Hierarchical Phrase-Based Statistical Machine Translation System

Mtech. Project Dissertation

by

Bibek Behera
113050043

under the guidance of

Prof. Pushpak Bhattacharyya

Report for the partial fulfillment of

M.Tech Project

Department of Computer Science and Engineering

Indian Institute of Technology, Bombay

Declaration

I declare that this written submission represents my ideas in my own words and where others' ideas or words have been included, I have adequately cited and referenced the original sources. I also declare that I have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in my submission. I understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

____

(Signature)

____

(Name of the student)

____

(Roll No.)

Date: __________________

Abstract

The aim of this thesis is to express fundamentals and concepts behind one of the emerging techniques in statistical machine translation (SMT) - hierarchical phrase based MT by implementing translation from Hindi to English. Basically hierarchical model extends phrase based models by considering subphrases with the aid of context free grammar (CFG). In other models, syntax based models bear a resemblance to hierarchical models since the former requires corpus annotated with linguistic phrases like noun phrase, verb phrase. Hierarchical model overcomes this weakness of syntax based models since it does not require annotated corpora at all. Most Indian languages lack annotated corpus, so hierarchical models can prove to be handy in Indian to English translation. In terms of real- time implementation and translation quality, hierarchical model can coexist and even compete with state of the art MT systems. An accuracy of 0.16 (BLEU score) establishes the effectiveness of this approach for Hindi to English translation.

Secondly, we discuss post editing techniques through implementation on the translation pipeline. Post editing techniques have recently emerged as a tool for improving quality of machine translation. In this thesis, we discuss translation for out of vocabulary (OOV) words, transliteration for named entities and grammar correction. OOV words are words that were not present in training data, but were present in test data. We deal with them using two approaches. Firstly, we check whether the word is a named entity and hence can be transliterated. Secondly, if a word is not a named entity, it is sent to the OOV module where it applies statistical technique like canonical correlation analysis (CCA) to translate an unknown Hindi word. The third approach that we discuss is grammar correction.

Grammar correction can be considered as a translation problem from incorrect text to correct text. Grammar correction typically follows two approaches: rule based and statistical. Rule based approaches handle each error differently, and no uniform framework seems to be in place. We introduce a novel technique that uses hierarchical phrase-based statistical machine translation (SMT) for grammar correction. SMT systems provide a uniform platform for any sequence transformation task. Over the years, grammar correction data in electronic form has increased dramatically in quality and quantity making SMT systems feasible for grammar correction. Moreover, better translation models like hierarchical phrase-based SMT can handle errors as complicated as reordering or insertion which were difficult

to deal with previously. Secondly, this SMT based correction technique is similar in spirit to human correction, because the system extracts grammar rules from the corpus and later uses these rules to translate incorrect sentences to correct sentences. We describe how to use Joshua, a hierarchical phrase-based SMT system for grammar correction. An accuracy of 0.77 (BLEU score) establishes the efficacy of our approach.

Acknowledgments

I would like to express my sincere gratitude to my guide Prof. Pushpak Bhattacharyya for his constant encouragement and corrective guidance. He has been my primary source of motivation and advice throughout my work. I would also like to thank Raj Dabre, Rucha, Govind for helping me in this work. I also like to acknowledge with thanks the helpful suggestions made by the whole Machine Translation group at IIT Bombay throughout this work.

Finally I thank my family: my parents and my sister. They were always interested in and supportive of my work. Their constant love and encouragement since the beginning of this long journey has made its completion possible.

Bibek Behera.
Department of Computer Science & Engineering,
IIT Bombay.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Machine Translation (MT) has undergone many changes after the inception of IBM model 1 Brown et al. [1990] and phrase based machine translation Koehn et al. [2003]. While the basic noisy channel model still persists, the evolution in machine translation is quite revolutionary in itself. This chapter gives brief introduction of three sub problems of this thesis. In section 1.1, we discuss the first problem, *i.e.*, hierarchical phrase based machine translation resolves some of the key issues existing in phrase based machine translation.

In the same section, we also discuss how the field of machine translation has developed due to emergence of many open source systems like Joshua and Moses, which are state of the art machine translation systems. Later in section 1.2, we discuss the second problem, *i.e.*, we try to resolve some of the deficits that still persist in the translation quality of hierarchical system, mostly by post-editing translation. Basically, we talk about translation of out of vocabulary words (OOV). OOV words are those words that are not present in the training corpus.

In section 1.3, we discuss the third problem, *i.e.*, one of the relatively new problem in MT domain called grammar correction as an application of hierarchical phrase based MT. We put forth an idea that grammar correction can be viewed as a translation problem using hierarchical phrase based machine translation. The next section 1.4 restates the three primary problem again in detail followed by organisation of the thesis in section 1.5.

1.1 Introducing hierarchical phrase based SMT for Indian to English language machine translation

Hierarchical model uses hierarchical phrases, phrases that contain sub-phrases. Sub-phrases play an important role in translation in the sense that they are a natural way of implementing translation. A person does not remember every phrase unlike phrase based MT system but remembers small phrases and some rules. A hierarchical MT system works in a similar fashion in the sense that it learns small phrases and rules for longer phrases from a parallel corpora. *The prime minister of India* and *national bird of India* both have the same structure on either side of *of i.e.*, X_1 of X_2 where X stands for phrase or non-terminal with reference to CFG.

Phrase based system learns translation for all phrases thus giving rise to a large phrase table. On the other hand, hierarchical system learns a rule that governs the translation for phrases containing *of* and learns translation for small phrases like *prime minister* and *national bird* thus reducing the size of grammar. Even though this is a statistical system, there is intelligence in the way it models translation.

The system takes a parallel corpus as input and feeds it to the MT pipeline. The pipeline includes word alignment, rule extraction, decoding, generating k-best lists, adding the language model, pruning candidate translation, which is elaborated in coming chapters, followed by experimentation and results. Every stage in the pipeline is an outcome of extensive research in the field of MT and together they form an arsenal of state-of-the-art technologies in the field of machine translation.

Lots of researchers have combined and built open source software like Joshua and Moses, which are discussed in details in chapter 6, that have implemented hierarchical models and factor-based models for machine translation. These software provide a platform for budding researchers to develop software for hierarchical models in particular and translation models in general.

Hierarchical models have been developed from syntax based machine translation system which requires annotated corpora for both languages. In the absence of annotated corpora, syntax based models are used to annotate corpus automatically. These models require a parallel corpora with annotated corpus in either language from the parallel corpora. If the system is working on Hindi to English translation, and English corpus already has annotated corpora, the system automatically annotates the Hindi corpora thereby introducing noisy annotations. Hierarchical models do not require annotated corpus, thereby the problems associated by dealing with a noisy corpus is handled.

1.2 Post editing techniques

After translation is done by the hierarchical phrase based system, the output is forwarded to the transliteration module. The sentence might contain untranslated word which lowers down the accuracy. Ex:- केजरीवाल. These untranslated words are categorized into two classes - mainly named entity (NE) and out of vocabulary (OOV). First we detect whether the untranslated word is named entity (NE). We have used supervised techniques to detect NE using gazetteer list and Edit distance. Every NE is then transliterated using trained CRF model. If a word still remains untranslated, that word is handled by an OOV module. Ex:- ऊन्चाई. OOVs are handled using projection techniques common in image processing field. This method uses mathematical tools like Canonical correlation analysis (CCA) to find projection parameters as explained in Haghighi et al. [2008]. In the testing stage we use these parameters learnt from data to obtain translation of unknown words.

1.3 Automated Grammar Correction Using Hierarchical Phrase-Based Statistical Machine Translation

Humans have a typical way of grammar correction. Not only do they follow grammar rules, but also they keep the meaning of the sentence in mind. Computers are not capable of understanding the meaning of sentences. So they make silly mistakes that human beings can easily avoid. Existing grammar correction systems are rule-based, but there are situations which require insertion of words or reordering. These types of errors do not fall into the category of errors such as article or preposition correction. Such errors are unpredictable from rules alone. Grammar correction techniques require automation to scale to the size of data available nowadays which can be achieved if the system is statistically driven.

In this thesis, we consider grammar correction as a translation problem. So we give erroneous sentences to translation system and the system returns us correct sentence. The corrections are learned by the translation system from a parallel training corpus. The system learns SCFG (synchronous context free grammar) rules Chiang [2005] during translation. Later it converts the erroneous sentence to a tree using the grammar rules of the incorrect side only and then applies correction rules to convert the tree as explained in 8.1.1. The yield of the tree generates the correct sentence.

1.4 Problem Statement

The problem statement comprises of three questions.

- Hierarchical phrase based system a better alternative to phrase based translation models for Hindi to English translation.
- Post-editing techniques can improve the quality of translation but real time implementation is time consuming.
- Grammar correction can be treated as a translation problem. Hierarchical models can be used to correct a grammatically incorrect sentence.

1.5 Organization of the thesis

Chapter 2 provides the background of the thesis *i.e.*, takes us through all sort of translation models. The remainder of the report is structured as follows. Chapter 3 reviews the hierarchical translation model originally presented by Chiang [2005]. Chapter 4 describes how decoders which implement this model can produce n-best list of translations, using the framework introduced in Huang and Chiang [2005]. Chapter 5 introduces the idea behind tuning in translation pipeline. Chapter 6 explores state-of- the-art open source machine translation systems Joshua and Moses. Chapter 7 discusses the post-editing techniques for machine translation. Chapter 8 brings forth the problem of grammar correction as a machine translation problem. In chapter 9, we discuss the issues related to data collection via crowd sourcing. In chapter 10, we report our experiments and evaluations done on Joshua and Indian to English Machine Translation (IELMT) along with improvements in results due to incorporating post editing techniques and grammar correction. In chapter 11, we publish the results followed by discussion on impact of hierarchical phrase-based MT on IELMT in chapter 12. Chapter 13 provides conclusion for the various modules of translation pipeline followed by future work.

Chapter 2

Literature Survey

In this chapter, we discuss the relevant work done in machine translation and grammar correction field.

2.1 Machine translation

Machine Translation has its roots in cold war which led Russian to English translation. But even after war was over, US government continued its effort in this field. But the research went in vain, when Automatic Language Processing Advisory Committee (ALPAC) report (1966) exposed that the MT project had hardly fulfilled the promises it made ten years back. In the 80s, this field again started to blossom when the computing power of machines had increased. This period was marked by the introduction of very exciting statistical models for MT.

2.1.1 Approaches

Machine translation is linguistically motivated because it aims at achieving the most appropriate translation from one language to other. This means that a MT system will attain success only after it attains natural language understanding. Generally speaking, rule-based approaches involve an intermediary symbolic language obtained from the source language. This intermediate language is translated to the foreign language. Depending upon how the intermediary symbolic language is obtained, an approach is categorized as Transfer-based machine translation or Interlingua based machine translation. These methods require extensive resources and annotated training set along with large number of rules.

Rule-based MT

Rule-based techniques are linguistically driven methods of MT in the sense that they require dictionary and grammar to understand the syntactic, semantic and morphological aspects of both languages. The main approach of these methods is to obtain the shortest path from one language to another using rules of grammar. Two approaches of rule-based MT are based on interlingua and transfer-based MT. Transfer-based machine translation is based on the idea of interlingua.

Interlingual MT

Interlingua is an intermediate symbolic language that captures the meaning of the sentence in source language, sufficient to convert that into target language. This intermediate symbolic language has no dependence on either source or target language while in transfer-based MT, the interlingua obtained is somewhat dependent on the language pair. The prime reason to go for interlingua is that if there are n languages, we need only 2n translation models instead of $\binom{n}{2}$. Each language is converted into the interlingua that contains the syntax, semantic and morphology and then the interlingua can be converted to any of the language. Another advantage is that people can develop the decoders and encoders independent of the source language. For example, for Chinese to Hindi translation and vice versa, Chinese to Interlingua decoder is programmed by scientist X who has no knowledge about Hindi language. Same goes for scientist Y who is developing Interlingua to Hindi decoder.

Dictionary-based MT

This approach refers to the usage of a dictionary to translate the sentence word-by-word without caring much about the context. It is the most simple of all MT systems. This system might be used to translate phrases for inventories or catalogs of products and services.

Statistical Machine Translation (SMT)

Statistical machine translation is based on statistical data calculated from parallel corpora. Examples of parallel corpora are Canadian Hansard corpus, the English-French record of the Canadian parliament. The idea is that if a word pair translation is more frequent in the training data, it is likely that this translation will get a better probability. The entire process works on the basic idea of counting and giving probability to each translation to evaluate the correctness of the translation.

Example-based Machine Translation (EBMT)

In this method, the idea of using statistical data from a parallel corpora is extended to the next level. The system looks for similar patterns that exist in the training data and gives a translation based on examples from the training data. The first EBMT system was developed by Nagao [1984] in 1984.

Hybrid Machine Translation

As the name suggests, it takes advantage of both rule-based and statistical approaches to devise a better translation technique. One approach is to obtain the translation using rule-based MT and then correct the translation using a statistical MT.

2.1.2 Major Issues in Machine Translation

In this part, we discuss some of the frequently encountered problems in MT.

Word sense disambiguation (WSD)

A word can have several senses. For example, bank can either mean riverbank or a financial institution. WSD tries to disambiguate the sense of the word either using shallow or deep techniques. Shallow techniques assume no previous knowledge about the word, but use statistics concerning the word sense by looking at neighboring words. Deep techniques have knowledge about the various senses of the word. Despite the knowledge backup, shallow techniques perform better compared to deep techniques.

Named entity recognition

Nouns come in different forms like persons, organizations, locations, expressions of times, quantities, monetary values, percentages, etc. The job of a Named Entity Recognizer (NER) is to correctly classify nouns into one of these categories. Although the job of a NER seems trivial, it has been observed that the best rule-based and statistical implementation of NER performs poorly in domains other than the one they are trained in. This has made the development of a universal NER mandatory. In the next section, we discuss phrase based machine translation model.

2.1.3 Phrase based model

Phrase-Based models (Koehn et al. [2003]) advanced the previous machine translation methods by generalizing translation. Earlier, the words were considered as a basic unit of translation. Phrase-Based methods introduced phrases as a basic unit of translation. So sentences were concatenation of two or more phrases. This approach is good at removal of translation error caused due to local reordering, translation of short idioms, insertions and deletions.

Noisy channel approach

Basic phrase-based model is an instance of the noisy channel approach (Brown et al. [1990]). The translation of a french sentence f into an English sentence e is modeled as:

$$\underset{e}{\operatorname{argmax}} P(e|f) = \underset{e}{\operatorname{argmax}} P(e) * P(f|e) \tag{2.1.1}$$

The translation model

1. Segment e into phrases $\bar{\mathrm{e}}_1 \ldots \bar{\mathrm{e}}_n$;

2. Reorder the $\bar{\mathrm{e}}_i$'s according to some distortion model;

3. Translate each of the $\bar{\mathrm{e}}_i$ into French phrases according to a model $P(\bar{\mathrm{f}}|\bar{\mathrm{e}})$ estimated from the training data.

Other phrase-based models

There are other phrase-based models such as the joint distribution P(e,f) or the one that makes P(e) or P(f|e) as features of log-linear model. Despite this fact the basic architecture consists of the same building blocks like phrase segmentation or generation, phrase reordering and phrase translation.

Salient features of a phrase-based model

Phrase-Based models are very good in performing translations at the phrase level that have been observed from the training data. The performance of translation hardly improves as the length of substring increases beyond three words because this method relies heavily on training data. So it fails to handle sparseness of data and provide translation for longer phrases. The distortion algorithm works on top of phrase model and reorders phrase irrespective of the words in their neighborhood.

Drawbacks of phrase-based models

Often it is required to capture translations that are relevant beyond the standard three word phrase. As an example, we consider a Chinese to English translation followed by an Odia to English translation and show how phrase-based translation cannot translate longer phrases and we need special structures.

A word by word translation

First we obtain a word by word translation for each language pair.

Chinese to English Aozhou$_1$ shi$_2$ yu$_3$ Bei$_4$ Han$_5$ you$_6$ bangjiao$_7$ de$_8$ shaoshu$_9$ guojia$_{10}$ zhiyi$_{11}$.
Australia$_1$ is$_2$ with$_3$ North$_4$ Korea$_5$ have$_6$ diplomatic$_7$ relations$_7$ that$_9$ few$_{10}$ countries$_{11}$ one$_{12}$ of$_{13}$.

Odia to English Australia$_1$ tee$_2$ alpa$_3$ desh$_4$ madhiyare$_5$ gotiye$_6$ emiti$_7$ jahar$_8$ uttar$_9$ korea$_{10}$ sangare$_{11}$ rajnaik$_{12}$ sampark$_{13}$ achi$_{14}$.
Australia$_1$ is$_2$ few$_3$ countries$_4$ of$_5$ one$_6$ that$_8$ Northr$_9$ Korea$_{10}$ with$_{11}$ diplomatic$_{12}$ relations$_{13}$ have$_{14}$.

2.1.4 Problem with Phrase based MT

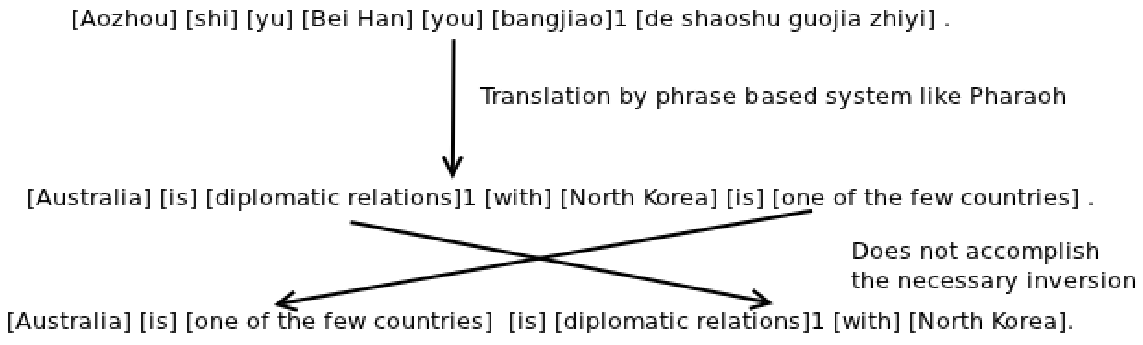

[Aozhou] [shi] [yu] [Bei Han] [you] [bangjiao]1 [de shaoshu guojia zhiyi] .
Translation by phrase based system like Pharaoh
[Australia] [is] [diplomatic relations]1 [with] [North Korea] [is] [one of the few countries] .
Does not accomplish the necessary inversion
[Australia] [is] [one of the few countries] [is] [diplomatic relations]1 [with] [North Korea].

Figure 2.1: Chinese to English phrase-based translation

When we ran phrase-based MT systems like Pharaoh on the Chinese sentence, we got the second sentence. Although it correctly translates “diplomatic relations with North Korea” and “one of the few countries”, it is not able to apply the necessary inversion of those two groups. Some other complicated reordering models like the lexical phrase reordering model might be able to accomplish such inversions, simpler distortion models will inevitably fail. The problem is not in the distortion model, but in identifying basic units of translation as we will discuss in Chapter 3.

2.2 Grammar correction

In this section, we discuss ongoing research in grammar correction. So far the work that has been done in grammar correction is based on identifying the grammar errors. Chodorow and Leacock [2000] used a ngram model for error detection by comparing correct ngrams with ngrams to be tested. Later classification techniques like Maximum entropy models have been proposed Izumi et al. [2003], Tetreault and Chodorow [2008]. These classifiers not only identify errors, but can correct them using probability values obtained from classifier for possible words. This method does not make use of the erroneous words. Thus, making the task of error correction similar to the task of filling the empty blanks. While editing sentences, humans often require the information in the erroneous words for grammar correction.

The work has also been done in using machine translation for grammar correction. Brockett et al. [2006] used phrasal based MT for noun correction of ESL students. Hermet and Désilets [2009] translated from native language L1 to L2 and back to L1 to correct grammar in their native languages obtained from translation to obtain parallel corpus. Translation techniques often suffered from lack of quality parallel corpora and also good translation systems. Brockett et al. [2006] mentioned that if high quality parallel corpus can be obtained, the task of grammar correction can be eased using a better translation model like hierarchical based machine translation. Also, the way it corrects the grammar can lead to new ways of application of grammar correction, like post-editing the translation outputs to obtain better translations.

Chapter 3

Hierarchical phrase based Machine Translation

In phrase based MT, the basic unit of translation is phrase. Hierarchical model brings sub-phrases into existence to remove the problems associated with phrase-based MT. Let us see an English to Hindi example. Consider the translation in Figure 3.1. We reduce this observation into a grammatical rule. A possible grammar rule is that the phrases on either side of the word *of* will be swapped when translating to Hindi. This is the advantage of using sub-phrases. In case of phrase level translation, this rotation is fixed only for a particular phrase and there are different rules for other phrases requiring similar rotation. This contributes to increasing redundant rules. We give some examples of phrase based translation to understand how redundancy is introduced in A.1

In phrase based MT, these redundant rules are stored in a dictionary. On the contrary, hierarchical machine translation replaces these rules by a single rule i.e.

$$X \rightarrow \langle\ X_1 \text{ का } X_2\ ,\ X_2 \text{ of } X_1\ \rangle$$

Every rule is associated with a weight w that expresses how probable the rule is in comparison to other rules with same rule in the Hindi side.
For ex:- भारत का राष्ट्रीय पक्षी {bhaarata kaa raastriiya pakshii} {India of National bird} $\rightarrow$ National bird of India bird
This example will have a similar expression on the Hindi side but different on the English side.

$$X \rightarrow \langle\ X_1 \text{ का } X_2,\ X_1\text{ ' s } X_2\ \rangle$$

Note that the ordering remains same.

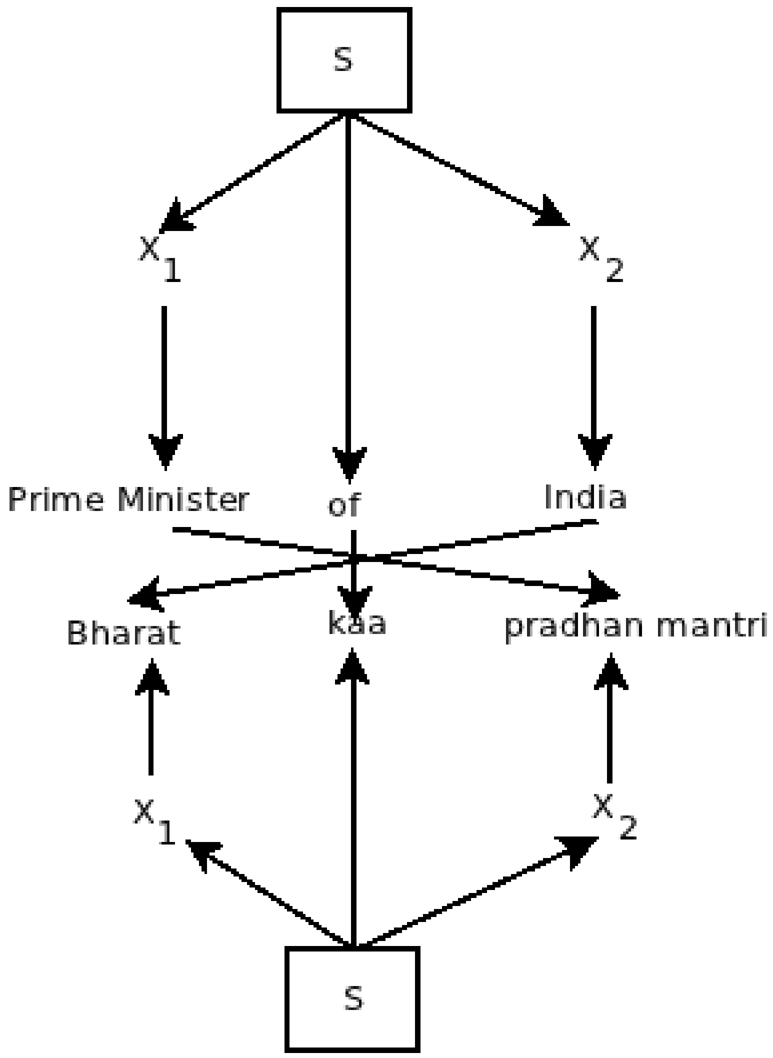


Figure 3.1: Hindi to English translation showing reordering

Basically, hierarchical model not only reduces the size of a grammar, but also combines the strength of a rule-based and a phrase-based machine translation system. This can be observed from the working of grammar extraction or decoding because hierarchical model uses rules to express longer phrases and phrases as it is for smaller phrases.

The grammar used for translation is very interesting in the sense that the system requires the same rules for parsing as well as translation. This kind of grammar is formally called synchronous context free grammar. Synchronization is required between sub-phrases because these sub-phrases need to have a number attached to them since they are essentially all X. X is the only symbol used as a non-terminal apart from the start state S. The numbering system is the way non-terminals are differentiated.

This model does not require parser at the Hindi side because all phrase are labelled as X. This is very important with respect to Indian languages, since none of the Indian languages have a good automated parser at the moment.

Phrase based systems are good at learning reordering of words. So the hierarchical model uses phrase based reordering technique to learn reordering of phrases. This can be achieved if the basic units of translation are combination of phrases and words. Systems using hierarchical models emphasize on the hypothesis that hierarchy may be implicit in the structure of a language. In the following sections, we demonstrate some grammar rules that can be automatically extracted from corpus.

Phrases are good for learning local reordering, translations of multi-word expressions, or deletion and insertions that are sensitive to local context. As we have seen in previous examples, a phrase based system can perform reordering with phrases that were present during training, but if it comes across unknown phrases that were actually not there in the corpus but are similar to a rule observed from the corpus, it will not provide the correct translation. This has been illustrated in A.2

3.1 Summarising the defects in phrase based model compared to hierarchical phrase based model

Phrase based models can perform well for translations that are localized to substrings and have been observed previously in the training corpus. Also learning phrases longer than three words hardly improves the performance because such phrases may be infrequent in the corpus due to data sparsity. The natural way seems to be learning small phrases and some grammatical rules and combining them to produce a translation.

There are also phrase based systems that try to introduce reordering termed as distortion independent of their content. But this is like fighting with your opponent blindfolded. Every reordering should be accompanied by the use of context.

All these problems are handled well by hierarchical phrase model. Certainly a leap above phrase based model, because hierarchical phrases can contain sub-phrases allowing for natural rotation of sub-phrases and learning of grammar rules.

The system learns these rules from parallel corpus without any syntactic annotation that is essential for Indian to English language MT (IELMT). The system adopts technology from syntax based machine translation system but includes the flavor of hierarchical phrases thus presenting a challenging problem.

3.2 Some notes about the system

The system that we describe later will be using rules called transfer rules. It learns such rules automatically from an unannotated bitext. Thus, this system does not require any kind of syntactic knowledge from the training data.

3.2.1 Synchronous context free grammar

Synchronous context free grammar is a kind of context free grammar that generates pair of strings.

Example:- S → I,मेन

This rule translates 'I' in English to मेन{main} in Hindi. This rule consists of terminals only i.e., words but rules may consist of terminals and non-terminals as described below.

$$VP \rightarrow \langle\ V_1\ NP_2, NP_2\ V_1\ \rangle$$

Use of synchronous CFG

The hierarchical phrase pairs can be seen as synchronous CFG. One might say that this approach is similar to syntax based MT. This is not true because the hierarchical phrase based MT system is trained on a parallel text without making any linguistic assumption that the data is annotated with part-of-speech.

Demonstrative Example

$$
\begin{array}{ll}
S \rightarrow \langle\ NP_1\ VP_2, NP_1\ VP_2 \rangle & (1) \\
VP \rightarrow \langle\ V_1\ NP_2, NP_2\ V_1\ \rangle & (2) \\
NP \rightarrow \langle\ \text{i, watashi wa}\ \rangle & (3) \\
NP \rightarrow \langle\ \text{the box, hako wo}\ \rangle & (4) \\
NP \rightarrow \langle\ \text{open, akemasu}\ \rangle & (5)
\end{array}
$$

How does this grammar work?

The parse tree begins with a start symbol in CFG but in synchronous CFG parser starts with a pair of start symbols.

Example:- $\langle\ S_{10}, S_{10}\ \rangle$

This rule means there are two parse trees instead of one. We number this symbols to avoid ambiguities when there are same elements (non terminals) occurring twice on both sides.

Example:- $\langle\ NP_{11}\ V_{13}\ NP_{14}, NP_{11}\ NP_{14}\ V_{13}\ \rangle$

Here we see that two NP symbols are co-occurring on the same side. If they are not indexed, there can be ambiguity over the correspondence of a non-terminal on the target side. This ambiguity is resolved by indexing the symbols. In this way, the non terminals are synchronized and hence this grammar is called synchronous grammar.

Next we substitute the rule for S based on the grammar.

$$
\begin{array}{l}
\langle\ NP_{11}\ V_{12}, NP_{11}\ VP_{12}\ \rangle \\
\Rightarrow \langle\ NP_{11}\ V_{13}\ NP_{14}, NP_{11}\ NP_{14}\ V_{13}\ \rangle \\
\Rightarrow \langle\ \text{i}\ V_{13}\ NP_{14}, NP_{11}\ \text{watashi wa}\ V_{13}\ \rangle \quad \text{(not allowed)} \\
\Rightarrow \langle\ \text{i}\ V_{13}\ NP_{14}, \text{watashi wa NP}_{14}\ \text{V}_{13}\ \rangle \\
\Rightarrow \langle\ \text{i open}\ NP_{14}, \text{watashi wa}\ NP_{14}\ \text{akemasu} \rangle \\
\Rightarrow \langle\ \text{i open the box, watashi wa hako wo akemasu} \rangle
\end{array}
$$

CFGs as pair of trees

The rules of synchronous CFG can be described as a pair of parse trees. The left hand side rules inside the rule region collectively gives grammar rules for obtaining a parse tree in english language. Consider following examples.

$S \rightarrow \langle\ NP_1\ VP_2\ \rangle$
$VP \rightarrow \langle\ V_1\ NP_2\ \rangle$
$NP \rightarrow \langle\ i\ \rangle$ (not allowed)
$NP \rightarrow \langle$ the box $\rangle$
$V \rightarrow \langle$ open$\rangle$

The parse trees look like in Fig3.2:

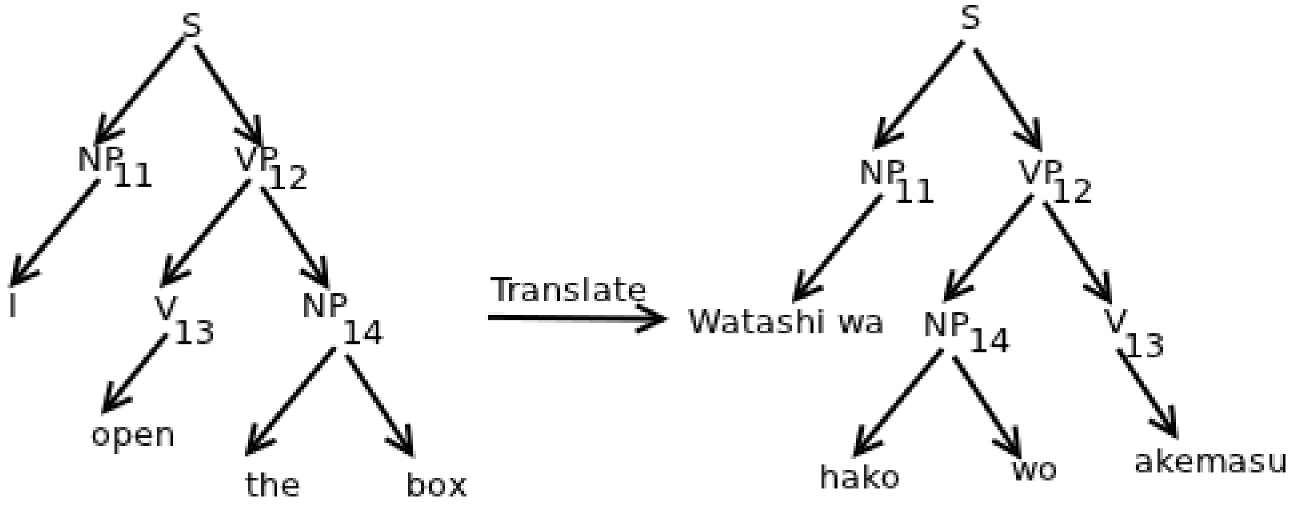


Figure 3.2: Parse tree for translation from English to Japanese

Once we have the parse tree in one language, we can construct the parse tree in other language. To accomplish the construction of the parse tree in target side, we need to apply the transfer rules and obtain the parse tree in the target language. In case there is reordering, the transfer rules cause the terminals or non terminals to rotate about a non terminal which has a corresponding rule in grammar for reordering. This has been demonstrated by the substitutions shown earlier.

3.2.2 The model

The system makes a departure from noisy channel approach to the more general log-linear model.

Log-linear model

The system evaluates a set of features for each rule it derives from the training data. Then it calculates the weight for each feature and obtains product to find the weight-age of each rule of the format X $\rightarrow \langle\gamma, \alpha\rangle$ according to this formula.

$$w(X \rightarrow \langle\gamma, \alpha\rangle) = \prod_i \phi_i(X \rightarrow \langle\gamma, \alpha\rangle)^{\lambda_i} \tag{3.2.1}$$

Note:- ϕ_i are the features and λ_i are the weights given to each feature.

There are five features similar to the ones found in Pharaoh's feature set. The features are :-

1. P($\gamma|\alpha$) and P($\alpha|\gamma$)
2. P$_w$ ($\gamma|\alpha$) and P$_w$ ($\alpha|\gamma$)
3. Phrase penalty

The feature have been divided in three sets in the manner in which they are evaluated.

Feature pair #1

$$P(\gamma|\alpha) = \frac{count(\gamma, \alpha)}{count(\alpha)} \tag{3.2.2}$$

$$P(\alpha|\gamma) = \frac{count(\gamma, \alpha)}{count(\gamma)} \tag{3.2.3}$$

The count of co-occurrences of phrase γ and α can be easily obtained from bitext simultaneously to obtain the probability. The former feature is found in noisy channel model but the latter feature was also found useful to obtain the alignment matrix discussed latter.

Lexical weights

P$_w$ ($\gamma|\alpha$) and P$_w$ ($\alpha|\gamma$) are features which estimate how well the words in phrase γ translate the words in phrase α Koehn et al. [2003].
w($\gamma|\alpha$) - probability distribution for lexical translation.

$$w(\gamma|\alpha) = \frac{count(\gamma, \alpha)}{count(\alpha)} \tag{3.2.4}$$

Given a phrase pair $\langle\gamma, \alpha\rangle$ and a word alignment *a* between the foreign word positions i = 1...n and the English word positions j = 0,1...m, the lexical weight P$_w$ is computed by

$$\prod_{i=1}^{n} \frac{1}{|\{j|(i,j) \in a\}|} \cdot \sum_{\forall (i,j) \in a} w(\gamma_i|\alpha_j) \tag{3.2.5}$$

Consider an example of translation of French phrase *f* and English phrase *e*, the alignment matrix is given as :

	f$_1$	**f**$_2$	**f**$_3$
Null	–	–	##
e$_1$	##	–	–
e$_2$	–	##	–
e$_3$	–	##	–

Table 3.2.1: Alignment matrix.

The alignment matrix provides the one to one mapping by filling the matrix with double hash for an alignment and double blank for non alignment. Based on the alignments and formula suggested above by Koehn, we obtain probability for translation of English phrase *e* to French phrase *f* given alignment *a* as in equation 3.2.6.

$$p_w(\bar{f}|\bar{e}, a) = p_w(f_1f_2f_3|e_1e_2e_3, a) = w(f_1|e_1) \times \frac{1}{2}(w(f_2|e_2) + w(f_2|e_3)) \times w(f_3|NULL) \tag{3.2.6}$$

Similarly we can obtain the probability in the opposite direction.

Phrase penalty

This feature is also similar to Koehn's phrase penalty which gives the model some flexibility in giving preference to shorter or longer derivations.

Final weight

Then the weight of D is the product of the weights of the rules used in the translation, multiplied by the following extra factors:

$$w(D) = \prod_{\langle r,i,j \rangle \in D} w(r) \times p_{lm}(e)^{\lambda_{lm}} \times exp(\lambda_{wp}|e|) \tag{3.2.7}$$

Where p_{lm} is the language model and $exp(\lambda_{wp}|e|)$, the word penalty gives some control over the length of the english output.

Chapter 4

Decoding

Basically the decoder is a CKY parser with beam search for mapping French derivations to English derivations.
Given a French sentence f, it finds the English yield of the single best derivation that has French yield f:

$$\hat{e} = \underset{D\ s.t\ f(D)=f}{\operatorname{argmax}}\ P(D) \tag{4.0.1}$$

This may not be the highest probability English string, which would require more expensive summation over derivations.
Over the next few sections I discuss the challenging technique to find the probability of single best English translation and the intricacies of decoder.

4.1 Basic Algorithm

A parser in this notation defines a space of weighted items, in which some items are designated axioms and some items are designated goals (the items to be proven), and a set of inference rules of the form

$$\frac{I_1 : w_1 ... I_k : w_k}{I : w} \phi \tag{4.1.1}$$

Which means that if all the items I$_i$ (called the antecedents) are provable, with weight w$_i$, then I (called the consequent) is provable with weight w, provided the condition ϕ holds.
In our previous example:

I$_1$($X\rightarrow$ भारत, India)	: w$_1$
I$_2$($X\rightarrow$ प्रधान मन्त्रि, Prime Minister)	: w$_2$
I$_3$($X\rightarrow X_1$ का X_2, X_2 of X_1)	: w$_3$

$$\frac{I_1 : w_1 \; I_2 : w_2 \; I_3 : w_3}{I : w_1 w_2 w_3} \tag{4.1.2}$$

Here is the derivation

I(*भारत का प्रधान मन्त्री* $\rightarrow$ *Prime Minister of India*)

More formally the well known CKY algorithm for CFGs in CNF can be thought of as a deductive proof system whose items can take one of two forms:

- $[X, i, j]$, indicating that a sub-tree rooted in X has been recognized spanning from i to j(that is spanning f_{i+1}^{j})
- X $\rightarrow \gamma$, if a rule X $\rightarrow \gamma$ belongs to the grammar G.

The axioms would be

$$\frac{}{X \rightarrow \gamma : w}(X \rightarrow \gamma) \in G \tag{4.1.3}$$

And the inference rules would be

$$\frac{Z \rightarrow f_{i+1} : w}{[z, i, i+1]} : w \tag{4.1.4}$$

$$\frac{Z \rightarrow XY : w \;\; [X, i, k] : w_1 \;\; [Y, k, j] : w_2}{[Z, i, j] : w_1 w_2 w3} \tag{4.1.5}$$

And the goal would be $[S, 0, n]$, where S is the start symbol of the grammar and n is the length of the input string f. Given a synchronous CFG, we could convert its French side grammar into Chomsky normal form, and then for each sentence, we could find the best parse using CKY. Then it would be a straight-forward matter to revert the best parse from Chomsky normal form into the original form and map it into its corresponding English tree,whose yield is the output translation. However, because we have already restricted the number of non-terminal symbols in our rules to two, it is more convenient to use a modified CKY algorithm that operates on our grammar directly, without any conversion to Chomsky normal form. Converting a CFG to CNF makes the grammar exponentially bigger, so it is better to keep the grammar, which is already a million lines as a CFG. In the next section, the above technique to transfer a tree to a string has been demonstrated with an Odia - English translation example. The section describes how to obtain grammar rules from a parallel corpus, *i.e.* training, then generating a tree for the Odia sentence, *i.e.* parsing, converting the tree in Odia to a tree in English, i.e. decoding and finally obtaining the yield of the tree in English, which is the translation.

4.2 Training

So far we have obtained a general idea about synchronous context free grammars and its usage. In the following section, we will explain the method deployed to obtain such grammar rules from a parallel corpora or bitext.

4.2.1 Illustration of word alignment algorithm

Consider the following example pair from Odia-English bitext.

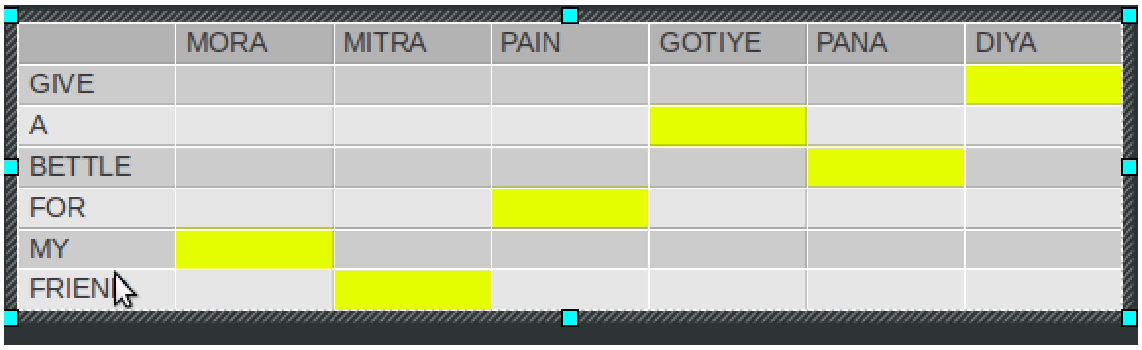
Odia: mora mitra pain gotiye pan diya
English: give a betel for my friend

Using an aligner, $O \rightarrow E$ alignment and $E \rightarrow O$ alignment are obtained, depicted as below. Taking a union of both alignments, an alignment matrix is obtained as

mora	my
mitra	friend
pain	for
gotiye	a
pana	betel
diya	give

Table 4.2.1: Odia to English Alignment

shown below.



	MORA	MITRA	PAIN	GOTIYE	PANA	DIYA
GIVE						🟨
A				🟨		
BETTLE					🟨	
FOR			🟨			
MY	🟨					
FRIEND		🟨				

Figure 4.1: Alignment matrix

4.2.2 Illustration of phrase alignment algorithm using heuristic

To obtain a phrase table, rules are used as stated below.

Rule 1. Given a word-aligned sentence pair
⟨f, e, ∼⟩, a rule $\langle f_i^j, e_{i'}^{j'} \rangle$ is an initial phrase pair of $\langle f, e, \sim \rangle$ if and only if:

$$\text{f}_k \sim e_{k'} \ \exists k \in [i,j] \ and \ k' \in [i',j'] \, ; (4.2.1)$$
$$\text{f}_k \neq e_{k'} \ \forall k \in [i,j] \ and \ k' \notin [i',j'] \, ; (4.2.2)$$
$$\text{f}_k \neq e_{k'} \ \forall k \notin [i,j] \ and \ k' \in [i',j'] \, ; (4.2.3)$$

The intuition behind this rule is that phrase f_i^j is translation of phrase $\text{e}_{i'}^{j'}$ if and only if there is some word in French sentence f at index k that is aligned to some word in English sentence at index k′. The second and third rule emphasizes that there is no word in f that is aligned to any word outside phrase e and there is no word in e that is aligned to any word outside phrase f.
Considering our previous example:

X → mora, my
X → mitra, friend
X → mora mitra, my friend
X → gotiye, a
X → pana, betel
X → diya, give
X → gotiye pana diya, give a betel

Other phrases can be made as well, but for the sake of translation, they are ignored. Returning to synchronous CFG, more complex rules need to be constructed that has sub-phrases (X) in them.

Rule 2. The rule is as follows:-
$\langle j, e_{i'}^{j'} \rangle$ is an initial phrase pair st $\gamma = \gamma_1 f_i^j \gamma_2$ and $\alpha = \alpha_1 e_{i'}^{j'} \alpha_2$ then X → $\langle \gamma_1 X_k \gamma_2, \alpha_1 X_k \alpha_2 \rangle$ is a rule, where K is an index not used in r.
Going back to our example,

Let r = X → ⟨mora mitra pain gotiye pan diya, give a betel for my friend⟩

If X $\rightarrow$ ⟨pain gotiye pan, a betel for⟩ is an initial phrase pair such that γ = γ_1 f_i^j γ_2, where γ_1 = mora mitra and γ_2 = diya and $\alpha = \alpha_1 e_{i'}^{j'} \alpha_2$ where α_1 = my friend and α_2 = give, then

X $\rightarrow$ ⟨ mora mitra X_1 diya, give X_1 my friend⟩

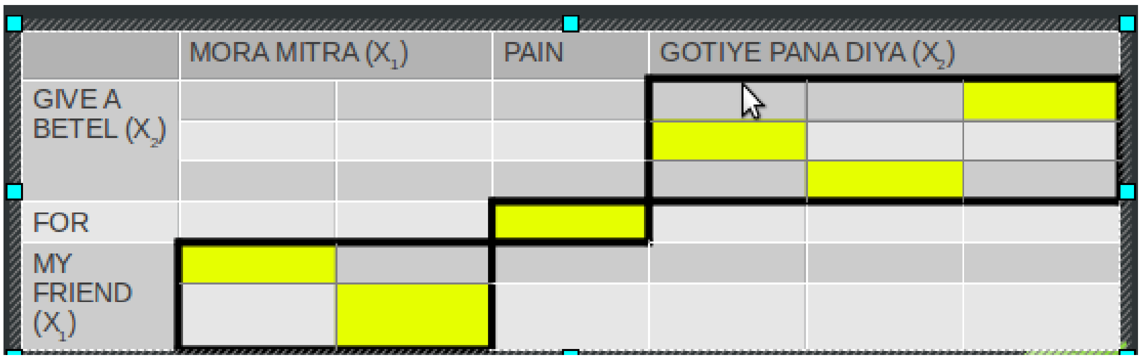


Figure 4.2: Phrase table

Note: The regions surrounded by black border indicates phrases and their phrase alignments.

4.2.3 Demerits of rule based phrase alignment and solutions to their problems

Notice that the algorithm forms general rules from specific rules. But such an algorithm could lead to unnecessary rules. Consider following example:

X$\rightarrow$ *mora mitra pain, for my friend*
X$\rightarrow$ *gotiye pana diya, give a betel*
X$\rightarrow$ *mora mitra pain gotiye pan diya, give a betel for my friend*
X$\rightarrow$ X_1 X_2, X_2 X_1

It is prohibited for nonterminals to be adjacent on the French side, a major cause of spurious ambiguity. Initial phrases are limited to a length of 10 words on either side. Rules can have at-most two nonterminals. Too many short phrases are not encouraged. A rule must have at-least one pair of aligned words.

4.2.4 Glue Rules

Glue rules facilitate the concatenation of two trees originating form the same nonterminal. Here are the two glue rules. $S \rightarrow S_1\ X_2, S_1\ X_2$

$S \rightarrow X_1, X_1$
These two rules in conjunction can be used to concatenate discontigous phrases.

4.2.5 Intuition behind using a SCFG

In the first step, we can extract CFG rules for source side language (Odia) from the SCFG rules, and parse the source side sentence with the CFG rules obtained. Let the transfer rules of a SCFG be:-
$X \rightarrow$ *diya, give*
$X \rightarrow$ *gotiye pana diya, give a betel*

Odia CFG
$X \rightarrow$ *diya*
$X \rightarrow$ *gotiye pana diya*

Given an Odia sentence we can obtain a parse tree. Let us go through a Odia to English translation and see what are the stages through which a sentence has to travel to reach the destination. Lets say a user gives our system a test sentence in Odia and is expecting an English sentence as given below.

Odia :-'Bhaina mora mitra pain gotiye pan diya.'
English-'Brother give a betel for my friend.'

4.3 Testing on Odia to English translation

So, input to the system is a sentence in Odia, and a set of SCFG rules extracted from training set. First the decoder filters only the relevant rules from the entire set of grammar rules as shown below.

SCFG for Odia to English translation
$S \rightarrow S_1\ X_2, S_1\ X_2$
$S \rightarrow X_1, X_1$
$X \rightarrow$ *Bhaina, brother*
$X \rightarrow X_1$ *pain* X_2. X_2 *for* X_1
$X \rightarrow$ *mora mitra, my friend*
$X \rightarrow$ *gotiye pana diya, give a betel*

These SCFG rules are converted to CFG rules for Odia language only. This is done

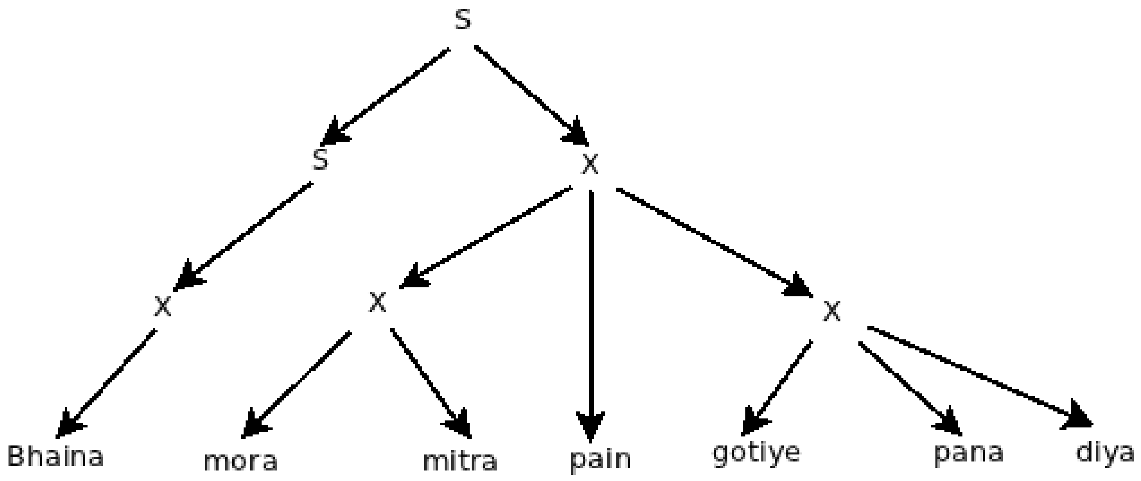


Figure 4.3: Parse Tree in Odia

by taking the source side rules because they are required to parse the given Odia sentence. **Corresponding CFG in Odia**
$S \rightarrow S_1\ X_2$
$S \rightarrow X$
$X \rightarrow$ *Bhaina*
$X \rightarrow X_1$ *pain* X_2
$X \rightarrow$ *mora mitra*
$X \rightarrow$ *gotiye pana diya*

Step 1:- Parse tree in Odia
Using a CKY parser, the tree in Figure 4.3 is obtained.

Step 2:- Apply transfer rules
We use the transfer rules one by one as shown below to map the Odia parse tree to an English parse tree as shown in Figure 4.4, 4.5, 4.6 and 4.7

$X \rightarrow$ Bhaina, brother (1)
$X \rightarrow X_1$ pain X_2. X_2 for X_1 (2)
$X \rightarrow$ mora mitra, my friend (3)
$X \rightarrow$ gotiye pana diya, give a betel (4)

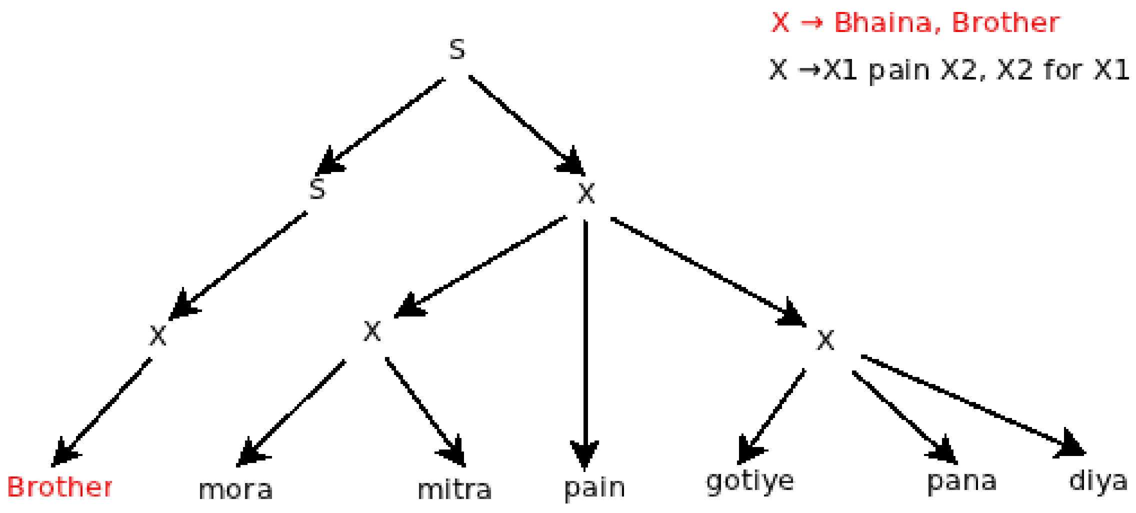


Figure 4.4: The right top corner shows one rule in red which has been applied while the second rule in white is next to be applied to the parse tree. The text mentioned in red implies that text has been translated to English while the text in white indicates that this text is yet to be translated.

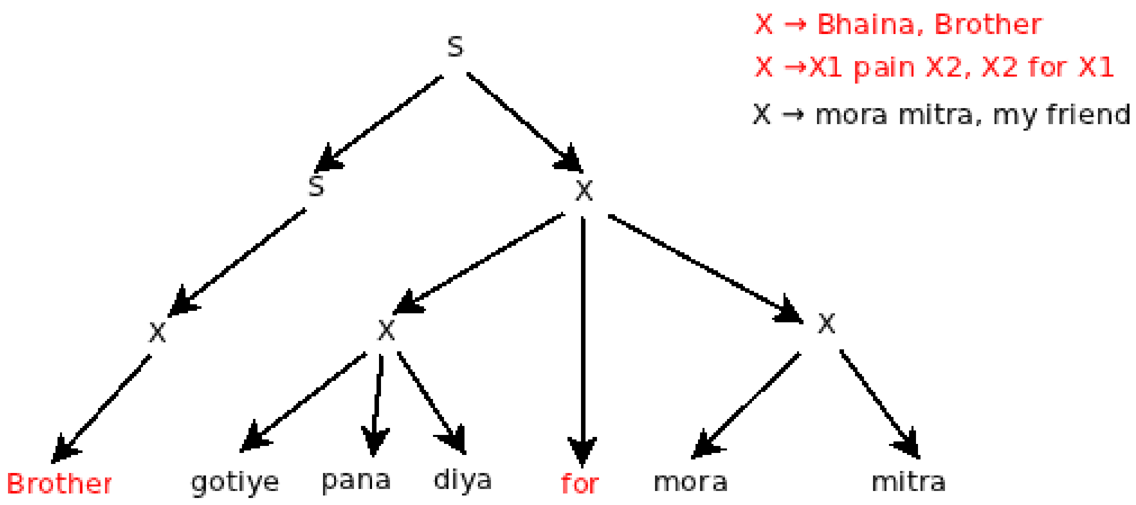


Figure 4.5: This rule replaces terminal pain by for and rotates subtree X_2 and X_1 about terminal for thus accounting for local reordering at phrase level.

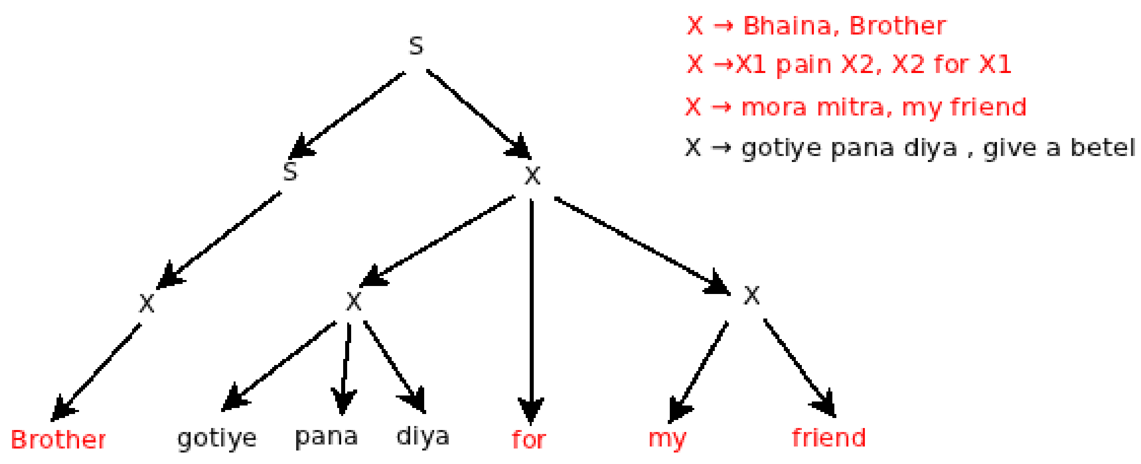


Figure 4.6: Parse Tree after applying rule #3.

Step 5:- Apply rule 4

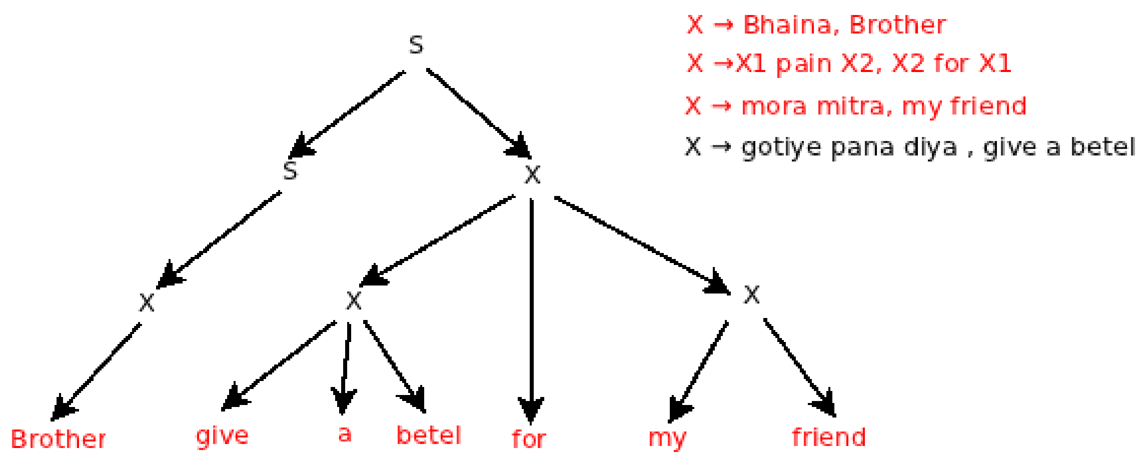


Figure 4.7: Parse Tree after applying rule #4.

Output
English:- “Brother give a betel for my friend.”

Chapter 5

Tuning

Once training is over, the parameters of the log-linear model have to be tuned to avoid overfitting on training data produce the most desirable translation on any test set. This process is called tuning. The basic assumption behind tuning is that the model must be tuned according to the evaluation techniques. The reason behind this assumption is that improvement in translation is directly proportional to improvement in evaluation methods. The evaluation methods must correlate with a human evaluator. There are many evaluation techniques, but the one that come closest to human evaluation are BLEU and NIST evaluation techniques. We have described BLEU later. The tuning technique mentioned here is called MERT (maximum error rate training).

Many state-of-the-art MT systems rely on several models to evaluate the goodness of a given candidate translation in the target language. The MT system proceeds by searching the highest-scoring candidate translation, as scored by the different model components, and return that candidate as the hypothesis translation. Each of these models need not be a probabilistic model but corresponds to a feature that is a function of a (candidate translation, foreign sentence) pair.

In case of log-linear model, each feature is assigned a weight. Och [2003] provides proof that while tuning these weights, the system should consider the evaluation metric by which the MT system will eventually be judged. This is done by choosing weights so as to improve the performance of the MT system on a development set commonly called as cross-validation set in machine learning domain, as measured by the same evaluation metric. The other contribution from Och is that he developed an efficient algorithm to find those weights.

5.1 Maximum Error Rate Training

This process is known as MERT phase in MT pipeline. Let us look at the log-linear model in MT systems and Och's efficient method before taking a look at ZMERT, a tool developed by Joshua team for the mert phase. We will discuss about Joshua MT system later.

5.1.1 Log-linear models in MT

Given a foreign sentence, the decoder aims to finding the best translation. So for a foreign sentence f the sentence with highest translation is given by

$$\hat{e} = \operatorname*{argmax}_{e} P(e \mid f) \tag{5.1.1}$$

Here the posterior probability Pr (e f) is modeled using the log-linear model. Such a model associates a sentence pair (e, f) with a feature vector

$$\Phi(e, f) = \{\Phi_1(e, f), \ldots \Phi_M(e, f)\} \tag{5.1.2}$$

and assigns a score

$$s_\Lambda(e, f) \overset{\text{def}}{=} \Lambda \cdot \Phi(e, f) = \sum_{m=1}^{M} \lambda_m \Phi_m(e, f) \tag{5.1.3}$$

for that sentence pair, where $\Lambda = \{\lambda_1 \ldots \lambda_m\}$ is the vector weight for the M features. Now the posterior is defined as:

$$P(e \mid f) \overset{\text{def}}{=} \frac{\exp(s_\Lambda(e, f))}{\sum_{e'} \exp(s_\Lambda(e', f))} \tag{5.1.4}$$

and therefore MT system selects the translation:

$$\hat{e} = \operatorname*{argmax}_{e} P(e \mid f) = \operatorname*{argmax}_{e} \frac{\exp(s_\Lambda(e, f))}{\sum_{e'} \exp(s_\Lambda(e', f))} = \operatorname*{argmax}_{e} s_\Lambda(e, f) \tag{5.1.5}$$

5.1.2 Parameter estimation using Och's method

Assume that we are moving along the dth dimension. Keeping the other dimensions fixed, the program moves along the dth dimension such that if there is a weight vector $\Lambda = \{\lambda_1 \ldots \lambda_d \ldots \lambda_M\}$, the new weight vector obtained by varying the dth dimension is optimal.

Consider a foreign sentence f and a list of candidate translations $\{e_1 \ldots e_K\}$. The best translation at a given Λ is the one that maximizes the score given by $s_\Lambda(e_K, f)$ defined as $\sum_{m=1}^{M} \lambda_m \Phi_m(e, f)$. The sum can be rewritten as $\lambda_d \Phi_d(e_K, f) + \sum_{m \neq d} \lambda_m \Phi_m(e, f)$. The second term is constant with respect to λ_d and so is $\Phi_d(e_K, f)$. This formulation is similar to a straight line equation and defined as follows.

$$s_\Lambda(e_K, f) = slope(e_K)\lambda_d + offset_\Lambda(e_K) \tag{5.1.6}$$

If λ_d is varied, then the score moves in a straight line for a sentence e$_k$. If a plot is drawn for all candidates, then the upper envelope indicates the best candidate at any given λ_d. The visualization is shown in 5.1. So the intersection points are our point of interest. If the intersection points are put in a set of critical values, where each point refers to 1-best change for a single sentence. Next time we need not rescore the candidate, but simply adjust the score as dictated by the candidate change associated with the intersection point.

The last decision making is that of making the choice for candidates for translation. If top 300 candidates are taken, search space is reduced since the top 300 candidates form a restricted set. Instead choosing the top candidates and optimizing the weight vector are done alternately, with new set of candidates merged with old set of candidates. The process is repeated till the weight vector converges, indicated by the lack of growth in size of candidate set.

5.2 ZMERT

ZMERT Zaidan [2009] is part of research and development at John Hopkins University to develop JOSHUA, an open source software package that implements hierarchical phrase based MT. The developers of JOSHUA desired to make it flexible and easy to use which were observed in the development of ZMERT, Joshua's MERT module. ZMERT is independently available as open source since it does not rely on any of Joshua's modules.

ZMERT works in a fashion described by Och. It takes a greedy approach for optimizing the weight vector along one of the M dimensions selecting the candidate that gives the maximum gain.

ZMERT is a tool that is easily integrable into any MT pipeline. This tool is easy to use and setup has a demonstrably efficient implementation. ZMERT has been developed with great care so that it can be used with any MT system without any modification to the MT code and without the requirement of extensive manuals, which is a situation that often arises in today's MT pipeline.

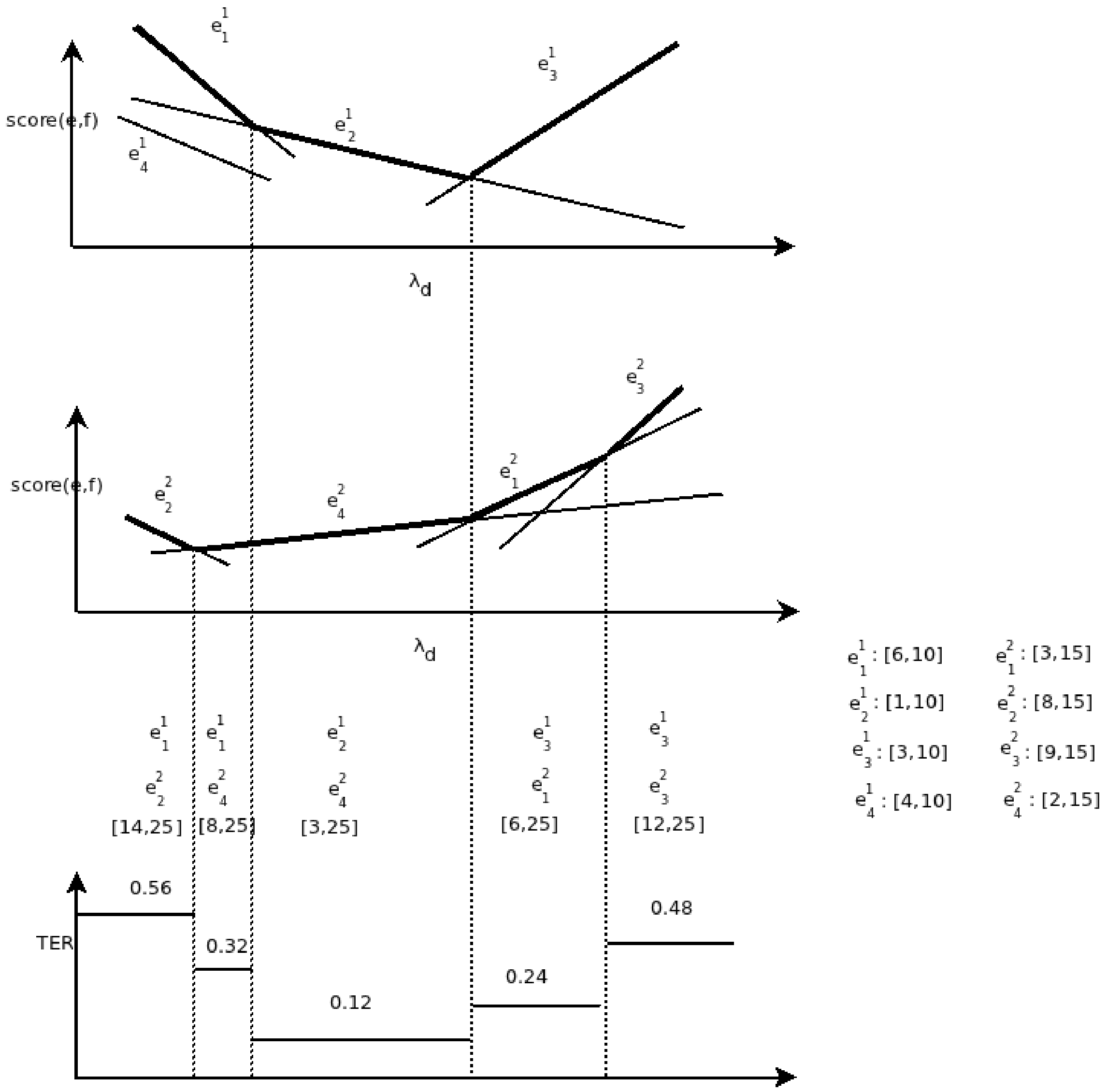


Figure 5.1: Och's method applied to a foreign sentence f

5.3 Existing MERT Implementations

There are plenty of MERT applications available as open source which could have been fit in the MERT module of JOSHUA. But the team decided to make one of their own primarily because the existing applications lacked in bits and pieces.

The first MERT implementation appears to have been used by Venugopal [2005]. The problem is that its written in MATLAB, which, like other interpreted languages, is quite slow. Secondly, MATLAB is a proprietary product of The MathWorks, which restricts the user space to people having license for using MATLAB.

ZMERT on the other hand is written in JAVA, hence is extremely fast. This also makes the user domain unrestricted because JAVA is freely available to all.

The second MERT implementation is observed in MERT module of Phramer Olteanu et al. [2006], an open source MT system written by Marian Olteanu. The MERT module is written in JAVA, but the module consists of as many as 31 files. Some of these are class definition such as evaluation metric, yet the MERT core consists of 15-20 files. Compared to this ZMERT has only 2 files. This makes ZMERT compilation almost trivial and running it quite easy.

Chapter 6

Open source hierarchical phrase based machine translation system

Large-scale parsing-based statistical machine translation (e.g. Chiang [2007], Quirk et al. [2005], Galley et al. [2006], Liu et al. [2006]) has made remarkable progress in the last few years. However most of the systems mentioned above are not open source and hence are not easily available for research. This results in a high barrier for new researcher to understand previous systems and improve them. In this scenario, open source can play a huge role in improving the number of experiments and magnitude of research going on in MT world. In the following topics, we present two of the well known open source hierarchical phrase-based MT systems.

6.1 JOSHUA

Joshua is an open source statistical MT toolkit. Joshua implements all of the algorithms required for synchronous CFGs: chart parsing, n gram language model integration, beam and cube pruning, and k-best extraction. The toolkit also includes a module for suffix array grammar extraction and minimum error rate training (MERT). To accommodate scalability, it uses parallel and distributed computing techniques. It has been demonstrated that the toolkit achieved state-of-the-art translation performance on the WMT09 French-English translation task.

6.1.1 Main functionalities

In this part, we have discussed the various functionalities of Joshua pipeline.

Training corpus sub sampling

Instead of using the entire corpus for extracting grammar, only a sample of the corpus is used as proposed by Kishore Papineni. This method works as follows:

for the sentences in the development and test set that are to be translated, every n gram up to length of 10 is gathered in a map W. Only those sentence pairs are selected from the training set that contains any n-gram found in W with a count of less than k. Every sentence that is selected causes an increment of the n-grams in W present in it by their count in that sentence. The reason is that similar sentences, *i.e.*, sentences containing the same n- grams will be rejected subsequently. This helps in reducing redundancy in new training set and less time taken while training.

Suffix Array Grammar Extraction

Hierarchical phrase-based MT requires grammar extracted from parallel corpus but in real translation tasks, grammar are too big and often violate memory constraints. In such tasks,feature calculation is damn expensive considering the time required; huge sets of extracted rules must be sorted in opposite direction to obtain features like translation probability p (f | e)and p (e | f) (Koehn et al. [2003]). In case the training data is changed, the extraction steps have to be re run. To alleviate such issues, a source language suffix array is used to extract only those rules that will be useful in translation following Callison-Burch et al. [2005]. This reduces the rule set compared to techniques that use the entire training set from extracting rules.

Decoding Algorithms

In this part, we describe the various sub-functionalities of the decoding algorithms as described in Li et al. [2010].

Grammar Formalism The decoder implements a synchronous context free grammar (SCFG) of the kind described by Heiro. (Chiang [2005]).

Chart Parsing Given a source sentence, the decoder produces 1-best and k-best translation using a CKY parser. The decoding algorithm maintains a chart, which contains an array of cells. Each cell in turn maintains a list of proven items. The parsing process starts with axioms, and proceeds by applying the inference rules repeatedly to prove new items until proving a goal item. Whenever the parser proves a new item, it adds the item to the appropriate chart cell. The item also maintains back pointer to antecedent items, which are used for k-best extraction.

Pruning Severe pruning is required to make decoding tractable. The decoder incorporates beam pruning and cube pruning (Chiang [2005]).

Hypergraph and k-best extraction For each source language sentence, the chart parsing algorithm produces a hypergraph, that contains an exponential set of likely derivation hypotheses. Using k-best algorithm, the decoder extracts the top k translations for each sentence.

Parallel and Distributed decoding They also work on parallel decoding and distributed language model using multi core and multi processor architecture and distributed computing techniques.

6.1.2 Language Model

They implement an ngram language model using a n-gram scoring function in Java. This java implementation can read ARPA fromat provided by SRILM toolkit and hence the decoder can be used independently from SRILM. They also developed their own code that allows the decoder to use the SRILM toolkit to read and score n-grams.

6.1.3 MERT

JOSHUA's MERT module is called ZMERT as described earlier. It provides a simple java implementation to efficiently determine weights for the log-linear model used for scoring translation candidates to maximize performance on a development set as measured by an automatic evaluation metric, such as BLEU.

6.2 Moses

Moses Koehn et al. [2007] is also an open source phrase-based MT system. Recently it has started developing hierarchical phrase-based MT to become a complete toolkit. Moses was developed prior to JOSHUA. Hence it brought in a completely out of the box translation toolkit for academic research. Developed by several scientists in the University of Edinburgh, it gave big boost to MT research. Also it brought new concepts like a pipeline in the era of MT systems wherein you just give a shell command, the pipeline is executed automatically making the system user friendly. The pipeline consists of three different stages training, testing and tuning.

The developers of Moses were concerned about phrase-based model's limitations which translated chucks of words without making any use of linguistic information like morphological, syntactic or semantic. So they integrated factor-based translation in which every word is morphologically analyzed and then translated. This certainly improves the quality of translation.

6.2.1 Factored Translation Model

Non factored SMT deals with chunks of words and has one phrase table as explained in **??**

6.3 Example of phrase based MT lagging

Translate:-
I am buying you a green cat.
“मै आप के लिये एक हरे रन्ग की बिल्ली खरीद रहा हून.
Using phrase dictionary.
I → *मै*
am buying → *खरीद रहा हून*
you → *आप के लिये*
a → *एक*
green cat → *हरे रन्ग की बिल्ली*

In factored translation , the phrases may be augmented with linguistic information like lemma or POS tags.

$$\begin{pmatrix} billi \\ NN \\ billi \\ sing/fem \end{pmatrix} \rightarrow \begin{pmatrix} cat \\ NN \\ cat \\ sing \end{pmatrix} \tag{6.3.1}$$

Mapping of source phrases to target phrases can be done in a number of steps so that different factors can be modelled separately thereby reducing dependecies between models and improving flexibility.

For ex:- sing/pl masc/fem should not depend on POS tag.

घरो → घर + “ओ ”→ Lemma⟨घर ⟩ POS⟨NN⟩ mod⟨pl⟩ $\overset{\text{translate to english}}{=}$ Lemma⟨house ⟩ POS⟨NN⟩ mod⟨pl⟩ → house + “s ”→ houses.

So the surface form was first transformed to lemma and surface forms, then the target was built from the lemma and other linguistic information. This reduces the size of phrase table considerably.

6.3.1 Toolkit

It consists of all the components needed to preprocess data, train the language models and the translation models. For tuning, it uses MERT and BLEU for evaluating the resulting translations. Moses uses GIZA++ for alignment and SRILM for language modeling. The toolkit is available online as open source under sourceforge.

The decoder is the core component of the toolkit which was adopted from Pharaoh to attract the interests of followers of Pharaoh. In order for the toolkit to be adopted by the community, and to make it easy for others to contribute to the project, the following principles were kept in mind:

- Accessibility
- Easy to maintain
- Flexibility
- Easy for distributed team development
- Portability

It was developed in C++ for efficiency and followed modular, object oriented design.

Chapter 7

Post Editing Techniques

In this chapter, we take a look at three post editing techniques for improving translation quality of our hierarchical phrase based system. We also show the overall model of our translation system including post-editing techniques.

7.1 Named entity Recognition

In this section, we discuss about collection, detection and correction of named entities.

7.1.1 Preparation of Gazetteer List

The gazetteer list has been manually collected from the Web by combining three entities such as *Common Indian Names*, *Common Indian Surnames* and *Names of places in India*. Common Indian names and surnames made up for 7943 entities while names of places summed to 350 entities. The sources were the websites provided in Appendix A.3

7.1.2 Named Entity Recognition

The approaches used for named entity recognition have varied between supervised and rule based such as :-

- Supervised Approach as discussed in Borthwick [1999]
- Non statistical tools - GATE (General Architecture for Text Engineering) provides a Nearly-New Information Extraction System (ANNIE) API for Named Entity Recognition.

The task of a NER system is to find named entities in a monolingual corpus of newspaper domain. Later these named entities can be used to form gazetteer list.

7.1.3 Algorithm for NER

```
Input: Hindi word
Output: Named entity or not Find all untranslated words in the output.
foreach $Untranslated\_Word$ in the tranlated ouput do
    if $Untranslated\_Word$ does not exist in the Gazetteer List then
        Index and get similar words from Gazetteer list
        D:= Find edit distance
        if D is within 2/3rd of the number of letters then
            $Untranslated\_Word$ is a named entity
        else
            $Untranslated\_Word$ is an OOV word
    else
        $Untranslated\_Word$ is a named entity;
end
```

Algorithm 1: Algorithm for names entity recognition

7.1.4 NER and OOV

All the recognized named entities are transliterated directly. All the OOV words are given to the OOV handling module.
Some examples of untranslated words:

औटो, लैन्डिन्गा, कोका कोका, पेप्सि, बिग कोला, केजरीवाल, आइआइटी, चौ-टाला, घोटाले, तीहाड, हेलिकोप्टर, हेमराज

Output:

NER + transliteration: auto, Pepsi, Bigkola, Kejriwal, Eiity, Chautala, Ghotale, Ti-had, Helicopter, hemraj

OOV: लैन्दिन्ग्, कोका कोका

आइआइती is transliterated to Eiity while the correct output is IIT. *Note:- Abbreviations like IIT cannot be handled by NER and OOV.*

7.1.5 Transliteration

Earlier we were using a word transliterator. Now we have completed the transliterator module. So it takes a file and converts any hindi word or number to english. For the word transliterator, we are using CRF based transliterator. But it does not translates numbers. So for devnagiri number translation, we wrote a java code that uses if then rules.

Transliteration module

This module has been uploaded in the server and could be used for public purposes. IT is well supplemented with readme files. But the transliterator is not good enough for abbreviations. आइआइती- aaiti, aaity, aschiti, aasti, aschity. Thus, we need a list of abbreviation. So the transliterator stage is preceeded by abbreviation detection and translation stage.

7.2 Handling OOV Words: Using Dictionary Mining

In this section, we discuss a very interesting way of translating out of vocabulary words.

7.2.1 Overview: Dictionary Mining

It is observed that domain divergence in test sentences causes increase in unseen words and degrades the translation performance. So, we are using a dictionary mined from comparable corpora. This mined dictionary will be integrated with the baseline Joshua MT system. The dictionary mining approach is based on Canonical Correlation Analysis.

7.2.2 Canonical Correlation Analysis

In our context, given a source and a target language side, CCA is a technique to find projection directions on both sides, so that when the same word concept is projected along these directions (independently from source side and from target side), then the projections are maximally aligned in the canonical space. The figure 7.1 illustrate this phenomenon for the word "samay" and "time". They have a common representation in the latent space or the z space or the canonical space.

7.2.3 Brief Overview of Steps

1. Extract feature vectors for top N words in both the languages.

2. Using the dictionary probabilities of seen words, we identify pairs of words whose feature vectors are used to learn the CCA projection directions.

3. We project all the words into the sub-space identified by CCA and mine translations for the OOV words.

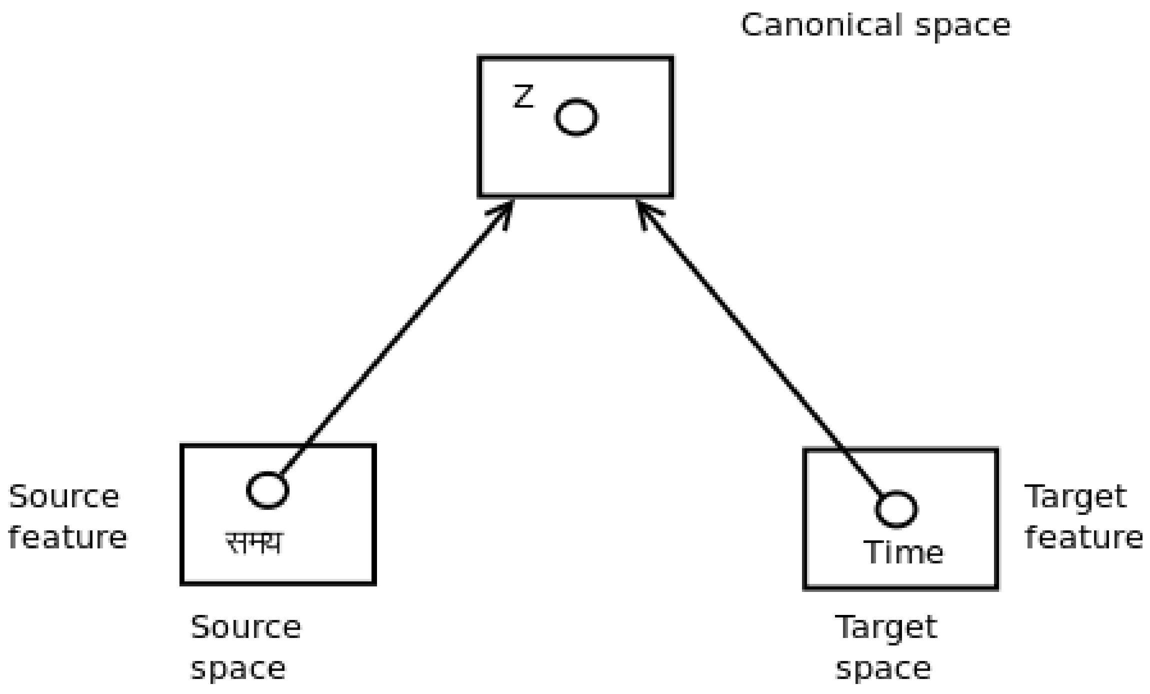


Figure 7.1: Canonical space

7.2.4 Features Used

Two types of features are used:

- Context features - Find count of the words occuring within a distance of 5 from a given word. Select only those words (and corresponding context vectors) which occur with at least 5 different words.
- Orthographic features - Find the count of occurrences of each 3-gram of a word within that word. eg. word time has orthographic features - #ti, tim, ime, me#.

Example of feature vectors are given in A.6

7.3 Grammar correction

It is observed that the translated output is often grammatically incorrect. In general, incorrect placement of function words, non-agreement between noun and verb, incorrect preposition, etc are the causes of grammatically incorrect output. To correct this, we add a post-processing module for grammar correction. This module in itself is an MT system, trained on a parallel corpus in which grammatically incorrect sentences are on the source side and the corrected sentences are on the target side. An example of grammar correction is given in A.7

7.4 Overall model

Figure 7.2 gives the overview of our system. The system takes newspaper headlines in Hindi for translation. Joshua MT system translates the newspaper headlines to English and forwards it to the transliteration module. The transliteration module is called by a python script. First the script detects whether there are any untranslated words. If there are untranslated words, and if they are NE, they are transliterated. Otherwise, they are forwarded to the OOV module. The OOV module which is already trained with a dictionary translated the OOV word using the projection vectors returned by CCA. This output which has no untranslated words is forwarded to the grammar correction module.

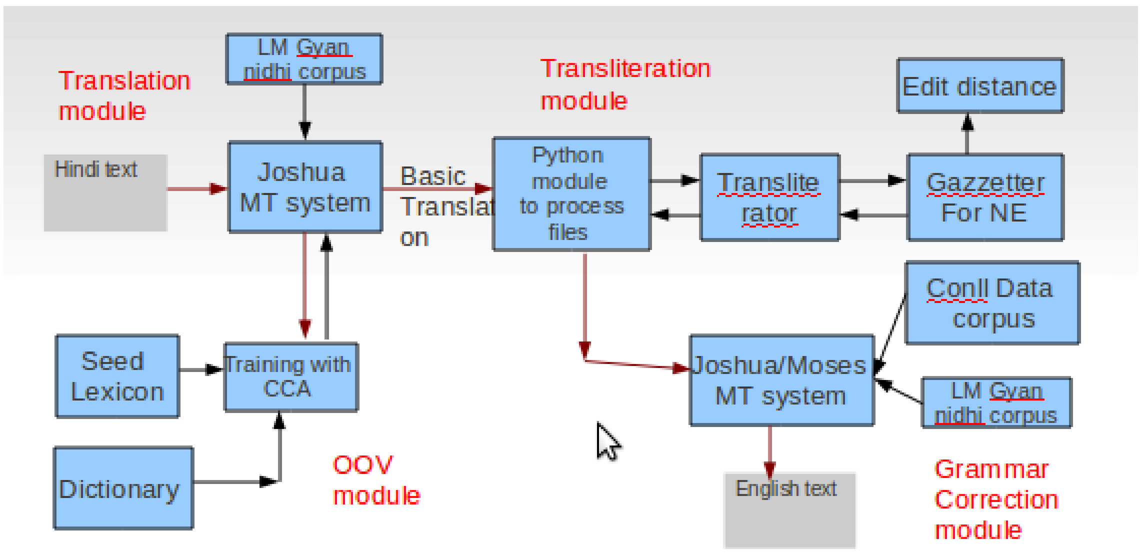


Figure 7.2: Overall model

Chapter 8

Automated Grammar Correction

First we discuss working of our system in section 8.1. In section 8.2, we discuss the various kinds of grammar rules extracted. Then we point out possible application of grammar correction in the final section 8.3.

8.1 Working

In this section, we discuss the intuition why grammar correction can be considered as SMT followed by implementation of the system.

8.1.1 Grammar correction as SMT

Grammar correction can be seen as translation of incorrect to correct sentence. Basically the translation system needs parallel corpus with incorrect and correct sentences. The system starts with alignment to obtain word to word translation probabilities. This procedure distributes translation probability of a single word into multiple words. Higher probability means that the word pairs have been seen more often in corpus together in parallel sentences than any other word. For the word *has* if *have* is given more probability than *had*, chances are that the corpus contains more pairs with correction from has to have.

The second stage is grammar extraction using hiero style of grammar Chiang [2005]. The grammar consists of non-terminal and terminal symbols only. Non-terminals are generalized form of phrases. These rules are in the form of SCFG rules. If the incorrect sentence is *few has arrived* and the correct sentence is *few have arrived*, the grammar rules extracted are :-

$[X]$ ||| few has $[X, 1]$ ||| few have $[X, 1]$ (1)
$[X]$ ||| arrived ||| arrived (2)

The first rule means that *few has* followed by a phrase may be translated to *few have* followed by translation of that phrase. Second rule suggests that any phrase that yields *arrived* can be translated to *arrived*.

After the grammar extraction is done, the left side of the grammar rules is stripped and used to generate the parse tree of the sentence *few has arrived*. Also, there is glue rule to combine two trees or just derive a non terminal.
Here are the left side rules:-

X $\rightarrow$ few has $[X, 1]$ (3)
X $\rightarrow$ arrived (4)
with the glue rules $\rightarrow$ S[1] X | X (5)

The glue rule is used to start the parsing process. It generates a sub-tree for the string *few has* and a non-terminal for *arrived*. Then the right side rules are (1) used to convert *few has* to *few have* as shown in Figure 8.1. While *arrived* remains as *arrived*.

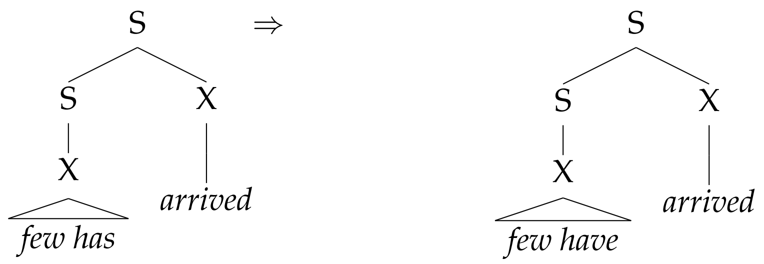


Figure 8.1: Parse tree for transformation from incorrect to correct sentences.

This is the essence of decoding in hierarchical machine translation.

8.1.2 Implementation

The translation system being used is Joshua Machine translation system Li et al. [2010]. We have not made any changes to the system. Various state of the art machine learning algorithms have been implemented in various stages of the translation pipeline. Joshua requires a training data that has a parallel corpus of aligned incorrect and correct sentence to train the system and extract grammar. Similarly it requires a tuning or development set which is a parallel corpus, but much smaller than training data to tune the parameters of the translation model,

[1]Here S means start of the tree

which is a log linear model. Joshua uses ZMERT Zaidan [2009] for tuning, i.e. finding the optimal weights for its seven features as mentioned by Chiang Chiang. The system also requires a testing set for evaluation. In the next section we look at how various grammar corrections have been handled.

8.2 Analysis of grammar rules extracted

Hierarchical models handles all sorts of errors in the same manner. Unlike previous implementation of grammar correction using noisy models, where each correction is handled separately. This is a straightforward advantage which has a single approach towards all errors. The various types of errors encountered are article choice errors, preposition errors, word-form choice errors, word insertion errors as mentioned in Park and Levy [2011]. Apart from these errors we also discuss error due to reordering and error due to unknown verbs which have not been implemented in previous models.

8.2.1 Article choice errors

Article *a* has been replaced by *the* before proper nouns like *a amazon* and *a himalayas*. The grammar rules are:-

$[X]$ ||| a himalayas $[X,1]$ ||| the himalayas $[X,1]$ (6)
$[X]$ ||| a amazon $[X,1]$ ||| the amazon $[X,1]$ (7)

The rules suggest that if *a himalayas* succeeded by a phrase $[X,1]$ can be replaced by *the himalayas* followed by the same phrase.

8.2.2 Preposition errors

Preposition *at* has been replaced by *in* before a place like *at central London*. The grammar rule is:-

$[X]$ ||| $[X,1]$ at central london ||| $[X,1]$ in central london

8.2.3 Unknown Verb correction

Lets say the training data has these sentences

He like milk $\rightarrow$ *He likes milk*
They hate the pollution $\rightarrow$ *They hate pollution*

This system will not be able to correct *He hate milk*, because hate needs to be corrected to hates and grammar has no rule for *hate* $\rightarrow$ *hates*. But has rule for *like* $\rightarrow$ *likes*. From these two rules grammar extractor wont be able to derive *hate* $\rightarrow$ *hates*. This can be solved by splitting *likes* to *like s*

He like milk $\rightarrow$ *He like s milk*

Now extractor will have a rule for this training sentence.

$[X]$ ||| $[X, 1]$ ||| $[X, 1]$ s
$[X]$ ||| hate ||| hate

Using these two rules it generates *hates* from *hate*.

8.2.4 Word insertion errors

As the name suggests these errors are due to missing words. For example:-

The court deemed necessary that she respond to the summons.
The court deemed <u>it</u> necessary that she respond to the summons.

Such a problem has been often encountered in SMT where unknown words are inserted due to language divergence. For this example the grammar rule extracted is :-

$[X]$ ||| $[X, 1]$ deemed $[X, 2]$ ||| $[X, 1]$ deemed it $[X, 2]$

8.2.5 Reordering errors

Reordering errors is somewhat new to the grammar correction fraternity. It can arise when we send the output of translation system to grammar correction. This output may be incorrectly ordered. For example:-

Translation of Hindi sentence:- सेन्त्रल लन्दन मे गिरा हेलिकोप्तर
Correct translation of this sentence is:- *helicopter crash in central London*
Output translation from Hindi-English translation system of this sentence:- *central down in london helicopter.*

If the output translation and correct translation is added to the training corpus of grammar correction system, we can obtain the correct translation for a sentence such as:-

central down in london helicopter $\rightarrow$ *helicopter down in central london*.

These kind of errors cannot be handled by rule based or maximum entropy model. In such scenarios hierarchical SMT based grammar correction models work better.

8.3 Application of grammar correction

One of the applications can be postprocessing like reordering correction as mentioned in 8.2.5.

8.3.1 Grammar correction after translation

Sometimes translation outputs are not grammatically correct. Grammar correction can be used to correct such mistakes in translation output. One of the common mistake is reordering problem that we have already discussed.

This problem can be solved using grammar correction but certainly conditional. We have to train the grammar correction system with the incorrect output from the previous translation system and the correct reference sentences. Here is a case study.

We made some dummy training files taking various combinations to enforce the correct reordering. Table 8.3.1 is a snapshot of the training file.

Incorrect	**Correct**
central down in london helicopter.	helicopter down in central london.
plane down at central london.	plane down <u>in</u> central london.
central in london helicopter fallen.	plane fallen in central london.

Table 8.3.1: Parallel corpus for grammar correction

8.3.2 Steps to grammar correction

First we train Joshua on this parallel corpus and call it grammar correction module. Secondly, any Hindi sentence is translated first on Hi-En [2] translation

[2]Hindi-English

system, then redirected to grammar correction module. Consider the example in A.5. Basically we need to train machine translation system with many disordered examples. We can first translate thousand sentences using a Hi-En translation system. Later we can provide grammar correction with the output of Hi-En system as incorrect corpus and the English sentences provided as reference to the Hi-En translation as the correct corpus.

8.4 Modular representation of entire system

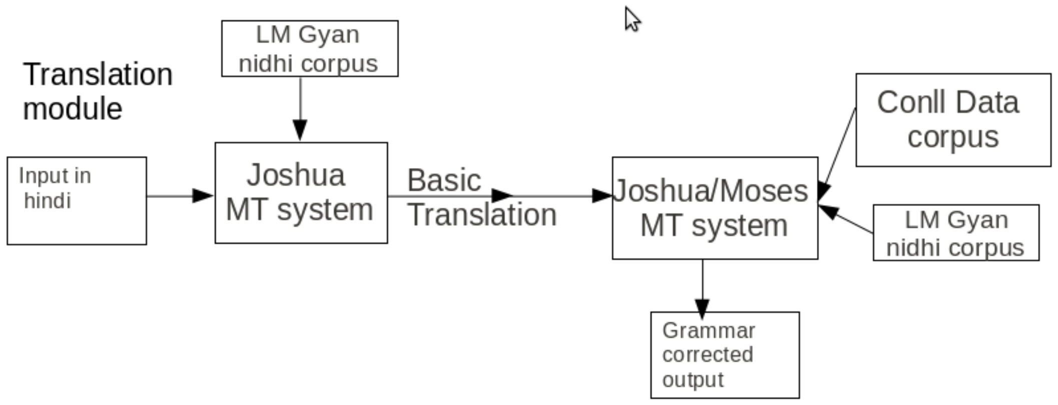


Figure 8.2: SMT system with postprocessing using Grammar correction

Figure 8.2 is a demo version of the entire system. The Translation module that is the Joshua Li et al. [2010] engine is first trained with Gyannidhi corpus which consists of two lacs sentences and is a parallel corpora. The Hindi sentence is fed to this system and the output is directed towards the grammar correction module. The grammar correction module is trained using Joshua system and parallel corpus obtained from a private firm. This parallel corpus is constructed from manually corrected data and hence is highly reliable. The language model is also trained with Gyannidhi corpus to enforce good translations. The output of this system is grammar corrected output. So the overall system is a cascaded system of two MT pipelines.

Chapter 9

Data collection

The team of Joshua conducted their experiments on low resource languages like Indian languages to evaluate the impact of hierarchical model on Indian language to English Machine translation. They obtained the parallel corpora for six Indian languages Hindi, Telugu, Tamil, Malayalam, Bengali and Urdu using crowd-sourcing techniques. Later they released this resource to the machine translation community. The crowd-sourcing techniques used by Joshua can be of immense importance to Indian researchers as well because lot of the Indian languages do not have parallel corpus.

9.1 Crowd-sourcing techniques

Source of document for translation task was the set of top-100 most-viewed documents from each language's wikipedia. These lists were obtained from page view statistics compiled from dammit.lt/wikistats over a one year period. Diverse set of topics including culture, person, places, Internet has been included. The parallel corpora was collected using a 3 step process designed to ensure the integrity of non-professional translations.

1. Building a bilingual dictionary.

2. These dictionaries were used to verify the collection of four different translations for each sentence.

3. As a measure of translation quality, voting was done by an independent set of people not involved in translation to rate the best translation from the four redundant translations.

9.2 Amazon Mechanical Turks Impact on collection of low cost translation

Amazon has a web interface called Mechanical Turk where workers for low cost translation are available. These workers are called Turkers as well. So the idea to create bilingual corpus is to present a set of 10 sentences to each turker. Based on his translations, he is evaluated. Too many incorrect translations mean that the turker is incompetent, so he is denied any more jobs.

Every job or task is called Human Intelligence Task (HIT). Workers were paid 0.7$ for each HIT. To discourage cheating through cut paste, each translation is provided as picture. For each sentence, 4 translations are obtained from different turkers. Evaluation of each worker's performance is done by comparison of translation to monotonic gloss, the percentage of empty translations, amount of time a worker takes to complete a task, geographic location of the worker and cross validation with reference translations obtained from other workers. Cost of translation per worker is much lower compared to professional translations. Germann [2001] says cost of professionally translated task is 0.3$ per word from Tamil to English. The translation obtained using Mechanical Turk was less than 0.01$ per word.

On the contrary low cost translations comes with low quality translations. The variance in quality of translation is higher than the consistently good translations from a professional translator. The problem with turker is they lack formal training, may give insufficient time and attention to the task and it is likely that their desire is to maximize their throughput (thereby their wage).

In the absence of professionally translated data, it is not possible to measure BLEU score of turkers.

9.3 Improve Training data

Additional references are required to increase quality.Translation of more foreign sentences are required to increase coverage. But results show that additional references did not improve quality. In the next section, we discuss issues related to some deficits found in the parallel corpus via crowd-sourcing.

9.4 Orthographic issues

Spelling of the same word may be different due to different realizations, phonetic variations, misspelling. Such discrepancies are found throughout in training and testing data. Translation by non-English speakers bring in such mistakes. Solution for such issues can cause significant improvement in translation.

9.5 Alignment

Different spellings cause incorrect alignment patterns. Because of the small size of the data such inconsistencies play a major role in increasing incorrect alignments. Here are some examples where spelling mistakes in the training corpus can cause incorrect alignment.

- वैधानिक प्रतिरोध {vaidhaanika pratirodha}{constitutional resistance}{constitutional resistance}$\rightarrow$ costitutional resistance
- और अगर दिये गये पीदा को वे स्वेच्छापूर्वक सह लिये {aura agara diye gaye piidaa ko ve svechaapuurvak saha liye}{and if given torture they willingly endure}{if they endure the torturing willingly}$\rightarrow$ if they *emdure* the torturing willingly
- उस्के अ त्याचारोन की घोसना {uske atyaacaarona kii ghosanaa}{his misdeeds of announcement}{announcement of his misdeeds}$\rightarrow$ annnouncement of his misdeeds
- ईस्वर का भय रखने वाले म्रु त्यु के आतन्क से नहि दरते | {iisvara kaa bhaya rakhane vaale mrutyu ke aatanka se nahi darate}{God of respect have death of terror not fear}{Those who resepect God, will not be afraid from terror of death$\rightarrow$ Those who resepect God, will not be afraid from terror of death.

9.5.1 Some observations obtained from the test on the parallel corpus

Here are the conclusion obtained by Post et al. [2012] on the Indian language parallel corpora on Joshua MT system. Berkeley aligner produces more reasonable looking alignments rather than the Moses heuristics. Mechanical Turk can be used for obtaining quality parallel corpus at a significantly lower cost. Increasing the number of reference translations does not improve the quality of translation. Mturk worker pool has worker capable of translating a number of low resource languages. Crowd source techniques with appropriate quality controls could be used to produce professional level translations.

With regards to IELMT, crowd-sourcing can be done cheaply without affecting quality if we follow the above techniques for quality control. The results shown by translation obtained using corpora obtained from crowd-sourcing gives a motivation to the Indian researcher that there should be effort in obtaining large quantity of parallel corpora.

Chapter 10

Experiments

This chapter is divided into four parts. Section 10.1 discusses experiments with translation system. In our experiments, we have used Joshua version 4, which is available for downloading at joshua 4.0 and Moses 1.0 also available at Moses 1.0. Section 10.2 discuss our experiments with OOV words and transliteration followed by grammar correction in the section 10.3.

10.1 Hi-en translation and evaluation

This section discusses our experiments on hierarchical and phrase based MT system. We start with dataset, evaluation technique and then the various settings we have used for evaluating our data.

10.1.1 Corpora

Indian Parallel Corpora was produced as a result of crowd-sourcing by Joshua group and has been made public through Post et al. [2012]. The speciality of this corpus is that it has been automatically generated by crowd through Amazon's Mechanical Turk. The whole concept of this project is to hire freelancers and obtain cheap translation. The project turned out to be a huge success as shown by the results in table 11.1.1.

Gyan Nidhi Corpus consists of 2,27,123 sentences, 1,95,165 words in English, 2,02,199 words in Hindi. Times Of India archives Monolingual English Corpus contains Times of India Headlines with 1,81,659 sentences. This corpus has been mined from the web automatically. The tuning corpus consists of about 1000 sentences from Gyan Nidhi Corpus while the testing corpus is also 1000 sentences from Gyan Nidhi corpus. The Gyan Nidhi corpus lacks in phrases and idioms.

Metric	System A	System B
Precision (1 gram)	3/6	6/6
Precision (2 gram)	1/5	4/5
Precision (3 gram)	0/4	2/4
Precision (4 gram)	0/3	1/3
Brevity penalty	6/7	6/7
BLEU	0%	52%

Table 10.1.1: Demonstration of fallacies of BLEU score

10.1.2 BLEU as evaluation method

We have used BLEU as our evaluation metric. BLEU considers n-gram overlap between machine translation output and reference translation. Then it computes precision for n-grams of size 1 to 4. It adds brevity penalty (for too short translations). The above technique has been formulated as in equation 10.1.1.

$$BLEU = min(1, \frac{output - length}{reference - length})(\prod_{i=1}^{4} precision_i)^{\frac{1}{4}} \tag{10.1.1}$$

A.8 shows an example that finds the bleu score for the below translation. System A gives a translation that is close to the reference, but System B gives a bad translation. But evaluation technique gives the second example better score. To improve the BLEU evaluation technique, it is required to have multiple reference translation as shown in A.9

Multiple Reference Translations

To account for variability, multiple reference translations should be used. The advantages of having multiple references are two fold:-

- n-grams may match in any of the references
- closest reference length used - brevity penalty is minimised

10.1.3 Translation models

We experimented with the following setup:-

Phrase Based Translation

We train Moses machine translation system on Gyan nidhi corpus and tested on Indian parallel corpus. This is the simplest machine translation model and is used as a benchmark to compare hierarchical MT system with phrase based MT system.

Factor Based Translation Model

We also train moses on factored Gyan Nidhi Corpus as an secondary effect to observe the change in translation quality after introducing factors. The setup used for factor based translation model makes use of various factors at the word level. Factors such as word, part of speech and stem are used to increase morphological information. Both source and target sides are populated with these factors. Various tools are used to form the factored corpus. These tools include Hindi POS tagger from CFILT, IIT Bombay, Hindi Rule-Based Stemmer from Lucene in the source side and Stanford POS Tagger, English Rule-Based Stemmer from Lucene in the target side.

The factored based machine translation system requires input as well as output factors in a particular format such as stem to stem mapping. This means stem of Hindi word is mapped to stem of English word. Stem to word + part of speech mapping means that stem of Hindi word is aligned to word and part of speech of English word. These are called translation factors. Similarly there are generation factors such as stem in Hindi side can generate part of speech in English side. Part of speech and stem of Hindi word generate word in English side. These factors are concisely presented in A.10

Hierarchical Phrase Based Translation

Our hierarchical MT system is trained with Joshua engine on Gyan nidhi corpus and tested with Indian parallel corpus. This system is our baseline system for comparison of hierarchical models with other models that include postprocessing units. Then we augment Joshua MT system with postprocessing like OOV handling and transliteration. We also add the grammar correction module to the output of postprocessing unit. The pipeline consists of hierarchical phrase based MT system, postprocessing unit and grammar correction system.

10.2 OOV translation

Calculating W$_x$ (projection vector) for Hindi feature vector and W$_y$ (projection vector) for English feature vector for each word pair is very time consuming. So, to simplify and make it fast, we first obtain a small translation dictionary of frequent words from the corpus. Training with this dictionary gives a rough estimate of W$_x$ and W$_y$.

Phrase table looks like Table 10.2.1:-

Hindi word	English word	lexical probability
ऊन्चाई	levels	0.0416667
ऊन्चाई	height	0.1333333
आवस्यकता	necessity	0.2307692
आवस्यकता	needs	0.1111111
आवस्यकता	need	0.1111111

Table 10.2.1: Lexical probability for Hindi to English translation

10.2.1 Training file

Converted the phrase table to a dictionary but most of the words are inappropriate:-

- निस्पन्द {nispanda } heightening
- अल्पजीवि {alpajiivi } witted
- चाल {caala } unbecoming
- वाहकपोत {vaahakapota} cruising

As we can see the resulting translation are not appropriate because, only Hindi to English lexicon are used. For obtaining dictionary we converted the Hindi to English lexicon obtained by Berkeley Aligner into word pairs by choosing the most probable translation.

10.3 Grammar Correction

Now we present the data set and evaluation techniques for our experiment with regards to grammar correction .

10.3.1 Data set

We took a small training file of 100 odd examples and ran Joshua on the training files. Tuning and testing is also done on the same set of files. Tuning basically finds the best weights for the log-linear model required to score the translation. Generally tuning and testing is done on a set different from training set. It generated a BLEU (Papineni et al. [2002]) score of 0.9874 when tested on the training data. BLEU score is used for evaluation in translation because it has been shown to have better correlation with human evaluators when compared to other evaluation metric. Later we ran grammar correction system on much larger data set to confirm the validity of result and scalability of system . We varied the training data from 1000 lines to 3000 lines keeping the testing and tuning set constant at 1000 lines.

Cleaning training corpus

This is a preprocessing step before training grammar correction system on the corpus. We have used CoNLL (Computational Natural Language Learning) corpus for training Joshua MT system. Without preprocessing the system gave a Bleu score of 96.92% on training data. This is primarily due to the presence of noisy data as pointed in A.4

For cleaning the data we divided the 50000 line CONLL corpus into three parts to divide work. First, we removed unchanged sentence pairs using a python script and brackets with a sed command. Remaining nuances are removed using meld. Meld is a python interface for diff that finds differences in parallel files. It took us about six hours to process the entire data and make it noise free. At the end we had about 0000 lines of clean data. The Bleu score of the Grammar Correction module improved to 98.12 %.

Table 10.3.1 explains the statistics obtained from the word alignment of unclean and clean data.

Direction of alignment	**Unclean corpus**	**Clean corpus**
Incorrect-Correct	629757.31	310814.04
Correct-Correct	489839.4	254840.081

Table 10.3.1: This figure gives an idea of how drastically the entropy of alignment falls after cleaning the corpus. The unclean corpus has a much higher negative log likelihood compared to clean corpus in both Incorrect to Correct and Correct to Incorrect alignment.

The Alignment algorithm runs for five iteration first with IBM model 1 and HMM in both direction, i.e. Correct to Incorrect and Incorrect to Correct to produce the following result. As we can see negative log likelihood is more for unclean corpus than clean corpus. This shows that clean corpus has better estimate of parameters.

Here are some errors found during testing of MT system trained with unclean data. Such errors didn't appear in clean data due to absence of brackets or citations which are not required for grammar correction.

Examples:-

1. Input:- $-lrb-$[1] mootee, 2009 $-rrb-$[2] due to these reasons
   Ref:- $-lrb-$ mootee, 2009 $-rrb-$ $\underline{.}$ due to these reasons
   Output:- $-lrb-$ mootee, 2009 $-rrb-$ due to these reasons
   Error is due to absence of . after $-rrb-$

2. Input:- Commercialization, diffusion and consequences ".
   Ref:- commercialization, diffusion and consequences $-lrb-$ p.? $-rrb-$.
   Output:- commercialization, diffusion and consequences $-lrb-$ citation $-rrb-$.
   Error is due to presence of citation instead of p.? between -lrb- and -rrb-

3. Input:- commercial aviation $-lrb-$ no year mentioned $-rrb-$
   Ref:- commercial aviation $-lrb-$ no year mentioned $-rrb-$
   Output:-commercial aviation $-lrb-$ no year mentioned .
   Error is due to presence of . instead of -rrb- after mentioned

The reason is that "mentioned ." (log P = -0.772518) comes more often in corpus than "mentioned -rrb-" which is not found at all. Certainly alignment plays a role here, but the key is that we should not allow noise to reduce accuracy in results. These errors can be removed because they have no role in grammar correction. This reduces data size and also time required to train the system.

[1]Generated by MT system for left curly bracket
[2]Generated by MT system for right curly bracket

Chapter 11

Results

This chapter discusses results on the three experiments *i.e.,* hierarchical phrase based MT system in section 11.1, OOV in section 11.2 and grammar correction in 11.3.

11.1 Hierarchical phrase based MT system

In the experiments conducted, we find that Joshua performs reasonably well as was claimed by the Joshua group as shown in table 11.1.1. We conduct translations from Hindi, Bengali, Tamil, Malayalam. The translation results are good but they are far away from human translations. Here is a sample translation in A.11.

Some of the observations:-

- Alignment issue
  Correct phrase:- *india had got success in space science.*
- Sparsity
  No translation for *पायलत* {paayalata} {pilot} and *वायूसेना* {vaayuusena} {airforce}.

Language pair	**BLEU 4 score**		
	Joshua (me)	**Joshua (team)**	**Google**
Hindi-English	16.31	17.29	25.21
Bengali-English	12.6	13.53	20.01
Tamil-English	8.02	9.81	13.51
Malayalam-English	7.1	13.72	NA

Table 11.1.1: Experiment on Joshua with various language pairs

- Lacks the awareness of a factor based model :-
  यात्री {yaatrii} {traveler} is singular but travelers is plural.

Improvements required
Increase corpora by crowdsourcing to account for sparsity in data. Better aligner than existing aligners like GIZA++ are required. Linguistic information should be included in the translation pipeline. Translation of out of vocabulary (OOV) words should be considered.

11.1.1 Evaluation of our system

So we compared all the models that we tested on various models like phrase based models, hierarchical models and factor based models.

Experiment	**Bleu score**
Phrase based MT using Moses	10.21
Factor based MT using Moses on Gyan Nidhi corpus	8.48
Hierarchical system on Joshua Corpus	14.24
Hierarchical system on Gyan Nidhi Corpus	15.96
Hierarchical system on Gyan Nidhi Corpus + OOV words feedback	16.47
Hierarchical system on Gyan Nidhi Corpus + OOV words feedback + Transliteration	16.96
Hierarchical system on Gyan Nidhi Corpus + OOV words feedback + Transliteration + Grammar correction	17.01

Table 11.1.2: Experiment with the OOV words, transliteration and Grammar correction.

Table 11.1.2 reveals that highest BLEU score is given by the Hierarchical Phrase based system (Joshua) followed by phrase-based and factor-based systems. Handling OOV words gives a significant rise in BLEU scores almost by a margin of 1 %.

11.2 OOV translation

#Word	#Length of feature vector	Context Vector Generation time	CCA
500	500	93 secs	8 secs
1000	500	105 secs	10 secs
2000	500	109 secs	10.3 secs
5000	2000	150 secs	900 secs

Table 11.2.1: Feature generation time and CCA Training time for varying word size, length of feature vector

Based on the result, we decide to have a dictionary of 1000 words and feature vector of size 500 to reduce the training time. Table 11.2.1 demonstrates the variation of CCA training time and context vector generation time with variation in size of word list and length of feature vector. Training time is dominated by CCA training. We need to implement a faster CCA training algorithm to reduce training time.

Techniques	Time in hours
With all possible pairs	24
With dictionary pairs	1

Table 11.2.2: OOV training time

Table 11.2.2 shows that using dictionary pairs reduces training time. The training process should be initiated with a lexicon.

11.3 Grammar correction

For baseline, we choose to evaluate directly on the test set and compare it on a training set of 1000 and 10000 sentences as shown in Fig 11.3.1.

Experiment	**Size of training corpus (sentences)**	**Size of tuning corpus (sentences)**	**Size of testing corpus (sentences)**	**BLEU score**	**increment**
Baseline	–	–	150	79.98	–
Dataset1	1000	150	150	80.07	0.09
Baseline	–	–	1000	79.95	–
Dataset2	10000	1000	1000	81.32	0.57

Table 11.3.1: Effect on BLEU score by using grammar correction system over baseline.

Tuning and testing corpus consist of 1000 sentences each. These experiments s Throughout the experiments we keep the same set of testing and tuning corpus. The results have been presented in table 11.3.2. The results show improvement in BLEU score with increase in size of training corpus. Some examples from the result have been given in Table 11.3.3.

Size of training corpus (sentences)	**Size of tuning corpus (sentences)**	**Size of testing corpus (sentences)**	**BLEU score**
1000	1000	1000	0.7668
2000	1000	1000	0.7679
3000	1000	1000	0.7744

Table 11.3.2: Effect on BLEU score by varying size of training corpus

In the next chapter, we look at the IELMT scenario and the possibilities for success of hierarchical MT on Indian language to English translation.

Incorrect	**Correct**
bar code is an important technology	<u>The</u> bar code is an important technology.
besides globalization, the various kinds of users shape it to an easily **implemented** technology	besides globalization, the various kinds of users shape it to an easily **used** technology.
collecting enough information before finding solutions will reward **you** with great **bonus**.	collecting enough information before finding solutions will reward **us** with great **benefits**.
till now, laptop is still a necessary tool in people' s daily life	**until** now, <u>the</u> laptop is still a necessary tool in people' s daily life
he finally invented the laptop that we use **every day now**	he finally invented the laptop that we **currently** use

Table 11.3.3: Some corrected examples from grammar correction

Chapter 12

Impact of hierarchical based MT system on IELMT

Phrase based MT gives low quality translation. Syntax based MT relies on annotated parallel corpora. In the Indian scenario, annotated corpora is not available in huge quantity. So there is sparsity of data. In the absence of corpora, syntax based MT annotates parallel sentences in Indian languages using annotations in English counterpart resulting in noisy annotations. In this scenario, hierarchical phrase-based MT comes into rescue because it does not require annotated corpora. Hence it can translate from any language be it Hindi, Gujarati, Marathi, Telugu, Tamil, etc. Therefore hierarchical phrase-based MT will have an impact on IELMT. Also the presence of open source systems like Joshua will have a direct impact on increasing research in this field. If research increases, so will the quality of translation. Lets look at the time taken by a translation and whether it is implementable for on line or real time translations.

12.1 Real-time implementation

Joshua uses parallel processing like grammar extraction and decoding in various stages in its pipeline. Yet, when it comes to translating in real time, it is slow compared to online translators like Google. Currently Joshua is giving a Hindi to English translation in about 1 minute. This is not fast enough to be implemented on a site giving instant translation from Indian languages. As far as quality is concerned people might find online translation useful for comparing with Google results.

A sample sentence on our system takes about 1 minute to translate. Figure 12.1 shows the distribution of time in various stages of the pipeline.

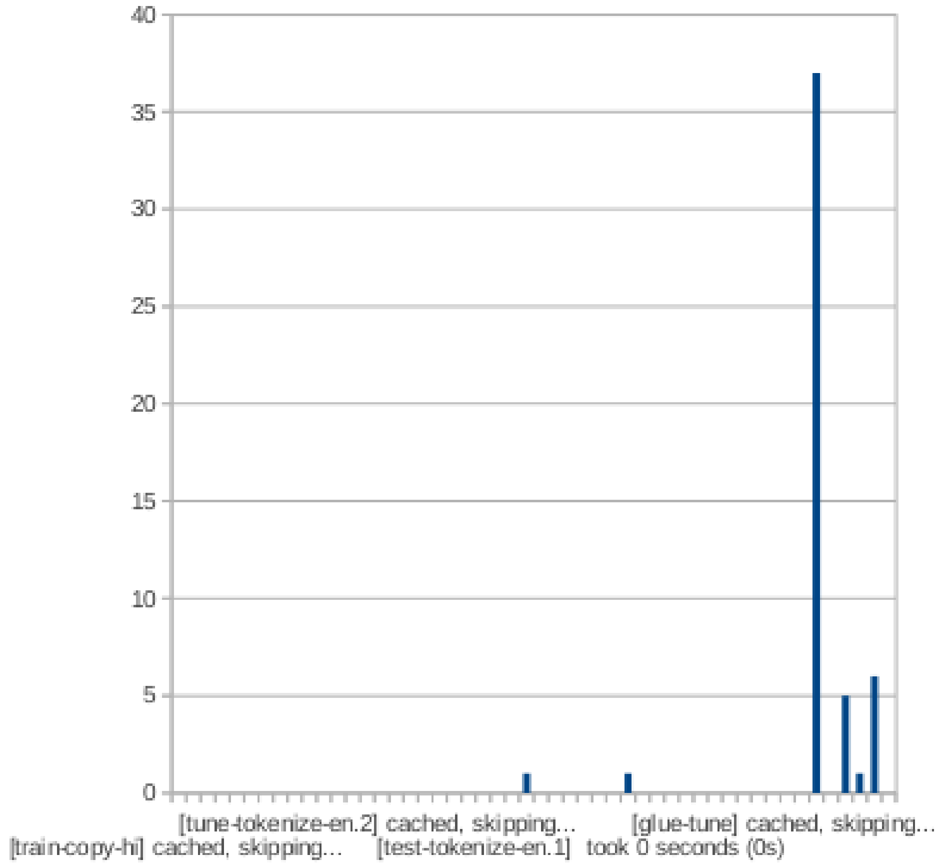


Figure 12.1: Distribution of time(s) among various stages.

Maximum time is taken for filtering the grammar. If this stage reduces translation time, Joshua could really be implemented as an online translation system and considering Indian scenario that would be useful to a lot of people.

Although the translation quality is not good, it can be improved by increasing the size of parallel corpora. We think that sparsity and alignment issues can be handled using a sufficiently large corpora but this may increase the translation time due to increase in size of grammar. The next chapter gives the conclusion of our work.

Chapter 13

Conclusion

In this chapter we discuss conclusions derived from experiments. Section 13.1 talks about results related to translation, section 13.2 talks about transliteration, section 13.3 talks about OOV and section 13.4 discusses grammar correction related conclusions. In this section, we also show that we have answered the question we initially raised in section 1.4.

13.1 Hierarchical phrase based MT

Hierarchical phrase-based MT demonstrates that language is inherently hierarchical. So there is no harm in using this technique for MT. Another observation is that a simple construct like synchronous CFG greatly reduces the complexity in deriving the parse tree. Moreover hierarchical model overcomes the difficulties faced by phrase-based models and practically it has been proved that hierarchical model outperforms phrase-based models.

Another beautiful aspect of this approach is that it is somewhat an amalgamation of rule-based and statistical approach. This gives us an intuition that MT might be a mixture of those two broad classifications i.e. rule-based and statistical machine translation. Syntax based model outperform phrase based model but they require large annotated corpora which are not available for Indian languages. On the other hand, hierarchical phrase-based MT will be useful for IELMT because it does not require annotated parallel corpora. So hierarchical phrase based system can act as an alternative to existing translation models for Hindi to English translation.

Most of the previous hierarchical phrase based systems were not available as open source. Nowadays the scenario has changed due to the arrival of open source systems like Joshua and Moses. These systems motivate researcher to pursue research in this branch of MT leading to more experiments and hence improvement in quality of MT. Algorithms like k-best translation extraction, pruning techniques and MERT have set a benchmark in the standard of algorithms MT demands.

Another thing worth mentioning is about obtaining low cost and quality parallel corpora using crowd sourcing techniques. The simplicity is in the idea that using a third party agent like Mechanical Turk, where one can hire people to translate some set of useful sentences is sensational in itself. This method reduces translation cost compared to translation obtained from professionals, of-course the quality of translation is not as good as professional translators. But with proper quality control, even crowd can contribute to a good quality parallel corpora.

13.2 Transliteration

Supervised model based on Edit distance is easy to implement but isn't accurate and is heuristic based. Right now our baseline is Edit distance supervised model. The threshold value for NER is 0.666 which means if $\frac{2}{3}^{rd}$ char match with words in gazetteer list, we consider word as NER. Consider the example, *Rajesh* $\sim$ *Romesh*. If Rajesh is in the corpus and we want transliteration for Romesh, we assume Romesh to be NER by edit distance algorithm and this gives a correct output. This threshold is language dependent and can vary for other language pairs. We believe this threshold can be a clue to easy NER detection.

13.3 OOV

Lexicons in both hi-en and en-hi should be used to prepare the training file for CCA. We observed that both training and tesing CCA is very slow. So, it needs sparse matrix algorithms like KCCAHardoon et al. [2004]. Also testing of an OOV word is cumbersome. Firstly, we need a fairly big corpus so that we get features for the OOV word. Secondly, we need to convert the feature vector of the OOV word to English. We were accomplishing this conversion by looking for the nearest word in the dictionary from the OOV word in the canonical space, but that is expensive to compute.

Results have shown that post-editing techniques like OOV, transliteration and grammar correction improved the BLEU score from 15.96 to 17.01. This increase in Bleu score suggests that incorporating post-editing techniques remarkably improves translation quality. But the concern is feasibility of real time translation using post-editing techniques. We find that including post-editing techniques require an overhead of 2-3 minutes for translation. This is bearable considering the minumum usage of resources in our system.

13.4 Grammar correction

As far as grammar correction is concerned, increasing training data definitely increases accuracy because patterns in grammar correction keep repeating even if test data is completely different from training set.The high accuracy is because we are translating in the same language domain. Moreover, the translation only requires small changes, so the bleu score is never low even if there is no correction. Cleaning the corpus is essential to improve accuracy as shown by analysis output of translation obtained from clean and unclean corpus.

Considering the grammar correction problem as a translation problem gives a new perspective to solving this long standing problem. By using hierarchical phrase based machine translation models, we try to mimic the human based approach of grammar correction.

13.5 Future Work

We have divided the future work into three classes - short, long and future on the basis of time required to complete them

13.5.1 Short Term

Out of vocabulary approach is slower for real time implementation. A faster approach like sparse CCA is required to improve the response time of a OOV module.

13.5.2 Medium Term

In other pending work, transliteration needs a much larger gazetteer list. A list of abbreviations and their translations are required because abbreviations do not follow the rules used by a transliterator to convert Hindi to English. Grammar correction using hierarchical models have resolved problems faced by rule based methods but they still require translation of out of vocabulary words. In this thesis, we have handled OOV translation for verbs, but grammar correction requires OOV translation for all words including verbs.

13.5.3 Long Term

Hierarchical models still suffer from lack of knowledge regarding morphology of the Hindi word. Factor based models take care of morphology by incorporating factors such as stem, number, gender. The improvement in translation can be expected if we incorporate factors in hierarchical models.

References

Borthwick, A. E. (1999). *A maximum entropy approach to named entity recognition*. PhD thesis, New York, NY, USA. AAI9945252.

Brockett, C., Dolan, W. B., and Gamon, M. (2006). Correcting esl errors using phrasal smt techniques. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, ACL-44, pages 249–256, Stroudsburg, PA, USA. Association for Computational Linguistics.

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

Callison-Burch, C., Bannard, C., and Schroeder, J. (2005). Scaling phrase-based statistical machine translation to larger corpora and longer phrases. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, ACL '05, pages 255–262, Stroudsburg, PA, USA. Association for Computational Linguistics.

Chiang, D. (2005). A hierarchical phrase-based model for statistical machine translation. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, ACL '05, pages 263–270, Stroudsburg, PA, USA. Association for Computational Linguistics.

Chiang, D. (2007). Hierarchical phrase-based translation. *Comput. Linguist.*, 33(2):201–228.

Chodorow, M. and Leacock, C. (2000). An unsupervised method for detecting grammatical errors. In *Proceedings of the 1st North American chapter of the Association for Computational Linguistics conference*, NAACL 2000, pages 140–147, Stroudsburg, PA, USA. Association for Computational Linguistics.

Galley, M., Graehl, J., Knight, K., Marcu, D., DeNeefe, S., Wang, W., and Thayer, I. (2006). Scalable inference and training of context-rich syntactic translation models. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, ACL-44, pages 961–968, Stroudsburg, PA, USA. Association for Computational Linguistics.

Germann, U. (2001). Building a statistical machine translation system from scratch: how much bang for the buck can we expect? In *Proceedings of the workshop on Data-driven methods in machine translation - Volume 14*, DMMT '01, pages 1–8, Stroudsburg, PA, USA. Association for Computational Linguistics.

Haghighi, A., Liang, P., Berg-Kirkpatrick, T., and Klein, D. (2008). Learning bilingual lexicons from monolingual corpora. In *Proceedings of ACL-08: HLT*, pages 771–779, Columbus, Ohio. Association for Computational Linguistics.

Hardoon, D. R., Szedmak, S. R., and Shawe-taylor, J. R. (2004). Canonical correlation analysis: An overview with application to learning methods. *Neural Comput.*, 16(12):2639–2664.

Hermet, M. and Désilets, A. (2009). Using first and second language models to correct preposition errors in second language authoring. In *Proceedings of the Fourth Workshop on Innovative Use of NLP for Building Educational Applications*, EdAppsNLP '09, pages 64–72, Stroudsburg, PA, USA. Association for Computational Linguistics.

Huang, L. and Chiang, D. (2005). Better k-best parsing. In *Proceedings of the Ninth International Workshop on Parsing Technology*, Parsing '05, pages 53–64, Stroudsburg, PA, USA. Association for Computational Linguistics.

Izumi, E., Uchimoto, K., Saiga, T., Supnithi, T., and Isahara, H. (2003). Automatic error detection in the japanese learners' english spoken data. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics - Volume 2*, ACL '03, pages 145–148, Stroudsburg, PA, USA. Association for Computational Linguistics.

Koehn, P., Hoang, H., Birch, A., Callison-Burch, C., Federico, M., Bertoldi, N., Cowan, B., Shen, W., Moran, C., Zens, R., Dyer, C., Bojar, O., Constantin, A., and Herbst, E. (2007). Moses: open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and*

Demonstration Sessions, ACL '07, pages 177–180, Stroudsburg, PA, USA. Association for Computational Linguistics.

Koehn, P., Och, F. J., and Marcu, D. (2003). 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*, NAACL '03, pages 48–54, Stroudsburg, PA, USA. Association for Computational Linguistics.

Li, Z., Callison-Burch, C., Dyer, C., Ganitkevitch, J., Irvine, A., Khudanpur, S., Schwartz, L., Thornton, W. N. G., Wang, Z., Weese, J., and Zaidan, O. F. (2010). Joshua 2.0: a toolkit for parsing-based machine translation with syntax, semirings, discriminative training and other goodies. In *Proceedings of the Joint Fifth Workshop on Statistical Machine Translation and MetricsMATR*, WMT '10, pages 133–137, Stroudsburg, PA, USA. Association for Computational Linguistics.

Liu, Y., Liu, Q., and Lin, S. (2006). Tree-to-string alignment template for statistical machine translation. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, ACL-44, pages 609–616, Stroudsburg, PA, USA. Association for Computational Linguistics.

Nagao, M. (1984). A framework of a mechanical translation between japanese and english by analogy principle. In *Proc. of the international NATO symposium on Artificial and human intelligence*, pages 173–180, New York, NY, USA. Elsevier North-Holland, Inc.

Och, F. J. (2003). Minimum error rate training in statistical machine translation. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics - Volume 1*, ACL '03, pages 160–167, Stroudsburg, PA, USA. Association for Computational Linguistics.

Olteanu, M., Davis, C., Volosen, I., and Moldovan, D. (2006). Phramer: an open source statistical phrase-based translator. In *Proceedings of the Workshop on Statistical Machine Translation*, StatMT '06, pages 146–149, Stroudsburg, PA, USA. Association for Computational Linguistics.

Papineni, K., Roukos, S., Ward, T., and Zhu, W.-J. (2002). 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. Association for Computational Linguistics.

Park, Y. A. and Levy, R. (2011). Automated whole sentence grammar correction using a noisy channel model. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, HLT '11, pages 934–944, Stroudsburg, PA, USA. Association for Computational Linguistics.

Post, M., Callison-Burch, C., and Osborne, M. (2012). Constructing parallel corpora for six indian languages via crowdsourcing. In *Proceedings of the Seventh Workshop on Statistical Machine Translation*, WMT '12, pages 401–409, Stroudsburg, PA, USA. Association for Computational Linguistics.

Quirk, C., Menezes, A., and Cherry, C. (2005). Dependency treelet translation: syntactically informed phrasal smt. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, ACL '05, pages 271–279, Stroudsburg, PA, USA. Association for Computational Linguistics.

Tetreault, J. R. and Chodorow, M. (2008). The ups and downs of preposition error detection in esl writing. In *Proceedings of the 22nd International Conference on Computational Linguistics - Volume 1*, COLING '08, pages 865–872, Stroudsburg, PA, USA. Association for Computational Linguistics.

Venugopal, A. (2005). Considerations in maximum mutual information and minimum classification error training for statistical machine translation. In *In the Proceedings of EAMT-05*, page 3031.

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

Appendices

Appendix A

A.1 Phrase based translation of a Hindi sentence to English sentence

भारत का प्रधान मन्त्री {bhaarata kaa pradhaana mantrii} {India of Prime Minister}	→ Prime Minister of India
जापान का प्रधान मन्त्री {jaapaana kaa pradhaana mantrii} {Japan of Prime Minister}	→ Prime Minister of Japan
चीन का प्रधान मन्त्री {ciina kaa pradhaana mantrii} {China of Prime Minister}	→ Finance Minister of China
भारत का राष्ट्रीय पक्षी {bhaarata kaa raastriiya pakshii} {India of National bird}	→ National bird of India

A.2 Example to establish reordering

For example :- (This mapping was observed during training)

भारत का प्रधान मन्त्री → Prime Minister of India

If a similar phrase appear during testing,

भारत का राष्ट्रीय पक्षी

Even if it had the translations of words in the above phrase,

भारत → India
का → of
राष्ट्रीय पक्षी → National bird

This will give an incorrect output like:-

भारत का राष्ट्रीय पक्षी → India of National bird

A.3 Websites for Gazetteer list

- babynames
- Surname
- http://en.wiktionary.org/wiki/Appendix:Indian_surnames_(Arora) (Bunt) (Chitpavan) (Deshastha_Brahmin) (Goan_Christian) (Paravar) (Shivalli)Indian surnames
- http://www.indiacom.com/yellowpage/telephonedirectories.aspTelephone directory

A.4 Examples of noisy data in CoNll corpus

1. HYPERLINK-`http://en.wikipedia.org/wiki/`
2. Bracketed information:- (DoD) {Common Access Card}
3. Citations:- (Ben, 2008)
4. Presence of sentence pairs without any changes.

   Ex:-*Our current population is 6 billion people and it is still growing exponentially* → *Our current population is 6 billion people and it is still growing exponentially.*

A.5 Grammar correction example

Input to Hi-En translation system is:-	*सेन्ट्रल लंडन में गिरा प्लेन*
Expected output is:-	*plane down in central london.*
Output from Hi-En translation system is:-	*central down in london plane.*
Input to Grammar correction is:-	*central down in london plane.*
The output is:-	*plane down in central london.*

A.6 Example of feature vector

brave

- Context Feature List for brave. The feature list is a tuple of ⟨ word, frequency ⟩ where the first word represents words present in the neighbourhood of brave and the second digit represents frequency of that word in the neighbourhood of brave over the entire corpus.

  Kashi 1, arjuna 1, by 2, who 3, of 5, are 1, died 1, karna 1 Akbar 1, son 1, he 1, like 1, much 1, and 2, warrior 1 . . . . . .

- Feature vector: Vector of the frequency of words present in the neighbourhood of brave. If there are 1000 words in the vocab list, then the feature vector generated will be of length 1000. If a word is absent, it is marked as 0. Otherwise the frequency of the word is saved in that position.

  0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
  0 3 . . . . . . 0 0 0 0 0 1 0 0 3 0 0 0 0 0 0 0 2 0

A.7 Example from grammar correction

- Text in the incorrect corpus:
  - central down in london helicopter.
  - plane down at central london.
- Text in the corrected corpus:
  - helicopter down in central london.
  - plane down in central london.
- Input:
  - central down in london plane.
- Output:
  - plane down in central london.

A.8 Single reference translation

System A: *Israeli officials responsibility of airport safety*
Reference: *Israeli officials are responsible for airport security*
SYSTEM B: *airport security* <u>*Israeli officials are responsible*</u>

A.9 Multiple Reference Translations

System:-
Israeli officials responsibility of airport safety

References:-

Israeli officials are responsible for airport security
Israel is in charge of the security at this airport
The security work for this airport is the responsibility of the Israel government
Israeli side was in charge of the security of this airport

A.10 Translation models

We experimented with the following setup:-

- Phrase Based Translation
  - Moses trained on Indian Parallel Corpora
  - Moses trained on Gyan Nidhi Corpus
- Hierarchical Phrase Based Translation
  - Joshua trained on Indian Parallel Corpora
  - Joshua trained on Gyan Nidhi Corpus
- Factor Based Translation Model
  - Moses Trained on Factored Gyan Nidhi Corpus

A.10.1 Factor-Based Translation Model

The setup used for factor based translation model is as follows:-

- Factors used:
  - word + POS tags + stem
- Tools used:
  - Hindi Side:
    - Hindi POS tagger from CFILT, IIT Bombay
    - Hindi Rule-Based Stemmer from Lucene
  - English Side:
    - Stanford POS Tagger
    - English Rule-Based Stemmer from Lucene

The configuration for factors has been done as follows:-

- Input Factors (Hindi side): word — POS — stem
- Output Factors (English side): word — POS
- Translation Factors:
  - Stem - stem
  - Stem - word, POS
  - Word - word, POS
- Generation Factors:
  - Stem POS
  - POS, stem word

A.11 Hindi-english translation

Hindi - *अंतरिक्ष में प्रथम भारतीय अप्रैल १९८४ में भारत ने अंतरिक्ष विज्ञान के क्षेत्र में एक और सफ़लता प्राप्त की जब पेहला भारतीय अंतरिक्ष यात्री राकेश शर्मा जो भारतीय वायूसेना के एक पाईलट थे अंतरिक्ष पहुँचे*
वैधानिक प्रतिरोध

Transliteration - *Antariksha mein pratham bharatiya: aprail 1984 mein bharat ne antariksha vigyan ke kshetra mein ek aur safalta prapta kee jab pehla bharatiya antariksha yatri Rakesh Sharma jo bharatiya vayusena ke ek paylat the antariksha pahunche.*

Gloss - *Space in first Indian: April 1984 in India had space science in field in one more success got did when first Indian space traveller Rakesh Sharma who Indian airforce of a pilot was space reached.*

English - *first indian: april in space in 1984 india had a in space science and got success when the first indian space travelers rakesh sharma which was a* पाईलट *of indian* वायूसेना *operation but the space.*