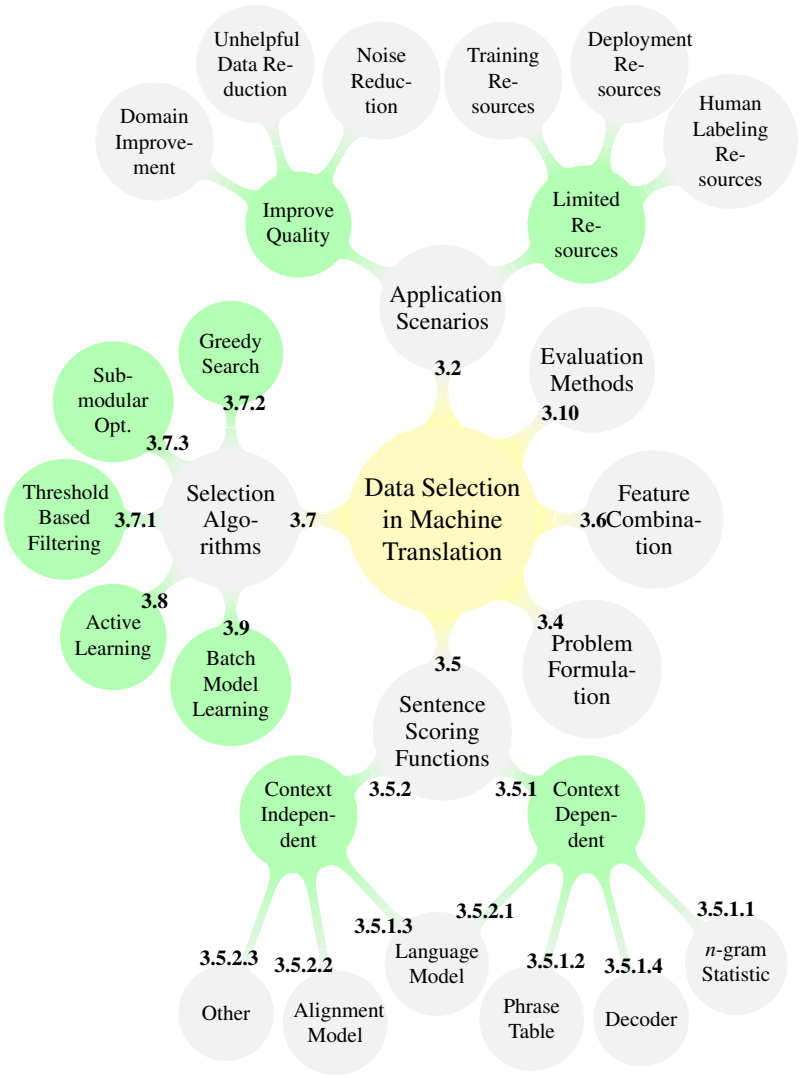


Figure 3.1 Taxonomy of data selection in Machine Translation[1]

data selection scoring functions is presented in Section 3.5 followed by feature combination and parameter tuning in Section 3.6. The scoring functions are organized based on the statistical model (e.g., language model, translation model, . . .) they use. Search methods used to solve optimization problems are listed in Sections 3.7, 3.8 and 3.9. Methods of evaluating the effectiveness of data selection methods are discussed in Section 3.10. We provide a comparative summary of all prior work in section 3.11, along with recommendations for applying these methods, and list related areas of research in Section 3.12.

3.2 Application Scenarios

Data selection is performed in a variety of scenarios . There are two broad application scenario categories: those where the goal is to minimize resource consumption, and those focused on quality improvements or noise reduction. These in turn can be broken down into sub-categories, as described below, all based on selection criteria and scoring functions, which will be discussed in more detail in Section 3.4:

1. **Satisfying Resource Constraints:** A common scenario in data selection for machine translation is that training data size needs to be reduced due to some resource constraints ([66, 40, 26]). Application scenarios in this category differ based on the constrained resource. Although model pruning can be used to satisfy deployment size constraints ([41]), selecting a subset of the training data can be used to satisfy any of the resource constraint listed below. With an exponential number of subsets to choose from, the goal is to choose a subset of the training data that will ultimately result in the highest translation quality.

    a) **Training Resources:** With the increasing availability of training data for statistical machine translation, training resource requirements (e.g., number of computers, main memory size or training time) increase as well. The goal here is to reduce training resource requirements by training on a subset of the training data with little or no impact on translation quality.

    b) **Deployment Size:** In scenarios where a translation system is hosted on devices with limited hardware capabilities, such as mobile devices, it is desirable to produce translation models that are small in size and can be hosted on such devices with minimum impact on translation quality. One method for achieving this objective is by selecting a subset of the training data ([33]). Alternatively, this objective can be achieved by pruning the translation models (Such methods are outside the scope of this thesis. See [110] for a comparison of different pruning methods).

c) **Manual Translation Cost:** When improving translation quality for low resource languages[2] where pre-existing parallel training data is limited, a common approach is to select a subset of a monolingual corpus to be manually translated by humans and added to the training data ([5, 6]). Since there is a cost element involved, the goal is to achieve highest improvement at a fixed cost or meet a predetermined quality bar with minimum cost.

2. **Quality Improvement:** Unlike the resource constraint scenario, the focus of applications that follow this scenario is on quality improvement. In these scenarios, the goal is to select a data subset which will result in a higher quality SMT[3] system compared to an SMT system trained on the full data set. This can be done by filtering out sentences that are noisy or that have a difficult-to-learn translation phenomena (e.g., phrase based translation systems often learn incorrect phrase translations from freely translated[4] text), or selecting a subset that is relevant to a domain different from the domain of the input data. Although data size will often be reduced in these scenarios, that is a side-effect, not the goal.

   a) **Noise reduction:** A common source of training data is from crawled multilingual web sites ([86]). Web data can be very noisy, hence, noise reduction has become a common preprocessing step for this kind of data ([32]). For parallel data, a pair of sentences that are not good translations of each other can be considered noise.

   b) **Reduction of Unhelpful Data:** Although the objective is similar and also partially achieved by noise reduction, it may be desirable to filter out data that are not traditionally classified as noise. For example, non-literal translations are not desirable for training phrase-based statistical MT systems (at least using current technology)..

[2]Low Resource Languages are languages that have relatively little monolingual or parallel data available for training, tuning and evaluation.

[3]SMT: Statistical Machine Translation

[4]Free translation or freely translated text contrary to literal, direct or word-for-word translation conveys the overall meaning of a sentence or phrase without necessarily a word-for-word correspondence between source and translated text [45].

c) **Domain Improvement:** A common approach for improving translation quality is to train domain specific translation models ([74, 8]). To that end, a subset of the training data that is more representative of the target domain can be selected. In this task, an in-domain data set is used to guide the training data selection process.

3.3 Notation and Terminology

Prior work in data selection in machine translation have borrowed terminology and notation from several related fields including active learning, machine learning and quality estimation. We list the terms used in the literature for each concept in data selection and present the terms and notations we use throughout this chapter.

Data selection in machine translation has also been referred to as data cleaning ([49, 80]), noise reduction ([95, 52, 81]), improving training data quality ([70, 2]) and active learning ([5, 7]). Although we use all of these terms in the context of corresponding scenarios or methods, we refer to the general task of selecting a subset of data for training as "data selection" independent of its purpose or method.

The data selection task is to select the "best" subset of the data for machine translation model training. We refer to the full set of available data as "**selection pool**" or S_{pool} and denote the "best" subset by **"optimum subset"**, S^*, regardless of the optimization criteria. The unit of data is always sentences or sentence pairs[5]. We do not specifically call out parallel data or sentence pairs versus monolingual data or sentences when it is apparent from the context. When distinction between two sides of a parallel corpus is necessary, we use the letter "*s*" with appropriate casing to indicate the source side of the parallel data and **"*t*"** for the target side. The selection pool is also called unlabeled data in some prior work ([43]). In some scenarios there is a seed parallel or monolingual corpus given as part of the data selection task. We call this data set the "**seed corpus**" and denote it by S_{seed}. In this scenario, the assumption is that the seed corpus is already

[5]The terms translation units or utterances could also be used, but they are functionally equivalent in this context.

selected and the task is to select a subset of the selection pool to be added to the seed corpus. Data selection tasks for domain adaptation are also given an "**in-domain**" development set, S_{in} and an "**out-of-domain**" development set, S_{out}. In some cases, the "selection pool" is also used as the out-of-domain development set. A single sentence in any of the sets defined above is lowercase s_i to denote the i^{th} sentence in the data set. In most proposed methods for data selection, one or more functions are used to evaluate the usefulness of a sentence or sentence pair. The literature calls such functions: features, feature functions, objective functions and "**scoring functions**". We chose the last term and denote it as $f(s_i)$. Many methods proposed for data selection use iterative selection algorithms where one or more sentences from the selection pool are selected in each iteration. While the algorithm has selected j sentences, the sentences that have been selected so far are called the **"selected pool"** and are denoted as S_1^j (sentences 1 through j). In this context it is assumed that after each iteration, the "selected pool" is added to the "seed corpus". The remaining sentences are noted as "**candidate pool**", $S_{candidate}$. Scoring functions that depend on the selected pool are called "**context dependent**" scoring functions and are denoted as $f(s_{j+1}|S_1^j)$.

Translation of sentence s_i using models trained on data set S is denoted as $t_i = T_{\mathcal{X}(S)}(s_i)$ where $\mathcal{X}(S)$ is any model trained on data set S (e.g., language model, $\mathcal{LM}(S)$, alignment model, $\mathcal{AM}(S)$, or translation model, $\mathcal{TM}(S)$). Lowercase s_1^m indicates words 1 through m in sentence, s. $P_{\mathcal{LM}(S)}(s_i)$ represents the probability of sentence s_i appearing in the evaluation set according to a language model trained on corpus S. $P_{\mathcal{TM}(S)}(s_i,t_i)$ and $P_{\mathcal{AM}(S)}(s_i,t_i)$ are interpreted similarly for a **translation model** and an **alignment model**.

Figure 3.2 demonstrates the role of each data set in the data selection process. The selection pool can be used to build statistical models (e.g., language model, translation model, ...) to provide statistical insight. These models are computed once and not updated. On the other hand, statistical models computed using the selected pool separately or combined with the seed corpus are updated as the selected pool changes during the data selection process. Scoring functions that depend on the models that need to be updated are called context dependent scoring functions. In addition, a decoder using models built by the selection or selected pool can be used as another input to

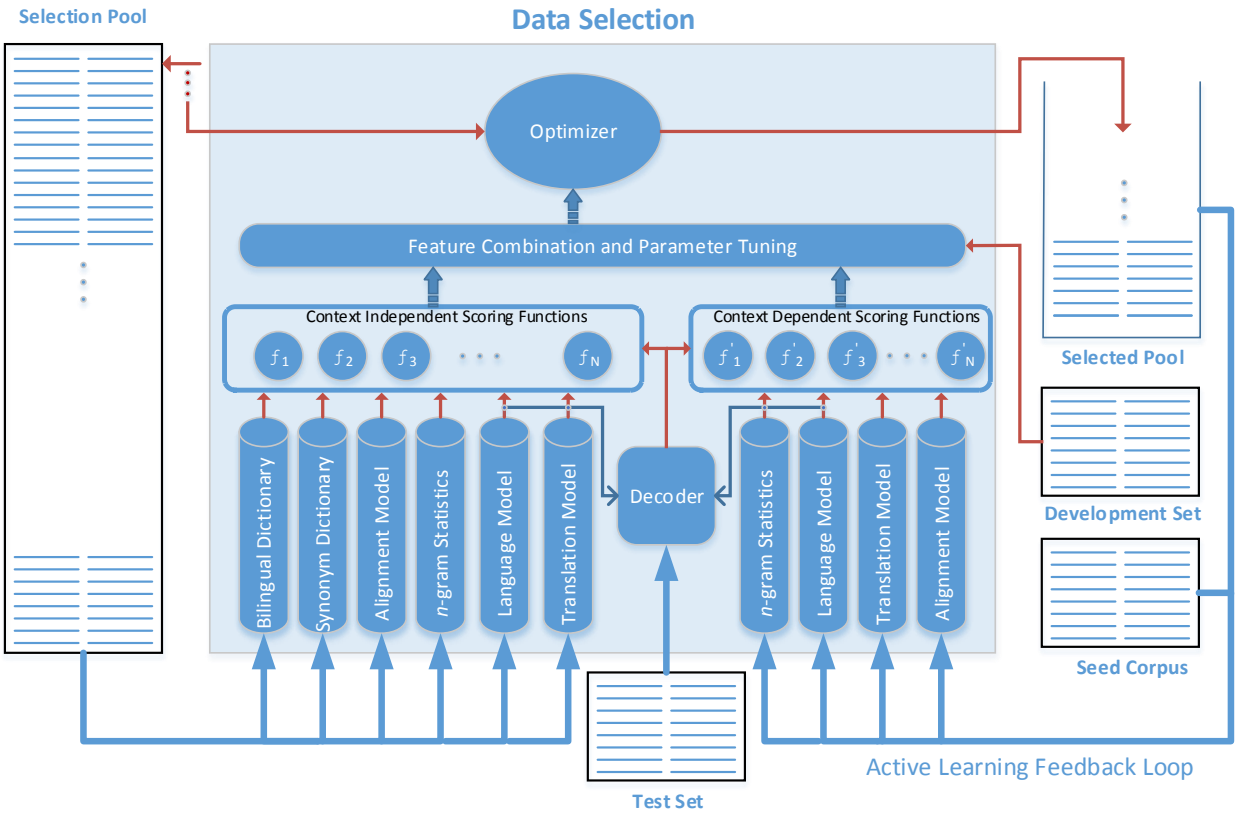


Figure 3.2 Data Selection: This diagram demonstrates the role of each data set in the data selection process.

context dependent or context independent scoring function. A development set can be used to train weights when more than one feature functions is used. Ultimately, the final scoring function is used to score sentences or sentence pairs in the selection pool with an optimizer used to select the optimum subset of the selection pool. The optimization step is often done by greedily adding sentence pairs from selection pool into the selected pool. For context dependent functions, the models or statistics need to be recomputed after the selection of each sentence pair, or N sentence pairs in the case of batch learning.

3.4 Problem Formulation

Data selection can be formulated as a constrained optimization problem. The constraint is that the optimum subset has a given maximum size, which is usually measured by the number of sentences or sentence pairs in the subset, denoted by $|S|$ (Size can also be measured in total number of

characters, tokens, token types, etc depending on the scenario.) The true scoring function that a data selection method aims to maximize measures the expected translation quality using data drawn from some desired target distribution (which could be of general domain or a specific target domain in the case of domain adaptation).

Below we formulate a constrained optimization problem in a generic form, which expresses what data selection methods aim to achieve in theory. Since it is impractical to solve this constrained optimization problem[6], different data selection methods use alternative formulations. The true scoring function is to select a subset, S, from the selection pool, S_{pool}, that maximizes expected translation quality[7] when translated using models trained on S for any data sample, D, that is drawn from some desired target distribution $\mathrm{P}(\mathrm{n}-\mathrm{gram})$, subject to some size or resource constraint C, as shown in Equation 3.1.

$$S^* = \underset{S \subset S_{pool}, \mathrm{Size}(S)<C}{\arg\max} \mathbb{E}\left[\mathrm{BLEU}\left(D_{ref}, T_{\chi(S)}\left(D_{src}\right)\right)\right] \tag{3.1}$$

D_{src} : Source side of D

D_{ref} : Reference translation for D_{src}

(3.2)

The value for $\mathbb{E}\left[\mathrm{BLEU}\left(D_{ref}, T_{\chi(S)}\left(D_{src}\right)\right)\right]$, is usually estimated using a blind test set, S_{Test}, drawn from the desired target distribution $P(n-gram)$(Equation 3.3).

$$S^* = \underset{S \subset S_{pool}, \mathrm{Size}(S)<C}{\arg\max} \mathrm{BLEU}\left(S_{Test_{ref}}, T_{\chi(S)}\left(S_{Test_{src}}\right)\right) \tag{3.3}$$

Consequently, the task of data selection is to find the best subset of sentences that will result in highest translation quality as approximated on a blind test set which is not available during training. Some application scenarios provide a constraint on the size of the subset. There are therefore two technical problems to solve:

[6]The number of subsets of size m from a set of size n is $\frac{n!}{m! \times (n-m)!}$. Since this value is exponential in $\frac{n}{k}$, it is impractical to do an exhaustive search enumerating all subsets.

[7] We will use BLEU as the automated metric here, but other measures are certainly viable. **Bil**ingual **E**valuation **U**nderstudy is the most popular automatic evaluation metric based on n-gram matches between translation output and reference translations ([57, 82])

1. Efficient estimation of the quality of an MT system trained on a set S using an unseen test set.

2. Finding the data subset S^* that maximizes this estimated quality.

A variety of models have been defined for the quality mentioned above. They either use

- a single scoring function on data subset S or
- a model which combines multiple scoring functions or
- in the case of active learning, an SMT system (or some elements of it, like a language or translation model) which is trained on the training data and evaluated against a blind test set.

An example of a scoring function is n-gram[8] coverage of the data subset with respect to the selection pool. Thus a subset that covers a larger portion of all n-grams is assumed to result in a higher quality MT system. Multiple scoring functions can also be combined using a linear model for estimation. Equation 3.4 below shows a formulation of the optimization problem from Equation 3.3 where a combination of scoring functions is used to model translation quality.

$$S^* = \underset{S \subset S_{pool}, \text{Size}(S) < C}{\arg\max} \sum_{i=1}^{\substack{Feature \\ Count}} \lambda_i \, \mathcal{F}_i(S) \tag{3.4}$$

$\mathcal{F}_i(S)$: corpus level scoring function

λ_i : feature weight

Once a model for scoring the goodness of data subsets has been defined, the remaining problem is searching for the best subset of a given maximum size. Depending on the scoring functions, the search problem can be extremely hard and necessitating approximate search techniques. For practical purposes, the scoring functions over data subsets can be decomposed into increments

[8]In this chapter, unless otherwise specified, word n-grams are referred to as simply n-grams for brevity. A word n-gram is a sequence of n words that appear in natural language text.

when adding each sentence one by one in a given order. Given this decomposition, a corpus subset can be selected incrementally by adding individual sentences. Equation (Equation 3.5) shows such a decomposition.

$$S^* = \underset{S \subset S_{pool}, \text{Size}(S) < C}{\arg\max} \sum_{j=1}^{|S|} \sum_{i=1}^{\substack{Feature \\ Count}} \lambda_i \, f_i(s_j | S_1^{j-1}) \tag{3.5}$$

$f_i(s_j|S_1^{j-1})$:context dependent sentence level scoring function

Equation 3.5 assumes conditional independence for the scoring function between sentences in the selection pool given the selected pool. On the other hand Equation 3.6 assumes complete independence. If the sentence level scores are independent of previously selected sentences (e.g., sentence length), a simple threshold based filtering algorithm will provide the optimum solution ([53]) to the constraint optimization problem with computational complexity of $\mathcal{O}(|U| \times \log|U|)$ for sorting sentences based on their feature function value and $\mathcal{O}(|S|)$ for satisfying the resource constraint. This is often the case for scoring functions motivated by data cleaning and noise reduction (For example, the source to target sentence length ratio.) Sentence level scoring functions that are motivated by minimizing resource constraints are often dependent on previously selected sentences (e.g., number of new n-grams in a sentence).

$$S^* = \underset{S \subset S_{pool}, \text{Size}(S) < C}{\arg\max} \sum_{s \in S} \sum_{i=1}^{\substack{Feature \\ Count}} \lambda_i \, f_i(s) \tag{3.6}$$

$f_i(s)$: sentence level scoring function

In such cases (Equation 3.6) an incremental search algorithm (e.g. greedy search) is often used to solve the constrained optimization problem. Selection algorithms are discussed in more detail in Section 3.7.

3.5 Sentence Level Scoring Functions

As explained in Section 3.4, the constrained optimization problem is often broken down using a combination of sentence level scoring functions to enable solving the optimization problem efficiently. These functions can be categorized as follows.

- **Context Independent Functions** which depend on nothing but the candidate sentences in question.
- **Context Dependent Functions** which depend on the selected pool.

Threshold filtering can only be used for context independent functions. For context dependent functions, a greedy incremental search algorithm is often used to solve the optimization problem.

3.5.1 Context Dependent Functions

The values or scores of context dependent functions depend on statistics from the selected data, in addition to the sentence being scored. In a typical data selection algorithm, all sentences are scored and one or more[9] sentences with highest score are added to the selected pool. This requires the statistics in use to be recomputed in addition to all candidate sentence scores. We categorize the scoring functions in this group based on the statistics or models that need to be calculated over the selected sentences in order to assign scores to candidate sentences.

3.5.1.1 N-Gram Coverage Functions

Functions in this group are motivated by finding a subset of the selection pool that best represents the entire set using n-gram statistics. In other words, when selecting a subset of training data, it is often desirable to preserve the vocabulary and its context (n-grams). The simplest way of using n-gram statistics for this purpose is to ensure all n-grams of up to a certain length are present in

[9] Batch-learning techniques are often used to make data selection methods practical. See Section 3.9 for details.

the selected subset. For this purpose, it is sufficient to keep the n-gram count for the selected pool. For each candidate sentence, the score is equal to the number of new n-grams it contains where the n-gram length is less than or equal to N.

$$f_1(s_j|S_1^{j-1}) = \frac{1}{norm(s_j)} \sum_{n=1}^{N} \sum_{ng \in \mathrm{NG}(s_j,n)} weight(ng, j-1) \tag{3.7}$$

$\mathrm{NG}(s,n)$: All n-grams of size n in sentence s

$weight(ng,m)$: Weight of n-gram ng given selected pool S_1^m are selected

$norm(s)$: Normalization factor for sentence s

Equation 3.7 covers much of the prior work depending on the choice of $weight(ng,m)$ and $norm(s)$. Eck et al. present several scoring functions with $weight(ng)$ equal to 1 ([33]). In their simplest scoring function they only count the number of new n-grams in a sentence and normalize by the length of the sentence (Equation 3.8).

$$weight_1(ng,m) = \begin{cases} 0 & ng \in \bigcup_{i=1}^{m} \mathrm{NG}(s_m,n) \\ 1 & \text{otherwise} \end{cases} \tag{3.8}$$

$$norm_1(s) = |\mathrm{NG}(s,n)|$$

In this approach rare and frequent n-grams are valued equally. In their effort to assign a higher weight to more frequent n-grams in the selection pool, S_{pool}, Eck et al. use the n-gram frequency as the weight in their improved scoring function (Equation 3.9). Using normalization factors of $|s|^i$ where i takes on values of 0, 1 and 2, are other variations attempted by this work.

$$weight_2(ng,m) = \begin{cases} 0 & ng \in \bigcup_{i=1}^{m} \mathrm{NG}(s_m,n) \\ \mathrm{Freq}(ng, S_{pool}) & \text{otherwise} \end{cases} \tag{3.9}$$

$$\mathrm{Freq}(ng,S) = \sum_{s \in S} \sum_{ng \in \mathrm{NG}(s,len(ng))} 1$$

A clear shortcoming of this function (Equation 3.9) is that once an n-gram exists in the selected sentences, it is of zero value when selecting new sentences. This does not allow for the scoring function to discriminate against an n-gram with frequent occurrences in the selected sentences versus an n-gram that has only appeared once. This is the motivation for introducing a feature decay function for the n-gram weight ([4, 16]).

$$weight_3(ng,m) = \text{Freq}(ng,S) * \mathrm{e}^{-\lambda * \text{Freq}(ng,\{s_1 \ldots s_m\})} \tag{3.10}$$

λ : Exponential decay hyper parameter

For their last variation on the weight function, Eck et al. use the cosine similarity between TF-IDF[10] vectors of the selected pool and sentences in the candidate pool ([33]). In an information retrieval setting for TF-IDF, all sentences in the selected pool play the role of the search query while each sentence in the candidate pool is considered a potential matching document. The goal in this case is to find the document (candidate sentence) that is most dissimilar to the search query (selected pool). Defining the weight function, $weight(ng,m)$, as follows, results in Equation 3.7 which uses cosine similarity of TF-IDF vectors as defined above, as the sentence scoring function.

$$\begin{aligned}
tf(ng,S) &= \text{Freq}(ng,S) \\
idf(ng,S) &= \frac{|S|}{\sum_{s_i \in S} \mathbb{1}\left(ng \in \text{NG}(s_i, len(ng))\right)} \\
\mathbb{1}\left(ng \in \text{NG}\right) &= \begin{cases} 1 & ng \in \text{NG} \\ 0 & \text{otherwise} \end{cases} \\
weight_4(ng,m) &= tf(ng, S_1^m) * \text{Log}\, idf(ng, S_{m+1}^{|S|})
\end{aligned} \tag{3.11}$$

Since the goal of the TF-IDF weight function is to find the most dissimilar sentence, the overall optimization problem (Equation 3.6) turns into a minimization, unlike other n-gram based scoring functions.

[10] Term Frequency - Inverse Document Frequency

3.5.1.2 Phrase-Table Based Functions

Scoring functions in this section require a phrase table. Depending on the function, the "seed corpus", the "selection pool", the "selected pool" or "candidate pool" or a combination of them are used to train the phrase table. When selecting from a monolingual pool of sentences, a full SMT system trained on the seed corpus can be used to turn the monolingual corpus into a parallel corpus to train the phrase table [43]. Haffari introduces the idea of *exploration* versus *exploitation* in this context ([43]). The idea is to "explore" by selecting sentences with new phrase pairs to expand coverage while "exploiting" phrase pairs that are not new, to improve their probability estimation. We use Table 3.1 to further explain this idea. Phrase pairs that occur frequently in the selection pool but do not occur or occur rarely in the seed corpus are of most value. If they do not exist in the seed corpus (A^*), they add new coverage (exploration). On the other hand, they provide better estimation if they occur rarely in the seed corpus (B^*, exploitation). Phrase pairs with low frequency in the selection pool can also be useful if they occur rarely in the seed corpus (D^*) or do not occur at all (C^*). Although this intuition is explained in the context of phrase-pairs it is applicable to n-gram and language model based scoring functions as well.

		Selection Pool		
	Phrase Pairs	high frequency	low frequency	zero frequency
Seed Corpus	high frequency			
	low frequency	B^*	D^*	
	zero frequency	A^*	C^*	

Table 3.1 Exploration versus Exploitation
A^*: Improved coverage for important[11] phrase pairs
B^*: Improved estimation for important phrase pairs
C^*: Improved coverage for phrase pairs
D^*: Improved estimation for phrase pairs

Haffari introduces a phrase pair utility function to capture exploration versus exploitation (Equation 3.12). The sentence scoring function is then computed using arithmetic (Equation 3.13)

[11] The assumption here is that a phrase pair that occurs frequently in the selection pool, is likely to be used in a translation task, and therefore important.

or geometric (Equation 3.14) average of the phrase pair utility function over all phrase pairs.

$$\text{PPUtil}(pp|(s,t)_1^{j-1}) = \frac{P(PP_1 = pp | PP_1 \in \bigcup_{k=1}^{j-1} \text{PPs}(s_k,t_k))}{P(PP_2 = pp | PP_2 \in \bigcup_{k=j}^{|S|} \text{PPs}(s_k,t_k))} \tag{3.12}$$

pp : A phrase pair mapping a phrase in one language to its translation in a second language.

$\text{PPs}(s,t)$: The set of all phrases pairs extracted from sentence pair (s,t).

$$f_2((s_j,t_j)|(s,t)_1^{j-1}) = \left(\prod_{pp \in \text{PPs}((s_j,t_j))} \text{Log PPUtil}(pp|(s,t)_1^{j-1}) \right)^{\frac{1}{|S|}} \tag{3.13}$$

$$f_3((s_j,t_j)|(s,t)_1^{j-1}) = \frac{1}{|S|} \sum_{pp \in \text{PPs}((s_j,t_j))} \text{PPUtil}(pp|(s,t)_1^{j-1}) \tag{3.14}$$

Ambati takes a different approach in using a phrase table for scoring sentences ([4]). The goal is to give a higher score to source phrases where the phrase table is more uncertain about their translation. Since entropy is a measure of uncertainty, Ambati defines phrasal entropy (Equation 3.15) to calculate the level of uncertainty a phrase table has regarding a source phrase.

$$\text{PhrEnt}^{12}(p_s|(s,t)_1^j) = \frac{\sum_{p_t \in \text{Trans}(p_s)} -P_{\mathcal{TM}(S_1^j)}(p_t|p_s) * \log P_{\mathcal{TM}(S_1^j)}(p_t|p_s)}{|\,\text{Trans}(p_s)|}$$

$\text{Trans}(p_s)$: Target phrases for phrase p_s according to phrase table trained on $(s,t)_1^j$ (3.15)

Using phrasal entropy, the scoring function for a sentence is defined as the sum of the phrasal entropy over all source phrases in a sentence. This scoring function can be extended to use two

[12] By definition conditional entropy of $H(Y|X)$ is defined as $\sum_{x \in X, y \in Y} p(x,y) \log p(y|x)$. But, the definition provided for PhrEnt here is missing a $P_{\mathcal{TM}(S_1^j)}(p_s)$ inside the summation but outside the log. We recognize this inaccuracy but keep the formula consistent with the referenced work ([4]).

different phrase tables (one trained from source language to target language and the other target to source). Using the second phrase table, uncertainty about target phrases can also be taken into account.

3.5.1.3 Language Model Based Functions

Language model based scoring functions are similar to n-gram based scoring functions. In both cases, n-grams are the basis for scoring sentences. The advantage of language model based scoring functions is their higher accuracy in estimating n-gram probability, especially when the data is sparse (This is due to back-off and smoothing strategies used in language modeling.) On the other hand, training a language model has a higher computational complexity compared to collecting n-gram statistics for n-gram based scoring functions.

In the absence of a phrase table (when parallel data is not available or less computational intensity is desirable), Haffari et al. suggests using a language model over all possible phrases (all n-grams of up to a certain size) to calculate the equivalent of the phrase pair utility function defined in Equation 3.12, n-gram utility. N-gram utility follows the same principle, but uses n-grams and two language models (one language model trained on the seed corpus and another trained on the selection pool) instead of phrase pairs and two phrase tables (Equation 3.16).

$$\text{NGramUtility}(ng|(s,t)_1^{j-1}) = \frac{P_{\mathcal{LM}(S_j^{|S|})}(NG_1 = ng)}{P_{\mathcal{LM}(S_1^{j-1})}(NG_2 = ng)} \tag{3.16}$$

$P_{\mathcal{LM}(S_1^{j-1})}(NG = ng)$: Probability of n-gram ng according to language model trained on S_1^{j-1}.

The sentence scoring function is then defined as an arithmetic or geometric average of the n-gram utility function of all n-grams up to a certain length derived from the sentence. In a slight variation to Equation 3.16, Mandal et al. use perplexity[13] instead of probability using the same

[13]The perplexity of a random variable is defined as two (or any given base number) to the power of its entropy. In natural language processing this function is commonly used as a measure of how surprised a language model is when observing a sequence of words.

language models ([71]).

Liu et al. attempt to assign the highest score to sentences with most informative n-grams ([69]). They measure informativeness of n-grams by n-gram[14] entropy. In addition, they multiply the entropy by the square root of the n-gram length to encourage longer phrases as they suggest longer phrases contribute to better translation.

$$\begin{aligned}\text{NGEntropy}(ng|S_1^{j-1}) = -\log\left(P_{\mathcal{LM}(S_j^{|S|})}(ng)\right) * \sqrt{len(ng)} \\ * \mathbb{1}\left(ng \in \text{NG}(S_j^{|S|})^{15}\right)\end{aligned} \tag{3.17}$$

Equation 3.17 uses individual entropy as a measure of informativeness. We would like to point out that a more appropriate use of entropy would be using it as a measure of uncertainty. For this purpose, given a n-gram ng of size m, $\{w_1 \dots w_m\}$, the entropy of the n-gram can be defined in the context of all n-grams of size $m+1$, $\{w_1 \dots w_{m+1}\}$, that have n-gram ng as their prefix (Equation 3.18).

$$\begin{aligned}\underset{\mathcal{LM}(S_1^j)}{\text{NGEntropy}}(w_1^m) = -\sum_{w \in \text{Vocab}(S_1^j)} P_{\mathcal{LM}(S_1^j)}(w|w_1^m) * \log\left(P_{\mathcal{LM}(S_1^j)}(w|w_1^m)\right) \\ \text{Vocab}(S_1^j) : \text{The set of all unique words in } S_1^j.\end{aligned} \tag{3.18}$$

Finally, Ambati attempts to make the probability distribution function of the final selected pool as close as possible to the probability distribution function of the selection pool. Two language models are trained for this purpose. The first language model is trained on the seed corpus and selected pool, while the second language model is trained on the selection pool. The aim is to

[14]For all practical purposes in their work ([69]), a phrase is the same as an n-gram as they consider phrases for a sentence to be all n-grams of up to a certain length.

[15]See Equation 3.7 for definition of $\text{NG}(S_j^{|S|})$.

minimize the Kullback-Leibler divergence[16] between these two probability distribution functions. To achieve this, the scoring function for a sentence equals the sum of its n-gram contributions to to the overall KL divergence (Equation 3.19).

$$\text{KL-Div}(ng) = P_{\mathcal{LM}(S_1^{j-1})}(ng) * \log \frac{P_{\mathcal{LM}(S_1^{j-1})}(ng)}{P_{\mathcal{LM}(S_1^{|S|})}(ng)} \tag{3.19}$$

Based on Equation 3.19, the sentence level scoring function is the sum of the KL-Div function over all possible n-grams in a sentence.

3.5.1.4 Decoder Based Functions

Decoder based scoring functions require a full SMT system to be trained over the seed corpus, the selection pool or both. If the goal is to select from a parallel pool of sentences, the simple approach is to use a full SMT system trained over the seed corpus to translate sentences in the selection pool. The scoring function for a sentence is based on the error or the distance between the produced translation and the target side of the sentence pair. The idea is that if the SMT system makes an error on a sentence, the sentence is likely to contain useful information. Most of the functions introduced in this category assume the selection pool is monolingual and therefore use other means to measure the likelihood that the decoder will produce an erroneous translation.

Haffari uses an SMT trained on the seed corpus to translate the monolingual selection pool. The scoring function is obtained by normalizing the model score produced by the decoder using sentence length. One drawback of this scoring function is that, although decoder score is an objective score when comparing translations of the same source sentence, it is not a comparable score when considering different source sentences. However, significant differences in this score can be a meaningful measure and can effectively be used for data selection ([44, 4]).

[16] Kullback-Leibler divergence is an information theoretic measure of the distance between two probability distributions.

Another approach to measuring translation error is using **Round Trip Translation Accuracy** (([44])). In this method two SMT systems are trained on the seed corpus, one in each direction. For each sentence in the selection pool, it is translated once from source to target and then from target to source. The scoring function is then the error function between the final translation and the original sentence.

Inter-System Disagreement is another approach developed by Mandal et al. ([71]). In this method different statistical machine translation systems (e.g., a hierarchical system as well as a phrase based system) are trained on a seed corpus. Next, sentences in a parallel development set are translated using all statistical translation systems. For each translation system, an inter-system disagreement error is calculated for its translation output using all other systems' output as references. The target side of the development set is then used to train a linear regression model that predicts the final translation error based on the inter-system disagreement error from each translation system. Once the linear regression model is learned, inter-system disagreement errors are calculated for the selection pool and fed into the linear regression model to produce the final scoring function. The idea is that the sentences where the translation output from different systems differ the most are the most poorly translated sentences and therefore most informative if added to training data with human translation.

The work of Ananthakrishnan et al. differs from all other work in this category, since it does not use a sentence level scoring function ([6]). Instead, they train a sentence level pairwise comparator classifier to sort sentences in the selection pool based on their estimated translation error reduction. The pairwise comparator uses n-gram features to compare sentences. However, we include this work in this section, since a decoder with models trained on the seed corpus is used to create the development set. First, a parallel corpus is required to create the development set. Next, the source side of the parallel corpus is decoded and the TER[17] is calculated for each sentence pair. Ananthakrishnan et al. suggest if the sentence with the highest TER is added to the seed corpus, it will likely reduce the translation error the most. Sorting the sentence pairs in the development

[17]Translation Edit Rate (TER) measures the amount of editing a human would need to perform on a machine translation output to exactly match a reference translation ([90]).

set according to their TER score produces the final development set. They train the pairwise comparator weights to best match the order of sentences in the development set. After the comparator is trained, it is used to sort the sentences. The scoring function in this case is the rank of each sentence in the sorted list.

Finally, Kauchak attempts to estimate the contribution of each sentence pair in the selection pool by sampling a large number of random subsets of the selection pool with replacement and training a statistical machine translation systems over each subset of the training data. After calculating the performance (e.g., BLEU score) of each system, a contribution score is estimated for reach sentence pair based on its membership in different subsets and the performance of those subsets. This method is very computationally intensive and its use is not feasible in practical scenarios. However, it can be calculated on small data sets and used as an oracle score to evaluate other data selection features or tune hyper parameters for various data selection methods ([51]).

3.5.2 Context Independent Functions

Context independent functions have mostly been inspired by features used in the area of Translation Quality Estimation ([98]). In Translation Quality Estimation, given a sentence and its translation, the quality of the translation is estimated in the absence of a reference translation. We have grouped these functions based on the statistic or model that is required for their computation.

3.5.2.1 Language Model Based Functions

Language models can be used in different ways to assess the usefulness of a sentence or sentence pair. Denkowski et al. use an external source and target language model trained on clean data to filter out ungrammatical sentences (Equation 3.20). In this case the scoring function is the length normalized language model score of the sentence ([32]). Allauzen et al. use the language model perplexity score of the sentence as its scoring function to filter out foreign language sentences ([3]) (Equation 3.21).

$$f_4(s_i) = P_{\mathcal{LM}(S_{\text{clean}})}(s_i) \tag{3.20}$$

$$f_5(s_i|S_{\text{clean}}) = 2^{H^{ind}_{\mathcal{LM}(S_{\text{clean}})}(s_i)} \tag{3.21}$$

$$H^{ind}_{\mathcal{LM}(S)}(s_i) = -P_{\mathcal{LM}(S)}(s_i) * \log P_{\mathcal{LM}(S)}(s_i) \qquad (\textit{individual entropy})$$

Yasuda et al. use a language model trained on an in-domain[18] development set to score sentences in the selection pool ([108]). In this approach the scoring function equals the geometrical average of the source and target entropy[19] of the sentence pair according to the in domain language models (Equation 3.20).

$$f_6(s_i|S_{\text{in}}) = \sqrt{H^{ind}_{\mathcal{LM}(S_{\text{in-src}})}(s_i) * H^{ind}_{\mathcal{LM}(S_{\text{in-tgt}})}(s_i)} \tag{3.22}$$

Since the language model used in computing the entropy is an estimate of the true probability distribution function, cross-entropy[20] is used instead. A uniform probability distribution function is used to estimate cross-entropy. The goal in this method is to give sentences similar to the in-domain development set a higher score. However, the shortcoming of this method is that it will also give generally popular sentences (that are also popular in the in-domain development set) a high score as well. This is not desirable in a domain-adaptation data selection task. Moore and Lewis address this issue by introducing a second language model trained on a general-domain development set ([74]). In their approach, they use the **cross-entropy difference** of the two language models to score sentences (Equation 3.23).

$$f_7(s_i|S_{\text{in}}, S_{\text{out}}) = H^{ind}_{\mathcal{LM}(S_{\text{in}})}(s_i) - H^{ind}_{\mathcal{LM}(S_{\text{out}})}(s_i) \tag{3.23}$$

[18] The seed corpus can be used for this purpose as well.

[19] Entropy, $H(X)$, measures the amount of information or uncertainty in a random variable ([72]).

[20] Cross-Entropy of a random variable, X, with the true probability distribution function of $P(X)$ and an estimated probability distribution function of $Q(X)$ is formally defined as the sum of the entropy of X plus the KL-divergence between $P(X)$ and $Q(X)$: $H(P(X), Q(X)) = H(X, P(X)) + D_{KL}(P(X)||Q(X))$ ([30]).

Axelrod et al. extend the cross-entropy difference method to include the target side of parallel data ([10]). In their method, **bilingual cross-entropy difference**, two separate in-domain and general-domain language models are trained for the source and target language. For a given sentence pair, its score is calculated by adding the cross-entropy difference of source and target together (Equation 3.24).

$$f_8(s_i|S_{\text{in}}, S_{\text{out}}) = f_7(s_{isrc}|S_{\text{in-src}}, S_{\text{out-src}}) + f_7(s_{itgt}|S_{\text{in-tgt}}, S_{\text{out-tgt}}) \tag{3.24}$$

3.5.2.2 Alignment Model Based Functions

Alignment[21] based scoring functions are only applicable to parallel training data selection. These scoring functions attempt to quantify the translation quality and accuracy of the sentence pair. Any word alignment model (e.g., HMM-based word alignment model [99]) can be used to compute this scoring function by aligning each word in a sentence to corresponding word(s) in its pair. A number of scoring functions can be computed using this alignment model.

Khadivi and Ney suggest training an alignment model[22] on all sentences in the selection pool. The score for each sentence equals the average Viterbi alignment probabilities according to source-to-target and target-to-source alignment models (Equation 3.25).

[21] A word alignment model is a statistical model trained and used to align individual words in a sentence to their translations in the translated sentence. A word-aligned sentence pair contains alignment links used to align words in one sentence to the other ([23]).

[22] In their work, Khadivi and Ney use IBM model 1, HMM and IBM model 2 in succession to train the final model. But, their scoring function does not depend on any particular alignment model. Hence, any alignment model can be used.

$$f_9((s_i)_1^m,(t_i)_1^n) = \frac{1}{m}\log\max_{a_1^m} P_{\mathcal{AM}}\left((s_i)_1^m, a_1^m|(t_i)_1^n\right) + \frac{1}{n}\log\max_{a_1^n} P_{\mathcal{AM}}\left((t_i)_1^n, a_1^n|(s_i)_1^m\right) \quad (3.25)$$

$(s_i)_1^m$: words 1 through m in sentence s_i.

a_1^m : target word alignment link for source words 1 through m.

Taghipour et al. introduce **alignment entropy** as a scoring function. Alignment entropy, as defined in Equation 3.26, is a smooth measure of uniform distribution of alignment links. In other words, given an alignment link count of n, how uncertain the model is about predicting the word corresponding to the link[23]. The highest value for alignment entropy is achieved when all words have the same number of alignment links associated with them.

$$f_{10}(s_i,t_i) = \frac{-1}{n}\sum_{(s_i)_1^n} p_1((s_i)_1^n) * \log p_1((s_i)_1^n) * \frac{-1}{m}\sum_{(t_i)_1^m} p_1((t_i)_1^m) * \log p_1((t_i)_1^m) \quad (3.26)$$

$$p_1((s_i)_k) = \frac{a((s_i)_k)}{\sum_k a((s_i)_k)}$$

$a((s_i)_k)$: The number of links that end on word $(s_i)_k$

They also use a number of features inspired by the work of Munteanu and Marcu on "exploitation of non-parallel corpus for machine translation" based on number of alignment links and word

[23] We think a more natural derivation for alignment entropy of a sentence is averaging alignment entropies of each source or target word according to all possible alignments. This will provide a measure of how uncertain an alignment model is about word alignments in a sentence. This is a more natural use of entropy compared to the uncertainty about a word given the number of Viterbi alignment links for the word.

fertility[24].

$$f_{11}(s_i) = N_{src}(s_i) - N_{tgt}(s_i) \tag{3.27}$$

$$f_{12}(s_i) = \frac{N_{src}(s_i)}{|s_i|} \tag{3.28}$$

$$f_{13}(s_i) = \frac{N_{tgt}(s_i)}{|s_i|} \tag{3.29}$$

$$f_{14}(s_i) = \arg\max_{w_i \in s_i} fertility(w_i) \tag{3.30}$$

$N_{src}(s)$: Number of null aligned source words in sentence s

$N_{tgt}(s)$: Number of null aligned target words in sentence s

Denkowski et al. add the following scoring functions to the list above ([32]).

$$f_{15}(s_i) = \frac{A_{src}(s_i)}{|s_i|} \tag{3.31}$$

$$f_{16}(s_i) = \frac{A_{tgt}(s_i)}{|s_i|} \tag{3.32}$$

$A_{src}(s)$: Number of aligned source words in sentence s

$A_{tgt}(s)$: Number of aligned target words in sentence s

Ambati uses **bidirectional alignment scores** to provide the score for a single alignment in a sentence pair ([4]). Denkowski et al. assign a sentence pair score by averaging these alignment scores ([32]) (Equation 3.33).

$$f_{17}(s_i, t_i) = \sqrt{P_{\mathcal{AM}}(s_i|t_i) * P_{\mathcal{AM}}(t_i|s_i)} \tag{3.33}$$

In addition, Ambati propose using the distance between the bidirectional alignment scores for the highest probability alignment and the second highest probability alignment for a source or target word as a measure of alignment uncertainty. The closer the distance, the more uncertain the alignment model is about the alignment for the word in question (Equation 3.34). Although, they

[24]In word alignment, fertility of a word is defined as the number of alignment links initiated from that word.

have not extended this scoring function to the sentence level, an average over all source or target words can be used to provide a sentence level scoring function.

3.5.2.3 Other Scoring Functions

We provide a list of sentence level scoring functions that do not fit the scoring function categories above. These include length-based functions, character-based functions and functions indicating literalness of translation. **Length-based functions** are primarily used to filter out noisy data. In the following length based functions, sentence length can be calculated as the number of tokens/words or number of characters. A development set can be used to tune thresholds for these length based functions [52, 95]. Alternatively, an unsupervised outlier threshold can be developed based on the candidate sentence pool itself ([32]).

1. Source sentence to target sentence length ratio and vice versa ([52, 95, 32])
2. Difference of source and target sentence lengths [95])
3. Sentences length ([32])
4. Length of longest token in the sentence ([32])

A few other simple scoring functions can also be used to filter out noisy data.

1. Sentence end punctuation agreement ([52])
2. Percentage of alphanumeric characters ([52, 32])
3. Existence of control characters and invalid unicode characters ([32])
4. Co-occurrence of special tokens such as email address, URL, date or number on both sides of a sentence pair ([95]).

In addition to the functions above, Han et al. introduce two functions to assess lexical and syntactic compatibility of a sentence pair ([45]). Their approach is to assign scores to sentence

pairs according to the potential for a statical machine translation to learn form the sentence pair. The idea is that sentence pairs with literal translations are more likely to be useful compared to an abstract or conceptual translation. First, they attempt to estimate the literalness of a translation given a sentence pair using lexical features. This feature is calculated by expanding the words of source and target sentences by a synonym dictionary and then counting the number of words that are translations of each other given a bilingual dictionary. They normalize this value by the total number of words and use it as the function score (Equation 3.34).

$$f_{18}(s_i,t_i) = \frac{\left|\left(\bigcup_k \underset{src\rightarrow tgt}{\text{BiDic}}\ \underset{src}{\text{Synonyms}}(s_{i_k})\right) \bigcap \left(\bigcup_k \underset{tgt}{\text{Synonyms}}(t_{i_k})\right)\right|}{\left|\bigcup_k \underset{tgt}{\text{Synonyms}}(t_{i_k})\right|} + \frac{\left|\left(\bigcup_k \underset{src}{\text{Synonyms}}(s_{i_k})\right) \bigcap \left(\bigcup_k \underset{tgt\rightarrow src}{\text{BiDic}}\ \underset{tgt}{\text{Synonyms}}(t_{i_k})\right)\right|}{\left|\bigcup_k \underset{src}{\text{Synonyms}}(s_{i_k})\right|} \tag{3.34}$$

$\underset{src\rightarrow tgt}{\text{BiDic}}(w_i)$: Translations for word w_i according to a given dictionary.

$\underset{src}{\text{Synonyms}}(w_i)$: Synonyms for word w_i accoring to a given dictionary.

s_{i_k} : k^{th} word in sentence s_i

In their second approach, Han et al. use part of speech tags to evaluate the grammatical compatibility of a sentence pair ([45]). First, they manually map common POS tags of the two languages (e.g., the "noun" POS tag in English is mapped to the "noun" POS tag in Chinese). For a given sentence pair, the number of occurrences of each source tag and its target counterpart tag is calculated. Their distance is computed using the ratio of the smaller count to the larger count . A weighted average[25] of the distance measures for all common tags is used to calculate the cross-lingual grammatical compatibility (Equation 3.35).

[25]The weights are trained using a manually analyzed development set of size 1000.

$$f_{19}(s_i,t_i) = \sum_{tag} \lambda_{tag} \frac{\min\left(\text{Count}(s_i,tag),\text{Count}(t_i,tag)\right)+1}{\max\left(\text{Count}(s_i,tag),\text{Count}(t_i,tag)\right)+1} \tag{3.35}$$

3.6 Feature Combination and Parameter Tuning

Some data selection scenarios and scoring functions require parameter tuning.

1. Scoring function parameter(s): Some scoring functions contain a parameter that needs to be tuned (e.g., [45]).

2. Feature Combination: Since different scoring functions capture different aspects of the data, it is often desirable to combine more than one function for data selection. For this purpose a combination model (e.g., linear) is required along with an algorithm to find optimum model parameters (e.g., [52]).

3. Threshold tuning: Data selection methods that use a threshold to filter/select data, require the threshold parameter to be tuned (e.g., [32]).

A common requirement for parameter tuning and feature combination is the availability of a *development set*. In a supervised learning scenario, human annotators are used to create the development set. For example, Han et al. use human annotators to label sentence pairs as literal translations versus conceptual translations to create a development set. Since this is an expensive and time consuming task, alternative techniques for development data creation in semi-supervised learning methods have been developed ([52, 95, 7]).

1. **Mixing clean and noisy data**: In their work, Khadivi and Ney add noisy data to clean data (labeled accordingly) as the development set to train their noise classification algorithm.

2. **Single Pass Active Learning**: In their effort to train weights for a pairwise sentence comparator classifier, Ananthakrishnan et al. use a parallel corpus ordered by TER[26] (between

[26]TER: Translation Edit Rate [90]

reference translations and translations of decoder using models trained on seed corpus) as their development set ([6]). Haffari uses a similar method to generate a development set to train their feature combination model weights ([43]).

3. **Using Selection Pool**: Denkowski et al. use the candidate selection pool of sentences to compute outlier detection parameters ([32]). They calculate average and standard deviation on the selection pool for a number of features (e.g., sentence length ratio and alphanumeric density) to filter out outliers.

3.7 Selection Algorithms

Given a constrained optimization problem formulation, consisting of scoring functions, a combination model and size constraints, a selection algorithm is used to solve the optimization problem.

3.7.1 Threshold Based Filtering

A context independent function can be computed for all sentence pairs in one pass and the subset of training data that passes a certain threshold can be selected in linear time. Methods that depend on a threshold tune the threshold prior to filtering. For example, Denkowski et al. sets the thresholds for length based filtering at $\text{avg}(s_1^N) \pm 2 * \text{std dev}(s_1^N)$. Other selection methods with context independent scoring functions iteratively select highest scoring functions until the constraint is met. These methods require the sentences to be sorted prior to filtering. For a linear filtering time, radix sort can be used. The work of Moore and Lewis is an example of constraint based filtering as sentences are sorted based on their cross-entropy difference before the filtering is applied. Classification based approaches such as [95, 52] follow the same paradigm. Since these approaches combine multiple features, they train a classifier on a development set and then apply the classifier to the selection pool to label sentences as selected versus not selected, or clean versus noisy.

3.7.2 Greedy Search

The selection method for most methods with context dependent scoring functions is greedy search. The highest scoring sentence is selected in each iteration and the models used for the scoring functions are updated. This process is repeated until a specified constraint (e.g., the total number of words in selected pool or a minimum quality bar) is met. [33], [44] and [6] are examples of this selection method. Due to high computational complexity, for practical purposes, greedy selection algorithms are often paired with batch-learning techniques (see Section 3.9).

3.7.3 Submodular Optimization

Kirchhoff and Bilmes have recently introduced submodularity for data selection in statistical machine translation ([53]). Submodular functions are a class of set functions that formalize the concept of "diminishing returns". This makes them a natural theoretical framework for data selection. Formally, a function, $\mathcal{F}$, is called submodular if adding an instance x to a set S increases $\mathcal{F}(S)$ more than (or equal) if it was added to a superset of S, $S \subset S'$ (Equation 3.36).

$$\mathcal{F}(S \cup \{x\}) - \mathcal{F}(S) \geq \mathcal{F}(S' \cup \{x\}) - \mathcal{F}(S') \tag{3.36}$$

In general, solving Equation 3.4 exactly is NP-complete. However, if the set function $\mathcal{F}$ is submodular the problem can be solved using a greedy algorithm with worst-case solution of $\mathcal{F}(X^*) > (1 - 1/e) * f(X_{opt})$. That is, in the worst case, the function value for the solution from the greedy algorithm is approximately 0.63 of the optimum solution [53]. Moreover, depending on the curvature of the submodular function, the worst case guarantee can be improved. Graph based submodular functions are a natural fit for data selection as each sentence is represented as a node in the graph and edge weights are computed based on the similarity of the two sentences. The order of graph construction complexity is quadratic in the number of sentences. This makes this class of submodular functions impractical for large data sets. Another class of submodular functions that are more practical for large data sets is based on bipartite graphs $G = (V, U, E, w)$. In this setting

sentences are the left vertices, V. Features (e.g., n-grams) are the right vertices, U, with weight w. Connecting each sentence to its features adds up to the set of edges, E. In this setting, Kirchhoff and Bilmes define a feature-based submodular function as follows.

$$f(X) = \sum_{u \in U} w_u \phi_u(m_u(X)) \tag{3.37}$$

Assuming w_u to be positive, $m_u(X)$ to be a non-negative modular function[27] and ϕ_u to be a non-negative, non-decreasing concave function, $\mathcal{F}(X)$ as defined in Equation 3.37 is submodular. This is a flexible framework for data selection that is scalable to large data sizes. Using this framework requires defining the components below:

1. U: U is the set of features present in all sentences in the selection pool. N-grams are a natural choice for this.

2. $m_u(x)$: This function calculates the relevance of each feature u to sentence x. Kirchhoff and Bilmes use $tfidf(u,x)$ for this function. The main constraint with this function is that it has to be modular.

3. $w(u)$: Each feature can have a weight function to present its importance. In a domain-adaptation task, the frequency of the n-gram in an in-domain development set can be used for this weight function.

4. $\phi_u(a)$: This decay function determines the rate of diminishing returns for redundant instances of a feature u. As this concave function becomes flat, the feature loses its ability to provide additional improvement to the value of a candidate subset. Kirchhoff and Bilmes experiment with square root and logarithmic functions.

The greedy algorithm proposed by Kirchhoff and Bilmes is similar the greedy algorithms mentioned in Section 3.7.2. Their algorithm chooses the sentence in the sentence pool that, if added to

[27] A set function is modular if and only if the value of the function over a set equals the sum of the function value over its individual elements.

the selected sentences, improves the submodular function's value over the selected sentences the most. This step is repeated until the budget constraint is met.

3.8 Active Learning

Active learning is a technique used in machine learning where the algorithm can choose the new data from which it learns [88]. This technique is often employed in problems where unlabeled data is abundant while the process of obtaining labeled data is expensive or time-consuming. In an active learning framework, the learning algorithm *queries* an *oracle*[28] for *new data points* to be labeled. First, the learning algorithm can obtain *new data points* in several ways.

1. **Pool Based**: The learning algorithm selects data points from a pool of unlabeled data.
2. **Stream Based**: All available unlabeled data is streamed through the learning algorithm and the learning algorithm can choose to query for the data point or discard.
3. **Query Synthesis**: The learning algorithm generates new data points to be labeled by the oracle.

The learning algorithm can use several strategies for selecting new data points such as Uncertainty Sampling, Query by Committee, Expected Model Change, Expected Error Reduction, Variance Reduction and Density-Weighted Methods ([88]). Active learning selects new data points in an iterative process. In each step, it selects new data point(s), queries the oracle for the label and learns from the results. In contrast, in passive learning, the learning algorithm has no contribution to the data selection and labeling process. From another perspective, in passive learning the data selection process does not use the learning algorithm to select new data points for labeling. Although, the data selection task for passive learning can also be iterative, the key distinction is that the learning algorithm does not play a role in the data selection task.

[28] An oracle can be a human or a human labeled data set that provides the true label for a query. In the context of machine translation, the oracle is a human translator or a parallel corpus where the source text has already been translated by humans.

In statistical machine translation a data selection method can be classified as an active learning method only if it follows the iterative data selection process described above. Context Dependent Functions listed in Section 3.5.1 fit this paradigm. In addition, the selection process must use an element of statistical machine translation learning to be considered active learning. Although only some prior work specifically refer to their method as active learning [44, 4, 7], all scoring functions listed in Sections 3.5.1.2, 3.5.1.3 and 3.5.1.4 fit the active learning framework.

3.9 Batch-Mode Learning

Batch-mode learning is applicable to both active learning as well as data selection for passive learning. The idea is to select new data points in batches rather than one at a time. Without batch-mode learning, after a single data point is selected and labeled by the oracle, the learning model (e.g., Language Model or Translation Model) or the data selection scoring function statistic (e.g., n-gram frequency in selected sentences) is updated with the new data point, and all data point scores are recomputed. This process can be very time consuming, especially if SMT models need to be retrained in order to recompute data point scores. Batch-mode learning addresses this concern by selecting multiple new data points at a time and therefore reducing the number of times the learning model has to be updated and sentence scores need to be recalculated. Using the top N sentences with highest scores often does not produce the best results since it fails to consider overlap between data points. This is commonly addressed by minimizing overlap between data points within a batch. To select a batch of size K, Ananthakrishnan et al. go through the data K times[29] ([6]). Each time they select the sentence with maximum score and update the scoring function to exclude features (in this case n-grams) used in scoring the selected sentences.

Inspired by the work of Dasgupta and Hsu on Hierarchical Sampling[30], Haffari et al. pro-

[29] In this batch-mode learning strategy the idea is to avoid retraining all SMT models after selection of each sentence. A less expensive update is used to improve the diversity of the batch where only the scoring function is updated. After the entire batch is selected, then all SMT models are updated.

[30] Hierarchical Sampling attempts to leverage the cluster structure of the data for sampling in an

pose Hierarchical Adaptive Sampling for feature combination and batch-learning at the same time ([43]). Their work differs from Hierarchical Sampling by not having a predefined static hierarchical cluster over the selection pool. Instead, the hierarchy is created dynamically in steps. Unlike Hierarchical Sampling, their algorithm always selects samples from the cluster node that has been selected for further partitioning. At each step, K sentences are randomly chosen from the selected cluster node and queried for human translations. The selected sentences are added to the set of selected sentence pairs and the SMT models are retrained on the larger set of selected sentence pairs. The two points below are essential to understand the rest of this algorithm.

1. Choice of cluster for further partitioning: A cluster node with lowest **average model score** when translated using the updated SMT models is chosen for further partitioning.

2. Partitioning mechanism for a chosen cluster node: The sentences in the partition are sorted according to their **similarity to the selected sentences**. The first α percentage of the sentences are put in the first child cluster node, and the rest in the second child cluster node.

In effect, Hierarchical Adaptive Sampling is combining the "average model score" feature with the "similarity to the selected sentences" feature. While the "average model score" feature selects the cluster that is most difficult to translate, the "similarity to the selected sentences" feature ensures diversity. However, in practice, these two features can be replaced by many of the scoring functions introduced in Section 3.5. In addition to its high computational complexity, another shortcoming of this approach is that it does not ensure diversity within a single batch. In a followup

active learning setting ([31]). In this work, a static hierarchical cluster structure of the unlabeled data is given. A set of cluster nodes for sampling is maintained throughout the algorithm. Initially, the sampling set only contains the root node of the cluster which contains all nodes. Random samples are drawn from the sampling set and queried from the oracle. Based on these queries, each node in the hierarchical cluster maintains statistics about its positive and negative labels. Cluster nodes in the sampling set with mixed labels and more pure child nodes are removed from the sampling set and their child nodes are added. These steps are repeated until all nodes reach a predefined level of purity. This method is motivated by addressing the "sampling bias" problem ([87]) in active learning and provides theoretical guarantees for a better learning performance than random sampling.

to their work referenced above, Ananthakrishnan et al. introduce three levels of scoring functions ([7]):

1. **Never Updated**: These functions are only computed once for sentences in the selection pool and are never updated. They use n-gram overlap with an in-domain development set as a measure of domain relevance for this purpose.

2. **Updated Per Batch**: These functions are updated after a batch of sentences are selected and queried for human translation. They compute translation difficulty by retraining the SMT models with the new batch and derive translation difficulty features based on the performance of new models for the next batch of sentences.

3. **Updated Per Sentence**: These functions are recomputed every time a single sentence is added to the pool. N-gram overlap with existing sentences in the batch (batch diversity) is used for this purpose and has to be updated per sentence.

Although only batch diversity is recomputed after the selection of each sentence, Ananthakrishnan et al. use all three functions above to select the highest scoring sentence in each step. This is continued until the predefined batch size is reached.

3.10 Evaluation Methods

All prior work evaluate the effectiveness of data selection methods by training on a selected parallel corpus and testing on a randomly held-out test set. In data selection for domain adaptation scenarios ([74, 8] the test set is an in-domain test set. As for most statistical machine translation tasks, BLEU is the most common evaluation method. While in some cases selecting a subset of the data outperforms cases using the entire data set [53, 33], the general trend is that selecting more data improves evaluation results. For this reason, comparison between different selection methods is often done by running each selection algorithm multiple times. Each time the selection constraint (e.g., maximum number of words selected) is set to a different percentage of the data.

An SMT system is trained on each selected subset and tested on the held-out test set. This provides several comparison points between the selection methods. The objective in cost focused selection methods is to meet a predefined quality bar at minimum cost ([19]). In these scenarios, different selection methods are compared based on their cost to reach a predefined BLEU score.

3.11 Comparative Summary of Cited Work

The work of Eck et al. is noteworthy as the first work published on data selection for statistical machine translation ([33]). Features introduced in this work such as n-gram coverage and $tf - idf$ are still the basis for much of the most recent work. A practical extension of their work is the Vocabulary Saturation Filter ([66]) which relaxes the dependency on previously selected sentences; this allows for linear-time application of n-gram coverage to very large data sets. Another extension to the work of Eck et al. is using feature decay functions ([5, 18]) to implement the idea of diminishing returns. While in their original work, Eck et al. maintained n-gram count statistics, other work ([44, 68]) used language models for more accurate probability estimation instead.

Moore and Lewis introduce cross-entropy difference which has become one of the most commonly used approaches in data selection in the domain adaptation setting [74]. Bilingual cross-entropy difference is a natural extension of this work that takes both sides of the parallel data into consideration ([8]).

The work of Haffari et al. is uniquely objective in that they use the performance of a translation system trained on the selected data in translating new data as a measure of translation difficulty and thus the usefulness of new data ([43]). Their approach to active learning is computationally intensive. In their first extension to this work, Ananthakrishnan et al. address the computational intensity problem by training a classifier to compare sentences based on their translation difficulty. In a follow up work they combine the approaches of Eck et al., Moore and Lewis and Haffari et al. to develop a multi-layer active learning strategy that combines translation difficulty, domain appropriateness and diversity into a single framework ([7]).

The work of Kirchhoff and Bilmes offers the most effective data selection search framework

with theoretical guarantees ([53]). Although they do not introduce any new features in their submodular optimization method for data selection, they offer a flexible framework where custom functions can be used for sentence features, relevance scores, feature weights and a decay function.

The work of Ambati et al. is unique in that, they introduce a customized active learning strategy for crowdsourcing while introducing new features such as phrasal entropy and KL-divergence [4]. Bloodgood and Callison-Burch offer a cost focused active learning strategy by asking a crowd for translation of specific phrases within a sentence instead of full sentence translation ([19]). Using this method they are able to outperform all previous approaches on a per word translation pricing scheme.

Taghipour et al. and Khadivi and Ney use a classifier based approach to filter out noisy data while Denkowski et al. use simple average and standard deviation statistics to achieve the same goal ([95, 52, 32]).

Han et al. have a unique approach to data selection where they use lexical and syntactic compatibility between source and target sentences to estimate the literalness of the translation ([45]). Their idea is to select more literal translations as a statistical machine translation system can only learn from literal translations and would not benefit from free translations.

Finally the work of Kauchak is unique as they provide the most objective measure of sentence pair usefulness for SMT, although their method is not practical for any real-world scenario ([51]). In their work, they create 500 random subsets from the selection pool to train 500 different sets of models. The usefulness of each sentence pair is estimated based on its membership in training data for different model sets and the model sets' performance.

Related works in the literature have been listed in Table 3.2 according to the main components of training data selection listed.

Reference	Scenario	Features	Algorithm
[33]	Low Resource Language[31]	N-gram Coverage TF-IDF	Greedy Search
[44]	Low Resource Language	Round Trip Translation Accuracy Phrase Pair Utility N-gram Utility N-gram Coverage Decoder Translation Model Score	Greedy Search HAS[32]
[71]	Low Resource Language	Inter-System Disagreement Language Models' Perplexity Ratio	Greedy Search Threshold Filtering
[61]	Domain Adaptation	TF-IDF	Threshold Filtering
[95]	Noise Reduction	Word Alignment Probability Word to Null Alignment Count Word to Null Alignment Ratio Alignment Entropy Top-3 Word Fertilities Factoid Co-Appearance Sentence Length Ratio LM Probability Difference LM Probability Ratio	Classifier
[32]	Noise Reduction	Language Model Score Combined Alignment Score Length Ratio Alphanumeric Intensity Maximum Token Length	Threshold Filtering
[6]	Low Resource Language	Expected Translation Error Reduction	Greedy Search Classification
[45]	Noise Reduction	Lexical Compatibility Grammatical Compatibility	Threshold Filtering
[104]	Training Resource Reduction	TF-IDF	Submodular Opt.
[69]	Training Resource Reduction	Weighted Phrase Entropy Weighted Sub-tree Entropy	Greedy Search
[74]	Domain Adaptation	Cross Entropy Difference	Threshold Filtering
[27]	Training Resource Reduction	N-gram Co-Occurrence	Greedy Search
[9]	Domain Adaptation	Cross Entropy Difference	Threshold Filtering
[40]	Training Resource Reduction	Average Alignment Probability	Greedy Search
[4]	Low Resource Language	Phrase Probability Unseen N-grams	Greedy Search
[52]	Noise Reduction	Sentence Length Ratio Alphanumeric Character Presence Sentence Ending Symbol Similarity Automatic Language Identification Alignment Probability	Classifier
[19]	Quality Improvement	Unseen N-Grams	Threshold Filtering
[51]	Low Resource Language	Estimated Contribution Score	Threshold Filtering Greedy Search
[80]	Noise Reduction	Minimum N-Gram Occurrence Sentence Length	Greedy Search
[3]	Noise Reduction	LM Perplexity	Threshold Filtering
[108]	Domain Adaptation	LM Perplexity	Threshold Filtering
[34]	Quality Improvement	N-gram Co-Occurrence	Greedy Search
[66]	Training Resource Reduction	Minimum N-Gram Occurrence Alignment Probability	Threshold Filtering

Table 3.2 Related Work

3.12 Related Research Areas

Data selection in machine translation is closely related to **quality estimation**. In a quality estimation task, given a sentence and its translation using an SMT decoder, the quality of the translation is estimated without a reference translation ([91]). This is a very similar task to data selection. In fact many features used in data selection have been inspired by features used in quality estimation ([32]).

Data selection is a practical method of adapting statistical machine translation models to a domain and is often used in **domain adaptation** tasks. Other domain adaptation techniques include translation model adaptation ([28]).

Most of the **active learning** literature in machine learning is focused on classification tasks with a limited number of features. Therefore, the focus is on selecting data points that are most useful in learning feature weights. Active learning for machine translation is unique in that the number of features to be learned is very large or, for all practical purposes, unlimited for large data sets. The active learning task in this case is to identify data points with the most useful features in addition to finding data points that are the most useful in estimating the weights for the features. This makes active learning for machine translation unique in the active learning literature.

Co-training and **self-training** are also closely related areas to data selection for machine translation ([107, 43]). Both techniques are used when labeled data is scarce and unlabeled data is abundant. In a simple co-training setting, two classifiers trained on labeled data classify the same unlabeled data set using two different but complimentary feature sets. If the classifiers agree in their labels, the data points and their labels are added as training data and classifiers are retrained. In self-training, unlabeled data is labeled using a single classifier trained on the limited labeled data. Data points with high classification confidence are added to the training as new data points. These techniques have been leveraged in data selection for machine translation as well. In his work, Haffari experiments with self-training while the work of [71] uses features that are closely

[31] Low Resource Language

[32] Hierarchical Adaptive Sampling

related to co-training.

Finally, another related research area is **model pruning** in statistical machine translation. In data selection scenarios with model size deployment constraints model pruning can be used as an alternative to data selection. The research in this area focuses on **phrase table pruning** ([106]) and **language model pruning**([41]).

3.13 Summary

In practice, performing a data selection task for statistical machine translation requires addressing two technical questions:

1. **Scoring Function:** How is the value of a sentence or sentence pair objectively evaluated?

2. **Selection Algorithm:** Given a scoring function and a constraint how is the optimum data subset selected?

In this chapter we reviewed a long list of objective functions along with a number of selection algorithms. This combination offers an overwhelming number of different configurations to perform a data selection task. However, in practice, the number of applicable solutions can be narrowed down significantly by the application scenario (Section 3.2) and data size.

Multiple data selection scenarios are often employed in the development of a single statistical machine translation system. For example, when developing a general purpose machine translation service for English↔French there are three different issues with training data that can be addressed using data selection.

1. Much of the available parallel data is collected through web crawling and is noisy.

2. There is too much parallel data to train and iterate on in a timely manner ([66]).

3. Even if we could train on all the data, if we were to deploy the SMT system on an offline mobile device, the model sizes are too large.

In this scenario the data can be first cleaned using context independent scoring functions listed in Equation 3.20-3.33 and Section 3.5.2.3. Investing in creating a development set to train a classifier ([95]) would enable training weights for multiple scoring functions. If few scoring functions are used, using average and standard deviation for threshold based filtering ([32]) can be effective as well.

The noise level of a sentence pair can be estimated independently using context independent scoring functions. However, when there is too much clean data, selecting a subset of the data that can perform the best on a blind test set is more complicated. Active learning provides an ideal solution as it is the most objective. On the other hand it requires retraining a full SMT system on every iteration which makes it impractical for large data sets. Active learning inherently uses context dependent scoring functions since it uses models built from the selected pool to evaluate new sentences from the selection pool. Other context dependent scoring functions ([33]) use incremental statistics from the selected pool to score new sentences in the selection pool. A general and efficient framework for the use of context dependent scoring functions is submodular optimization ([53]). For very large data sets that a computational complexity of $O(N \times \log N)$ is not practical, VSF[33] ([66]) offers a linear solution. The deployment resource restriction scenario is similar to training resource limitation scenario with the exception that phrase table or language model pruning is a viable alternative to data selection.

Another common scenario for the data selection task in SMT is developing an SMT system for low resource languages. In this scenario there is a small parallel seed corpus and a large monolingual selection pool. The idea is to select a limited number of sentences from the selection pool to be manually translated by humans and added to the seed corpus. In this scenario active learning strategies have proved to be the most successful ([7, 4, 44]. The active learning strategies can be used with any of the context dependent scoring functions reviewed in Section 3.5.1 while batch-mode learning strategies reviewed in Section 3.9 can be used to speed up the data selection process.

[33]VSF: Vocabulary Saturation Filter ([66])

CHAPTER 4

TRANSLATION DIRECTION DETECTION

The final goal of this dissertation is to select a subset of parallel training data that results in translation models that yield highest quality translation output. As described in Chapter 3 this task is often broken down to sentence level feature functions. By definition, a selected parallel data subset cannot be immediately and independently evaluated without relying on the downstream translation model training task and test set. The downstream task itself can take on many variations (e.g., training a phrase based system [109] versus hierarchical phrased based translation model [29] or tree-to-string translation model [85]) with many tuning parameters (maximum phrase length, maximum distortion, pruning parameters, ...) in addition to the choice of development set and turning method (e.g., PRO [47], MERT [79] or MIRA [102]). Finally, there are different evaluation methods (BLEU [83], METEOR [12] or human evaluation [83]). In addition to all variations in training a statistical machine translation system, the training process is also computationally intensive. The aforementioned factors makes an objective, fair and accurate comparison between a wide range of feature functions intractable. Therefore in this chapter we selected a classification task for parallel data that can be objectively, independently and accurately evaluated without the need for a any downstream task. This enables us to explore a wide range of sentence pair feature functions in different settings (same domain versus cross-domain) and accurately measure their effectiveness. The only task that meets the criteria mentioned above is the "Translation Direction Detection" task. In this chapter, we define the task, review related work, introduce a range of new features and evaluate their effectiveness in an existing data set. In addition, we develop a three way cross-domain data set and evaluate the effectiveness of previously introduced features as well as new feature in this cross-domain setting. The best performing features are used in Chapter 5 and Chapter **??** to improve the performance of the data selection task for statistical machine translation.

4.1 Introduction

It has been known for many years in linguistics that translated text has distinct patterns compared to original or authored text [11]. The term "Translationese" is often used to refer to the characteristics of translated text. Patterns of Translationese can be categorized as follows [101]:

1. **Simplification**: The process of translation is often coupled with a simplification process at several levels. For example, there tends to be less lexical variety in translated text and rare words are often avoided.

2. **Explicitation**: Translators often have to be more explicit in their translations due to lack of the cultural context that speakers of the source language have. Another manifestation of this pattern is making arguments more explicit which can be observed in the heavy use of cohesive markers like "therefore" and "moreover" in translated text [59].

3. **Normalization**: Translated text often contains more formal and repeating language.

4. **Interference**: A translator is likely to produce a translation that is structurally and grammatically closer to the source text or their native language.

In Figure 4.1 the size of a word in the "Translated" section is proportional to the difference between the frequency of the word in original and in the translated text [37]. For example, it is apparent that the word "the" is over-represented in translated English as noted by other research [101]. In addition, cohesive markers are clearly more common in translated text.

In the past few years there has been work on machine learning techniques for identifying Translationese. Standard machine learning algorithms like SVMs [13] and Bayesian Logistic Regression [59] have been employed to train classifiers for one of the following tasks:

i. Given a chunk of text in a specific language, classify it as "Original" or "Translated".

ii. Given a chunk of translated text, predict the source language of the translation.

iii. Given a text chunk pair and their languages, predict the direction of translation.

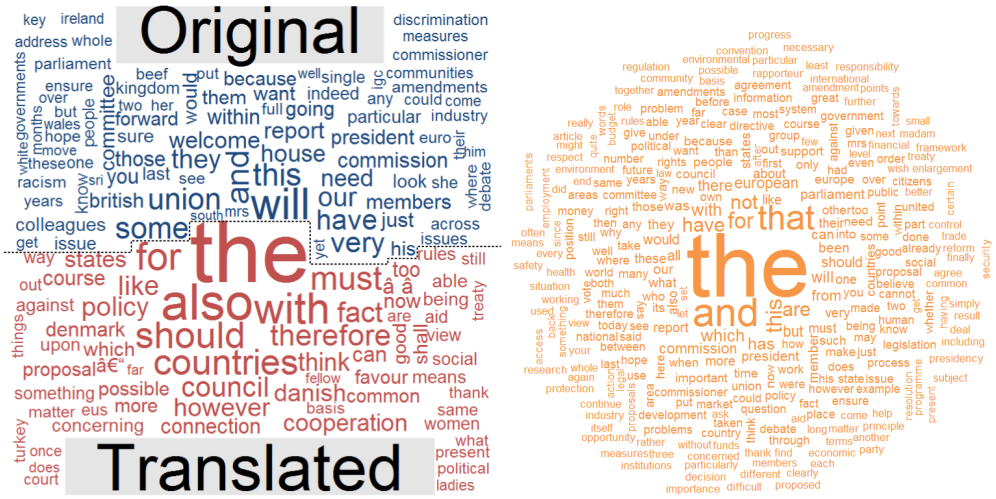


Figure 4.1 EuroParl Word Cloud Data Visualization (Translated vs Original) [1]

In addition to our stated motivation of exploring the effectiveness of different feature functions in parallel data classification, there are two motivations stated in prior research for the tasks above: first, empirical validation of linguistic theories about Translationese [101], and second, improving statistical machine translation by leveraging the knowledge of the translation direction in training and test data [63, 65, 64]. Few parallel corpora including a customized version of EuroParl [48] and a processed version of Hansard [60] are labeled for translated versus original text. Using these limited resources, it has been shown that taking the translation direction into account when training a statistical machine translation system can improve translation quality [65]. However, improving statistical machine translation using translation direction information has been limited by several factors.

1. **Limited Labeled Data**: The amount of labeled data is limited by language and domain and therefore by itself is not enough to make a significant improvement in statistical machine translation.

2. **Cross-Domain Scalability**: Current methods of Translationese detection do not scale across

[1] This word cloud was created using the *wordcloud* and *tm* **R** packages [37] from EuroParl parallel data annotated for translation direction [48] obtained from http://www.hucompute.org/ressourcen/corpora/56.

different corpora. For example, a classifier trained on EuroParl corpus [56] had in-domain accuracy of 92.7% but out-of-domain accuracy of 64.8% [59].

3. **Text Chunk Size**: The reported high accuracy of Translationese detection is based on relatively large (approximately 1500 tokens) text chunks [59]. When similar tasks are performed at the sentence level the accuracy drops by 15 percentage points or more [60]. Figure 4.2 shows how detection accuracy drops with the reduction of the input text chunk size. Since parallel data are often available at the sentence level or small chunks of text, existing detection methods aren't suitable for this type of data.

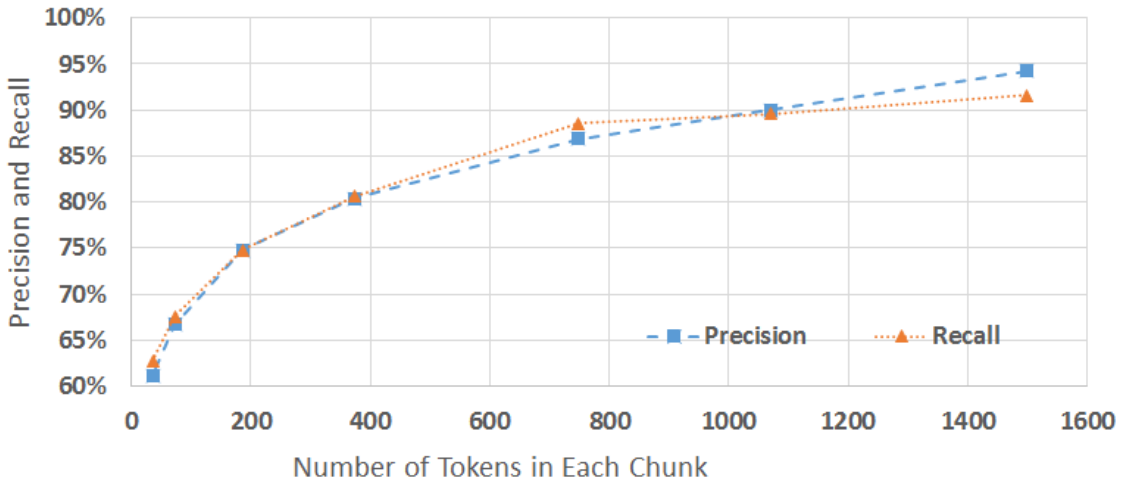


Figure 4.2 Effects of Chunk Size on Translationese Detection Accuracy[2]

Motivated by these limitations and the aforementioned goal of this chapter, in this work we focus on improving sentence-level classification accuracy by using non-domain-specific bilingual features at the sentence level. In addition to improving accuracy, these fine-grained features may be better able to confirm existing theories or discover new linguistic phenomena that occur in the translation process. We use a fast linear classifier trained with online learning, Vowpal Wabbit [62]. The Hansard French-English dataset [60] is used for training and test data in all experiments.

[2] This is a reproduction of the results of Koppel and Ordan [59] using function word frequencies as features for a logistic regression classifier. Based on the description of how text chunks were created, the results of the paper (92.7% accuracy) are based on text chunk sizes of approximately 1500 tokens.

4.2 Related Work

Volansky et al. [101] provide a comprehensive list of monolingual features used for Translationese detection. These features include POS *n*-grams, character *n*-grams, function word frequency, punctuation frequency, mean word length, mean sentence length, word *n*-grams and type/token ratio. While distinct patterns of Translationese have been studied widely in the past, the work of Baroni and Bernardini [13] is the first to introduce a computational method for detecting Translationese with high accuracy. Prior work has shown in-domain accuracy can be very high at the chunk-level if fully lexicalized features are used [101], but then the phenomena learned are clearly not generalizable across domains. For example, in Figure 4.1, it can be observed that content words like "commission", "council" or "union" can be used effectively for classification while they do not capture any general linguistic phenomena and are unlikely to scale to other corpora. This is also confirmed by an average human performance of 72.7% precision with 82.1% recall on a similar task where the test subjects were not familiar with the domain and were not able to use domain-specific lexical features [13]. A more general feature set still with high in-domain accuracy is POS tags with lexicalization of function words [13, 60]. We build on this feature set and explore bilingual features.

We are aware of only one prior work that presented a cross-domain evaluation. Koppel and Ordan [59] use a logistic regression classifier with function word unigram frequencies to achieve 92.7% accuracy with ten fold cross validation on the EuroParl [56] corpus and 86.3% on the IHT corpus. However testing the EuroParl trained classifier on the IHT corpus yields an accuracy of 64.8% (and the accuracy is 58.8% when the classifier is trained on IHT and tested on EuroParl). The classifiers in this study are trained and tested on text blocks of approximately 1500 tokens, and there is no comparative evaluation of models using different feature sets.

We are also aware of one prior works that investigate Translationese detection accuracy at the sentence level. Kurokawa et al [60] use the Hansard English-French corpus for their experiments. For sentence level translation direction detection they reach F-score of 77% using word *n*-grams and stay slightly below 70% F-score with POS *n*-grams using an SVM classifier. This is also

the only work to consider features of the two parallel chunks (one original, one translated). They simply used the union of the n-gram mixed-POS[3] features of the two sides; these are monolingual features of the original and translated text and do not look at translation phenomena directly. Their work is also the only work to look at sentence level detection accuracy and report 15 percentage points drop in accuracy when going from chunk level to sentence level classification.

Our work in this chapter is unique in several ways: 1) We explore several bilingual features not explored before, 2) We improve on the best results reported for sentence-level classification accuracy and 3) We develop a cross-domain data set and outperform all prior methods in all cross-domain settings.

4.3 Bilingual Features for Translation Direction Classification

We are interested in learning common localized linguistic phenomena that occur during the translation process when translating in one direction but not the other.

4.3.1 POS Tag MTUs

Minimal translation units (MTUs) for a sentence pair are defined as pairs of source and target word sets that satisfy the following conditions [84, 111].

1. No alignment links between distinct MTUs.

2. MTUs are not decomposable into smaller MTUs without violating the previous rule.

We use POS tags to capture linguistic structures and MTUs to map linguistic structures of the two languages. To obtain POS MTUs from a parallel corpus, first, the parallel corpus is word aligned. Next, the source and target side of the corpus are tagged independently. Finally, words are replaced with their corresponding POS tag in word-aligned sentence pairs. MTUs were extracted from the POS tagged word-aligned sentence pairs from left to right and listed in source order.

[3]Only replacing content words with their POS tags while leaving function words as is.

Unigram, bi-gram, and higher order n-gram features were built over this sequence of POS MTUs. For example, for the sentence pair in Figure 4.4, the following POS MTUs will be extracted: VBZ⇒D, PRP⇒(N,V), RB⇒ADV, JJ⇒N, .⇒PUNC.

4.3.2 Distortion

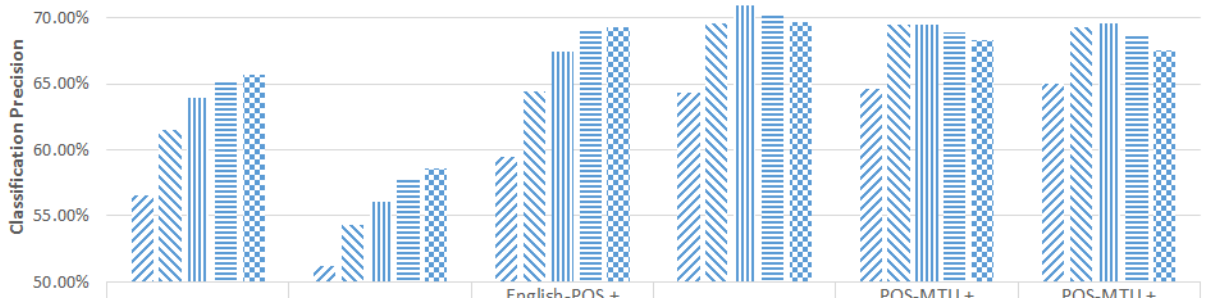
In addition to the mapping of linguistic structures, another interesting phenomenon is the reordering of linguistic structures during translation. One hypothesis is that when translating from a fixed-order to a free-order language, the order of the target will be very influenced by the source (almost monotone translation), but when translating into a fixed order language, more re-ordering is required to ensure grammaticality of the target. To capture this pattern we add distortion to POS Tag MTU features. We experiment with absolute distortion (word position difference between source and target of a link) as well as HMM distortion (word position difference between the target of a link and the target of the previous link). We bin the distortions into three bins: "$= 0$", "> 0" and "< 0", to reduce sparsity.

4.4 Single Domain Experiments

For the translation direction detection task explained in section 4.1, we use a fast linear classifier trained with online learning, Vowpal Wabbit [62]. Training data and classification features are explained in section 4.4.1 and 4.4.2.

4.4.1 Data

For this task we require a parallel corpus with sentence pairs available in both directions (sentences authored in language A and then translated to language B and vice versa). While the customized version of EuroParl [48] contains sentence pairs for many language pairs, none of the language pairs have sentence pairs available in both directions (e.g., it does contain sentences authored in English and translated into French but not vice versa). The Canadian Hansard corpus on the other



	English-POS	French-POS	English-POS + French-POS	POS-MTU	POS-MTU + Absolute Distortion	POS-MTU + HMM Distorition
1-gram	56.59%	51.20%	59.46%	64.37%	64.58%	65.01%
2-gram	61.48%	54.27%	64.43%	69.60%	69.47%	69.29%
3-gram	63.93%	56.04%	67.43%	70.95%	69.45%	69.57%
4-gram	65.28%	57.72%	69.01%	70.29%	68.91%	68.58%
5-gram	65.71%	58.61%	69.32%	69.63%	68.36%	67.55%

Figure 4.3 Sentence level translation direction detection precision using different features with n-gram lengths of 1 through 5.

hand fits the requirement as it has 742,408 sentence pairs translated from French to English and 2,203,504 sentences pairs that were translated from English to French [60]. We use the Hansard data for training classifiers. For training the HMM word alignment model used to define features, we use a larger set of ten billion words of parallel text from the WMT English-French corpus.

4.4.2 Preprocessing and Feature Extraction

We used a language filter[4], deduplication filter[5] and length ratio filter to clean the data. After filtering we were left with 1,890,603 English-French sentence pairs and 640,117 French-English sentence pairs. The Stanford POS tagger [96] was used to tag the English and the French sides of the corpus. The HMM alignment model [100] trained on WMT data was used to word-align the Hansard corpus while replacing words with their corresponding POS tags. Due to differences in word breaking between the POS tagger tool and our word alignment tool there were some mismatches. For simplicity we dropped the entire sentence pair whenever a token mismatch occurred. This left us with 401,569 POS tag aligned sentence pairs in the French to English direction and

[4]A character n-gram language model is used to detect the language of source and target side text and filter them out if they do not match their annotated language.

[5]Duplicate sentences pairs are filtered out.

1,184,702 pairs in the other direction. We chose to create a balanced dataset and reduced the number of English-French sentences to 401,679 with 20,000 sentence pairs held out for testing in each direction.

	POS MTU (E⇒F)	FE#	EF#	Example
1	NNPS⇒(N,C)	336	12	quebecers(NNPS) ⇒ québécoises(N) et(C) des québécois
2	IN⇒(CL,V)	69	1027	a few days ago(IN) ⇒ il y(CL) a(V) quelques
3	PRP⇒(N,V)	18	663	he(PRP) is ⇒ le député(N) à(V)
4	(NNP,POS)⇒A	155	28	quebec(NNP) 's(POS) history ⇒ histoire québécoises(A)
5	(FW,FW)⇒ADV	7	195	pro(FW) bono(FW) work ⇒ bénévolement(ADV) travailler
6	(RB,MD)⇒V	2	112	money alone(RB) could(MD) solve ⇒ argent suffirait(V) à résoudre

Table 4.1 POS MTU features with highest weight. FE# indicates the number of times this feature appeared when translating from French to English.[6]

4.4.3 Results

The results of our experiments on the translation direction detection task are listed in Table 4.5. We would like to point out several results from the table. First, when using only unigram features, the highest accuracy is achieved by the "POS-MTU + HMM Distortion" feature, which uses POS minimal translation units together with distortion. The highest accuracy overall if obtained by a "POS-MTU" trigram model, showing the advantage of bilingual features over prior work using only a union of monolingual features (reproduced by the "English-POS + French-POS" configuration). While higher order features generally show better in-domain accuracy, the advantage of low-order bilingual features might be even higher in cross-domain classification.

4.4.4 Analysis

An interesting aspect of this work is that it is able to extract features that can be linguistically interpreted. Although linguistic analysis of these features is outside the scope of this work, we list POS MTU features with highest positive or negative weights in Table 4.1. Although the top feature,

[6]For description of English POS tags see [73] and [1] for French

Label	Description
ENU.LEX	English word *n*-grams
FRA.LEX	French word *n*-grams
ENU.POS	English POS Tag *n*-grams
FRA.POS	French POS Tag *n*-grmas
ENU.BC	English Brown Cluster *n*-grams
FRA.BC	French Brown Cluster *n*-grams
POS.MTU	POS MTU *n*-grams
BC.MTU	Brown Cluster MTU *n*-grams

Table 4.2 Classification features and their labels.

`NNPS⇒(N,C)`[7], in this context is originating from a common phrase used by French speaking members of the Canadian Parliament, québécoises et des québécois, it does highlight an underlying linguistic phenomenon that is not specific to the Canadian Parliament. When translating a plural noun from English to French it is likely that only the masculine form of the noun appears, while if it was authored in French with both forms of the nouns, a single plural noun would appear in English as English doesn't have masculine and feminine forms of the word. A more complete form of this feature would have been `NNPS⇒(N,C,N)`, but since word alignment models, in general, discourage one-to-many alignments, the extracted MTU only covers the first noun and conjunction.

4.5 Cross-Domain Experiments

The goal this section is to compare novel and previously introduced features in a cross-domain setting. Due to the volume of experiments required for comparison, for an initial study, we select a limited number of feature sets for comparison. Prior works claim POS *n*-gram features capture linguistic phenomena of translation and should generalize across domains [60]. We chose source and target POS *n*-gram features for $n = 1 \ldots 5$ to test this claim. Another feature we have chosen is the best performing feature from single domain experiments: POS MTU[8] *n*-gram features. POS MTUs incorporate source and target side information in addition to word alignment. Prior work has also claimed lexical features such as word *n*-grams do not generalize across domains due to

[7] `NNPS`: Plural Noun, `N`: Noun, `C`:Conjunction

[8] Minimal Translation Units [84]

Corpus	Authored Language	Translation Language	Training Sentences	Test Sentences
EuroParl	English	French	62k	6k
EuroParl	French	English	43k	4k
Hansard	English	French	1,697k	169k
Hansard	French	English	567k	56k
Hansard-Committees	English	French	2,930k	292k
Hansard-Committees	French	English	636k	63k

Table 4.3 Cross-Domain Data Sets

corpus specific vocabulary [101]. We test this hypothesis using source and target word *n*-gram features. Using *n*-grams of length 1 through 5 we run 45 (Nine data matrix entries times *n*-gram lengths of five) experiments for each feature set mentioned above.

In addition to the features mentioned above, we make a small modification to the feature used to obtain the best single domain sentence level performance to derive a new type of features. We introduce Brown cluster [22] MTUs instead. Our use of Brown clusters is inspired by recent success on their use in statistical machine translation systems [15]. Finally, we also include source and target Brown cluster *n*-grams as a comparison point to better understand their effectiveness compared to POS *n*-grams and their contribution to the effectiveness of Brown cluster MTUs.

Given these 8 feature types summarized in Table 4.2, *n*-gram lengths of up to 5 and the 3×3 data matrix explained in the next section, we run 360 experiments for this cross-domain study.

4.5.1 Data Matrix

We chose the English-French language pair for our cross-domain experiments based on prior work and availability of labeled data. Existing sentence-parallel datasets used for training machine translation systems, do not normally contain gold-standard translation direction information, and additional processing is necessary to compile a dataset with such information (labels). Kurokawa et al [60] extract translation direction information from the English-French Hansard parallel dataset using speaker language tags. We use this dataset, and treat the two sections "main parliamentary proceedings" and "committee hearings" as two different corpora. These two corpora have slightly different domains, although they share many common topics as well. We additionally choose a

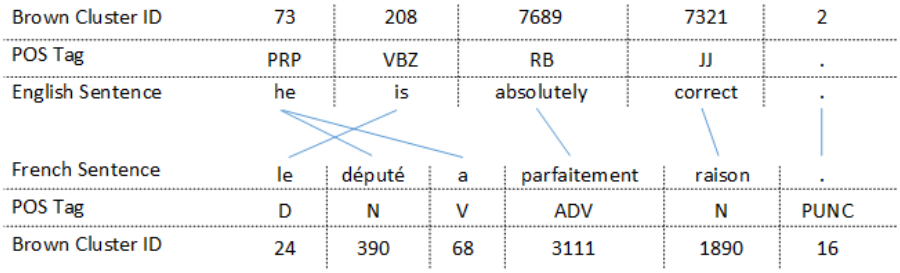


Figure 4.4 POS Tagged and Brown Cluster Aligned Sentence Pairs

third corpus, whose domain is more distinct from these two, from the EuroParl English-French corpus. [48] provided a customized version of Europarl with translation direction labels, but this dataset only contains sentences that were authored in English and translated to French, and does not contain examples for which the original language of authoring was French. We thus prepare a new dataset from EuroParl and will make it publicly available for use. The original unprocessed version of EuroParl [56] contains speaker language tags (original language of authoring) for the French and English sides of the parallel corpus. We filter out inconsistencies in the corpus. First, we filter out sections where the language tag is missing from one or both sides. We also filter out sections with conflicting language tags. Parallel sections with different number of sentences are also discarded to maintain sentence alignment. This leaves us with three data sets (two Hansard and one EuroParl) with translation direction information available, and which contain sentences authored in both languages. We hold out 10% of each data set for testing and use the rest for training. Our 3×3 corpus data matrix consists of all nine combinations of training on one corpus and testing on another (Table 4.3).

4.5.2 Preprocessing and Feature Extraction

Similar to preprocessing steps for single domain experiments, first we clean all data sets using the following simple techniques.

- Sentences with low alphanumeric density are discarded.
- A character n-gram based language detection tool is used to identify the language of each

sentence. We discard sentences with a detected language other than their label.

- We discard sentences with invalid unicode characters or control characters.
- Sentences longer than 2000 characters are excluded.

Next, an HMM word alignment model [100] trained on the WMT English-French corpus [20] word-aligns sentence pairs. We discard sentence pairs where the word alignment fails. We use the Stanford POS tagger [96] for English and French to tag all sentence pairs. A copy of the alignment file with words replaced with their POS tags is also generated. French and English Brown clusters are trained separately on the French and English sides of the WMT English-French corpus [20]. The produced models assign cluster IDs to words in each sentence pair. We create a copy of the alignment file with cluster IDs instead of words as well.

The classifier of our choice (Section 4.5.3) extracts n-gram features with n specified as an option. In preparation for classifier training and testing, feature extraction only needs to produce the unigram features while preserving the order (n-grams of higher length are automatically extracted by the classifier).

POS, word, and Brown cluster n-gram features are generated by using the respective representation for sequences of tokens in the sentences. For POS and Brown cluster MTU features, the sequence of MTUs is defined as the left-to-right in source order sequence (due to reordering, the exact enumeration order of MTUs matters). For example, for the sentence pair in Figure 4.4, the sequence of Brown cluster MTUs is: 73⇒(390,68), 208⇒24, 7689⇒3111, 7321⇒1890, 2⇒16.

4.5.3 Online Classification

We chose the Vowpal Wabbit [62] (VW) online linear classifier since it is fast, scalable and it has special (bag of words and n-gram generation) options for text classification. We found that VW was comparable in accuracy to a batch logistic regression classifier. For training and testing the classifier, we created balanced datasets with the same number of training examples in both directions. This was achieved by randomly removing sentence pairs from the English to French

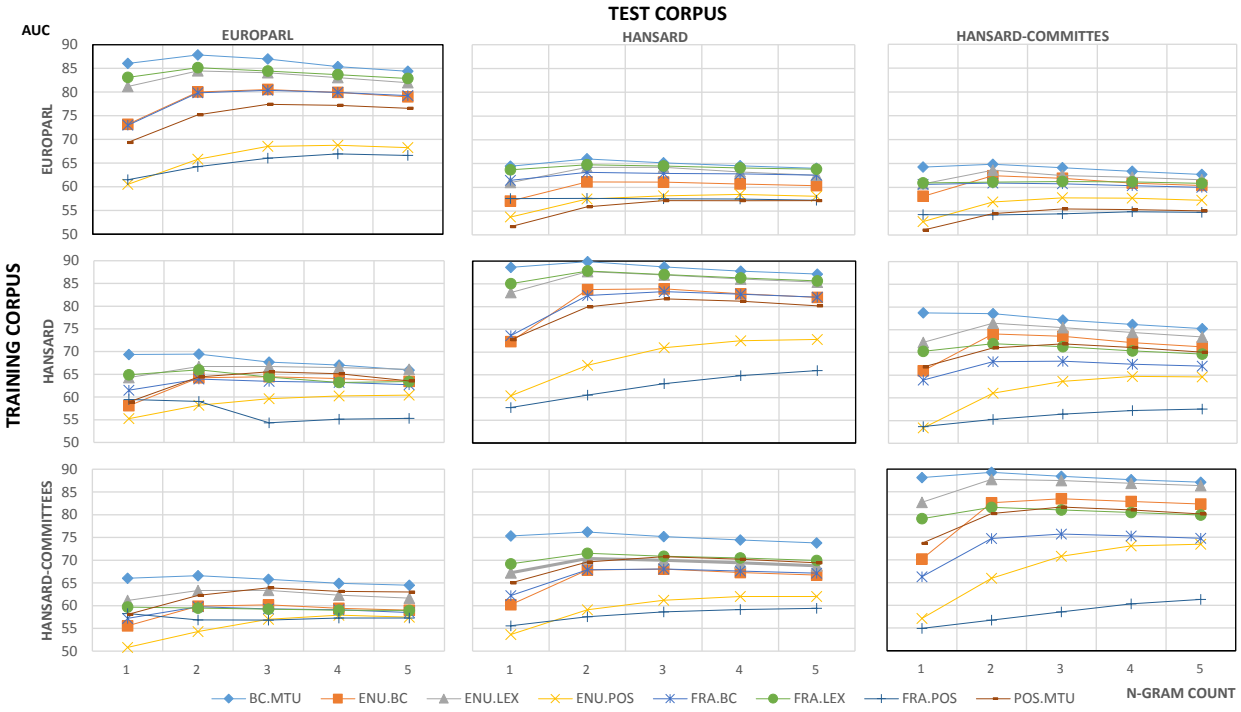


Figure 4.5 Comparing area under the ROC curve for the translation direction detection task when training and testing on different corpora using each of the eight feature sets. See Table 4.2 for experiment label description.

direction until it matches the French to English direction. For example, 636k sentence pairs are randomly chosen from the 2,930k sentence pairs in English to French Hansard-Committees corpus to match the number of examples in the French to English direction.

4.5.4 Evaluation Method

We are interested in comparing the performance of various feature sets in translation direction detection. Performance evaluation of different classification features objectively is challenging in the absence of a downstream task. Specifically, depending on the preferred balance between precision and recall, different features can be superior. Ideally an ROC graph [35] visualizes the tradeoff between precision and recall and can serve as an objective comparison between different classification feature sets. However, it is not practical to present ROC graphs for 360 experiments. Hence, we resort to the Area Under the ROC graph (AUC) measure as a good measure to provide an objective comparison. Theoretically, the area under the curve can be interpreted as the probability

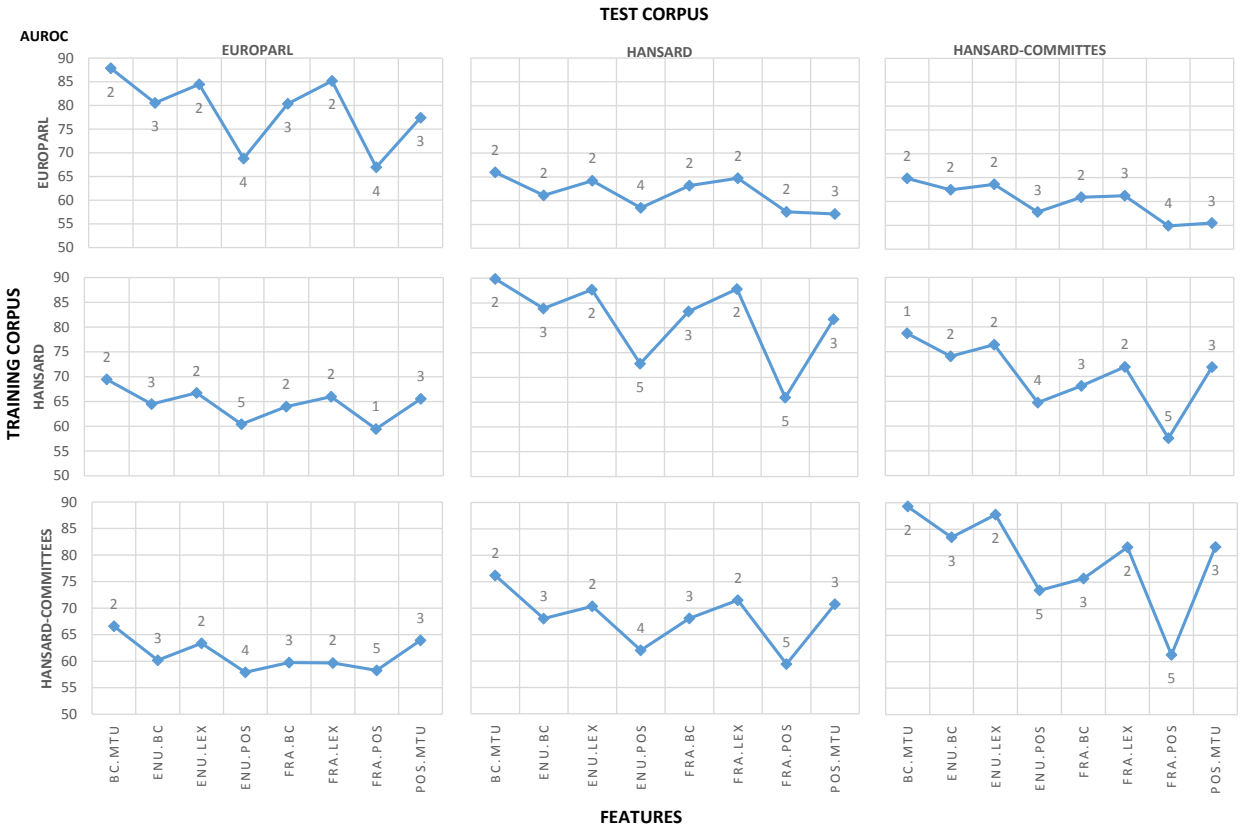


Figure 4.6 Translation detection performance matrix for training and testing on three different corpora. Each row corresponds to results of training on a specific corpus while test results are laid out in a column of the 3×3 matrix. Unlike Figure 4.5 where we report AUC values for all n-grams, in this diagram we only report the highest AUC value for each feature. The number next to each data point indicates the n-gram length that achieves the highest performance for that feature. See Table 4.2 for experiment label description.

that the classifier scores a random negative example higher than a random positive example [35]. As a point of reference, we also provide F-scores for experimental settings that are comparable to the prior work reviewed in Section **??**.

4.5.5 Results

Figure 4.5 presents AUC points for all experiments. Rows and columns are labeled with corpus names for training and test data sets respectively. For example, the graph on the third row and first column corresponds to training on the Hansard-Committees corpus and testing on EuroParl. Within each graph we compare the AUC performance of different features with n-gram lengths of

1 through 5.

Graphs on the diagonal demonstrate higher performance compared to off diagonal graphs. This confirms the basic assumption that cross-domain translation direction detection is a more difficult task as the graphs on the diagonal correspond to in-domain detection. The overall performance is also higher when trained on the Hansard corpus and tested on Hansard-Commitee and vice versa. This is becuase the Hansard corpus is more similar to the Hansard-Committees corpus compared to the EuroParl corpus. It is also observable that the variation in performance of different features diminishes as the training and test corpora become more dissimilar. For instance, this phenomenon can be observed on the second row of graphs where the features are most spread out when tested on the Hansard corpus. They are less spread out when tested on the Hansard-Committees corpus, and compressed together when tested on the EuroParl corpus. The same phenomenon can be observed for classifiers trained on other corpora.

For different feature types, different n-gram order of the features is best, depending on the feature granularity. To make it easier to observe patterns in the performance of different feature types, Figure 4.7 shows the performance for each feature type and each train-test corpus combination as a single point, by using the best n-gram order for that feature/data combination. Each of the 9 train/test data combinations is shown as a curve over feature types.

We can see that MTU features (which look at both languages at the same time) outperform individual source or target features (POS or Brown cluster) for all datasets. Brown clusters are unsupervised and can provide different levels of granularity. On the other hand, POS tags usually provide a fixed granularity and require lexicons or labeled data to train. We see that Brown clusters outperform corresponding POS tags across data settings. As an example, when training and testing on the Hansard corpus FRA.BC outperforms FRA.POS by close to 20 AUC points.

Lexical features outperform monolingual POS and Brown cluster features in most settings although their advantages diminish as the training and test corpus become more dissimilar. This is somewhat contrary to prior claims that lexical features will not generalize well across domains – we see that lexical features do capture important generalizations across domains and models that

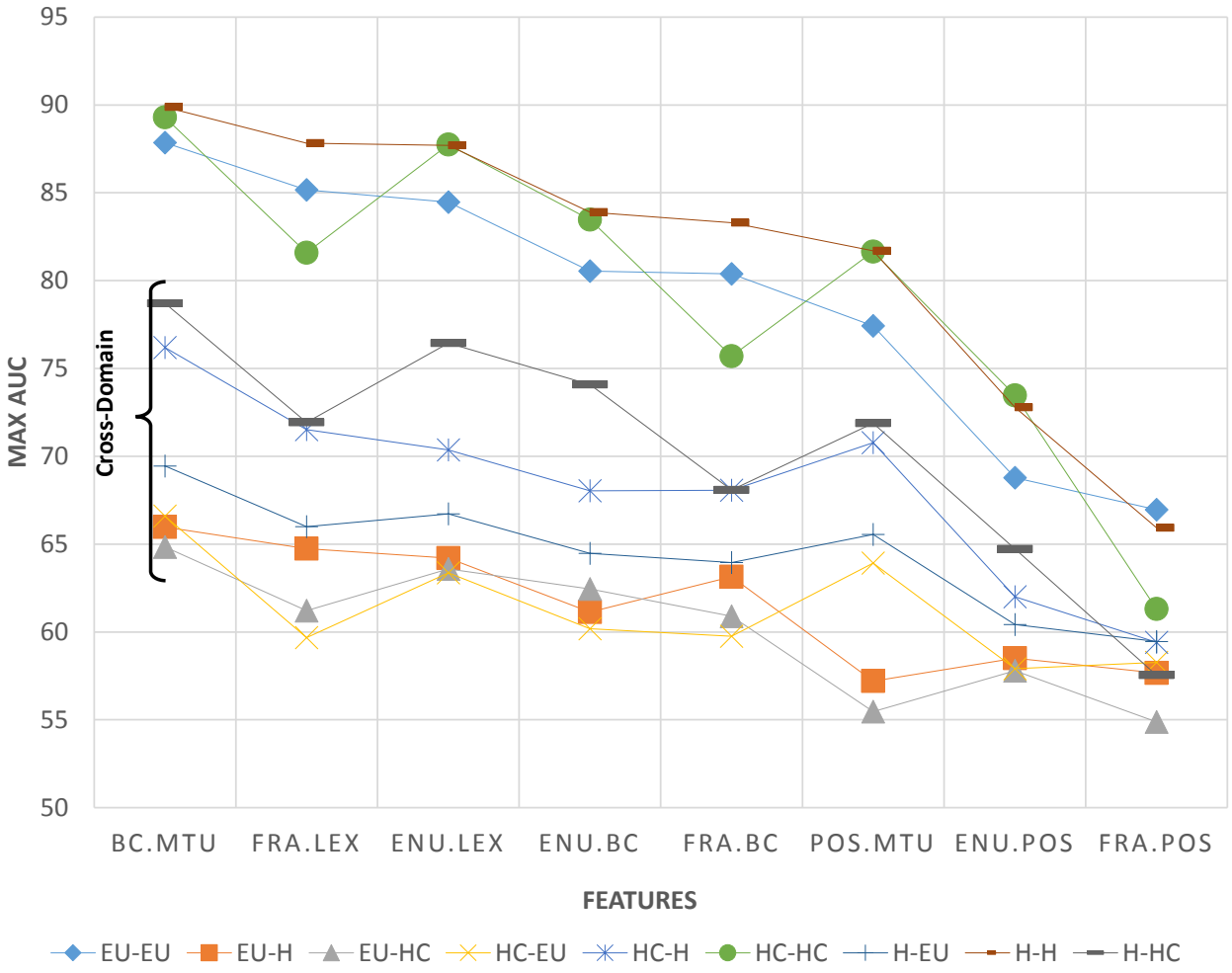


Figure 4.7 Translation detection performance matrix for training and testing on three different corpora - We ran experiments for *n*-grams of up to length five for each feature (See Table 4.2 for feature label descriptions). Unlike Figure 4.5 where we report AUC values for all *n*-gram lengths, in this graph we only present the highest AUC number for each feature. Each marker type indicates a training and test set combination. The format of experiment labels in the legend is [TrainingSet]-[TestSet] and **EU**: EuroParl, **H**: Hansard, **HC**: Hansard Committees. For example, EU-HC means training on EuroParl corpus and testing on Hansard Committees corpus.

use only POS tag features have lower performance, both in and out-of-domain.

Figure 4.8 shows the rank of each feature amongst all 8 different features for each entry in the cross-corpus data matrix (Similar to Figure 4.7 the highest performing *n*-gram length has been chosen for each feature). Brown cluster MTUs outperform all other features with rank one in all dataset combinations. Source and target POS tag features are the lowest performing features in 8 out of 9 data set combinations. The POS.MTU has its lowest ranks (7 and 8) when it is trained on the EuroParl corpus and its highest ranks (2 and 3) when trained on the Hansard-Committees corpus. High number of features in POS.MTU requires a large data set for training. The variation in performance for POS.MTU can be explained by the significant difference in training data size between EuroParl and Hansard-Committees. Finally, while FRA.LEX and ENU.LEX are mostly in rank 2 and 3 (after BC.MTU) they have their lowest ranks (6 and 4) in corss-corpus settings (HC-EU and HC-H).

Finally, we report precision and recall numbers to enable comparison between our experiments and previous work reported in Section 4.2. When training and testing on the Hansard corpus, BC.MTU achieves 0.80 precision with 0.85 recall. In comparison, ENU.POS achieves 0.65 precision with 0.64 recall and POS.MTU achieves 0.73 precision and 0.74 recall. These are the highest performance of each feature with *n*-grams of up to length 5.

4.6 Conclusion and Future Work

From among eight studied sets of features, Brown cluster MTUs were the most effective at identifying translation direction at the sentence level. They were superior in both in-domain and cross-domain settings. Although English-Lexical features did not perform as well as Brown cluster MTUs, they performed better than most other methods. In future work, we plan to investigate lexical MTUs and to consider feature sets containing any subset of the eight or more basic feature types we have considered here. With these experiments we hope to gain further insight into the performance of feature sets in in out out-of-domain settings and to improve the state-of-the-art in realistic translation direction detection tasks. Additionally, we plan to use this classifier to extend

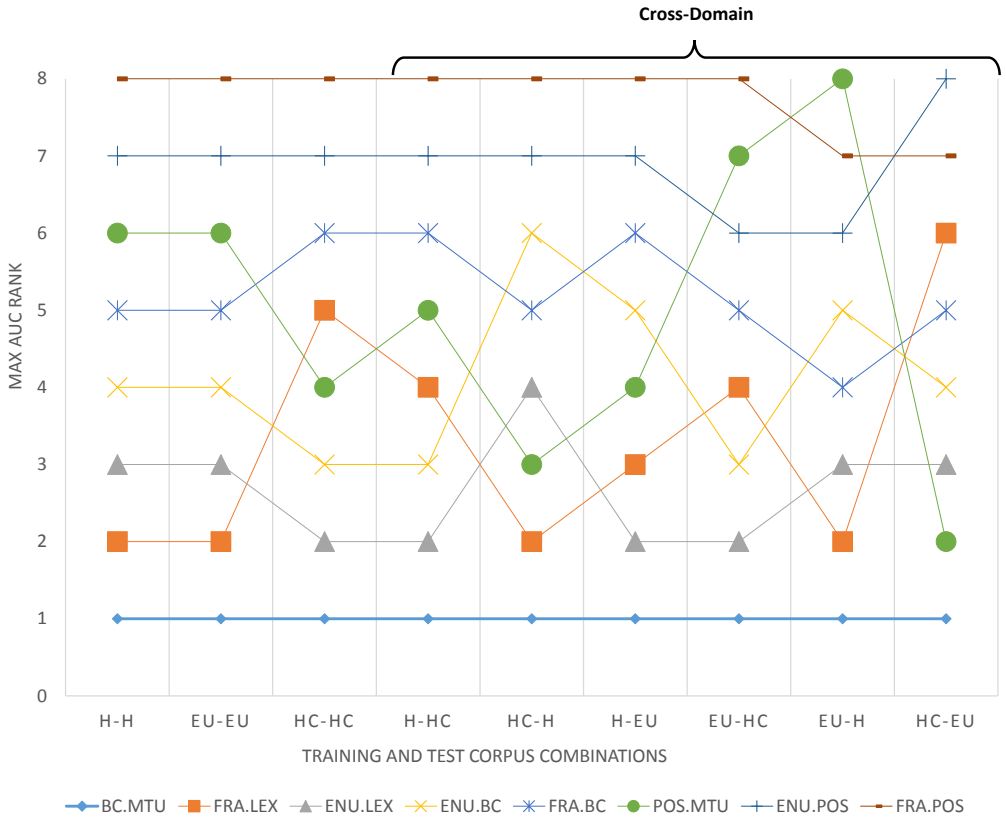


Figure 4.8 Translation direction detection AUC performance rank for each training and test set combination. For corpus combination abbreviations see description of Figure 4.7. For feature label descriptions see Table 4.2.

the work of Twitto-Shmuel ([97]) by building a more accurate and larger parallel corpus labeled for translation direction to further improve SMT quality.

CHAPTER 5

EFFECTIVE AND EFFICIENT DATA SELECTION

Lack of an *effective* and *efficient* data selection algorithm is apparent from the literature review in Chapter 3. Most *effective* data selection methods involve retraining a full set of SMT models at every step of the data selection process (e.g., [43, 7]). These methods are barely practical for small data sets with 100K sentences in training data, let alone medium or large data sets with over few million sentences. The contribution of more *efficient* methods is in two folds: 1) Introducing a surrogate function (e.g., maximum number of unseen token types covered with minimum number of sentences ([33])) to eliminate the need for retraining a full SMT system and 2) providing a mechanism to optimize the surrogate objective function (e.g., Greedy Search, also in [33]). In this chapter we challenge the significance of achieving the optimum value for the simple surrogate function described above by providing a simple and efficient algorithm that achieves similar results using the same surrogate function and a linear search algorithm that does not provide any theoretical guarantees for achieving the optimum solution. We run experiments at several data set sizes and compare it with random selection. In addition, we experiment with different orderings of the data and provide experimental results.

5.1 Vocabulary Saturation Filter

The most common feature function that has been used with some variations (see Chapter 3) for data selection in statistical machine translation is the number of new n-grams in a sentence pair. The objective of the previously introduced search algorithms (e.g., [33]) is to maximize the number of unique n-grams in a subset of the data with a constraint on the total number of tokens. As explained in Chapter 3 this is done using a greedy algorithm with the complexity of $O(|S_{SelectionPool}|^2)$. In contrast we propose a linear algorithm, $O(|S_{SelectionPool}|)$, that achieves comparable results. A simplified version of this algorithm (Vocabulary Saturation Filter or VSF) is making a single

pass through the data and selecting sentence pairs such that for all n-grams there is at least one selected sentence that contains that n-gram. The algorithm would go through the sentence pairs in an arbitrary order and only select a sentence pair, if it contains a new n-gram. To increase the flexibility of the algorithm we introduce a "minimum occurrence" threshold, t, which is equal to one in the algorithm described above. The algorithm will select a sentence if and only if it contains at least one n-gram that has occurred less than t times in the *selected pool*. The implementation of VSF is described in **Algorithm 1** below.

```
Input: SelectionPool, t, ℓ
Output: SelectedPool
foreach sp ∈ SelectionPool do
    NG_src ← EnumNgrams(sp.src, ℓ);
    NG_tgt ← EnumNgrams(sp.tgt, ℓ);
    selected ← false;
    foreach ng ∈ NG_src do
        if SrcCnt [ng]<t then
            selected ← true;
            break;
        end
    end
    if not selected then
        foreach ng ∈ NG_tgt do
            if TgtCnt [ng]<t then
                selected ← true;
                break;
            end
        end
    end
    if selected then
        SelectedPool.Add(sp);
        foreach ng ∈ NG_src do
            SrcCnt [ng]++;
        end
        foreach ng ∈ NG_tgt do
            TgtCnt [ng]++;
        end
    end
end
```

Algorithm 1: Pseudocode for implementing VSF. ℓ: n-gram length, t: n-gram threshold, sp: sentence pair.

5.2 Comparison to near Optimum Solution

Although the feature function "number of new n-grams per sentence" was first introduced by Eck et al., Kirchhoff and Bilmes have shown the function to be sub-modular and hence a greedy search algorithm guarantees achieving constant factor approximation of the optimum solution ([53, 33]). Since the complexity of the greedy algorithm is $O(|S_{SelectionPool}|^2)$, it is impractical for large data sets. Therefore we seek to discover whether the extra complexity of finding near optimum solution impacts the final translation quality of the SMT system compared to using a linear selection algorithm such as VSF.

$$f_j(sentence) = \frac{\sum_{n=1}^{j} \left[\sum_{\substack{unseen \\ n-grams}} Freq(n-gram) \right]}{|sentence|} \tag{5.1}$$

For comparison the feature function in Equation 5.1, introduced by Eck et al., is used in the greedy search algorithm.

5.2.1 Experimental Setup

To compare VSF against the greedy algorithm proposed by Eck et al. we selected the English-Lithuanian parallel corpus from JRC-ACQUIS ([92]). We selected the corpus for the following reasons:

- VSF performance on this particular data set was at its lowest compared to a number of other data sets, so there was room for improvement by a potentially better algorithm.
- With almost 50 million tokens combined (English and Lithuanian) we were able to optimize the greedy algorithm proposed by Eck et al. and run it on this data set in a reasonable amount of time. The experiments run by the original paper in 2005 were run on only 800,000 tokens.

We also used an HMM alignment model [100] for word alignment and a tree-to-string translation system ([85]) for training the translation model along with a 4-gram language model.

5.2.2 Results

Using the greedy algorithm with n-gram length set to one ($j \leftarrow 1$ in Equation 5.1) only 10% (5,020,194 tokens total) of the data is sorted, since all n-grams of size one have been observed by that point and the weight function for the remaining sentences returns zero. In other words, since there are no unseen unigrams after 10% of the data has been sorted, in Equation 5.1, the numerator becomes zero there after and therefore the remaining 90% of sentence pairs are not sorted. This must be taken into consideration when examining the comparison between unigram VSF and the Eck algorithm with n-gram length set to one in Figure 5.1. VSF with its lowest setting, that is threshold $t = 1$, selects 20% of the data, so this chart may not be a fair comparison between the two algorithms.

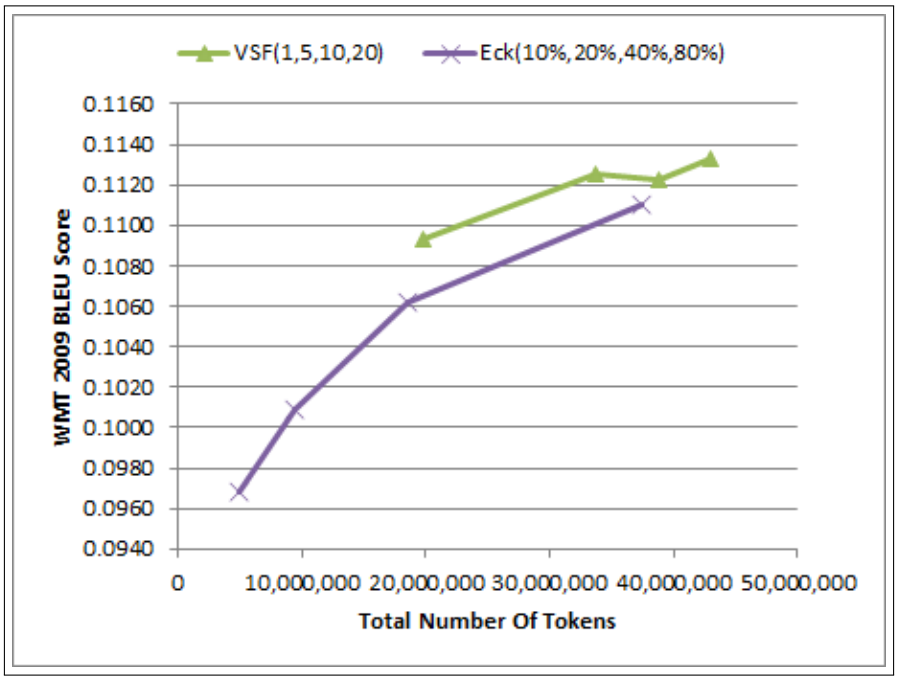


Figure 5.1 Unigram Eck vs. Unigram VSF

In an attempt to do a fairer comparison, we also experimented with n-grams of length two in the Eck algorithm, where 50% of the data can be sorted (since all unigrams and bigrams are observed by that point). As seen in Figure 5.2, the BLEU scores for the Eck and VSF systems built on the

similar sized data score very closely on the WMT 2009 test set.[1]

In addition, we developed the following two extensions to the greedy algorithm, none of which resulted in a significant gain in BLEU score over VSF with *n*-gram lengths set up to three.

- Incorporating target sentence *n*-grams in addition to source side sentence *n*-grams.
- Dividing the weight of an *n*-gram (its frequency) by a constant number each time a sentence that contains the *n*-gram is selected, as opposed to setting the weight of an *n*-gram to zero after it has been seen for the first time.[2]

In relatively small data sets there is not a significant difference between the two algorithms. The greedy algorithm does not scale to larger data sets and higher *n*-grams. Since a principal focus of our work is on scaling to very large data sets, and since the greedy algorithm could not scale to even moderately sized data sets, we decided to continue our focus on VSF and improvements to that algorithm.

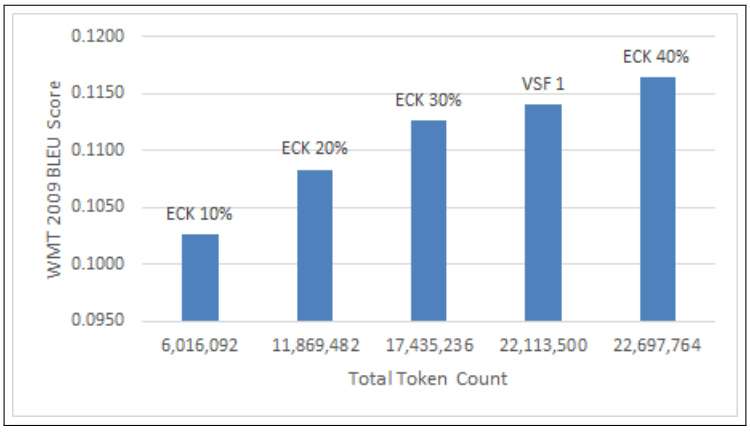


Figure 5.2 Bigram Eck vs. Unigram VSF

[1] The careful reader may note that there is no official WMT2009 ([25]) test set for Lithuanian, since Lithuanian is not (yet) a language used in the WMT competition. The test set mentioned here was created from a 1,000 sentence sample from the English-side of the WMT2009 test sets, which we then manually translated into Lithuanian.

[2] Further details of the modifications to the greedy algorithm are not discussed here as they did not yield improvements over the baseline algorithm.

5.3 Data Order

As apparent from the VSF algorithm description, the order in which the algorithm goes through the data plays a key role in the final selected data set. We explored few different ordering of the data.

1. Longest sentence first: The motivation for sorting sentences in descending order of length is that longer sentences are more likely to have diverse vocabulary and by including them first, the algorithm will chose more vocabulary items with fewer total number of sentences or tokens.

2. Shortest sentence first: Word alignment is more accurate for shorter sentences as there are fewer total number of alignments to chose from and therefore phrases extracted from shorter sentences tend to be less erroneous. The idea is that sorting sentences in ascending order of length will allow to extract more accurate phrase pairs for vocabulary items and result in a higher quality machine translation system.

3. Highest alignment score first: Although word alignment is a resource intensive task, it is possible to use an existing word alignment model to extract Viterbi alignments. The Viterbi alignment score can then be used as a direct and more accurate measure of alignment quality instead of sentence length.

While experiments were run for all three orderings, in addition to a random order, only using alignment score ordering yielded an improvement over random ordering. We report experiment results for random and alignment score ordering in the next section.

5.4 Experiment Setup

Training and tuning tools used for experiments are the same as described in Section 5.2.1 above. We chose English-French language pair as it is the language pair with highest number of parallel sentence pairs available from the WMT corpus. A 5-gram French language model trained over the

$t =$	Random	VSF	Ordered VSF
1	1.83 M	1.83 M	1.68 M
2	2.53 M	2.53 M	2.34 M
5	3.62 M	3.62 M	3.35 M
10	4.62 M	4.62 M	4.29 M
20	5.83 M	5.83 M	5.44 M
40	7.26 M	7.26 M	6.83 M
60	8.21 M	8.21 M	7.78 M
100	9.53 M	9.53 M	9.13 M
150	10.67 M	10.67 M	10.33 M
200	11.53 M	11.53 M	11.23 M
250	12.22 M	12.22 M	11.97 M
All	22.5 M		

Table 5.1 English-side Sentence Counts (in millions) for different thresholds for VSF, VSF after ordering the data based on normalized combined alignment score and random baselines.

French side of the EnFrGW corpus was kept constant and used for all experiments (only parallel training data is changed throughout these experiments). Minimum Error Rate Training (MERT) [79] was used for tuning the lambda values for all systems, using the official WMT2010 dev data. We used WMT 2009 and 2010 test sets, used in the WMT competitions in the respective years as test sets.

To discern the effects of VSF at different degrees of "saturation", we tried VSF with different threshold values t, ranging from 1 to 250. For each t value we actually ran VSF twice. In the first case, we did no explicit sorting of the data. In the second case, we ranked the data using the method described in Section 5.3.

Finally, we created random baselines for each t, where each random baseline is paired with the relevant VSF run, controlled for the number of sentences (since t has no relevance for random samples). The different t values and the resulting training data sizes (sentence and word counts) are shown in Tables 5.1 and 5.2.

Since our interest in this study is scaling parallel data, for all trainings we used the same language model, which was built over all training data (the French side of the full EnFrGW). Because monolingual training scales much more readily than parallel, this seemed reasonable. Further, using one language model controls one parameter that would otherwise fluctuate across

$t =$	Random	VSF	Ordered VSF
1	46.1 M	64.52 M	65.74 M
2	63.99 M	87.41 M	88.12 M
5	91.55 M	121.3 M	120.86 M
10	116.83 M	151.53 M	149.95 M
20	147.31 M	186.99 M	184.14 M
40	183.46 M	228.14 M	224.29 M
60	207.42 M	254.89 M	250.68 M
100	240.88 M	291.45 M	287.02 M
150	269.77 M	322.5 M	318.33 M
200	291.4 M	345.37 M	341.69 M
250	308.83 M	363.44 M	360.32 M
All	583.97 M		

Table 5.2 English-side Word Counts for different thresholds for VSF, VSF after ordering the data based on normalized combined alignment score and random baselines.

experiments. The result is a much more focused view on parallel training diffs.

5.5 Experiment Results

We trained models over each set of data. In addition to calculating BLEU scores for each resulting set of models in (Table 5.4), we also compared OOV[3] rates (Table 5.5) and performance differences (Table 5.3). The former is another window into the "quality" of the resulting models, in that it describes vocabulary coverage (in other words, how much vocabulary is recovered from the full data). The latter gives some indication regarding the time savings after running VSF at different thresholds.

On the WMT2009 data set, both sets of VSF models outperformed the relevant random baselines. On the WMT2010 test set, the pre-sorted VSF outperformed all random baselines, with the unsorted VSF outperforming most random baselines, except at t=60 and t=200 for WMT2010. Clearly, the $t = 200$ results show that there is less value in VSF as we approach the total data size.

The most instructive baseline, however, is the one built over all training data. It is quite obvious that at low threshold values, the sampled data is not a close approximation of the full data: not

[3]OOV: Out Of Vocablary rate indicates the number of tokens from the source side of a test set that did not match any phrases from the phrase table trained on parallel training data.

$t =$	Random	VSF	Ordered VSF
1	1:07	2:17	1:56
2	1:33	2:55	2:39
5	2:15	4:05	3:47
10	2:43	4:49	4:50
20	3:23	5:25	5:14
40	4:12	6:16	5:56
60	4:45	6:41	7:15
100	5:31	7:32	7:55
150	6:07	8:20	8:18
200	6:36	8:31	8:52
250	7:30	9:19	9:11
All	13:12		

Table 5.3 Word alignment times (hh:mm) for different thresholds for VSF, VSF after model score ordering, and a random baseline

		WMT2009			WMT2010	
$t =$	Random	VSF	S+VSF	Random	VSF	S+VSF
1	23.76	23.83	23.84	25.69	25.78	25.68
2	23.91	24.04	24.07	25.76	26.21	26.14
5	24.05	24.29	24.40	26.10	26.40	26.32
10	24.15	24.37	24.45	26.21	26.63	26.32
20	24.20	24.40	24.55	26.30	26.46	26.56
40	24.37	24.43	24.65	26.40	26.55	26.53
60	24.32	24.43	24.64	26.56	26.56	26.61
100	24.37	24.49	24.71	26.46	26.75	26.70
150	24.37	24.61	24.71	26.67	26.67	26.70
200	24.48	24.63	24.69	26.56	26.65	26.78
250	24.41	24.57	24.85	26.62	26.74	26.68
All	24.37			26.54		

Table 5.4 BLEU Score results for VSF, S+VSF (Sorted VSF), and Random Baseline at different thresholds t.

enough vocabulary and contextual information is preserved for the data to be taken as a proxy for the full data. However, with t values around 20-60 we recover enough BLEU and OOVs to make the datasets reasonable proxies. Further, because we see a relative reduction in data sizes of 32-44%, model size reductions of 27-39%, and performance improvements of 41-50% at these t values further argues for the value of VSF at these settings. Even at $t = 250$, we have training data that is 54% of the full data size, yet fully recovers BLEU.

		WMT2009			WMT2010	
$t =$	Random	VSF	S+VSF	Random	VSF	S+VSF
1	630	424	450	609	420	445
2	588	374	395	559	385	393
5	520	343	347	492	350	356
10	494	336	335	458	344	344
20	453	335	335	432	339	341
40	423	330	331	403	336	337
60	419	329	330	407	333	336
100	389	330	329	391	333	335
150	397	330	330	384	332	332
200	381	328	330	371	331	332
250	356	329	328	370	333	331
All	325			331		

Table 5.5 OOV rates for VSF, S+VSF (Sorted VSF), and Random Baseline at different thresholds t.

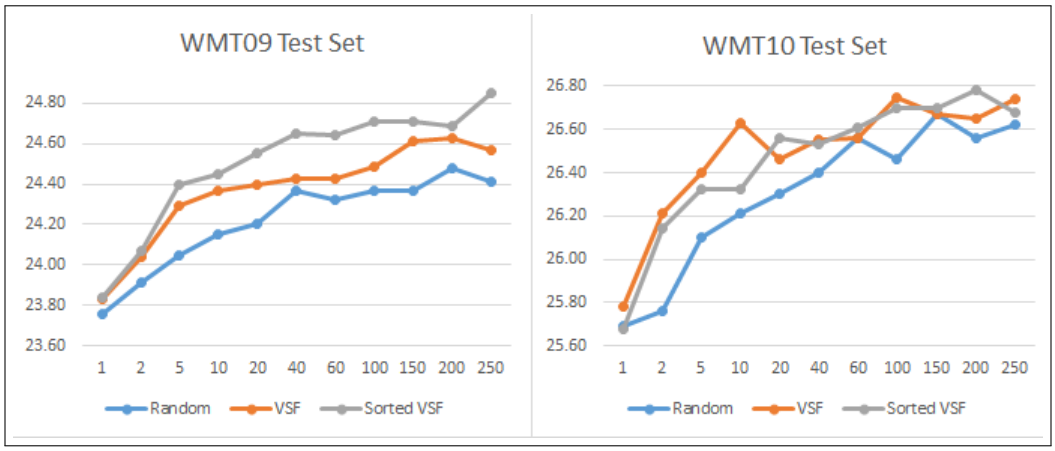


Figure 5.3 Comparative BLEU scores for two VSF implementations, against a randomly sampled baseline.

5.6 Effects of VSF on n-gram distribution

In an attempt to better understand the behavior of VSF and how VSF changes the n-gram distributions of vocabulary items in a sample as compared to the full corpus, we created log_2-scale scatter plots, as seen in Figure 5.5. In these plots, unigram frequencies of unfiltered data (*i.e.*, the full corpus, EnFrGW) are on the vertical axis, and unigram frequencies of the VSF filtered data are on the horizontal axis. The three plots show three different settings for t. There following observations can be made about these plots:

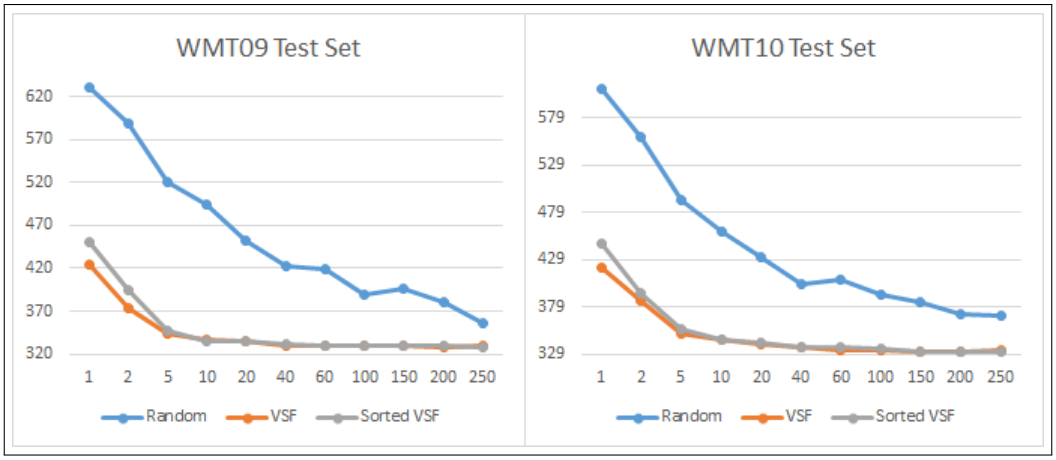


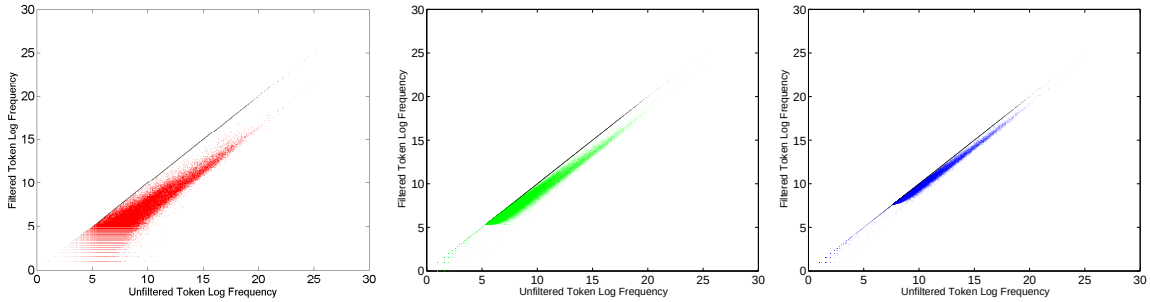
Figure 5.4 Comparative OOV rates for two VSF implementations, against a randomly sampled baseline.

1. On the horizontal axis before we reach $log_2(t)$, all data points fall on the $x = y$ line.

2. As the threshold increases the scatter plot gets closer to the $x = y$ line.

3. VSF has the highest impact on the "medium" frequency unigrams, that is, those with a frequency higher than the threshold.

The third point speaks the most to the effects that VSF has on data: Very low frequency items, specifically those with frequencies below the threshold t, are unaffected by the algorithm, since we guarantee including all contexts in which they occur. Low frequency items are at the lower left of the plots, and their frequencies follow the $x = y$ line (point 1 above). Medium frequency items, however, specifically those with frequencies immediately above t, are the most affected by the algorithm. The "bulge" in the plots shows where these medium frequency items begin, and one can see plainly that their distributions are perturbed. The "bulge" dissipates as frequencies increase, until the effects diminish as we approach much higher frequencies. The latter is a consequence of a simplifying heuristic applied in VSF (as described in Section 5.1): t is not a hard ceiling, but rather a soft one. Vocabulary items that occur very frequently in a corpus will be counted many more times than t; for very high frequency items, their sampled distributions may approach those observed in the full corpus, and converge on the $x = y$ line. We suspect that the BLEU loss that results from the application of VSF is the result of the perturbed distributions for medium



(a) VSF $t = 1$ (b) VSF $t = 40$ (c) VSF $t = 200$

Figure 5.5 log_2-scale Unigram Frequency scatter plot before VSF versus after VSF

frequency items. Adjusting to higher t values decreases the degree of the perturbation, as noted in the second point, which likewise recovers some of the BLEU loss observed in lower settings. We investigate this phenomena further in Chapter 6.

5.7 Conclusion

In this chapter we introduced a new data selection technique (Vocabulary Saturation Filter). While this method is highly scalable and linear in data size, we showed that it is comparable to the state of the art in data selection. We conducted several experiments with various data sizes and showed that it performs better than random data selection. In addition, we experiment with various orderings of the data and showed that ordering the data based on alignment score results in higher translation quality output. We compared and analyzed the effects of data selection on vocabulary distribution and suggested using a variable threshold as an improvement. We implement variable thresholds and analyze vocabulary distribution in Chapter 6.

CHAPTER 6

THE SATURATION FILTER FRAMEWORK

We demonstrated the advantages of Vocabulary Saturation Filter in Chapter 5. In the saturation filter framework the goal is to have enough samples for each feature such that it can be learned efficiently. For example, in the case of a lexical feature, the goal is to observe enough instances of the lexical item such that it can be translated correctly in a sentence. Two questions arise from this goal:

1. What are the key features of a sentence pair that correspond to its contribution to higher quality machine translation output?

2. How many instances of each feature in a sentence pair should be observed in the training data for it to be learned effectively by the statistical translation system?

VSF (as described in Chapter 5) only uses source lexical vocabulary items as features of a sentence pair and a constant number (threshold) for each feature to be learned effectively. The idea being that if all source vocabulary items have been seen at least N times, the statistical translation system is able to translate them with minimum error. Although the choice of feature and threshold was quite simple, we showed it is very effective and comparable to the state of the art while being highly scalable.

In this chapter we expand the feature space to cover word alignment and semantic information using Minimal Translation Units, Brown Clusters and several combinations of them. We also explore higher n-gram orders in this feature space to incorporate context information. In addition, we explore different variable threshold functions such as individual entropy and log frequency and uniform threshold. The goals of using variable threshold is three folds:

1. As shown in Chapter 5, data selection changes the empirical probability distribution of words in the selected data compared to the selection pool, causing a sampling bias. Using variable

threshold functions we are able to minimize this sampling bias (Section 6.5.2).

2. When the number of unique features is high compared to the number of sentence pairs in the selection pool, using the smallest constant threshold of "one" can result in selecting most or all of the data. Using a variable threshold will enable using richer features while selecting an arbitrary amount of data (Section 6.3).

3. A variable threshold allows for giving a higher weight to the most important features. This will result in more instances for these features and ultimately allow for a higher accuracy estimation for them (Section 6.2).

4. Using a variable threshold also enables incorporating arbitrary side information in the selection algorithm such as alignment score and cross entropy difference ([74]) in the case of domain adaptation.

Finally, we develop a scalable, incremental and adaptive algorithm that is able to select a subset of the data according to the feature set and threshold function with an arbitrary desired size. This is done by producing incremental bins with predefined granularity. The subset is then selected by selecting the number of desired bins in the order produced by the algorithm.

6.1 Feature Space

The saturation filter framework is able to use any feature that can be calculated for a sentence pair, independent of other sentence pairs. In the context of the SFF[1] each unique feature signifies new information (or new instance of a random variable) that can be used to estimate the statistical translation models more accurately. We explored several sentence-pair features used in Chapter 4 for parallel data analysis. These features have three main components:

1. **Source and Target Lexical Items**: Lexical items are the basic features used in VSF.

[1] Saturation Filter Framework

2. **Brown Clusters**: Brown clusters are statistically learned features that can be used as independent features or combined with other features to provide semantic awareness to them.

3. **Minimal Translation Units**: Minimal Translation Units enhance the feature space by incorporating word alignment information.

Please see Chapter 4 for detailed description of the individual features above. These features can be combined in many different ways. Since evaluation of each feature combination takes several weeks (including variations in threshold functions) to train statistical machine translation models for evaluation, it is not practical to explore all variations. Therefore, we have selected a few variations that are theoretically motivated.

6.1.1 Source Vocabulary

This feature is selected as a baseline from Chapter 5. Features are individual words in the source side of the parallel data.

6.1.2 Source and Target Vocabulary

This is a simple variation of VSF, as it incorporates source and target vocabulary as separate individual features from the source and target side of the parallel data. This is also included as a baseline as Chapter 5 only includes experiments for the basic version of VSF (Unigram Source Vocabulary).

6.1.3 Bigram Source Vocabulary

Bigram source vocabulary is another simple variations of VSF and used here as a comparison point to the baseline. In this feature all pairs of neighboring lexical items in the source side of the parallel data are used as features. This feature has the natural effect of giving more frequent lexical items a higher threshold since they naturally participate in more unique bi–grams.

6.1.4 Source Vocabulary with Context

For this feature we take the Brown cluster of the previous and next word for each lexical item to create a new feature. This feature enables us to distinguish between the same lexical item appearing in different semantic contexts. This allows the data selection algorithm to ensure all lexical items have been seen in different semantic contexts and therefore ensure the final statistical translation model can provide translations for words in different contexts accordingly and hence provide higher quality machine translation output. Similar to bi–gram source vocabulary, as more frequent words often appear in more diverse contexts, this feature results in a naturally higher threshold for more frequent lexical items (compared to source vocabulary with uniform threshold).

6.1.5 Minimal Translation Unit

This is the first feature that incorporates word alignment information into the data selection process. Although incorporating source and target vocabulary enables the saturation filter framework to cover source and target vocabulary, using MTUs[2] ensures all possible translations of each source word are considered in the data selection process. Please see Chapter 4 for more details.

6.1.6 Minimal Translation Unit with Target Brown Cluster

Using MTUs with target context is a variation between Source Vocabulary with Context and MTU. While source vocabulary with context, considers the word classes surrounding the source word, MTU with target Brown cluster considers source word and the word class of its translation. Using this feature it is possible to distinguish between different senses of a word when the target language does not share the same senses for its translation. For example, it can distinguish between the case where the English word "bank" is aligned to the French word "banque" and the French word "bank". The potential advantage of this feature over MTUs is that it can make this distinction with a smaller feature space. The number of possible features for MTU with target Brown clus-

[2]Minimal Translation Units

ter is |SourceVocabulary| × |WordClasses| while it is |SourceVocabulary| × |TargetVocabulary| for MTU[3]. Since the number of word classes is many orders of magnitudes smaller than the vocabulary size, this can increase efficiency as well as reduce noise due to bad alignments.

6.2 Threshold Function

In the saturation filter framework, the feature threshold can be any arbitrary function as long as it doesn't change throughout the selection process. This restriction allows for scalable implementation of the algorithm as it can be run on different partitions of the data independently and aggregate the results afterward. Ng and Jordan ([77]) have shown that the number of samples required to reach asymptotic error for generative classifiers (such as Naive Bayes) is a logarithmic function in the number of features. Statistical machine translation can be formulated as a set of generative classifiers each predicting one target word given source words as features. Using this formulation and results from Ng and Jordan, machine translation reaches asymptotic error as a logarithm function of the number source features observed for each target word. For this reason, we define the general form of the threshold function as shown in Equation 6.1 to include a logarithm function. This formulation allows for the logarithm of number of source features per target words to be used in the threshold functions.

$$t(f) = h(f) \times \log g(f) \tag{6.1}$$

$h(f)$: Tuning function that can be tuned to achieve desired data size.

$g(f)$: An arbitrary function that can use statistics of feature f.

f : feature

Although many variations of the functions $h(f)$ and $g(f)$ can be used, motivated by prior work ([33, 17, 66]) we introduce the following variable threshold functions.

[3]In Table 6.2, MTU with uniform threshold of one selects 12.88 million sentence pairs while MTU with target Brown clusters selects 11.76 million sentences with the same threshold.

1. **Uniform**: This function is used as a baseline comparison to VSF where the feature threshold is a constant function for all features. In this variation, $h(f) \times \log g(f)$ is set to a constant integer number for all features ($g(f) = 2, h(f) = c$).

2. **Individual Entropy**: Data selection can also be referred to as data compression. The idea is to have a richer vocabulary (features) in a smaller set of data while preserving the original probability distribution. Setting $g(f) = P(f) = \frac{Count(f)}{\sum_{f \in F}(Count(f))}$ and $h(f) = -k \times P(f)$ will turn the threshold function into individual entropy which is a natural and intuitive function to use as we approach the problem as a data compression problem. The "Individual Entropy" of an instance of a random variable, $-P(X = x) \times \log P(X = x)$, is the theoretical limit of the average number of bits used to represent random variable $X = x$ in a compressed format. The argument for using this function is to treat data selection as a data compression problem and set the threshold limit for feature $F = f$ to be a constant factor of the individual entropy of $P(F = f)$.

3. **Logarithmic**: Using result from the work of Ng and Jordan, we like to set the threshold for the number of target words to be included in selected data to be the logarithm of the number of source features for target words. Since higher number of source words is correlated with higher number of source words per target word, we use the frequency of source words (or features in general) as a threshold function. The intuition behind this strategy is that it is more important to learn frequent features as they are more likely to also appear in test data. In this case, $g(f) = Count(f)$ and $h(f) = k$.

6.3 Incremental Algorithm

A practical shortcoming of the VSF selection algorithm was tuning the threshold value. We solve this problem using an incremental algorithm which splits the data into several ordered partitions. Depending on the desired data size, a number of bins are selected and the rest are discarded. We achieve this by defining a "Threshold Growth Function", $G(t)$. We iterate through the data multiple

times. In each iteration, we select new parallel sentences using the threshold function to create a new partition and remove them from the selection pool. In the next iteration, we increase the threshold using the growth function and repeat while all the data is partitioned. The tradeoff in defining the growth function is between flexibility and efficiency. A slow growth function will create smaller partitions allowing to select the desired amount of data with higher accuracy, but it will take more iterations to partition all the data. On the other hand, a fast growth function will take less iterations to run but will split the data into fewer partitions which offers less flexibility in achieving the desired data size. In contrast, with VSF one would have to set the threshold value t to a desired value, run the algorithm and then based on the amount of data selected, increase or decrease the threshold value and run the algorithm again, until desired amount of data is selected. Computational complexity of this incremental algorithm is $O(N \times C)$ where N is the number of sentences in selection pool and C is the number of times partitions created by the algorithm (Value of C is dependent on $G(t)$). Although this algorithm does not require the data to be all loaded in memory, in our implementation, to increase efficiency, we load all the data in memory and gradually remove it from memory as we select each partition and write it to the disk. Therefore, for our implementation the memory requirement is $O(N)$.

6.4 Experiment Setup

For statistical machine translation training, similar to Chapter 5, we use English-French language pair as it is the language pair with highest number of parallel sentence pairs available from the WMT corpus ([21]) with 22.5 million sentence pairs. We also used an HMM alignment model [100] for word alignment and a phrasal decoder ([75]) using a distortion limit of 5 and a hierarchical lexical reordering model ([39]). We use a 4-gram language model trained on the target side of the training data using Kneser–Ney smoothing ([54]). The language model is kept constant for all experiments. We only train the translation model followed by Minimum Error Rate Training (MERT) [79] for tuning the lambda values for all systems, using the official WMT2010 development data set ([21]).

Label	Description
W_SRC	Source Vocabulary
W_BI	Source and Target Vocabulary
WBi_SRC	Source Bi-gram Vocabulary
CW_SRC	Source Vocabulary with Context
MTU_BI	Minimal Translation Unit
CMTU_TGT	Minimal Translation Unit with Target Brown Cluster
Random	Random Selection

Table 6.1 Data selection features and their labels.

For data selection, in all experiments we use the incremental algorithm with an exponential growth function, $G(t_i) = 2 \times G(t_{i-1})$, where i is the iteration number. We attempt to have data sets selected with sizes at exponential intervals of 1M, 2M, 4M, 8M, 16M and all the data (22.5M) for the baseline. For constant threshold experiments, in cases with high number of unique features, the minimum threshold of "one" selects a large percentage of the data (For example, the MTU feature selects 13M sentence pairs with a constant threshold of one). In these cases, while not always possible, we have tried to keep the data set sizes as close to the exponential intervals listed above as possible. For the threshold function we experiment with uniform threshold, logarithm of frequency and individual entropy as detailed in Section 6.2. The data selection features used in our experiments are listed in Table 6.1.

6.5 Experiment Results

The goal of these experiments are to evaluate the effectiveness of each feature and variable threshold function. First we compare all features using a constant threshold function using the exponential growth function described above. This results in 22 unique subsets of the entire 22.5M sentence pair data set (See Table 6.2).

Feature	Data Sizes
Random	1.00M, 2.00M, 4.00M, 8.00M, 16.00M, 22.50M (all)
W_SRC	0.98M, 1.97M, 4.15M, 8.08M, 15.85M
W_BI[4]	2.19M, 3.99M, 7.90M, 15.95M
WBi_SRC[5]	9.38M, 16.33M
CW_SRC[6]	16.54M
MTU_BI[7]	12.88M, 16.30M
CMTU_TGT[8]	11.76M, 16.54M

Table 6.2 Incremental Feature Saturation Filter algorithm sizes for each feature.

We also compare the individual entropy, logarithm of frequency and constant threshold functions (Table 6.3).

Feature	Threshold Function	Data Sizes
W_SRC	Constant	0.97M, 1.97M, 4.15M, 8.08M, 15.84M
W_SRC	Individual Entropy	1.17M, 2.30M, 4.66M, 9.35M, 17.60M
W_SRC	Logarithm of Frequency	0.91M, 2.43M, 4.56M, 8.02M, 15.68M
WBi_SRC	Constant	9.38M, 16.33M
WBi_SRC	Individual Entropy	1.39M, 2.30M, 4.37M, 8.46M, 15.46M
WBi_SRC	Logarithm of Frequency	3.23M, 4.41M, 8.23M, 16.65M
CMTU_TGT	Logarithm of Frequency	0.68M, 1.90M, 4.41M, 6.29M, 15.97M
MTU_BI	Logarithm of Frequency	0.54M, 1.61M, 6.01M, 11.03M, 15.32M

Table 6.3 Incremental Saturation Filter algorithm sizes for different threshold functions.

Different selection techniques are evaluated using the metrics below.

1. **BLEU score**: We report the BLEU score of each translation system on WMT2009 and WMT2013 test sets ([21]).

2. **Unique Source Phrases**: The number of unique source phrases extracted from parallel data.

3. **Unique Phrase Pairs**: The total number of phrase pairs extracted from parallel data.

[4]Constant threshold of "one" selects 2.19M sentence pairs. Thus a 1M data set is not available for this feature.

[5]Constant threshold of "one" selects 9.38M sentence pairs.

[6]Constant threshold of "one" selects 16.64M sentence pairs.

[7]Constant threshold of "one" selects 12.88M sentence pairs.

[8]Constant threshold of "one" selects 11.76M sentence pairs.

4. **Average Target Phrase Per Source**: The ratio of the two metrics above (unique phrase pairs divided by unique source phrases).

5. **Empirical Distribution Distance**: The Jensen–Shannon divergence between empirical probability distribution functions of selected data and selection pool (Equation 6.2).

$$
\begin{aligned}
JSD(P||Q) &= \frac{1}{2}D_{KL}(P||M) + \frac{1}{2}D_{KL}(Q||M) \\
M &= \frac{1}{2}(P+Q) \\
D_{KL}(P||Q) &= \sum_i P(i)\log\frac{P(i)}{Q(i)}
\end{aligned}
\tag{6.2}
$$

6.5.1 Feature Comparison

First, we compare the effectiveness of different features in the saturation filter framework using constant thresholds. The BLEU score comparison between features are presented in Figure 6.1 and Figure 6.2 using WMT2009 and WMT2013 test sets.

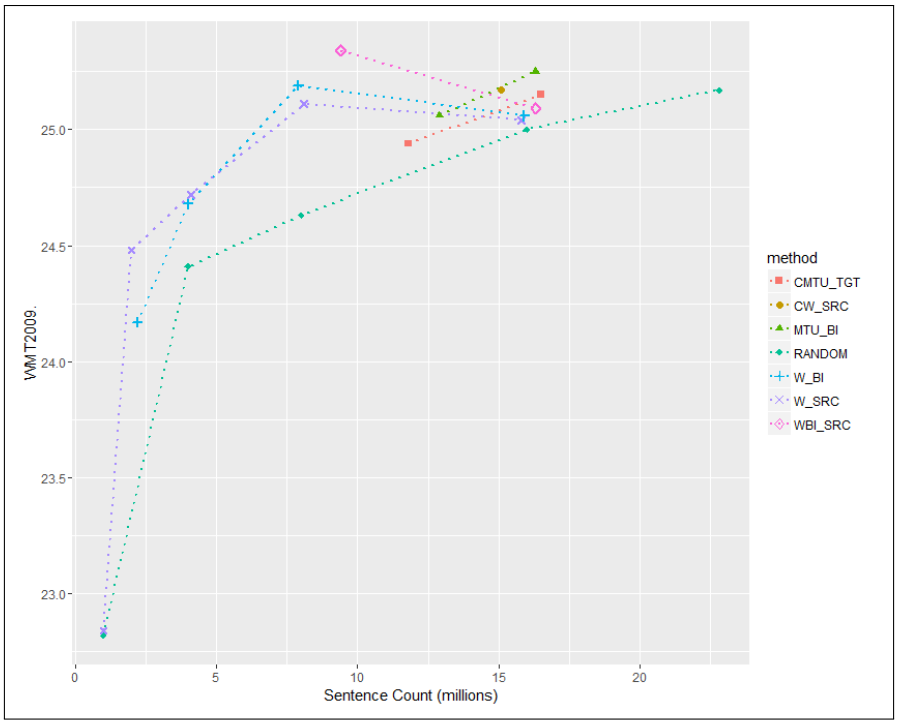


Figure 6.1 BLEU score comparison of different data selection features with constant threshold with the WMT2009 test set.

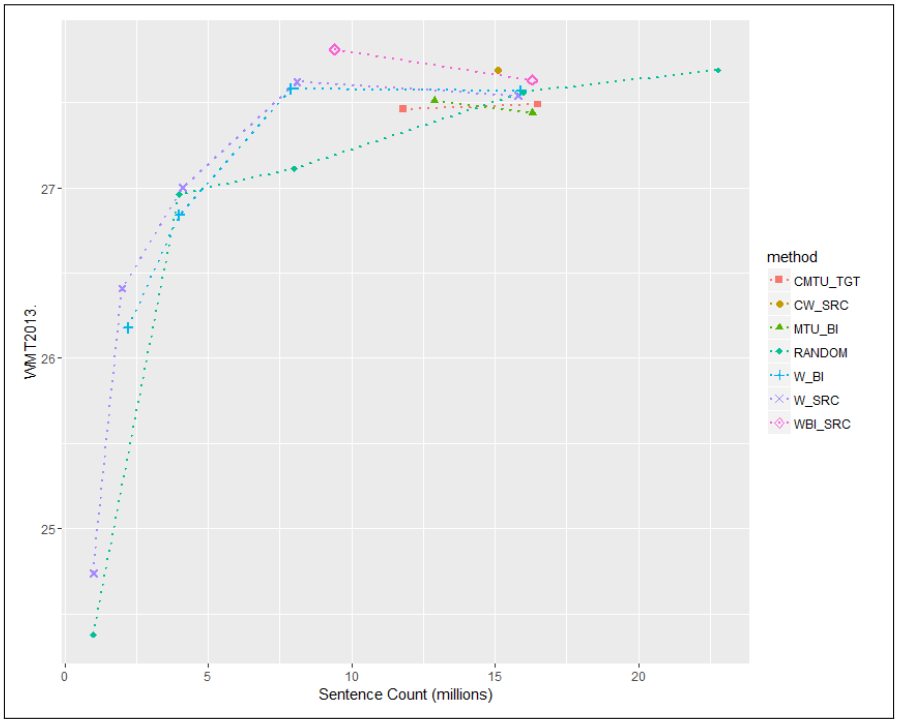


Figure 6.2 BLEU score comparison of different data selection features with constant threshold with the WMT2013 test set.

We further compare different data selection features using total number of phrase pairs (Figure 6.4), number of unique source phrases (Figure 6.3) and the average number of target phrases per source phrase (Figure 6.5).

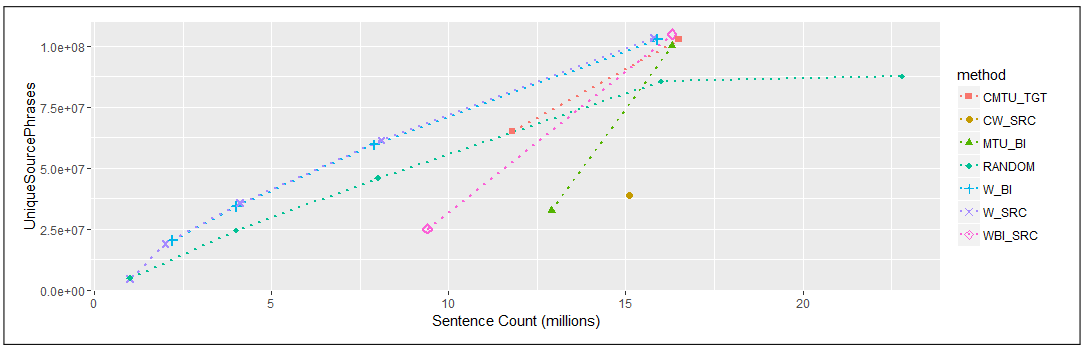


Figure 6.3 Comparison of total number of unique source phrases extracted for each data selection feature.

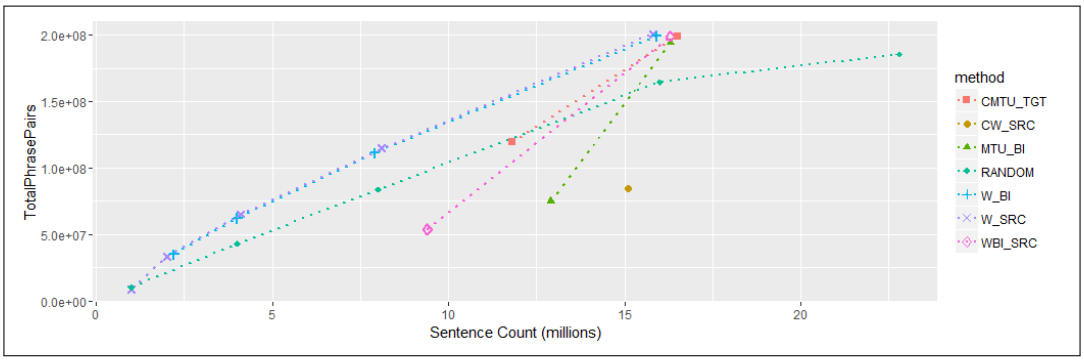


Figure 6.4 Comparison of total number of unique phrase pairs extracted for each data selection feature.

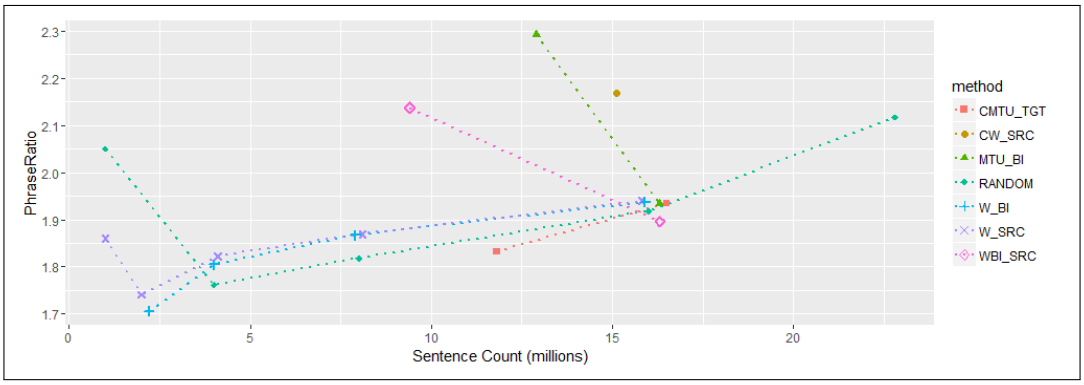


Figure 6.5 Average number of target phrases per source phrase for each data selection feature.

It is notable that all features perform better than random data selection. In addition we make the following observations when comparing different features.

1. Features with source context (WBi_Src and CW_Src) have the highest BLEU score performance given their size (Figure 6.1 and Figure 6.2).

2. MTU feature provides most target phrases per source with least amount of data (Figure 6.5).

3. Source word with Brown cluster context (CW_SRC) has the least number of unique source phrases with highest phrase ratio and comparable or higher overall BLEU score. This suggests this feature is able to filter out noisy or unimportant source phrases.

6.5.2 Threshold Function Comparison

As explained earlier, the empirical probability distribution of words change as a result of data selection (also known as sampling bias). Figure 6.6 shows this phenomena in random data selection. We use a density scatter plot to demonstrate changes to the distribution. In a density scatter plot darker colors are used to show concentration or overlapping data points. For example in Figure 6.6 for data size of 1M, the dark blue starting at the origin moving along the horizontal axis represents the many vocabulary items that were observed in the selection pool but are missing in randomly selected data set size of 1 million sentence pairs. As more data is selected (4M, 8M and 16M), the dark blue moves closer to the $x = y$ line as the word probabilities in selection pool and selected data get closer. It is also clear that the distance between the distributions (measured using Jensen–Shannon Divergence) drops as the selection size increases.

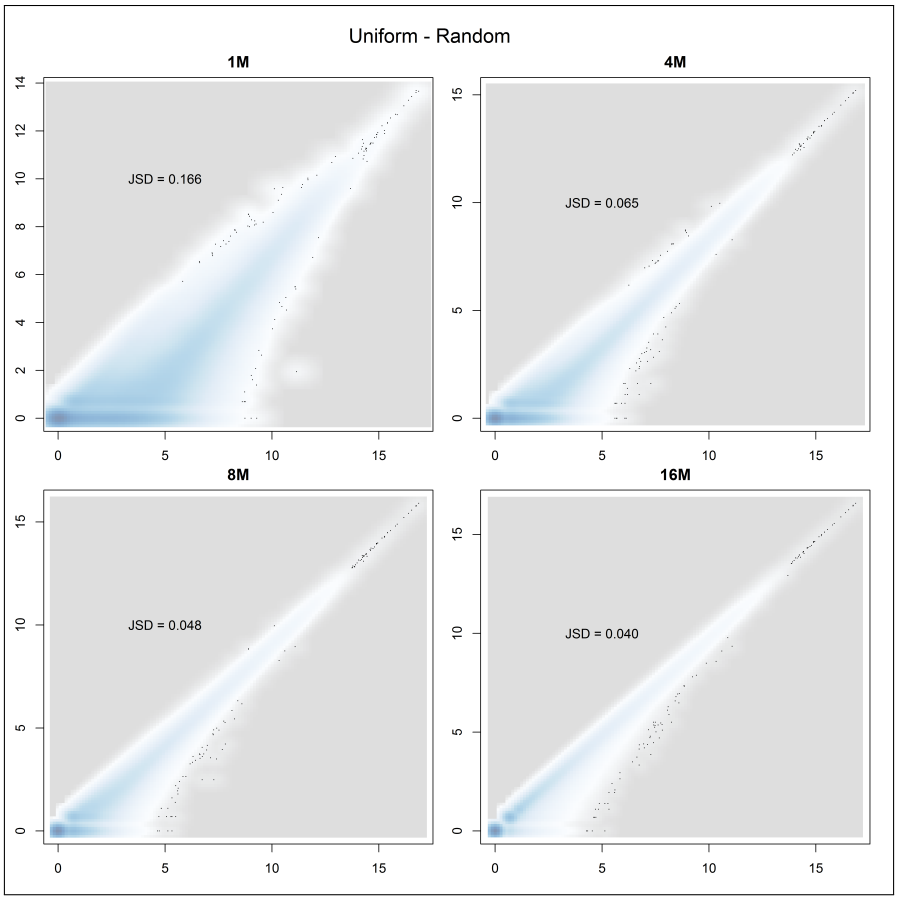


Figure 6.6 Sampling bias in random data selection. X-Axis is logarithm of frequency for words in all the data. Y-Axis is logarithm of frequency for words in selected data. The graph is generated for subsets of the data in sizes 1M, 4M, 8M and 16M (JSD: Jensen–Shannon Divergence).

Using a constant threshold causes the sampling bias to increase (compared to random selection). We measure the distance between the two distributions using Jensen-Shannon Divergence (marked on the plot with JSD). This can be observed in Figure 6.7. The constant threshold frequency can also be observed in this graph in the data set size of 4M as points get scattered away from the $x = y$ line after $x = 5$. This is due to a cut-off frequency of $2^5 = 32$. Any word with frequency of less than 32 continues to have the same frequency in the selected data while words with higher frequencies start to move away from the $x = y$ axis.

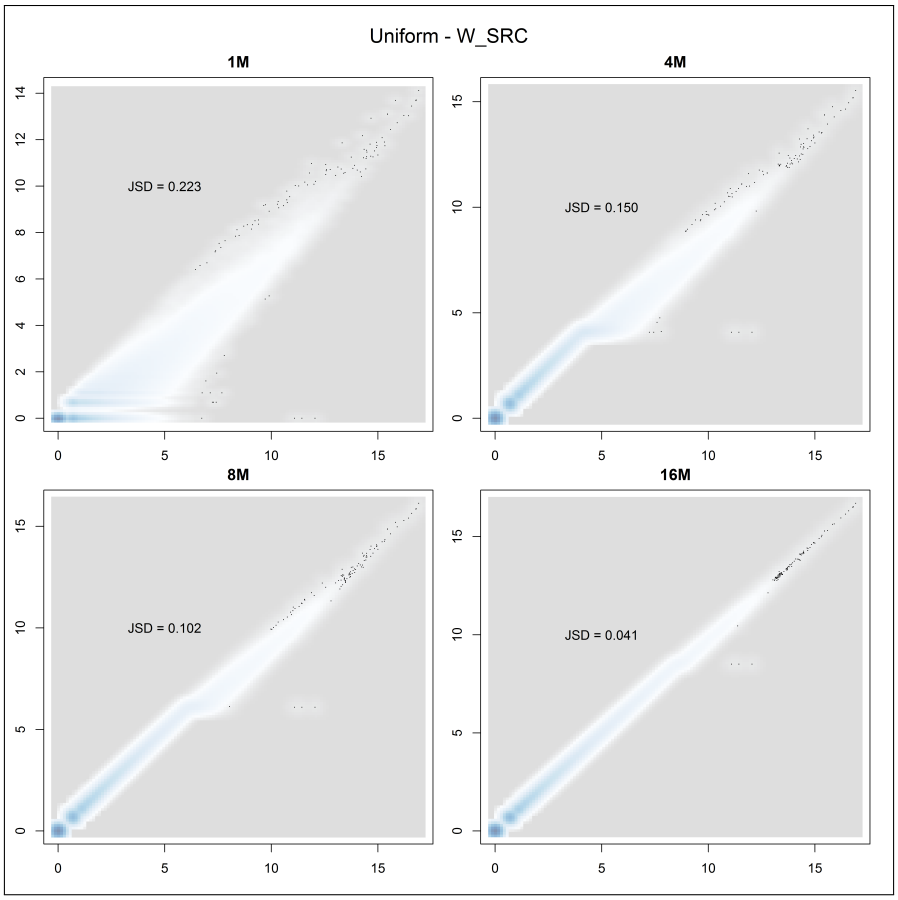


Figure 6.7 Sampling bias in uniform threshold function. X-Axis is logarithm of frequency for words in all the data. Y-Axis is logarithm of frequency for words in selected data. The graph is generated for subsets of the data in sizes 1M, 4M, 8M and 16M (JSD: Jensen–Shannon Divergence).

The sampling bias is improved using a logarithm of frequency function, but the Jensen-Shannon divergence is still higher than random data selection (Figure 6.8). The comparison with uniform can be observed clearly in the difference between the 4M data size between Figure 6.8 and Figure 6.7. Since the logarithm of frequency threshold function has a higher threshold for higher frequency words, data points move away from the $x = y$ line with a milder slope after the $x = 5$ line.

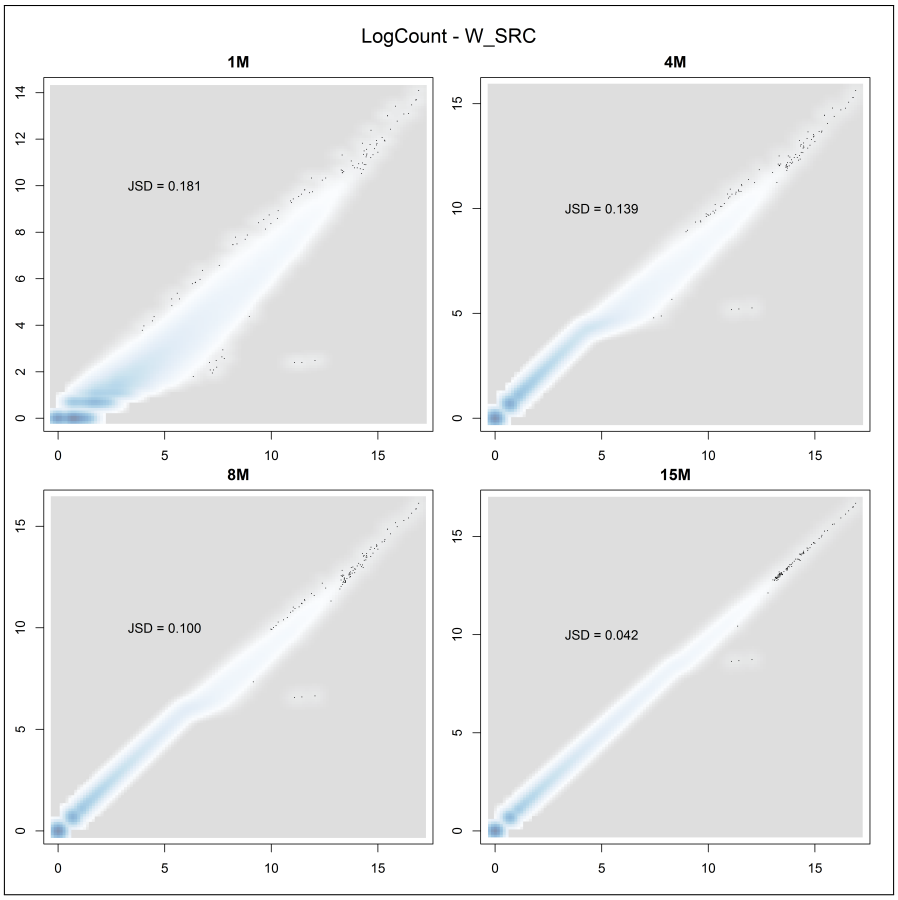


Figure 6.8 Sampling bias in data selection using logarithm of frequency threshold function. X-Axis is logarithm of frequency for words in all the data. Y-Axis is logarithm of frequency for words in selected data. The graph is generated for subsets of the data in sizes 1M, 4M, 8M and 15M (JSD: Jensen–Shannon Divergence).

Using an entropy threshold function, the Jensen-Shannon divergence improves dramatically as it is nearly half of what it is for random data selection (Figure 6.9). This can be observed by the relative closeness of data points to the $x = y$ axis for all data sizes compared to their counter parts in other threshold function figures (Figure 6.7, Figure 6.8, Figure 6.6).

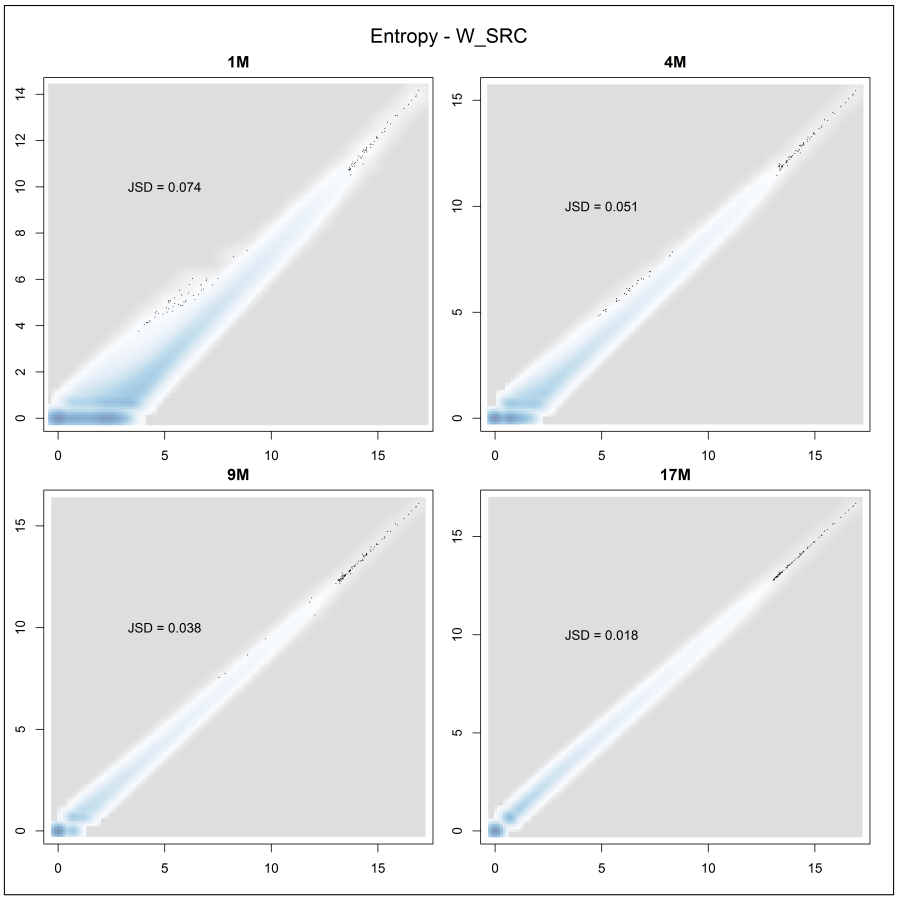


Figure 6.9 Sampling bias in data selection using entropy threshold function. X-Axis is logarithm of frequency for words in all the data. Y-Axis is logarithm of frequency for words in selected data. The graph is generated for subsets of the data in sizes 1M, 4M, 9M and 17M (JSD: Jensen–Shannon Divergence).

The entropy threshold function preserves the vocabulary distribution better than random selection, uniform threshold and logarithm of frequency threshold function. Next, we compare these methods using BLEU score on the WMT2009 and the SpeechEX1 conversational ([67]) test sets. First, we compare the threshold functions using the source vocabulary feature (Figure 6.10 and Figure 6.11).

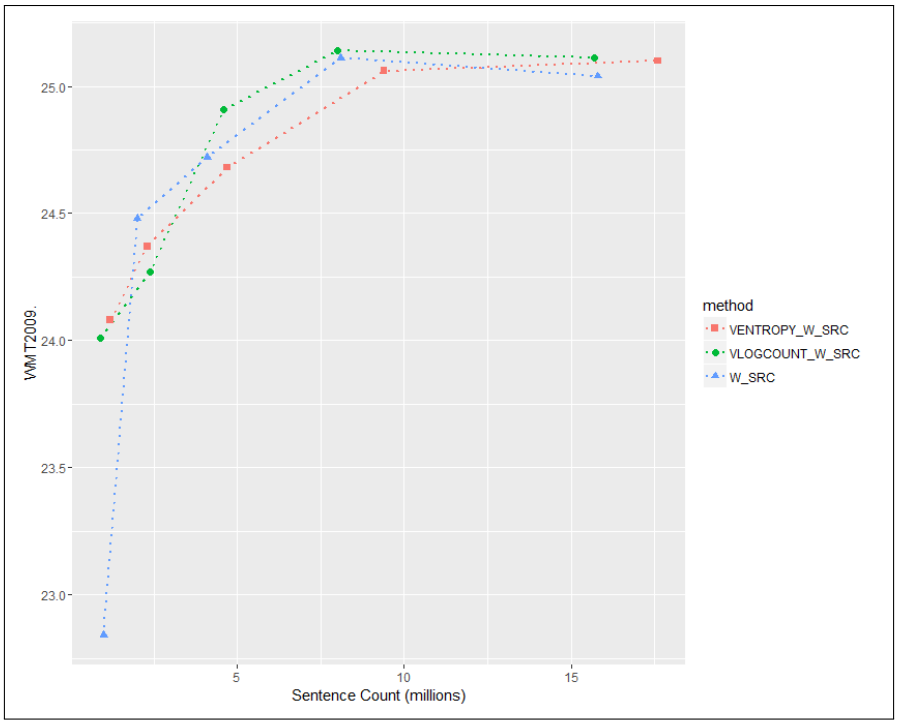


Figure 6.10 BLEU score comparison of different data selection threshold function using source vocabulary feature with the WMT2009 test set.

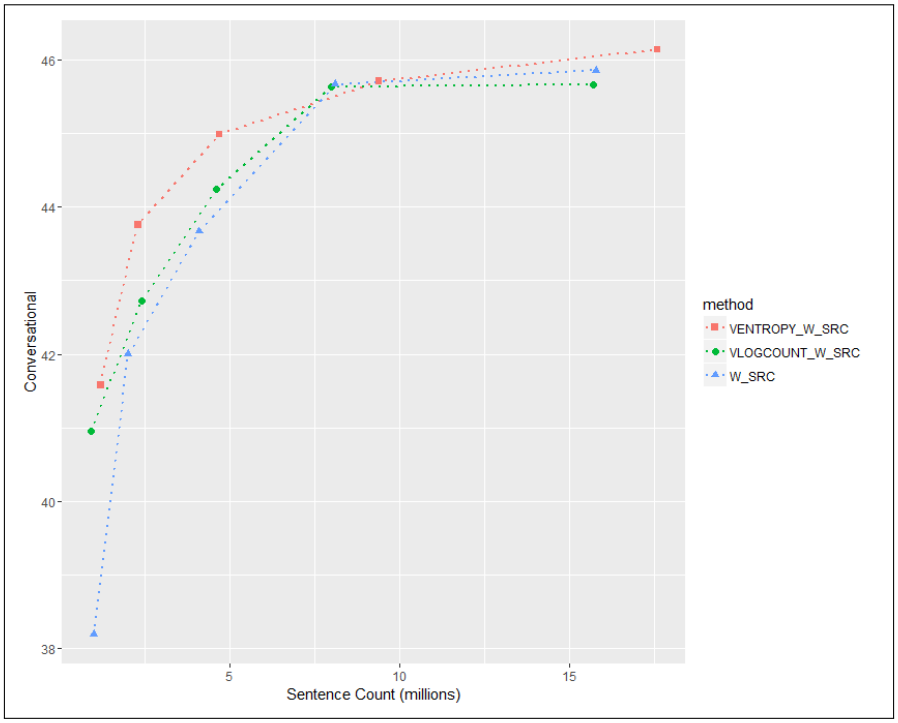


Figure 6.11 BLEU score comparison of different data selection threshold function using source vocabulary feature with the SpeechEX1 conversational test set ([67]).

The entropy threshold function and logarithm of frequency threshold functions outperform uniform threshold at small selection sizes in the WMT2009 test set but perform similarly at larger selection sizes. However, in the conversational test set the entropy threshold function outperforms other methods for most selection sizes. We confirm these findings by running the same experiment using bigram source vocabulary (Figure 6.12 and Figure 6.13) as this feature is one of the best performing features in the feature comparison plots (Figure 6.1 and Figure 6.2).

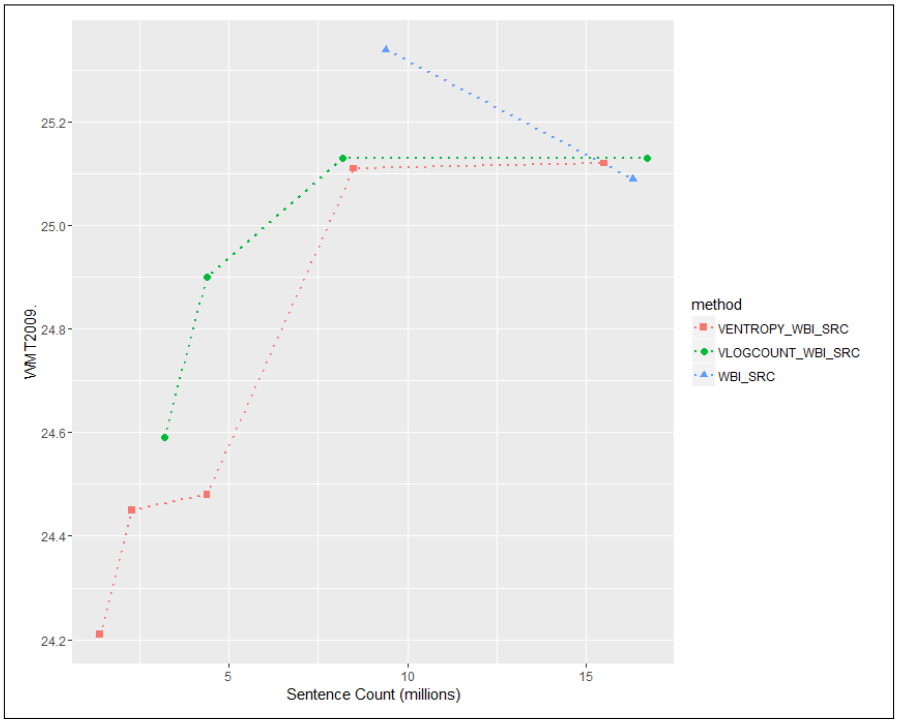


Figure 6.12 BLEU score comparison of different data selection threshold function using source vocabulary feature with the WMT2009 test set.

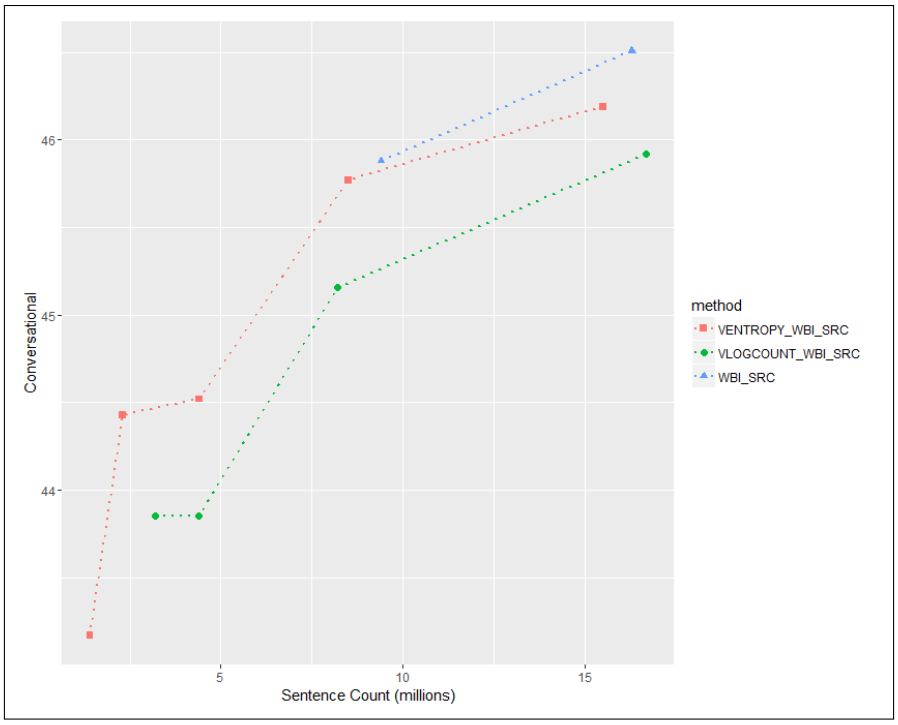


Figure 6.13 BLEU score comparison of different data selection threshold function using source vocabulary feature with the SpeechEX1 conversational test set ([67]).

We further explore phrase table statistics to better understand the differences in different threshold functions. In Figure 6.14 and Figure 6.15 the entropy threshold function has a higher average number of target phrases per source phrase in the 1 million selection size, despite having a lower unique number of phrases and total number of phrase pairs (Figure 6.16, Figure 6.17, Figure 6.18 and Figure 6.19) for both features (Source Vocabulary and Bigram Source Vocabulary).

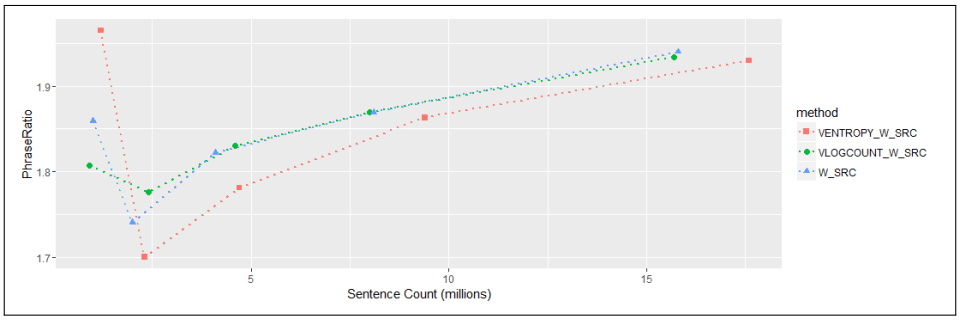


Figure 6.14 Threshold functions comparison for average number of target phrases per source phrases extracted for source vocabulary feature.

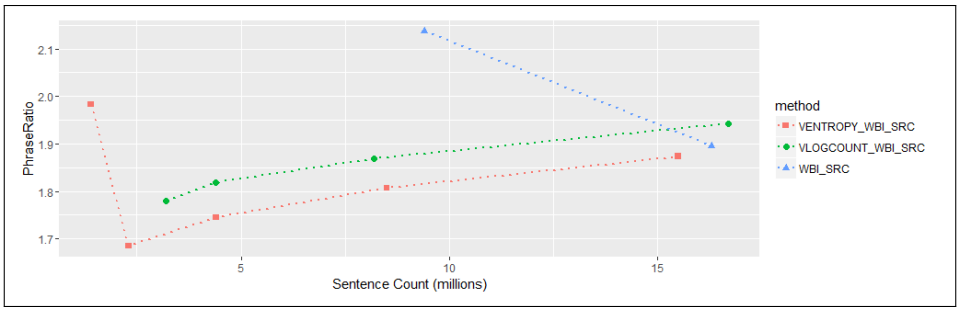


Figure 6.15 Threshold functions comparison for average number of target phrases per source phrases extracted for bigram source vocabulary feature.

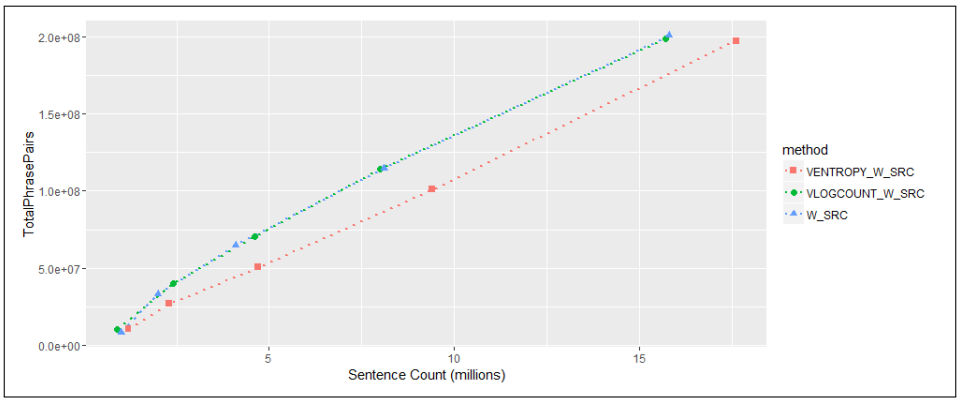


Figure 6.16 Threshold functions comparison for total number of phrase pairs extracted for source vocabulary feature.

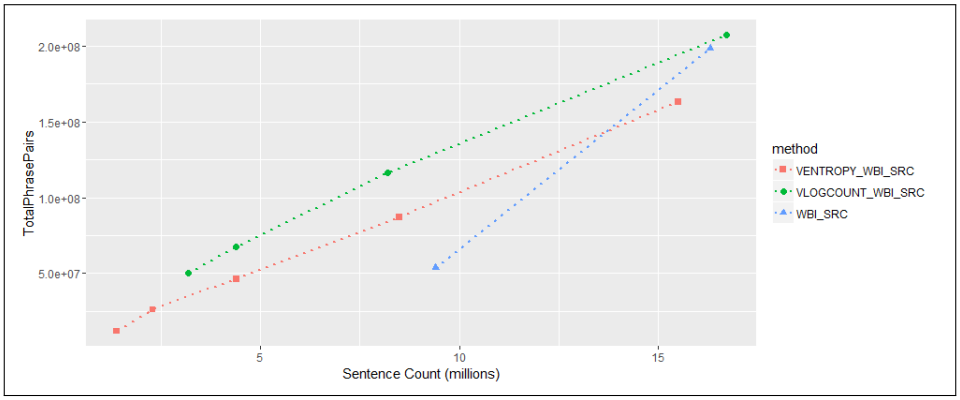


Figure 6.17 Threshold functions comparison for total number of phrase pairs extracted for bigram source vocabulary feature.

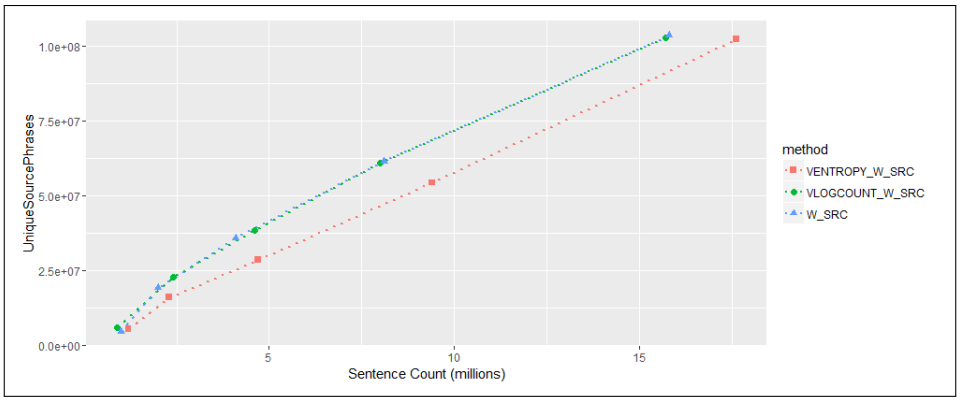


Figure 6.18 Threshold functions comparison for number of unique source phrases extracted for source vocabulary feature.

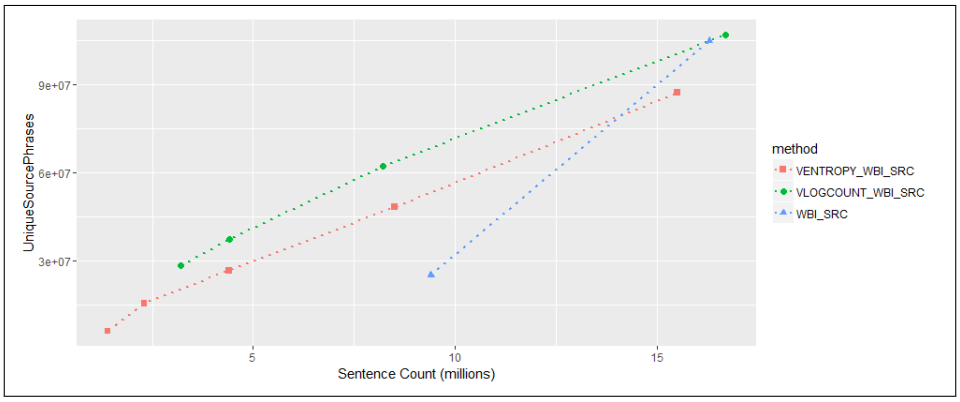


Figure 6.19 Threshold functions comparison for number of unique source phrases extracted for bigram source vocabulary feature.

The entropy threshold function excels in two areas: 1) in small selection sizes and 2) in test sets that consists of mostly high frequency vocabulary items (i.e. the SpeechEX1 test set ([67])). In the case of small selection sizes it is clear that despite having fewer or equal number phrase pairs, the data selected using the entropy threshold function results in a phrase table with higher average number of target phrases per source phrase. As the selected data set grows and less frequent words are also selected, the average number of target phrases per source phrase in phrase tables trained on data selected using different threshold functions converge. This finding is in line with higher BLEU score in the conversational test set as it mostly contains high frequency vocabulary items.

One of the limitations of the uniform threshold function is that in the case of features with high number of unique values (e.g. MTU), a threshold value of one selects most of the data (57% in the

case of MTU). Variable threshold functions can overcome this limitation by giving higher priority to more frequent or more important feature instances. We run two experiments with such features (MTU and MTU with Brown clusters) to demonstrate this. Figure 6.20 and Figure 6.21 show how BLEU score performance is effected using the logarithm of frequency threshold function at various data sizes with the WMT2009 test set.

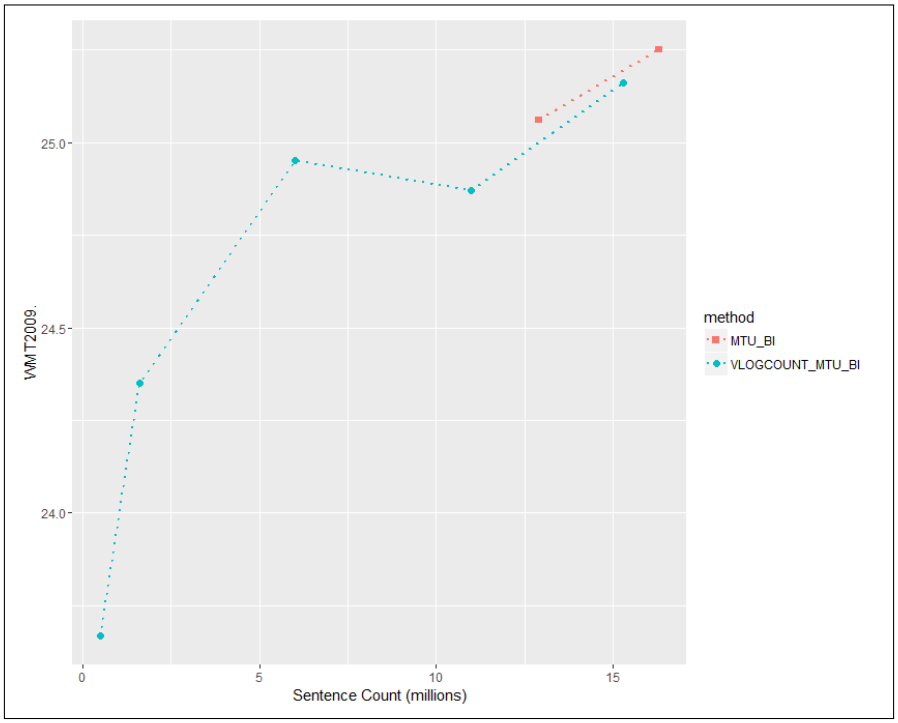


Figure 6.20 BLEU score comparison for the selection of lower data sizes using logarithm of frequency threshold function and MTU feature at different selection sizes.

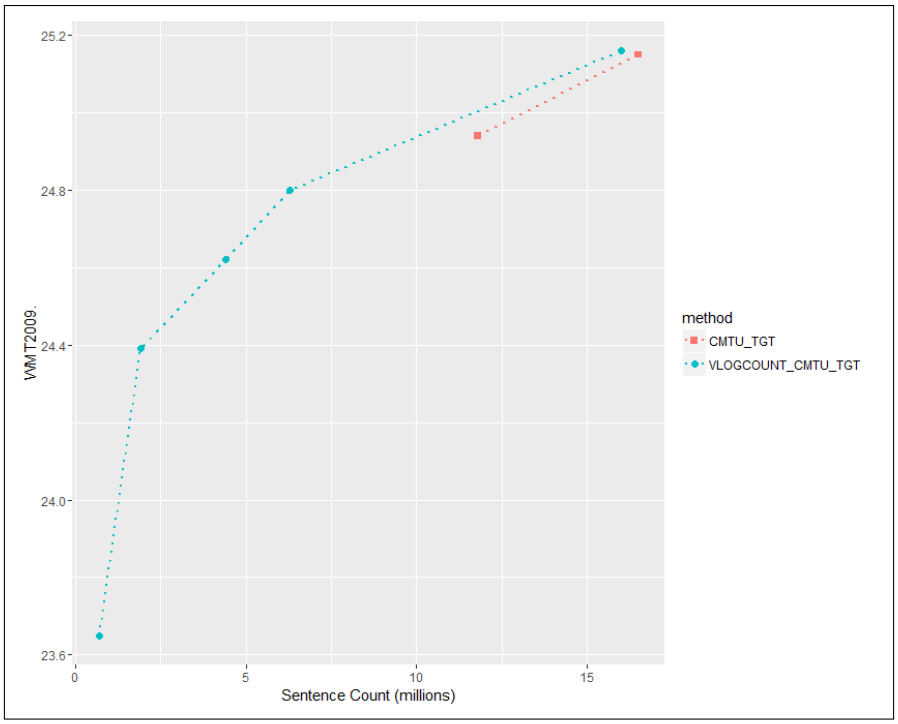


Figure 6.21 BLEU score comparison for the selection of lower data sizes using logarithm of frequency threshold function and MTU with target Brown cluster feature at different selection sizes.

Figure 6.21 and Figure 6.20 demonstrate although MTU and MTU with target Brown cluster are not able to select data sizes smaller than approximately 50% of the data, this can be achieved using a variable threshold function while the BLEU score performance drops off smoothly similar to features with less number of unique features.

6.6 Conclusion

In this chapter we extended the VSF algorithm to a flexible and efficient data selection framework which is able to take advantage of highly complex features while remaining efficient. In addition, saturation filter framework uses an incremental algorithm that allows for selecting data sets with any desirable size. While comparing usage of different features within this framework we showed features that leverage source context perform best while using Brown clusters as context has a noise filtering effect as well. In addition, while exploring different threshold functions we showed the entropy threshold function preserves the vocabulary distribution the best using Jensen–Shannon

divergence metric. The entropy threshold function also provides the highest BLEU score when selecting small data sizes as well as test sets with limited but high frequency vocabulary items.

CHAPTER 7

CONCLUSION

In this thesis we explored parallel data analysis techniques while introducing various sentence level features for the translation direction detection task. We presented a set of features that out perform all prior work. Furthermore, we developed the first cross-domain data set for the translation direction detection task and introduced new features that out perform all prior work in all cross-domain data sets. We presented a comprehensive overview of the scientific literature on the data selection task for statistical machine translation. In Chapter 5, we developed a highly scalable data selection method (VSF) that out performed state of the art in large data sets. Finally in Chapter 6, we introduced a flexible and scalable data selection framework that can leverage sentence pair level features as well as various threshold functions. We leveraged the features we developed in Chapter 4 and explored advantages and disadvantages of each feature. We further analyzed the effects of data selection on vocabulary distribution and demonstrated using an entropy based threshold function best preserves vocabulary distribution (using Jensen–Shannon divergence). In future work, we plan to develop a comparative translation model analysis tool to further explore the effects of data selection on translation models and understand some of the effects observed in translation model statistics (such as noise reduction effects with source vocabulary and Brown cluster contexts). A common problem with large data sets is noise. We plan on extending this work to perform data cleaning as well. On the algorithmic side, we plan on extending this algorithm to a distributed computing environment to improve its scalability.

BIBLIOGRAPHY

BIBLIOGRAPHY

[1] Anne Abeillé, Lionel Clément, and François Toussenel. Building a treebank for french. In Anne Abeillé, editor, *Treebanks*, volume 20 of *Text, Speech and Language Technology*, pages 165–187. Springer Netherlands, 2003. ISBN 978-1-4020-1335-5.

[2] Yvonne Adesam. *The Multilingual Forest : Investigating High-quality Parallel Corpus Development*. PhD thesis, Stockholm University, Department of Linguistics, 2012.

[3] Alexandre Allauzen, Hélène Bonneau-Maynard, Hai-Son Le, Aurélien Max, Guillaume Wisniewski, Fran\ccois Yvon, Gilles Adda, Josep M. Crego, Adrien Lardilleux, Thomas Lavergne, and Artem Sokolov. LIMSI @ WMT11. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, WMT '11, page 309–315, Stroudsburg, PA, USA, 2011. Association for Computational Linguistics. ISBN 978-1-937284-12-1.

[4] V. Ambati. *Active Learning and Crowdsourcing for Machine Translation in Low Resource Scenarios*. PhD thesis, University of Southern California, 2011.

[5] V. Ambati, S. Vogel, and J. Carbonell. Active learning and crowd-sourcing for machine translation. *Language Resources and Evaluation (LREC)*, 7:2169–2174, 2010.

[6] S. Ananthakrishnan, R. Prasad, D. Stallard, and P. Natarajan. Discriminative sample selection for statistical machine translation. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, page 626–635, 2010.

[7] S. Ananthakrishnan, R. Prasad, D. Stallard, and P. Natarajan. A semi-supervised batch-mode active learning strategy for improved statistical machine translation. In *Proceedings of the Fourteenth Conference on Computational Natural Language Learning*, page 126–134, 2010.

[8] A. Axelrod, X. He, and J. Gao. Domain adaptation via pseudo in-domain data selection. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, page 355–362, 2011.

[9] A. Axelrod, Q.J. Li, and W.D. Lewis. Applications of data selection via cross-entropy difference for real-world statistical machine translation. *Proceedings IWSLT 2012*, 2012.

[10] Amittai Axelrod, Xiaodong He, and Jianfeng Gao. Domain adaptation via pseudo in-domain data selection. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 355–362. Association for Computational Linguistics, 2011.

[11] Mona Baker. Corpus linguistics and translation studies: Implications and applications. *Text and technology: in honour of John Sinclair*, 233:250, 1993.

[12] Satanjeev Banerjee and Alon Lavie. Meteor: An automatic metric for mt evaluation with improved correlation with human judgments. In *Proceedings of the acl workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization*, volume 29, pages 65–72, 2005.

[13] Marco Baroni and Silvia Bernardini. A new approach to the study of translationese: Machine-learning the difference between original and translated text. *Literary and Linguistic Computing*, 21(3):259–274, 2006.

[14] Michael K Bergman. White paper: the deep web: surfacing hidden value. *Journal of electronic publishing*, 7(1), 2001.

[15] Austin Matthews Waleed Ammar Archna Bhatia, Weston Feely, Greg Hanneman Eva Schlinger Swabha Swayamdipta, Yulia Tsvetkov, and Alon Lavie Chris Dyer. The cmu machine translation systems at wmt 2014. *ACL 2014*, page 142, 2014.

[16] Ergun Biçici and Deniz Yuret. Instance selection for machine translation using feature decay algorithms. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, pages 272–283. Association for Computational Linguistics, 2011.

[17] Ergun Bicici and Deniz Yuret. Instance selection for machine translation using feature decay algorithms. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, page 272–283, 2011.

[18] Ergun Biçici, Qun Liu, and Andy Way. Parallel fda5 for fast deployment of accurate statistical machine translation systems. In *Proceedings of ACL 2014 Nith Workshop on Statistical Machine Translation*. Association for Computational Linguistics, 2014.

[19] Michael Bloodgood and Chris Callison-Burch. Bucking the trend: Large-scale cost-focused active learning for statistical machine translation. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, page 854–864, Uppsala, Sweden, July 2010. Association for Computational Linguistics.

[20] Ondrej Bojar, Christian Buck, Chris Callison-Burch, Christian Federmann, Barry Haddow, Philipp Koehn, Christof Monz, Matt Post, Radu Soricut, and Lucia Specia. Findings of the 2013 workshop on statistical machine translation. In *Proceedings of the Eighth Workshop on Statistical Machine Translation*, pages 1–44, 2013.

[21] Ondrej Bojar, Christian Buck, Christian Federmann, Barry Haddow, Philipp Koehn, Johannes Leveling, Christof Monz, Pavel Pecina, Matt Post, Herve Saint-Amand, et al. Findings of the 2014 workshop on statistical machine translation. In *Proceedings of the Ninth Workshop on Statistical Machine Translation*, pages 12–58. Association for Computational Linguistics Baltimore, MD, USA, 2014.

[22] Peter F Brown, Peter V Desouza, Robert L Mercer, Vincent J Della Pietra, and Jenifer C Lai. Class-based n-gram models of natural language. *Computational linguistics*, 18(4):467–479, 1992.

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

[24] C. Callison-Burch and M. Dredze. Creating speech and language data with amazon's mechanical turk. In *Proceedings of the NAACL HLT 2010 Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk*, page 1–12, 2010.

[25] Chris Callison-Burch, Philipp Koehn, Christof Monz, and Josh Schroeder. Findings of the 2009 Workshop on Statistical Machine Translation. In *Proceedings of the Fourth Workshop on Statistical Machine Translation*, Athens, Greece, March 2009. Association for Computational Linguistics.

[26] W. Chao and Z. Li. A graph-based bilingual corpus selection approach for SMT. In *Proceedings of the 25th Pacific Asia Conference on Language, Information and Computation*, 2011.

[27] WenHan Chao and ZhouJun Li. Improved graph-based bilingual corpus selection with sentence pair ranking for statistical machine translation. In *2011 23rd IEEE International Conference on Tools with Artificial Intelligence (ICTAI)*, pages 446 –451, November 2011.

[28] Boxing Chen, Roland Kuhn, and George Foster. A comparison of mixture and vector space techniques for translation model adaptation. In *Proceedings of AMTA*. Association for Computational Linguistics, 2014.

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

[30] Thomas M. Cover and Joy A. Thomas. *Elements of Information Theory*. Wiley-Interscience, New York, NY, USA, 1991. ISBN 0-471-06259-6.

[31] Sanjoy Dasgupta and Daniel Hsu. Hierarchical sampling for active learning. In *Proceedings of the 25th international conference on Machine learning*, pages 208–215. ACM, 2008.

[32] Michael Denkowski, Greg Hanneman, and Alon Lavie. The CMU-Avenue french-english translation system. In *Proceedings of the NAACL 2012 Workshop on Statistical Machine Translation*, 2012.

[33] M. Eck, S. Vogel, and A. Waibel. Low cost portability for statistical machine translation based in n-gram frequency and tf-idf. In *International Workshop on Spoken Language Translation (IWSLT)*, 2005.

[34] Sauleh Eetemadi and Hayder Radha. Effects of parallel corpus selection on statistical machine translation quality. In *Proceedings of the Pacific Northwest Regional NLP Workshop*, 2010.

[35] Tom Fawcett. An introduction to roc analysis. *Pattern recognition letters*, 27(8):861–874, 2006.

[36] Dan Feldman, Melanie Schmidt, and Christian Sohler. Turning big data into tiny data: Constant-size coresets for k-means, pca and projective clustering. In *Proceedings of the Twenty-Fourth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1434–1453. SIAM, 2013.

[37] Ian Fellows. *wordcloud: Word Clouds*, 2013. R package version 2.4.

[38] Alex Franz and Thorsten Brants. All our n-gram are belong to you. *Google Machine Translation Team*, 20, 2006.

[39] Michel Galley and Christopher D Manning. A simple and effective hierarchical phrase reordering model. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 848–856. Association for Computational Linguistics, 2008.

[40] R. Gangadharaiah, R. Brown, and J. Carbonell. Active learning in example-based machine translation. In *Proceedings of the 17th Nordic Conference of Computational Linguistics NODALIDA*, 2009.

[41] Joshua Goodman and Jianfeng Gao. Language model size reduction by pruning and clustering. In *INTERSPEECH*, pages 110–113, 2000.

[42] Google. Google Books Ngram Viewer. books.google.com/ngrams|. URL `http://books.google.com/ngrams`.

[43] G. Haffari. *Machine Learning Approaches for Dealing with Limited Bilingual Training Data in Statistical Machine Translation*. PhD thesis, SIMON FRASER UNIVERSITY, 2009.

[44] G. Haffari, M. Roy, and A. Sarkar. Active learning for statistical phrase-based machine translation. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, page 415–423, 2009.

[45] Xiwu Han, Hanzhang Li, and Tiejun Zhao. Train the machine with what it can learn: corpus selection for SMT. In *Proceedings of the 2nd Workshop on Building and Using Comparable Corpora: from Parallel to Non-parallel Corpora*, BUCC '09, page 27–33, Stroudsburg, PA, USA, 2009. Association for Computational Linguistics. ISBN 978-1-932432-53-4.

[46] Kenneth Heafield. Kenlm: Faster and smaller language model queries. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, pages 187–197. Association for Computational Linguistics, 2011.

[47] Mark Hopkins and Jonathan May. Tuning as ranking. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1352–1362. Association for Computational Linguistics, 2011.

[48] Zahurul Islam and Alexander Mehler. Customization of the europarl corpus for translation studies. In *LREC*, page 2505–2510, 2012.

[49] J. Jiang, A. Way, and J. Carson-Berndsen. Lattice score based data cleaning for phrase-based statistical machine translation. 2010.

[50] Dan Jurafsky and James H Martin. *Speech & language processing*. Pearson Education India, 2000.

[51] D. Kauchak. *Contributions to research on machine translation*. ProQuest, 2006.

[52] S. Khadivi and H. Ney. Automatic filtering of bilingual corpora for statistical machine translation. *Natural Language Processing and Information Systems*, page 263–274, 2005.

[53] Katrin Kirchhoff and Jeff Bilmes. Submodularity for data selection in machine translation. In *Empirical Methods in Natural Language Processing (EMNLP)*, October 2014.

[54] Reinhard Kneser and Hermann Ney. Improved backing-off for m-gram language modeling. In *Acoustics, Speech, and Signal Processing, 1995. ICASSP-95., 1995 International Conference on*, volume 1, pages 181–184. IEEE, 1995.

[55] Kevin Knight. A statistical mt tutorial workbook, 1999.

[56] Philipp Koehn. Europarl: A Parallel Corpus for Statistical Machine Translation. In *Conference Proceedings: the tenth Machine Translation Summit*, pages 79–86, Phuket, Thailand, 2005. AAMT, AAMT.

[57] Philipp Koehn. *Statistical machine translation*. Cambridge University Press, 2009.

[58] Philipp Koehn, Franz Josef Och, and Daniel Marcu. Statistical phrase-based translation. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*, pages 48–54. Association for Computational Linguistics, 2003.

[59] Moshe Koppel and Noam Ordan. Translationese and its dialects. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, page 1318–1326. Association for Computational Linguistics, 2011.

[60] David Kurokawa, Cyril Goutte, and Pierre Isabelle. Automatic detection of translated text and its impact on machine translation. *Proceedings. MT Summit XII, The twelfth Machine Translation Summit International Association for Machine Translation hosted by the Association for Machine Translation in the Americas*, 2009.

[61] Y. Lü, J. Huang, and Q. Liu. Improving statistical machine translation performance by training data selection and optimization. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, page 343–350, 2007.

[62] J Langford, L Li, and A Strehl. *Vowpal wabbit online learning project*, 2007.

[63] Gennadi Lembersky, Noam Ordan, and Shuly Wintner. Adapting translation models to translationese improves SMT. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, page 255–265. Association for Computational Linguistics, 2012.

[64] Gennadi Lembersky, Noam Ordan, and Shuly Wintner. Language models for machine translation: Original vs. translated texts. *Computational Linguistics*, 38(4):799–825, 2012.

[65] Gennadi Lembersky, Noam Ordan, and Shuly Wintner. Improving statistical machine translation by adapting translation models to translationese. 2013.

[66] William Lewis and Sauleh Eetemadi. Dramatically reducing training data size through vocabulary saturation. In *Proceedings of the Eighth Workshop on Statistical Machine Translation*, page 281–291, Sofia, Bulgaria, August 2013. Association for Computational Linguistics.

[67] William D. Lewis, Christian Federmann, and Ying Xin. Applying cross-entropy difference for selecting parallel training data from publicly available sources for conversational machine translation. In *Proceedings of IWSLT 2015*, December 2015. URL `http://research.microsoft.com/apps/pubs/default.aspx?id=259751`.

[68] P. Liu, Y. Zhou, and C. Zong. Approach to selecting best development set for phrase-based statistical machine translation. *Proc. of the 23rd PACLIC, Hongkong*, page 325–334, 2009.

[69] P. Liu, Y. Zhou, and C. Zong. Data selection for statistical machine translation. In *Natural Language Processing and Knowledge Engineering (NLP-KE), 2010 International Conference on*, page 1–5, 2010.

[70] P. Liu, Y. Zhou, and C. Q. Zong. Approaches to improving corpus quality for statistical machine translation. In *Machine Learning and Cybernetics (ICMLC), 2010 International Conference on*, volume 6, page 3293–3298, 2010.

[71] A. Mandal, D. Vergyri, W. Wang, J. Zheng, A. Stolcke, G. Tur, D. Hakkani-Tur, and N. F. Ayan. Efficient data selection for machine translation. In *Spoken Language Technology Workshop, 2008. SLT 2008. IEEE*, page 261–264, 2008.

[72] Christopher D Manning and Hinrich Schütze. *Foundations of statistical natural language processing*. MIT press, 1999.

[73] Mitchell P. Marcus, Mary Ann Marcinkiewicz, and Beatrice Santorini. Building a large annotated corpus of english: The penn treebank. *Comput. Linguist.*, 19(2):313–330, June 1993. ISSN 0891-2017.

[74] Robert C. Moore and William Lewis. Intelligent selection of language model training data. In *Proceedings of the ACL 2010 Conference Short Papers*, page 220–224, 2010.

[75] Robert C. Moore and Chris Quirk. Faster beam-search decoding for phrasal statistical machine translation. In *Proceedings of MT Summit XI*. European Association for Machine Translation, September 2007.

[76] Dragos Stefan Munteanu and Daniel Marcu. Improving machine translation performance by exploiting non-parallel corpora. *Computational Linguistics*, 31(4):477–504, 2005.

[77] Andrew Ng and Michael Jordan. On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes. *Advances in NIPS 14*, 2002.

[78] Alexandros Ntoulas, Marc Najork, Mark Manasse, and Dennis Fetterly. Detecting spam web pages through content analysis. In *Proceedings of the 15th international conference on World Wide Web*, pages 83–92. ACM, 2006.

[79] Franz Josef Och. Minimum error rate training in statistical machine translation. In *Proceedings of the 41st ACL*, Sapporo, Japan, 2003.

[80] T. Okita. Data cleaning for word alignment. In *Proceedings of the ACL-IJCNLP 2009 Student Research Workshop*, page 72–80, 2009.

[81] T. Okita, S. K. Naskar, and A. Way. Noise reduction experiments in machine translation.

[82] 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 on Association for Computational Linguistics*, ACL '02, pages 311–318, Stroudsburg, PA, USA, 2002. Association for Computational Linguistics.

[83] 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 on association for computational linguistics*, pages 311–318. Association for Computational Linguistics, 2002.

[84] Chris Quirk and Arul Menezes. Do we need phrases?: Challenging the conventional wisdom in statistical machine translation. In *Proceedings of the Main Conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, HLT-NAACL '06, pages 9–16, Stroudsburg, PA, USA, 2006. Association for Computational Linguistics.

[85] Chris Quirk, Arul Menezes, and Colin Cherry. Dependency tree translation: Syntactically informed phrasal smt. In *Proceedings of ACL 2005*, 2005.

[86] Philip Resnik. Mining the web for bilingual text. In *Proceedings of the 37th annual meeting of the Association for Computational Linguistics on Computational Linguistics*, pages 527–534. Association for Computational Linguistics, 1999.

[87] Hinrich Schütze, Emre Velipasaoglu, and Jan O Pedersen. Performance thresholding in practical text classification. In *Proceedings of the 15th ACM international conference on Information and knowledge management*, pages 662–671. ACM, 2006.

[88] Burr Settles. Active learning literature survey. *University of Wisconsin, Madison*, 52:55–66, 2010.

[89] Claude Elwood Shannon. A mathematical theory of communication. *ACM SIGMOBILE Mobile Computing and Communications Review*, 2001.

[90] Matthew Snover, Bonnie Dorr, Richard Schwartz, Linnea Micciulla, and John Makhoul. A study of translation edit rate with targeted human annotation. In *Proceedings of association for machine translation in the Americas*, pages 223–231, 2006.

[91] L. Specia, D. Raj, and M. Turchi. Machine translation evaluation versus quality estimation. *Machine translation*, 24(1):39–50, 2010.

[92] Ralf Steinberger, Bruno Pouliquen, Anna Widiger, Camelia Ignat, Tomaž Erjavec, and Dan Tufiş. The JRC-Acquis: a multilingual aligned parallel corpus with 20+ languages. In *In Proceedings of the 5th International Conference on Language Resources and Evaluation (LREC'2006*, page 2142–2147, 2006.

[93] Andreas Stolcke et al. Srilm-an extensible language modeling toolkit. In *INTERSPEECH*, 2002.

[94] Ilya Sutskever, Oriol Vinyals, and Quoc VV Le. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112, 2014.

[95] K. Taghipour, N. Afhami, S. Khadivi, and S. Shiry. A discriminative approach to filter out noisy sentence pairs from bilingual corpora. In *2010 5th International Symposium on Telecommunications (IST)*, pages 537 –541, December 2010.

[96] Kristina Toutanova and Christopher D. Manning. Enriching the knowledge sources used in a maximum entropy part-of-speech tagger. In *Proceedings of the 2000 Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora: Held in Conjunction with the 38th Annual Meeting of the Association for Computational Linguistics - Volume 13*, EMNLP '00, pages 63–70, Stroudsburg, PA, USA, 2000. Association for Computational Linguistics.

[97] Naama Twitto-Shmuel. *Improving Statistical Machine Translation by Automatic Identification of Translationese*. PhD thesis, University of Haifa, 2013.

[98] Nicola Ueffing and Hermann Ney. Word-level confidence estimation for machine translation. *Computational Linguistics*, 33(1):9–40, 2007.

[99] Stephan Vogel, Hermann Ney, and Christoph Tillmann. Hmm-based word alignment in statistical translation. In *Proceedings of the 16th Conference on Computational Linguistics - Volume 2*, COLING '96, pages 836–841, Stroudsburg, PA, USA, 1996. Association for Computational Linguistics.

[100] Stephan Vogel, Hermann Ney, and Christoph Tillmann. Hmm-based word alignment in statistical translation. In *Proceedings of the 16th conference on Computational linguistics-Volume 2*, pages 836–841. Association for Computational Linguistics, 1996.

[101] Vered Volansky, Noam Ordan, and Shuly Wintner. On the features of translationese. *Literary and Linguistic Computing*, page fqt031, 2013.

[102] Taro Watanabe, Jun Suzuki, Hajime Tsukada, and Hideki Isozaki. Online large-margin training for statistical machine translation. In *In Proc. of EMNLP*, 2007.

[103] Warren Weaver. Translation. *Machine translation of languages*, 14:15–23, 1955.

[104] Kai Wei, Yuzong Liu, Katrin Kirchhoff, and Jeff Bilmes. Using document summarization techniques for speech data subset selection. In *Proceedings of NAACL-HLT*, page 721–726, 2013.

[105] Worldwidewebsize. The size of the World Wide Web (WorldWideWebSize.com). WorldWideWebSize.com |. URL http://www.worldwidewebsize.com/.

[106] Joern Wuebker, Arne Mauser, and Hermann Ney. Training phrase translation models with leaving-one-out. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 475–484. Association for Computational Linguistics, 2010.

[107] David Yarowsky. Unsupervised word sense disambiguation rivaling supervised methods. In *Proceedings of the 33rd annual meeting on Association for Computational Linguistics*, pages 189–196. Association for Computational Linguistics, 1995.

[108] K. Yasuda, R. Zhang, H. Yamamoto, and E. Sumita. Method of selecting training data to build a compact and efficient translation model. In *Proceedings of the Third International Joint Conference on Natural Language Processing*, volume 2, page 655–660, 2008.

[109] Richard Zens, Franz Josef Och, and Hermann Ney. Phrase-based statistical machine translation. In *KI 2002: Advances in Artificial Intelligence*, pages 18–32. Springer, 2002.

[110] Richard Zens, Daisy Stanton, and Peng Xu. A systematic comparison of phrase table pruning techniques. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 972–983. Association for Computational Linguistics, 2012.

[111] Hui Zhang, Kristina Toutanova, Chris Quirk, and Jianfeng Gao. Beyond left-to-right: Multiple decomposition structures for smt. In *HLT-NAACL*, pages 12–21, 2013.