Tribhuvan University
Institute of Science and Technology

A Chunk Alignment Model for Statistical Machine Translation on English-Nepali Parallel Corpus

Dissertation

Submitted to

Central Department of Computer Science and Information Technology,
Tribhuvan University, Kirtipur, Kathmandu, Nepal

In partial fulfillment of the requirements
for the Master's Degree in Computer Science and Information Technology

by

Yoga Raj Joshi

May, 2010

Tribhuvan University

Institute of Science and Technology

A Chunk Alignment Model for Statistical Machine Translation On English-Nepali Parallel Corpus

Dissertation

Submitted to

Central Department of Computer Science and Information Technology
Kirtipur, Kathmandu, Nepal

In partial fulfillment of the requirements
for the Master's Degree in Computer Science and Information Technology

By
Yoga Raj Joshi
May, 2010

Supervisor
Prof. Dr. Shashidhar Ram Joshi
Department of Electronics and Computer Engineering
Institute of Engineering, Pulchowk, Nepal
(Head)

Tribhuvan University

Institute of Science and Technology

A Chunk Alignment Model for Statistical Machine Translation On English-Nepali Parallel Corpus

Dissertation

Submitted to

Central Department of Computer Science and Information Technology
Kirtipur, Kathmandu, Nepal

In partial fulfillment of the requirements
for the Master's Degree in Computer Science and Information Technology

by

Yoga Raj Joshi

May, 2010

Tribhuvan University

Institute of Science and Technology

Central Department of Computer Science and Information Technology

Supervisor's Recommendation

I hereby recommend that the dissertation prepared under my supervision by **Mr. Yoga Raj Joshi** entitled **"A Chunk Alignment Model for Statistical Machine Translation on English-Nepali Parallel Corpus"** be accepted as fulfilling in partial requirements for the degree of M. Sc. in Computer Science and Information Technology.

Prof. Dr. Shashidhar Ram Joshi
Department of Electronics and Computer Engineering,
Institute Of Engineering, Pulchowk, Nepal
(Head)

Date:_____

Tribhuvan University

Institute of Science and Technology

Central Department of Computer Science and Information Technology

LETTER OF APPROVAL

We certify that we have read this dissertation work and in our opinion it is satisfactory in the scope and quality as a dissertation in the partial fulfillment for the requirement of Master of Science in Computer Science and Information Technology.

Evaluation Committee

Prof. Dr. Jeevan Jyoti Nakarmi Head, Central Department of Computer Science and Information Technology Tribhuvan University	**Prof. Dr. Shashidhar Ram Joshi** Head,Department of Electronics and Computer Engineering, Institute of Engineering Pulchowk, Nepal (Supervisor)
(External Examiner)	**(Internal Examiner)**

Date: _____

Tribhuvan University

Institute of Science and Technology

Central Department of Computer Science and Information Technology

Student's Declaration

I hereby declare that I am the only author of this work and that no sources other than the listed here have been used in this work.

… … … … … … …
Yoga Raj Joshi
Date: May, 2010

Dedication

To my parents

Acknowledgement

I deeply extend my heartily acknowledgement to my respected teacher and dissertation advisor Prof. Dr. Shashidhar Ram Joshi, Department of Electronics and Computer Engineering, Institute of Engineering, Pulchowk, for his wholehearted cooperation, encouragement and strong guidelines throughout the preparation of this study. With this regard, I wish to extend my sincere appreciation to respected Head of the Central Department of Computer Science and Information Technology, Prof. Dr. Jeevan Jyoti Nakarmi for his kind help, encouragement and constructive suggestions. I am also grateful to former heads Prof. Dr. Devi Datta Paudyal and Dr. Tanka Nath Dhamala for their valuable advice and support during my study period.

I owe special debt of gratitude and deeply grateful to Mr. Laxmi Prasad Khatiwada, Prof. Dr. Yogendra Prasad Yadav (Former Head, Central Department of Linguistic), Mr. Balaram Prasain (Lecturer, Central Department of Linguistic), and Mr. Bal Krishna Bal for their kind support during this study. They provide me ample time for discussion on issue related to the study and valuable suggestion during my study. I am very much grateful and thankful to the Madan Puraskar Pustakalaya for the valuable support.

I am very much grateful and thankful to all the respected teachers, Prof. Dr. Onkar P. Sharma (Marist College, USA), Dr. Subarna Shakya, Prof. Sudarshan Karanjit, Asst. Prof. Min Bahadur Khati, Mr. Sammujjwal Bhandari, Mr. Hemanta G.C., Mr. Dinesh Bajracharya, Mr. Kedarjung Thapa and others for granting me broad knowledge and inspirations within the time period of two years of my master degree study.

Friends Mr. Mekh Raj Jaishi, Mr. Suraj Karki and all my class fellows are worthy of my gratefulness for their direct or indirect support in completion of my dissertation.

Yoga Raj Joshi

May, 2010

Abstract

Chunk alignment is the process of mapping the chunk correspondence between the source language text and the target language text which will further help on the other application of the Natural Language Processing tasks including the Machine Translation. This dissertation mainly focuses on the process of chunking and the alignment between the chunks of the two parallel sentences of English and Nepali language. The alignment model first performs tagging by using the TnT tagger and then does chunking of the both tagged sentences using the grammatical rules of bracketing the chunks. And finally it gives the alignment between the source chunks and target chunks using the statistical information gathered from bilingual sentence aligned corpus. The IBM model-1 has been used for the alignment algorithm in which the total alignment probability only depends on the lexicon probability for the given sentence pair. The knowledge-base for statistical information in the alignment is generated by training the bilingual sentence aligned corpus using Expectation Maximization (EM) algorithm. The model performs chunking using a rule based technique and does the alignment by the help of statistics generated from bilingual sentence aligned training corpus; hence the model is of hybrid nature.

Our alignment model does rely on the information from tagging and chunking process of the given sentence pair. Hence the model accuracy not only depends on the alignment algorithm and the training corpus but also depends on the former two models tagger and the clunker. It has been observed that the training corpus having the large quantity of high quality bilingual data and the sufficient chunking rules for bracketing the chunks of the given sentence pair will enhance the accuracy of the model.

Table of Contents

List of Tables

List of Figures

List of Abbreviations

AI	Artificial Intelligence
EBMT	Example Based Machine Translation
EM	Expectation Maximization
HMM	Hidden Markov Model
ITG	Inversion Transduction Grammar
KB	Knowledge Base
MAP	Maximum A Posterior
MT	Machine Translation
NLP	Natural Language Processing
POS	Part of Speech
RBMT	Rule Based Machine Translation
SL	Source Language
SMT	Statistical Machine Translation
TAM	Tense Aspect and Modality
TL	Target Language

CHAPTER I

INTRODUCTION

1.1 Machine Translation

Machine Translation (MT) is the automatic translation of text from one natural language (the source) to another (the target). An MT system expects texts in a specific language as input and produces a text with a corresponding meaning in a different language as output. Hence, machine translation is a decision problem where we have to decide on the best of target language text matching a source language text.

The field of Machine Translation is almost as old as the modern digital computer. In 1949 Warren Weaver suggested that the problem be attacked with statistical methods [45]. A speech of Warren Weaver in 1949, "I have a text in front of me which is written in Russian but I am going to pretend that it is really written in English and that it has been coded in some strange symbols. All I need to do is strip off the code in order to retrieve the information contained in the text", gives the idea of MT. Statistical machine translation was re-introduced in 1991 by researchers at IBM's Thomas J. Watson Research Center [6] and has contributed to the significant resurgence in interest in machine translation in recent years.

To do fluent translation, a translator (human or machine) must read the original text, understand the situation to which it is referring and find a corresponding text in a target language that does a good job of describing the same or similar situation. Often this involves a choice for example the English word 'you' when referring to single person can be translated in to Nepali as either "तिमी" or "तँपाई". Translations sometimes find it difficult to make the choice.

What makes MT so hard? An important reason is that natural languages are highly complex. Many words have various meanings and different possible translations. Sentences might have various readings and the relationships between linguistic entities are often vague. In some languages such as Chinese or Japanese, not even the word boundaries are given. Certain grammatical relations in one language might not exist in another language and sentences involving these relations need to be significantly reformulated.

1.2 Approaches on Machine Translation

Several criteria can be used to classify machine translation approaches, yet the most popular classification is done attending to the level of linguistic analysis required by the system to produce translations. Usually, this can be graphically expressed by the machine translation pyramid in Figure 1.1.

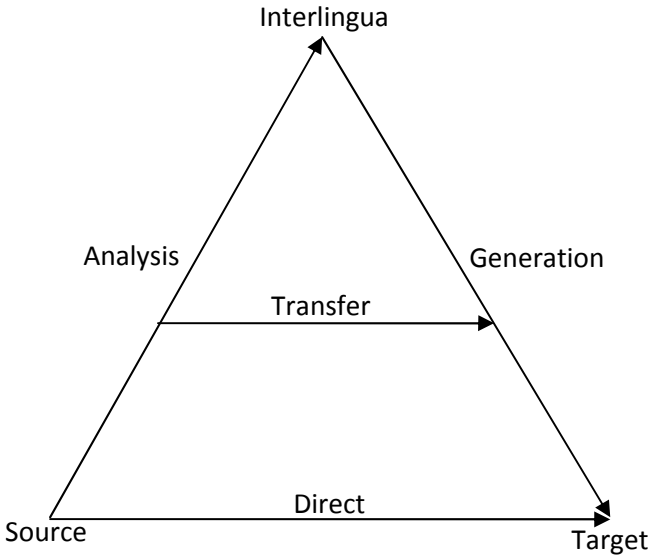


Fig 1.1: Machine Translation Pyramid

Generally speaking, the bottom of the pyramid represents those systems which do not perform any kind of linguistic analysis of the source sentence in order to produce a target sentence. Moving upwards, the systems which carry out some analysis (usually by means of morphs-syntax based rules) are to be found. Finally, on top of the pyramid a semantic analysis of the source sentence turns the translation task into generating a target sentence according to the obtained semantic representation.

1.2.1 Direct translation

This approach solves translation on a word-by-word basis, and it was followed by the early MT systems, which included a very shallow morph-syntactic analysis. Today, this preliminary approach has been abandoned, even in the framework of corpus-based approaches. This approach needs more amount of annotated and aligned data (corpus) for translation.

1.2.2 Transfer-based translation

The rationale behind the transfer-based approach is that, once we grammatically analyze a given sentence, we can pass this grammar on to the grammatical representation of this sentence in another language. In order to do so, rules to convert source text into some structure, rules to transfer the source structure into a target structure, and rules to generate target text from it are needed. Lexical rules need to be introduced as well.

Usually, rules are collected manually, thus involving a great deal of expert human labor and knowledge of comparative grammar of the language pair. Apart from that, when several competing rules can be applied, it is difficult for the systems to priorities them, as there is no natural way of weighing them. In this approach we keep a database of translation rules (or examples), and whenever the rule (or example) matches, the text is translated directly. Transfer can occur at the lexical, syntactic or semantic level.

This approach was massively followed in the 1980s, and despite much research effort, high quality MT was only achieved for limited domains [19].

1.2.3 Interlingua-based translation

An Interlingua is a knowledge representation formalism that is independent of the way particular language expresses meaning. An Interlingua has the added advantage that it efficiently addresses the problem of translating for a large number of languages. Instead of building O (n^2) translation system for all possible pairs of languages, one only has to build O (n) system to translate between each language and the interlingua.

This approach advocates for the deepest analysis of the source sentence, reaching a language of semantic representation named Interlingua. This conceptual language, which needs to be developed, has the advantage that, once the source meaning is captured by it, in theory we can express it in any number of target languages, so long as a generation engine for each of them exists.

The difficulty of creating a conceptual language capable of bearing the particular semantics of all languages is an enormous task, which in fact has only been achieved in very limited domains. Apart from that, the requirement that the whole input sentence needs to be understood before proceeding onto translating it, has proved to make these engines less robust to the grammatical incorrectness of informal language. There may be chances of ambiguity that has to be resolved to translate from a natural language to a knowledge representation language.

1.2.4 Corpus-based approaches

In contrast to the previous approaches, these systems extract the information needed to generate translations from parallel corpora that include many sentences which have already been translated by human translators. The advantage is that, once the required techniques have been developed for a given language pair, in theory it should be relatively simple to transpose them to another language pair, so long as sufficient parallel training data is available.

Among the many corpus-based approaches, the most relevant ones are Example-Based Machine Translation (EBMT) and Statistical Machine Translation (SMT), although the differences between them are constantly under debate. Example-based MT makes use of parallel corpora to extract a database of translation examples, which are compared to the input sentence in order to translate. By choosing and combining these examples in an appropriate way, a translation of the input sentence can be provided.

In SMT, this process is accomplished by focusing on purely statistical parameters and a set of translation and language models, among other data-driven features. Although this approach initially worked on a word-to-word basis and could therefore be classified as a direct method, nowadays several engines attempt to include a certain degree of linguistic analysis into the SMT approach, slightly climbing up the aforementioned MT pyramid.

1.2.5 Statistical Machine Translation

Statistical Machine Translation has come to denote an approach to the whole translation problem that is based on finding the most probable translation of a sentence, using data gathered from a bilingual corpus. As an example of a bilingual corpus, Hansord[1] is a record of parliamentary debate. Canada, Hong Kong and other countries produce bilingual Mansards.

The goal is the translation of a text given in some source language into a target language. We are given a source ('English') sentence $e_1^m = \mathrm{e}_1 \ldots \mathrm{e}_\mathrm{i} \ldots \mathrm{e}_\mathrm{m}$ which is to be translated into a target ('Nepali') sentence $n_1^n = \mathrm{n}_1 \ldots \mathrm{n}_\mathrm{j} \ldots \mathrm{n}_\mathrm{n}$. Among all possible target sentences, we will choose the sentence with the highest probability:

$$\hat{n}_1^n = \arg\max_{n_1^n} \left\{ P\left(n_1^n \mid e_1^m\right) \right\}$$

[1] Named after William Hansard, who first published the British parliamentary debates in 1811

The argmax operation denotes the search problem, i.e. the generation of the output sentence in the target language.

According to authors in [21], SMT is the name for a class of approaches that do just the translation of text from one language to other, by building the probabilistic models of faithfulness[2] and fluency[3], and then combining these models to choose the most probable translation. If we choose the product of faithfulness and fluency as our quality metric, we could model the translation from source language sentence S to a target language sentence T as:

$$Best-translation\ \hat{T} = \arg\max_T\ faithfu\ln ess(T,S) * fluency(T)$$

Here, English is used as source language and Nepali is used as target language. Hence E is used as source language sentence and N as Nepali language sentence.

In probabilistic model, the best target language (Nepali) sentence N= $n_1^n = \mathrm{n}_1 \ldots \mathrm{n}_\mathrm{j} \ldots \mathrm{n}_\mathrm{n}$ is the one whose probability P (N|E) is the highest, where E= $e_1^m = \mathrm{e}_1 \ldots \mathrm{e}_\mathrm{i} \ldots \mathrm{e}_\mathrm{m}$ is the source language (English) sentence. Where e_i is the i^{th} chunk of English sentence and n_j is the j^{th} chunk of Nepali sentence and m and n are the length of English and Nepali sentence respectively. The length is measured in terms of number of chunks present in a sentence. This can be expressed by the application of Bayes' rule:

$$\begin{aligned} Best-translation\ \hat{N} &= \arg\max_N\ P(N \mid E) \\ &= \arg\max_N \frac{P(E \mid N) * P(N)}{P(E)} \\ &= \arg\max_N\ P(E \mid N) * P(N) \end{aligned}$$

This rule says we should consider all possible target language (Nepali) sentences N and choose the one that maximizes the product $P(E \mid N) * P(N)$.

We can ignore the denominator P (E) inside the argmax operation since we are choosing the best Nepali sentence for a fixed English sentence E, and hence P (E) is a constant.

[2] Faithfulness is obtained from Translation Model which says how probable a source language sentence is as a translation, given a target language sentence.
[3] Language model gives the Fluency of the string which says how probable a given sentence is in target language.

The factor P (N) is the Language Model for Nepali language; it says how probable a given sentence is in Nepali. P (E|N) is the Translation Model; it says how probable an English sentence is as a translation, given a Nepali sentence.

We pretend that English (source language) sentence e, we must translate is a corrupted version of some Nepali (target language) sentence n, and our task is to discover the hidden (target language) sentence n that generates our observation sentence e. This idea was suggested by Weaver.

The language model P (N) can be any model that gives a probability to a sentence. With a large corpus, for example Nepali corpus of 100 thousands sentences, if the sentence 'ऊ बिधार्थी हो' appears 100 times, then P (ऊ बिधार्थी हो) = 100/100000 = 0.001. But even with 100 thousands examples most sentence count will be zero. Therefore the familiar n-gram (for e.g. bi-gram) language model can be used, in which the probability of Nepali sentence of the words $n_1 \ldots n_m$ is

$$P(n_1 \ldots n_m) = \prod_{i=1}^{m} P(n_i \mid n_{i-1})$$
$$= P(n_1) * P(n_2 \mid n_1) * P(n_3 \mid n_2) * \ldots * P(n_m \mid n_{m-1})$$

We will need to know bi-gram probabilities such as P (बिधार्थी|ऊ). This captures only a very local notion of syntax where a word depends on just the previous word.

The Translation model P (E|N) is more difficult to come by. For one thing, we don't have to ready collection of (English, Nepali) sentence pairs from which to train. For another, the complexity of the model is greater, because it consider the cross product of the sentences rather than just individual sentences.

Basically Statistical Machine Translation system must deal with the following three problems:

- Modeling Problem: How to depict the process of generating a sentence in a source language, and the process used by a channel to generate a target sentence upon receiving a source sentence? The former is the problem of language modeling, and the later is the problem of translation modeling. They provide a framework for calculating P (N) and P (E|N).

- Learning Problem: Given a statistical language model P (N) and a statistical translation model P (E|N), how to estimate the parameters in these models from a bilingual corpus of sentences?

- Decoding Problem: With a fully specified (framework and parameters) language and translation model, given a source sentence, how to efficiently search for the target sentence that satisfies.

In statistical machine translation, it is necessary to model the translation probability $P(e_1^m \mid n_1^n)$, which describes the relationship between a source language sentence e_1^m and a target language sentence n_1^n. The simple model is to translate a sentence, just translate each word individually and independently in left to right order. Thus in unigram word choice model, it makes it easy to compute the translation:

$$P(E \mid N) = \prod_{i=1}^{n} P(e_i \mid n_i)$$

In few cases this model works fine. However in most of cases, the model fails for the language pair like English-Nepali which include different word order; for e.g. the English sentence 'go home' is translation of Nepali sentence 'घर जाऊ'. In this sentence, the translation of English word 'go' is 'जाऊ' and 'home' is 'घर'. Since the word order is different, hence the model fails. Another problem is that the word choice is not always one to one. To handle the fact that words are not translated one to one, the model introduces the notion of 'fertility' of words. A word with fertility n gets copied over n times and the each of n copies gets translated independently. Another solution is to bracketing the words in the grammatical unit chunk and translates these chunks as translating the words independently.

In statistical alignment models $P(e_1^m, a_1^m \mid n_1^n)$, a hidden alignment variable $a_1^m = a_1 \ldots a_j \ldots a_m$; where $a_j \in \{1 \ldots n\}$ is introduced that describes a mapping from source position j to target position $i=a_j$. The relationship between a translation model and alignment model is given by:

$$P(e_1^m \mid n_1^n) = \sum_{a_1^m} P(e_1^m, a_1^m \mid n_1^n)$$

Typically, the search is performed using the so-called maximum approximation:

$$\hat{n}_1^n = \arg\max_{n_1^n} \left\{ P(n_1^n) * \sum_{a_1^m} P(e_1^m, a_1^m \mid n_1^n) \right\}$$
$$\approx \arg\max_{n_1^n} \left\{ P(n_1^n) * \max_{a_1^m} P(e_1^m, a_1^m \mid n_1^n) \right\}$$

Hence, the search space consists of the set of all possible target language sentences n_1^n and all possible alignments a_1^m.

Since its revival more than a decade ago when IBM researchers presented the Candide SMT system [6, 7], the statistical approach to machine translation has seen an increasing interest among both natural language and speech processing research communities.

1.3 Corpora and Parallel Corpora

Corpora are the term used on Linguistics, which corresponds to a (finite) collection of texts (in a specific language). A collection of documents in more than one language is called multilingual corpora. A parallel corpus is a collection of texts in different languages where one of them is the original text and the other is their translations. A bilingual corpus is a collection of texts in two different languages where each of one is translation of other.

Parallel corpora are very important resources for tasks in the translation field like linguistic studies, information retrieval systems development or natural language processing. In order to be useful, these resources must be available in reasonable quantities, because most application methods are based on statistics. The quality of the results depends a lot on the size of the corpora, which means robust tools are needed to build and process them.

The alignment at sentence and word levels makes parallel corpora both more interesting and more useful.

- Aligned bilingual corpora have been proved useful in many ways including machine translation, sense disambiguation and bilingual lexicography.

1.4 Statistical Alignment in Statistical Machine Translation

The traditional score based approach to alignment aims to find the one best alignment of a set of sequences, which is then used as the 'true' alignment. Statistical alignment fundamentally breaks with this view. It is based on stochastic models of sequence evolution. In general, this means that every alignment is possible. However, depending on parameters there may be a vast difference in how probable two different alignments are. In statistical alignment, the score of an alignment is the probability of obtaining the alignment under the evolutionary model, so rather than just providing a means for sorting alignments to be able to nominate one as the best, the scores of statistical alignment defines a distribution over plausible alignments. This has several advantages, among them quantifying the uncertainty of alignments and allowing joint modeling and inference of alignment, alignment parameters and phylogeny in a sound probabilistic framework.

In general statistical alignment algorithms try to use word distribution information in text to find some relationship between possible translations. Most of the algorithms are rely on the frequency of the co-occurrences of the words with in the text. To determine these values, texts should first go into a series of segmenting, preprocessing and archiving activities. Then parameter estimation and comparing similarities are involved in determining the distribution relations. There is different level of alignment to achieve the actual translation of text from one language to another.

1.5 Text Alignment

Text Alignment is the task of identifying correspondences between the texts written in two different languages. Statistical Machine Translation is the data driven approach of finding the correspondence in two different languages. The aligned text will play the role of data in SMT. Hence text alignment plays an important role to make bilingual corpora which will be very useful in Statistical Machine Translation. Text alignment is done in different levels; it includes document alignment, paragraph alignment, sentence alignment, word alignment or chunk alignment etc.

1.5.1 Document Alignment

Document alignment is the process of finding the document pair that is translation of one another from the collection of bilingual texts.

1.5.2 Paragraph Alignment

Paragraph are often aligned sequentially, i.e. first paragraph of one language to first paragraph of another and so on. This might not be always true. Insertions, deletion, splitting and merging may appear on translating the paragraph of different language. Paragraph marker is used to separate the different paragraph on the document. Sometimes the use of the cognates and collocation is also used to recognize translation paragraphs.

1.5.3 Sentence Alignment

Parallel text provides the maximum utility when it is sentence aligned. The sentence alignment task is to identify correspondences between sentences in one language and sentences in the other language. This task is a first step toward the more ambitious task finding correspondences among words. Sentence alignment is not trivial because translators do not always translate one sentence in the input into one sentence in the output. Another problem is that of crossing dependencies, where the order of sentences is changed in the translation.

There are well established algorithms for aligning sentences across parallel corpora. Some are pure length based approaches, some are lexicon based, and some are a mixture of the two approaches

The length based approach works remarkably well on language pairs with high length correlation, such as French and English. Its performance degrades quickly, however, when the length correlation breaks down, such as in the case of Chinese and English. Among the various length based algorithms Gale and Church Algorithm [16] is the famous one. The Gale-and-Church Algorithm is basically dependent on the length of the sentence in terms of characters and the Brown's algorithm [8] is dependent on the length of the sentence in terms of words. Dynamic programming is then used to search for the best alignment in both the algorithms. These algorithms are based on the idea that long sentences will be translated into long sentences and short sentences into short ones.

Even with language pairs with high length correlation, the Gale-Church algorithm may fail at regions that contain many sentences with similar length. A number of algorithms, such as [47], try to overcome the weaknesses of length based approaches by utilizing lexical information from translation lexicons, and/or through the identification of cognates. Lexicon based methods are based on the lexical resources such as bilingual lexicon (bilingual dictionary). The other resources that are used include the Chunker for both the languages.

The algorithm first breaks the sentences of both the languages into small units called chunks. To find an alignment for a sentence in the source language, it is matched with a set of possible sentences in the target language and the scores are assigned for each comparison. The score of match of two sentences is calculated by finding out the number of chunks that match between the two sentences. The algorithm then carries out the alignment of sentences using these scores. The precision of the alignment is 94.3%. [4].

For sentence alignment paragraph alignment is performed first, and then sentence within a paragraph are aligned. Paragraphs within a document can be aligned manually by inserting the paragraph marker within the document.

1.5.4 Word Alignment

Word alignment is the natural language processing task of identifying translation relationships among the words in a bitext. For a word alignment system the texts are first segmented into smaller units which are themselves aligned. Word alignment is typically done after sentence alignment of already identified pairs of sentences that are translations of one another. Bitext word alignment is an important supporting task for most methods of statistical machine translation; the parameters of statistical machine translation models are typically estimated by observing word-aligned bitexts, and conversely automatic word alignment is typically done by choosing that alignment which best fits a statistical machine translation model. Circular application of these two ideas results in an instance of the expectation-maximization algorithm.

In the word alignment algorithm, any word of the target language is taken to be possible translation for each source langauge word. The probablity of some target language word to be a translţion of source langauge word then depends on the frequency with which both co-occur at the same or similar positions in the parallel corpus. The probabilities are estimated from the use of EM alogorithm and a Viterbi search is carried out to compute the most probable sequence of word translation pairs.

In 1991, Gale and Church [17] introduced the idea of using measures of association for finding translations of words based on information in parallel text. They begin by carrying out sentence alignment, which is the problem of determining which sentences are translations of each other. In fact this is a much simpler problem than finding the translations of words, since long sentences in one language tend to translate as long sentences in another language, and the order in which sentences appear doesn’t usually change radically in a translation.

The original K-vec algorithm proposed by Fung and Church [15] works only for parallel corpus and makes use of the word position and frequency feature to find word correspondences.

Most current SMT systems [22, 32] use a generative model for word alignment such as the freely available GIZA++ [31], which is an implementation of the IBM word alignment models [6] in C++. These models treat word alignment as a hidden process, and maximize the probability of the observed sentence pairs using the expectation maximization (EM) algorithm.

Singh and Chinnappa [11] uses the information about the cognates which is specially relevant for Indian languages because these languages have a lot of borrowed and inherited words which are common to more than one language. They implemented the IBM models and added the Dice coefficient similarity measures as a parameter to the EM algorithm to improve the performance of word alignment accuracy.

There are several word-alignment strategies for major languages such as English and French. One of the examples is found in [16]. Considerable effort has also been made to align English-Chinese translation texts [15, 47].

The task of aligning words has been dominated mostly by statistical approaches based on the distribution of words in text. The assumption behind using the statistics of words as an indication of possible association between terms is hinged on the assumption that translation words are comparably distributed in parallel texts. In practice, word alignment is much more difficult than sentence alignment.

Phrase alignment [25, 49] falls between word and sentence alignments, but it is usually resolved subsequent to word alignment.

1.5.5 Chunk Alignment

Natural languages are made up of words that are fully formed by rules of generation that vary across languages. In morphologically complex languages a word might be inflected to contain lots of information in it, while in simpler languages the same information might be expressed using several words. This leads into a situation where, given complex-simple bilingual text pairs, we often have word alignments of m: 1 which is not easy to extract using automated systems. English-Nepali bitext fall in this category. Bracketing the words into the appropriate chunks helps to handle this situation. The chunks are often aligned 1:1 which is easy to extract using automated system but this may not true for all pair of languages.

Chunk alignment is the extended idea of word alignment, the process of mapping the chunk correspondence between the source language text and the target language text. Chunk is a group of related words which forms the meaningful constituent in sentence. It is a grammatical unit of a sentence. Words can also be treated as a chunk of characters. Chunks are used to refer to a single concept. A sentence can thus be looked as a sequence of chunks; each chunk adding information to the sentence. Here the chunks can be seen as the building blocks of a sentence in a conceptual term. They are often used to do the analysis of the sentence.

According to Abney [1], a typical chunk consists of a single content word surrounded by a constellation of function words. And Bharati et al. [5] define the chunk as- "A minimal (non recursive) phrase (partial structure) consisting of correlated, inseparable words/entities, such that the intra chunk dependencies are not distorted".

Before aligning the chunks in the bitext, the first step is dividing the sentence in to the smaller units called chunks. The process is known as chunking. Chunking is sometimes called shallow parsing which involves dividing the sentences in to non-overlapping elements on the basis of very superficial analysis. It includes discovering the main constituents of the sentence (NCH, VCH, ADJCH). Shallow parsing usually identifies non-recursive constituents, also called chunks [1] such as non recursive noun phrases, verb phrases so on. Shallow parsing which always follows the tagging process is used as the fast and reliable preprocessing phase for full or partial parsing. Tagging is the process of association of word category (tag) to the word of a sentence.

Example:

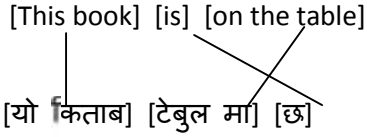


Fig1.2: English/Nepali Bitext alignment at sentence and chunk level. Bracketing denotes the Chunk alignment.

Detection of corresponding chunks between two sentences that are translations of each other is usually an intermediate step of SMT but also has been shown useful for other applications of bilingual lexicons (phrase level dictionary), and projection of resources and cross language information retrieval. In addition it is also used to solve the computational linguistics tasks such as disambiguation problems.

CHAPTER II

BACKGROUND AND PROBLEM DEFINITION

2.1 Background

Research on text alignment has largely focused on aligning either sentences or words, i.e. most approaches either compute which sentence of a source and a target form a translational pair, or they use sentence alignment as a preprocessing step to align on the word level.

Additionally, emphasis was laid on the development of language independent algorithms. Ideally, such algorithms would not be tailored to align a specific language pair, but would be applicable to any two languages. Language independence has also been favored with respect to linguistic resources in that alignment should do without e.g. using pre-existing dictionaries. Hence there is a dominance of purely statistical approaches.

A lot of work has been done in the field of alignment in SMT and much is being done. All statistical translation models are based on the idea of word alignment [21]. Hiemstra uses a pure bag of words model [18]. Most of the statistical translations models are based on the IBM Models are built from bilingual corpora without considering the structural aspect of the language [6]. Different word alignment techniques have been developed usually based on statistical information [4]. Standard word alignment approaches like the ones by [6, 7, 41] make use of statistical models to devise word alignment. Brown et. all in [7] have been the first to publish a word alignment procedure. It consists of cascade of five statistical translation models of increasing complexity. The first model of [7], IBM-1 treats every sentence as a bag of words, where the position of word in a sentence does not have influence on its translational probability. IBM-2 to IBM-5 refines this notion by introducing the statistical weights such as distortion and fertility to account for word order phenomena and one to many alignments.

The two competing standard alignment models by [41] correspond most closely to the IBM-1 model: The HMM model by [7] treats a sentence mainly as a bag of words but the probability of alignment is influenced by the preceding alignment. [4] have presented an algorithm for aligning the sentence in bilingual corpora using lexical information and certain heuristics. [31] compared the different alignment models for SMT by measuring the quality of alignment model using the quality of Viterbi

alignment compared to manually-produced alignment. A new statistical method called bilingual chunking for structure alignment has been proposed by Wang et. al.[42].

2.2 Problem Definition

A key issue in modeling the sentence translation probability P(E|N) is the question of how we define the correspondence between the words of Nepali sentence and the words of English sentence. The idea here is extended to the chunk level. Chunk alignment lies between word alignment and sentence alignment. When a person speed-reads through a text, he or she looks for key phrases rather than fully parse the sentence and the stress falls either in the beginning of the chunk or at the end of the chunk. And one of the psycholinguistic evidence is; while the human translator translates the sentence from one language to other he/she pick the chunk of a source sentence at a time in his/her mind and translate it to the target language. And another thing of bracketing the sentence at the chunk level reduces the length of the sentence and hence reduces the complexity of the alignment. Hence the motivation is towards the chunk level alignment. The basic idea of this approach is to develop a model of the translation process with the chunk alignment as a hidden variable of this process, to apply statistical estimation theory to compute the 'optimal' model parameters, and to perform alignment search to compute the best chunk alignment. In typical cases, we can assume a sort of pair wise dependence by considering all chunk pairs (e_i , n_j) for a given sentence pair (E,N). We can further constrain the model by assigning each English chunk to exactly one Nepali chunk and vice-versa. Models describing this type of the dependences are referred to as the alignment models.

Formally, the following definition of alignment at chunk level is used:

We are given an English (source language) sentence E= $e_1^m = \mathrm{e}_1 \ldots \mathrm{e}_\mathrm{i} \ldots \mathrm{e}_\mathrm{m}$ and a Nepali (target language) sentence N= $n_1^n = \mathrm{n}_1 \ldots \mathrm{n}_\mathrm{j} \ldots \mathrm{n}_\mathrm{n}$ that have to be aligned. We define an alignment between the two sentences as a subset of the Cartesian product of the chunk position; that is, an alignment A is defined as:

$$A \subseteq \{ (i, j) : i = 1 \ldots m;\ j = 1 \ldots n \}$$

The alignment mapping consists of associations i→j, which assigns a chunk e_i in position i to a chunk n_j in position j=a_i.

The objective of the statistical alignment model is to describe the relationship between a source language sentence and a target language sentence adequately.

For a given sentence pair there is a large number of alignment, but the problem is to find out the best alignment:

$$\hat{a}_1^m = \arg\max_{a_1^m} P(e_1^m, a_1^m \mid n_1^n)$$

The alignment $\hat{a}_1^m$ is also called the Viterbi Alignment of the sentence pair (e_1^m , n_1^n). We propose to measure the quality of the alignment model using the quality of the Viterbi Alignment compared to a manually produced reference alignment.

2.3 Applications and Overview

The applications of aligned texts are extremely diverse, and include compiling translation memories; deriving dictionaries and bilingual terminology lists, extracting knowledge for cross-language information retrieval, extracting examples for machine translation and computer assisted teaching or contrastive linguists, etc. It is these applications that encouraged researchers to invest more effort on the area. A review of the need and relevance of aligned texts for some of the application areas is presented.

2.3.1 Machine translation

One of the major applications for which data-driven bilingual lexica are used is machine translation system. Machine translation systems require bilingual lexica from which to get translations of terms. The well known data-driven approaches to machine translation are basically two: Example based machine translation (EBMT) [35, 37] and statistical machine translation (SMT) [2, 9]. The basis for SMT is basically word translations. Lately the use of bigram or trigram models [22, 23] has been practiced because translations are not always one to one correspondence of words and also because such sequences of strings or phrases lead to sentences faster than the word to word translations. In example based translation systems, translation data of bigger chunks are used. The idea behind example based translation system is a kind of translation gathering made by a second language

learner where one translates an entire expression, phrases or clauses heard or read before to construct new translations. The type of bilingual lexicon required by machine translation systems is thus a database of translation units of words or phrases or clauses that were used in previous translations. Efficient algorithms that do align sentences or sub-sentence chunks of any level are thus very important tools to achieve such structures from parallel corpora.

2.3.2 Bilingual lexicography

Word alignment can be used to create bilingual translation dictionaries of the vocabulary included in parallel corpora. Phrase or chunk alignment is used to create phrase level dictionary. Bilingual dictionaries generated from translation corpora are of utmost importance in providing translation of terms in different contexts. Domain specific dictionaries without doubt are built exhaustively and precisely from translation documents in certain domains.

2.3.3 Computer-assisted language learning

Computer-assisted Language learning (CALL) refers to programs designed to help people learn foreign languages. CALL is an innovative approach of second language acquisition. Natural language processing has been enlisted in several ways in CALL: including carrying out bilingual text alignment so that a learner who encounters an unknown word in a second language can see how it was rendered in translation [29]. Parallel texts can help the language learners to know the grammatical patterns of a language in advance. The grammar of two languages can be compared with the help of aligned texts so that the language learners can take benefit on language understanding.

2.3.4 Machine-aided human translation

Existing translations are extremely valuable resources that can be exploited with software systems to improve the efficiency of human translation. A human translator can learn how certain source language expressions have been translated from existing translations for improving translation, and

improving consistency of translation. Translation memory, dictionaries with multi-word units and non-compositional compounds are useful resources for such purposes [26]. Translation texts are providing such tools readily.

Human translators get benefit from machine translation in different ways. Translators may access a bilingual terminology or they can use a translation memory or they directly submit the parts of the text to a machine translation server to get the complete translation. Hence the previously translated texts can be used as a helpful document to the human translators.

2.3.5 Cross-language information retrieval

Information retrieval systems that retrieve documents from more than one language can use bilingual lexica by which query words are translated and the search is carried out in different languages. Cross-language information retrieval is a subfield of information retrieval dealing with retrieving information written in a language different from the language of the user's query. For example, a user may pose their query in English but retrieve relevant documents written in Nepali.

Domain specific bilingual lexica, particularly, provide very useful support in getting the sense of words in a specified context. Such kind of bilingual dictionaries are simply generated by searching repeated co-occurrence.

To use the automatically extracted dictionary for information retrieval, each of the words in the original query is substituted by the possible translations into a new query in the other language. This new query is then used for monolingual retrieval in the document collection. Retrieval performance can be enhanced by giving weights to the more likely translations of each term as obtained in the frequency information in the bilingual dictionary. Such approaches are supposed to have a performance approaching monolingual accuracy [10].

2.3.5 Word sense disambiguation

The task of word sense disambiguation (WSD) is to determine the correct meaning, or sense of a word in context. Word sense disambiguation is the process of identifying which sense of a word is used in any given sentence, when the word has a number of distinct senses. It is a

fundamental problem in natural language processing (NLP). The ability to disambiguate word sense accurately is important for applications such as machine translation, information retrieval, etc. Corpus-based supervised machine learning methods have been used to tackle the WSD task [33]. Among the various approaches to WSD, the supervised learning approach is the most successful to date. One source to look for potential training data for WSD is parallel texts [14]. Given a word-aligned parallel corpus, the different translations in a target language serve as the sense-tags of an ambiguous word in the source language. The outcome of word sense disambiguation of a source language word is the selection of a target word, which directly corresponds to word selection in machine translation.

2.3.6 Paraphrasing

Paraphrasing means to express someone else's ideas in our own language without changing its original meaning. The main way to paraphrase is to change the structure of the paragraph and to change the words including it. It becomes sometimes difficulties in Natural Language Processing because of the fact that there are many ways to express the same message. Recent studies show that by way of alignment techniques for phrase-based statistical machine translation, paraphrases in one language can be identified using a phrase in another language as a pivot [3].

So much about the applications of aligned bilingual translation texts, a lot has been done to come up with efficient techniques and methods of aligning existing translation texts without which the application described are simply theoretical.

2.4 Review of Alignment Models or Alignment Algorithms

2.4.1 Statistical Word Alignment models

One of the fundamental goals of SMT is describing word alignment. Alignment at word level specifies how word order changes when a sentence is translated into another language.

In this section, we will give an overview of the commonly used statistical word alignment models. We are given a source language sentence e: = e_1^m which has to be translated into a target language

sentence n: = n_1^n . According to the classical source-channel approach, we will choose the sentence with the highest probability among all possible target language sentences:

$$\hat{n} = \arg\max_n \{P(n \mid e)\}$$
$$= \arg\max_n \{P(n) * P(e \mid n)\}$$

This decomposition into two knowledge sources allows for an independent modeling of target language model P (n) and translation model P (e |n).

In statistical machine translation, we try to model the translation probability $P(e_1^m \mid n_1^n)$, which describes the relationship between a source language string e_1^m and a target language string n_1^n . In statistical alignment models $P(e_1^m, a_1^m \mid n_1^n)$, a hidden alignment variable $a_1^m = a_1 ... a_j ... a_m$; where $a_j \in \{1 ... n\}$ is introduced that describes a mapping from source position j to target position i=a_j. The relationship between a translation model and alignment model is given by:

$$P(e_1^m \mid n_1^n) = \sum_{a_1^m} P(e_1^m, a_1^m \mid n_1^n)$$

The alignment a_1^m may contain alignment a_j=0 with the empty word n_0 to account for source words that are not aligned with any target word. Usually, we use restricted alignments in the sense that each source word is aligned to at most one target word.

In general, the statistical model depends on the set of unknown parameters that is learned from training data. The art of the statistical modeling is to develop specific statistical models that capture the relevant properties of the considered problem domain. Here in the alignment problem, the statistical alignment model has to describe the relationship between a source language string and a target language string adequately.

To train the unknown parameters, we are given a parallel training corpus consisting of large amount of sentence pairs. The EM algorithm described in [13] is used to perform the maximization of the parameter. Note that the use of the EM algorithm is not essential for the statistical approach, but only a useful tool for solving the parameter estimation problem.

Although for a given sentence pair there is a large number of alignments, we can always find a best alignment:

$$\hat{a}_1^m = \arg\max_{a_1^m} P(a_1^m \mid e_1^m, n_1^n) = \arg\max_{a_1^m} P(e_1^m, a_1^m \mid n_1^n)$$

The alignment a_1^m is also called Viterbi alignment of the sentence pair (e_1^m, n_1^n). The quality of this Viterbi alignment can be measured by comparing it to a manually produced reference alignment.

2.4.2 Computation of the Viterbi alignment

Och and Ney in [31] developed an algorithm to compute the Viterbi alignment for each alignment model. There is a simple polynomial algorithm for the model 1 and model 2.

$$\begin{aligned}
\hat{a}_1^m &= \arg\max_{a_1^m} P(e_1^m, a_1^m \mid n_1^n) \\
&= \arg\max_{a_1^m} \left\{ P(m \mid n) \cdot \prod_{j=1}^{m} \left[P(a_j \mid j, n) \cdot P(e_j \mid n_{a_j}) \right] \right\} \\
&= \left[\arg\max_{a_j} \left\{ P(a_j \mid j, n) \cdot P(e_j \mid n_{a_j}) \right\} \right]_{j=1}^{m}
\end{aligned}$$

For the Model 1, if we make the additional simplifying assumption that the distribution $P(a_j \mid j, n)$ is uniform; the only parameters that are required in order to compute the optimal alignment are the word translation parameters $P(e_j \mid n_{a_j})$. Notice that this independence assumption reduces the model to a set of m (source positions) independent decisions each with n + 1 (target word positions) possible outcomes. n+1 is the number of target words plus the empty target word. The Maximization over the $(n+1)^m$ different alignment decomposes into m maximization of (n+1) lexicon probabilities.

2.4.3 Statistical Generative word alignment models

Generative models simulate the process of generating the observations, given some hidden variables. It defines the joint distributions of the observations and the hidden variables, and learning the parameters for maximizing the data likelihood, typically through Expectation Maximization or its variants models

The IBM models in Brown et al. [6] are typically generative models, laying down a foundation for statistical machine translation. The models learn word translation pairs from parallel corpora through the Expectation Maximization (EM) algorithm.

These models approximate the translation process as a process of manipulating bilingual word pairs: permutations, substitutions, and insertions/deletions with NULL token introduced into the generative process. To be more specific, the generative story is approximately as follows: Delete or duplicate n times for each target word according to a fertility table; second, once the desired length of the target sentence is reached, add the necessary number of NULL words to generate the source words, which have no corresponding target translations; try to map the source word with target word in a one-to-one fashion which is preferred by the noisy channel model; fourth after all the word pairs are connected, the resulting source word string is permuted into a possibly different positions as controlled by a distortion table. These mixture models are mainly located near the paths directly from source to target at the bottom of the Vauqois pyramid.

To assign probabilities to target strings given source strings P(t|s), most statistical translation models a hidden random variable: word alignment a which specifies how words between source and target strings are to be aligned. A generative translation model describes how the target string t is generated from the source string s stochastically, which determines the conditional likelihood of "complete" data P(t,a|s). The conditional likelihood of "incomplete" data, i.e. sentence translation probability, is obtained by considering all possible word alignments $P(t \mid s) = \sum_a P(t, a \mid s)$.

Model parameters are usually estimated from a collection of sequence pairs via the Expectation Maximization (EM) algorithm [13].

Given a pair of sentence (s,t), the best word alignment under the model is given by the Maximum A Posterior (MAP) criterion

$$\hat{a} = \arg\max_a P(a \mid s, t) = \arg\max_a P(t, a \mid s).$$

The problem of word alignment is generally defined as building word links between a bilingual sentence-aligned corpus. Formally, for each sentence pair ($s = s_1^I, t = t_1^J$) in the corpus, the problem is to produce a bag of word links A = {(i,j)}, where i is the word index of sentence s and j is the word index of sentence t. The definition applies to general word alignment models. Statistical generative models [6, 31, 41] find a source word for each target word. This is expressed by the hidden random variable $a = a_1^J$, which maps each target word t_j to the source word at position a_j, j = 1,2,…,J; when $a_j = 0$, t_j has no correspondence in the source string. Therefore, the bag of word links generated is A = {(a_j, j)|j = 1,2,…,J}.

This notion implies that any one target word can be linked to at most one source word.

2.4.3.1 IBM Models

Brown et al. [6] developed five statistical models of translation, IBM Models 1 through 5, and parameter estimation techniques for them. These models all use the many-to-one alignment structure. The IBM models are word-based models and represent the first generation of SMT models. The models were designed to be used in a pipeline, where each model is bootstrapped from the previous model. Model complexity is increased gradually with more parameters to estimate.

Model 1:

Given the target sentence $t_1 \ldots t_I$ and source sentence $s_1 \ldots s_J$ from the parallel corpus, we want to find the best alignment a, where a is a vector a = $\{a_j, a_{j+1}, a_{j+2}, \ldots, a_J\}$j=1 to J. The value of a_j represents the position of the target word t_{aj} (a_j = i) to which s_j corresponds. We add a spurious NULL word to the target sentence at position 0. Thus there are $(I+1)^J$ possible alignments.

Since in model-1 word positions are not considered, the probability of alignment of any two positions is a constant equal to $\frac{1}{(I+1)^J}$ for a particular sentence pair. The probability of generating source word given a target word is given as:

$$P(s,a \mid t) = \frac{1}{(I+1)^J} \prod_{j=1}^{J} T(s_j \mid t_{a_j})$$

Model-2: Distortion or Alignment Parameter

Given source and target sentence lengths J and I, probability that ith target word is connected to jth source word, the distortion probability is given as P***(i \| j, I, J)***. Now, the probability of an alignment ***a***, given the target sentence and the lengths of the source and target sentences is:

$$P(a \mid t; I, J) = \prod_{j=1}^{J} P(a_j \mid j, I, J)$$

where $\boldsymbol{a = \{a_1, \ldots, a_J\}}$ and $\boldsymbol{a_j}$ is the position in target sentence that aligns to the jth position in the source sentence, i.e., $\boldsymbol{a_j}$ is ***i***.

Now the probability of generating a target word with alignment ***a***, given the source word and the lengths of the source and target sentences can be calculated as:

$$P(s,a \mid t) = \prod_{j=1}^{J} P(a_j \mid j, I, J) * P(s_j \mid t_{a_j})$$

Model-1 and Model-2 assume that the target word string is generated independently from the source string. For each target position, a source position is chosen randomly; then, the target word is sampled from the chosen source word translation table. In Model-1, source positions are selected uniformly, while in Model-2 they depend on the actual position and the length of the two strings. Model 1 makes very strong conditional independence assumptions on word placement and generation. Model 2 relaxes one of the assumptions of Model 1, by making the location of the target word which generated each source word dependent on the absolute locations of the two words.

Formally, let P(i|j; I, J) be the probability of the i^{th} target word choosing the j^{th} source word.

Then the conditional likelihood is given by

$$P(s,a \mid t) = \prod_{j=1}^{J} P(a_j \mid j, I, J) * P(s_j \mid t_{a_j})$$

Model-1 is a special case of Model-2 where P(i|j;I,J) = $\frac{1}{(I+1)}$ regardless of i and j.

Fertility based Models

Fertility based alignment models (Model 3, 4, and 5) have a significantly more complicated structure than the simple models 1 and 2. Model 3, 4, and 5 all use the important concept of fertility. For each source word, fertility models first decide how many target words it generates. Fertility is the number of (zero or more) source words that will be generated from target word t_i and is dependent only on t_i. It is needed to generate source words from the NULL target word. These models introduce the concept of spurious words. The words which have no corresponding target word are called spurious words.

The fertility models of [6] explicitly model the probability $P(¤ \mid e)$ that the English word e_i is aligned to $¤_i = \sum_j ¢(a_j, i)$ French words.

Model 3 is a zero-order alignment model like Model 2 including in addition fertility parameters. Model 4 of [6] is also a first-order alignment model (along the source positions) like the HMM, but includes also fertilities. In Model 4 the alignment position j of an English word depends on the alignment position of the previous English word (with non-zero fertility) j'. It models a jump distance j - j ' (for consecutive English words) while in the HMM a jump distance i - i ' (for consecutive words in another language) is modeled. The full description of Model 4 [6] is rather complicated as there have to be considered the cases that English words have fertility larger than one and that English words have fertility zero.

A special problem in Model 3 and Model 4 concerns the deficiency of the model. This results in problems in re-estimation of the parameter which describes the fertility of the empty word. In normal EM- training, this parameter is steadily decreasing, producing too many alignments with the empty word. Therefore we set the probability for aligning a source word with the empty word at a suitably chosen constant value.

Once the fertility of the source word is determined, the target words are generated by translating its aligned source word. As the Model 1 translation will be based only on the source words. Spurious target words will be generated by translating the NULL word. Then the target word position is determined by the distortion probability, which is conditioned on the source and target sentence lengths. Model 4 is used in much of the work in Statistical Machine Translation published in the last several years. Model 4 is a generalization of Model 3 where the alignment model uses relative positions rather than absolute positions. The alignment model is again inverted from that used by Model 1 and Model 2. The detail description of these models is found on [6].

2.4.3.2 HMM-based Statistical word alignment models

In Hidden markov model for word alignment, words in one language are treated as states while words in the other language are regarded as observations. The HMM word alignment model[41] constructs a Markov space by treating each source word s_i as a state and building fully connected transitions between states. The HMM word alignment model uses an alignment model which has relative positions, like IBM Model 4, rather than using an alignment model involving absolute positions which are used with models like IBM Model 2. It observes the target words, which are the emissions of the HMM, and we know that there are I states, the source words. The transition parameters are tied by distance. For example, suppose we already emitted the target word at position j from state i. The transition probability of transitioning from state i to i' (which would mean

that we would emit the target word at position j + 1 from i') is conditioned on the signed distance i'-i. The more detail explanation can also be found in [31].

The target sentence t_1^I is an observation sequence. Target words are emitted one by one from left to right after each state transition. The state output probability distribution is the word to word translation table t(t|s), called t-table. Given these, the conditional likelihood functions read as

$$P(s,a \mid t) = P(a \mid t)P(s \mid t,a) = \prod_{j=1}^{J} P(a_j \mid a_{j-1};I)P(s_j \mid t_{a_j})$$

Like HMM in acoustic modeling, an efficient forward-backward algorithm can be applied to train parameters of HMM word alignment models. The Viterbi algorithm can be used to find the most likely state sequence, which corresponds to the best word alignment under the model.

To handle target words for which no source words are responsible, i.e. target words aligned to the empty word, Och and Ney [31] expanded the Markov network by doubling the state space. Target words emitted by the added states are aligned to the empty word. The Markov transition matrix is adjusted accordingly to allow transition to the added states.

Toutanova et al [40] extended the HMM based word alignment models in several ways. One of them is augmenting bilingual sentence pairs with part-of-speech tags as linguistic constraints for word alignments.

A detailed description of the popular translation models IBM-1 to IBM-5 [6], as well as the Hidden-Markov alignment model (HMM) [41] can be found in [31].

The EM-algorithm [13] is used to train the free lexicon parameters p (e|n).

2.4.4 Problems in Word Alignment

The initial assumption for word alignment is that we have a sentence aligned parallel corpus of two languages. Now, given a parallel sentence pair, we can link (align) words that are translations of one another. There may be a large number of possible alignments, but we need to find the best alignment as shown below:

Ram is a good boy

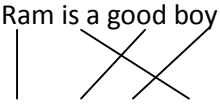


Fig 2.1: An example of word alignment on English-Nepali parallel sentence

In approaches based on IBM models, the problem of word alignment is divided into several different problems. The first problem is to find the most likely translations of an SL word, irrespective of positions. This part is taken care of by the translation model. The model alone has many applications. For example, since this model gives probable of word translations, we can use this model to make the task of building a bilingual dictionary easier. The second problem is to align positions in the SL sentence with positions in the TL sentence. This problem is addressed by the distortion model. It takes care of the differences in word orders of the two languages. The third problem is to find out how many TL words are generated by one SL word. Note that an SL word may sometimes generate no TL word, or a TL word may be generated by no SL word (NULL insertion). The fertility model is supposed to account for this. The first three models corresponding to these problems form the core of the IBM model based generative SMT. Examples of these are shown in Figure-4.

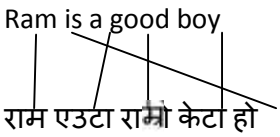
Unlike European languages, most of the Indian languages are morphologically rich and have the feature of compounding, thereby making the problem different in terms of SMT. When we are trying to align two European languages, we are much more likely to get one-to-one alignments, but when at least one of the languages is an Indian language, this is less likely. In other words, the problem is much harder for the fertility model. One-to-many or many-to-one translations are much more likely and so is NULL insertion.

Since English is an SVO language and Indian languages are SOV with respect to the word order, alignment of word positions may also be more difficult when one language in an Indian language and the other is a European language, like English. This will make the task of the distortion model harder. But this will not be a problem if both the languages are Indian languages.

Apart from compounding, tense, aspect and modality (TAM) of Indian language verbs also are a cause of errors in alignment. This is because the TAM information is distributed over several words, which causes problems for the fertility model. This is, in fact, one of major factors in reducing the alignment accuracy.

However, there are some aspects which, if used properly, may allow us to get good accuracy with approached bases on IBM models. As mentioned earlier, Indian languages have a lot of borrowed

and inherited words which are common to more than one language. Using a list of cognates or aligning cognates on the fly using better techniques like the ones used on the [5, 1], we can increase the accuracy of alignment. If a bilingual dictionary is available, we can use that to initialize the EM algorithm.Ex-1. Translation (one to one alignment):



Ex-2. Distortion (word order) and NULL insertion ('spurious' words):

Ex-3. Fertility:

Fig 2.2: Problems in word alignment which the first three IBM models try to solve

2.4.5 Summarization

The baseline alignment model does not allow a source word to be aligned with two or more target words. Therefore the resulting Viterbi alignment of the standard alignment models has a systematic loss in recall. Och and Ney in [32] describe various methods for performing a symmetrization of their directed statistical alignment models by applying a heuristic post processing step that combines the alignments in both translation directions (source to target, target to source). To solve this problem, they compute two Viterbi alignment A_1 and A_2 for both direction and combine (symmetrize) A_1 and A_2 in to one alignment matrix A using the various combination methods like intersection, union etc.

2.4.6 Phrase Alignment

In phrase-based models the unit of translation may be any contiguous sequence of words, called a *phrase.* Null alignment and fertility is not considered in the phrase based model. Each source phrase

is nonempty and translates to exactly one nonempty target phrase. The intuition of the phrase based SMT is to use phrases as well as single words as the functional units of the translation.

One major disadvantage of single-word based approaches is that contextual information is not taken into account. The lexicon probabilities are based only on single words. For many words, the translation depends heavily on the surrounding words. In the single-word based translation approach, this disambiguation is addressed by the language model only, which is often not capable of doing this.

One way to incorporate the context into the translation model is to learn translations for whole phrases instead of single words. So, the basic idea of phrase-based translation is to segment the given source sentence into phrases, then translate each phrase and finally compose the target sentence from these phrase translations.

It is straightforward to extend word alignment to phrase alignment: two phrases are said to be aligned if their words align. Deriving phrase pairs from word alignment is now widely used in phrase-based SMT. Parameters of statistical word alignment model are estimated from bitext, and the model is used to generate word alignments over the same bitext. Phrase pairs are extracted from the aligned bitext and used in the SMT system. With this approach, the quality of the underlying word alignments can exert a strong influence on phrase based SMT system performance. The common practice therefore is to extract phrase pairs from the best attainable word alignments. Currently, IBM model-4 alignments [6] as produced by GIZA++ [31] are often the best that can be obtained, especially with the large bitext.

Given a pair of source and target language sentence it needs to determine which phrase in the source language sentence is translated by which phrase in the target language sentence. Such a mapping is called phrase alignment. It is easy to compute the phrase alignment probability, if we have a large training set with each pair of sentences labeled with phrase alignment. We could just count the number of times each phrase pair occurred and normalize it to get the appropriate probabilities. The author in [31] the experiment is done in French-English parallel corpus by applying the following formula.

$$P(\bar{f},\bar{e}) = \frac{count(\bar{f},\bar{e})}{\sum_{F} count(f,\bar{e})}$$

Phrase pair $(\bar{f},\bar{e})$, together with its probability $P(\bar{f},\bar{e})$, is stored in a large phrase translation table.

We don’t have large hand-labeled phrase alignment training set; so we can extract aligned phrases from the low level alignment called the word alignment.

Many natural language phenomena go beyond single-word dependencies i.e. a Nepali compound word may translate by two or more English words. Similar problems occur in the case of proverbs, which typically have a completely different translation. In addition, the translation of prepositions, articles and particles strongly depends on the context.

The term phrase is being used in different ways with regards to alignment, denoting both linguistic phrases and word n-grams of varying length.

The system somehow has to learn which phrases are translations of each other. Most of the system uses the following approach: first, they train statistical alignment models using GIZA++ and compute the Viterbi word alignment of the training corpus. This is done for both translation directions. They combine both alignments using a refined heuristic (intersection, unions) to obtain a symmetrized word alignment matrix. This alignment matrix is the starting point for the phrase extraction. The following criterion defines the set of bilingual phrases of the sentence pair and the alignment matrix ‘A’ subset of J ×I that is used in the translation system.

2.4.7 The Alignment Template Approach

A variant of phrase-based models is the *alignment template model* [30, 32]. In this model explicit phrase-to-phrase translations are not used. Instead, each phrase is first associated with an alignment template, which is a reordering of the words in the phrase based on word categories rather than specific word identities. The words in the phrase are then translated using word-to-word translation, and the alignment templates are reordered as in phrase-based models.

The statistical machine-translation method described in [30] is based on a word aligned training corpus and thereby makes use of single word based alignment models.

A general deficiency of the baseline alignment model is that they are only able to model correspondences between single words. A more systematic approach is to consider whole phrases rather than single words as the basis for the alignment models. In other words, a whole group of adjacent words in the source sentence may be aligned with the whole group of adjacent words in the target language. The key element of this approach is the *alignment templates* which are pairs of phrases together with an alignment between the words within the phrases. An alignment template z

is a triple $(\overline{S},\overline{T},\overline{A})$ which describes the alignment $\overline{A}$ between a source class sequence $\overline{S}$ and a target class sequence $\overline{T}$. The alignment $\overline{A}$ is represented as a matrix with binary values. Matrix element with value 1 means that the words at the corresponding positions are aligned and the value 0 means that the words are not aligned. If a source word is not aligned to a target word then it is aligned to the empty target word which shall be at the imaginary position i = 0. This alignment representation is generalized of the baseline alignments described in [6] and allows for many-to-many alignments. For more detail about the alignment template approach see [30]. The advantage of the alignment template approach over word based statistical translation models is that word context and local re-orderings are explicitly taken into account. It is observed that this approach produces better translations than the single-word based models. The alignment templates are automatically trained using a parallel training corpus.

2.4.8 Syntax-based Models

The statistical MT systems (like IBM models 1,2,3) are based on word alignment where words are used as an elementary units. And the phrase-based systems improve on these word-based systems by using larger units. Thus capturing larger contexts and providing more natural unit for representing language divergences.

Some approaches in MT has focused on ways to move even further up the Vaquois hierarchy from simple phrases to larger and hierarchical syntactic structures. These approaches attempt to assign a parallel syntactic structure to a pair of sentences in different languages with a goal of translating the sentences by applying reordering operation on the trees.

The mathematical structure of these parallel structures is known as transduction grammar also called synchronous grammar which is a generalization of finite state transducers. Most of which are generalization of context free grammars to the two language situations. Example is an ITG (Inversion Transduction Grammar). Each non-terminal generates two separate strings.

N → watch|घड़ि

PP → <POP NN>

This angle bracket means the right hand side symbols are in both order.

i.e. PP → POP NN|NN POP

In terms of the motivations, syntax based models have a grammar-channel view of the parallel data. The models range from the simplest one such as Bilingual Bracketing [46] to more complicated ones including tree-to-tree and tree-to-string models in [48]. Yamada et al [48] modeled the translation process from a parse tree of source language into target language sentence. But the computational complexity during alignment is very high because it must handle hierarchical structure. Some methods use parsed sentences in parallel sentence aligned corpora to extract transfer rules or examples [43]. Watanabe et al. in [43] describe a system and methods for finding structural correspondences from the paired dependency structures of a source sentence and its translation in a target language. The system they have developed finds word correspondences first, and then finds phrasal correspondences based on word correspondences.

However parse-to-parse matching, which regards parsing and alignment as two separate and successive procedures, suffers from grammatical inconsistency between languages. Moreover, it costs a lot to develop parsers of both the source and target language. To deal with the difficulties in parse-to-parse matching, Wu [46] utilizes inversion transduction grammar (ITG) for bilingual parsing. Bilingual parsing approach looks upon the parsing and alignment as a single procedure which simultaneously encodes both parsing and transferring information. It is however difficult to write broad coverage 'bilingual grammar' for bilingual parsing. The simple bilingual bracketing grammar works well in many cases of word alignments and as well as word reordering constraints in decoding algorithms.

2.4.9 Linguistic alignment methods

Although initially SMT systems did not incorporate any linguistic analysis and worked at the surface level of word forms, an increasing number of research efforts are introducing a certain degree of linguistic knowledge into their statistical framework. Linguistic alignment methods are basically highly dependent on linguistic knowledge such as cognates [22] and external knowledge base such as machine readable dictionaries or glossaries [28]. These methods perform highly for related languages that share the same vocabulary to a large extent. In cases where cognates are insignificant subset of the languages considered, they certainly would fail. Kupiec [24] aligned noun phrases in English and French bilingual corpus using part-of-speech tagger for each half of the texts.

Cherry and Lin use syntactic analysis to guide and restrict the word alignment process [36]. The advantage of using available syntactic information for word alignment is that it helps to overcome data sparseness problem. Although a token (word) may be rare, its syntactic category may not and hence there may be sufficient statistical information to align at the phrase level. Subsequently, the

phrase level information can be used to compute alignments for the tokens within the aligned phrases. The syntactic function of a token as modifier head etc. can equally simplify and guide the alignment process considerably. However it is unclear whether such an approach performs well for language pairs where syntactic and functional differences are greater than between for example English and French.

Shortcomings of the statistical approach [36] word alignment quality suffers due to three problematic phenomena: the amount of rare words typically found in corpora, word order differences between the two languages to be aligned and existence of multiword units.

Morphology: Morphological analyzers help to distinguish between lexical components of words which are accountable for the semantics of the words and grammatical words. During text alignment this helps to align grammatical components and lexical components separately. Hence the use of effective morphological analyzers can particularly be useful to assist statistical alignment systems. Specially, this is true for languages that are highly inflected.

Part of Speech: Part of speech information is important during text alignment and later for disambiguating ambiguous translations. In translation sentences if a word with a certain part of speech is recognized, then one can project that the words with other part of speech may not be its translation. The use of POS information for improving statistical alignment quality of the HMM-based model is described in [40], where they introduce additional lexicon probability for POS tags in both languages, but actually are not going beyond full forms.

Syntax: One approach of aligning words/phrases of a source sentence to that of the target is by projecting the parse tree of one to the other. When such tools are found in both languages successful alignments can be obtained to a large extent [48].

Dictionaries: Dictionary based alignment algorithms try to recognize words in sentences as given in dictionaries [27, 28]. Such alignment methods may not necessarily depend on complete dictionaries but with some entries that would allow recognition of anchor words which would reduce the number of remaining alignments to be made.

Cognates: Cognates in historically related languages have been used to align texts [27, 28]. This alignment method has produced good results on highly related languages. Melamed in [28] argues if orthographic cognates cannot be recognized on languages that do have different writing system

phonetic cognates can be used. But again how exhaustive these cognates are in comparison with word types in a language is a question that may not have a positive answer.

2.4.10 Chunk Level Alignment

Extracting chunk pairs is an alignment problem that falls somewhere between word alignment and sentence alignment. It incorporates and extends well-established techniques for bitext sentence alignment, with the aim of aligning text at the sub-sentence level. There is a practical benefit of doing this. Shorter bitext segments can lead to quicker training of MT systems since MT training tends to run faster on shorter sentences. There is also the evidence that word alignment achieved over chunks pairs aligned at the sub-sentence level produce better results than word alignment over sentence pairs.

Many approaches have been proposed to align chunk pairs with in bitext. Watanabe et al [44] describes an alternative translation model based on a text chunk under the framework for Statistical Machine Translation. The translation model suggested by them first perform chunking i.e. a source sentence is first chunked and then each chunk is translated in to target language with local word alignment. And next, translated chunks are reordered to match the target language constrains. They perform the chunk alignment by performing the local alignment in side the chunks.

Wang and Zhou [42] proposed a statistical method called bilingual chunking for structure alignment in the sentence pair. They perform alignment on chunks rather than alignment in the hierarchical structure like sub trees in other approaches. This model for structure alignment comprises three integrated components: Chunking model of both languages and the *crossing constraint*; which uses chunk as the structure. The *crossing constraint* requests a chunk in one language only correspond to at most one chunk in the other language. According to Wu [46], crossing constraint can be defined as:

For non-recursive phrases: Suppose two words w1 and w2 in language-1 correspond to two words v1 and v2 in language-2, respectively, and w1 and w2 belong to the same phrase of language-1. Then v1 and v2 must also belong to the same phrase of language-2.

The alignment is finished through a simultaneous bilingual chunking algorithm. Using the constrains of chunk correspondence between source language and target language, their algorithm reduces the search space, support time synchronous dynamic programming algorithm, and lead to highly consistent chunking.

Hwang et al. [20] describe the chunk alignment algorithm for Japanese and Korean sentences. The algorithm on chunk alignment module consists of 3 steps: first, word alignment based on bilingual dictionaries; Second, statistical chunk alignment constrained by previous word alignment results and chunk boundaries of the source language; and third, statistical word alignment in the aligned chunk pairs.

Aiming to overcome the shortcomings of word-based methods, some alignment algorithm based on phrases, chunks or structures have been proposed [30, 46, 42]. These modifications are advantageous and lead to more fluent translations since chunk-based translations capture local reordering phenomena. However, these alignment algorithms have been based on complex syntax information. For example, by incorporating parsing technology with crossing constraints or have been narrowly focused on certain special kinds of phrases. These methods have proven to yield poor performance when dealing with long sentences. Further, the methods heavily depend on the performance of the associated tools, such as parsers, part-of-speech taggers. In order to address these shortcomings effectively, Zhou et al [50] purpose a new algorithm called multi-layer filtering (MLF) for automatically aligning bilingual chunks. They use this algorithm on the Chinese-English parallel corpus. Multiple layers are used to extract bilingual chunks according to different feature of chunks in the bilingual corpus. And the alignment chunks are one-to-one corresponding with each other. The chunking and alignment algorithm described in this paper does not rely on the information from tagging, parsing, syntax analyzing or segmenting for the Chinese corpus.

2.5 POS Tagging and its Approaches

Parts of Speech (POS) tagging is a process of assigning accurate syntactic categories (noun, verb, adjective etc.) to every word in the text and plays fundamental role in various Natural Language Processing (NLP) application such as speech recognition, information extraction machine translation and word sense disambiguation etc. [21]. POS tagging particularly plays very important role in word-free languages because such languages have relatively complex morphological structure of sentences than other languages. Indic languages including Nepali and Urdu are good candidate examples of such word-free languages.

There are mainly two approaches of POS tagging.

2.5.1 Rule Based Approach

The basic principle of rule based approaches is that the knowledge base consists of a set of linguistic generalizations, known most commonly as rules or constraints. Each rule contain the instructions for an operation to be performed, and context describing the where the rule should be applied. And these rules are responsible to provide the appropriate tags to the text.

2.5.2 Probabilistic Approach

Probabilistic approaches rely on the statistical information concerning the frequency with which the sequences of tags occur is gathered from long stretches of running text. The knowledge base for tagging the text is generated by training the tagged corpus. The algorithms such as Hidden Markov Model, N-gram models etc. are used in probabilistic approaches for tagging the given text. The accuracy of probabilistic approach of tagging is mainly depends on the training corpus. The more contexts and more the data in the training corpus will increase the accuracy of the probabilistic tagger.

2.6 Shallow Parsing or Chunking

Shallow Parsing is a natural language processing technique that attempts to provide some machine understanding of the structure of a sentence, but without parsing it fully into a parsed tree form. Shallow parsing is also called partial parsing, and most often refers to the task of chunking. Chunking is a process of dividing a text in syntactically correlated parts of words. It is an intermediate step towards the full parting. Text chunking is useful preprocessing step for parsing. Text chunking consists of dividing a text into phrases in such a way that syntactically related words become member of the same phrase. These phrases are non-overlapping which means that one word can only be a member of one chunk [39].

Steven Abney pioneered the notion of chunk parsing. According to him [1] chunking is a kind of shallow syntactic analysis where certain sequences of words in a sentence are identified as forming phrases of various types, such as noun phrases, verb phrases, adjective phrases etc. He begins to explore his idea by giving the following example of bracketing the sentence in to the appropriate chunks.

[I begin] [with an intuition]: [when I read] [a sentence], [I read it] [a chunk] [at a time]

The output of a chunker is a division of the text's sentences into series of words that together consists a grammatical unit (mostly noun, verb, or preposition phrase, with less-frequent occurrences of adverb, adjective, clause, and other phrases). The output is different from that of a fully parsed tree because it consists of series of words that do not overlap and that do not contain each other. This makes chunking an easier Natural Language Processing task than parsing. It can be said that chunking is a middle step between identifying the part of speech of individual words in a sentence, and providing a full parsed tree of it.

It can be argued that chunking actually reflects aspects in human language processing. The following psycholinguistic evidence or arguments can be given:

- When a person speed-reads through a text, he or she do not fully parse the sentence but instead look for "key phrases". These key phrases are actually chunks, which mean that chunking is the machine analogy for speed reading.
- When a speaker of a language utters a sentence, the uttered sentence's prosody features, intonation changes and pauses, often match the edges of chunks as they would have been output by a shallow parser.

Since human translator understands the source sentence taking a chunk at a time and generates the corresponding chunk in another language by translating whole phrase rather than translating word by word, we can take a chunk as a unit of machine translation. Hence, chunking is useful for Machine Translation. Besides this chunking can be useful for information retrieval, information extraction, and question answering, since a complete chunk (Noun, Verb or Preposition Phrase) is likely to be semantically relevant for the requested information.

There are mainly two approaches on chunking.

2.6.1 Rule-based Approach

Chunker based on rule-based approach mainly relies on the hand-crafted patterns of deciding the chunk boundaries in a sentence. A simple context-free grammar is quite adequate to describe the structure of chunks. The rules may be the regular expression based on POS tags which will capture the syntactic chunks in a POS tagged sentence. Most of the approaches of chunking take an input text that has been tagged by a part-of-speech tagger. Hence the performance of chunker depends on the performance of POS tagger. Madan Puraskar Pustakalaya develops a rule based chunker which is described in [34].

2.6.2 Statistical Approach

These approaches are based on machine learning techniques. In this method chunker uses a large corpus of text that has been hand-chunked and uses some machine learning algorithm to extract the chunks from the test data. Here the training data will be the chunk annotated corpus. N-grams or Hidden Markov models can be used as an algorithm for the chunking purpose.

CHAPTER III

IMPLEMENTATION

This chapter describes the implementation detail and the different tools used for the development of the chunk alignment model. The Java Programming language has been used to implement the chunker and the aligner of the model. The TnT tagger is used as a tagging tool.

3.1 Specification of the Model

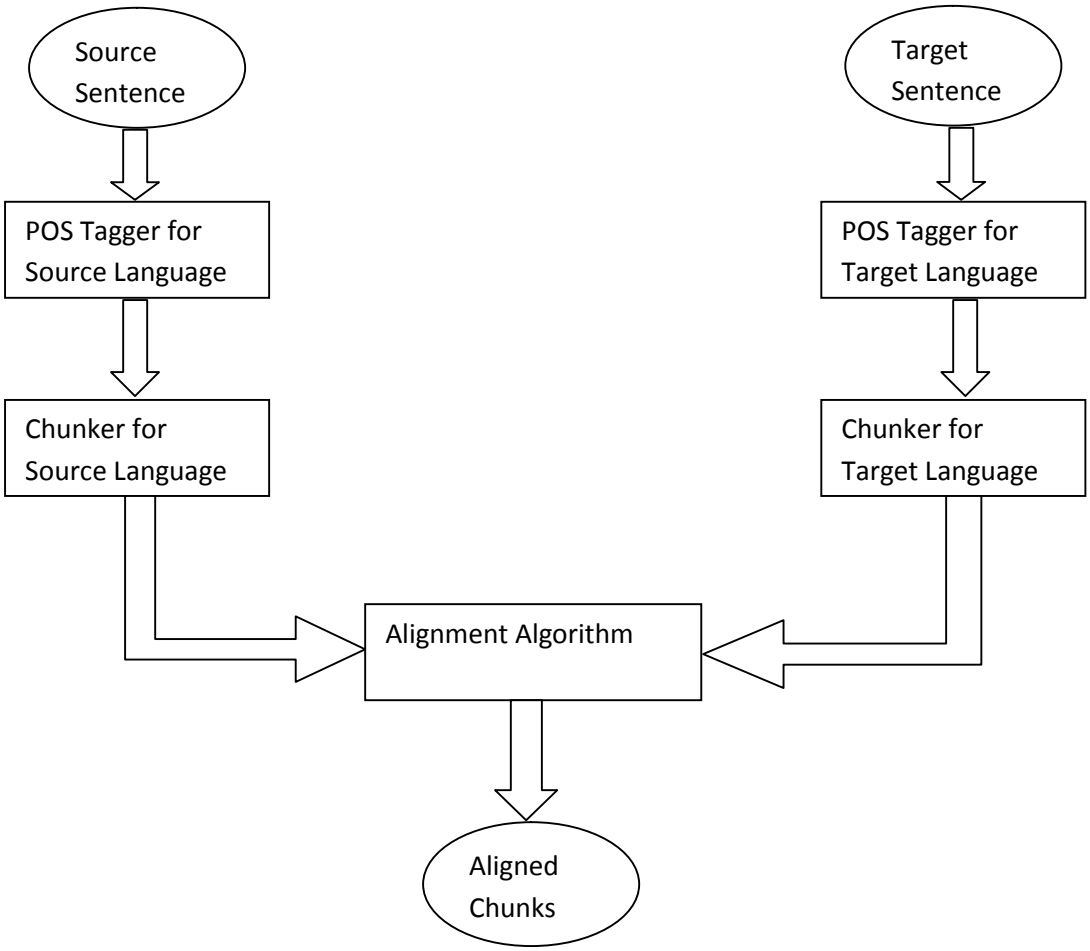


Figure 3.1: The Architecture of the Chunk Alignment Model

3.2 Description of the Model

The total work is mainly divided into the series of steps of tagging, chunking and alignment of the chunk annotated sentence pair. The model first takes a sentence pair on both languages and feeds into the tagger used by the model. Here TnT tagger [38] is used as a POS tagger for providing the part-of-speech category to the every words of the sentence. The chunker will then used for bracketing the words of a sentence and provide them a chunk category based on the chunk rules. The chunk rules are the regular expressions in which, the symbols are POS tags of the words associated with the words of the sentence. The chunker here used is thus a regular expression based chunker. The aligner will then align the chunks of both sentences based on the knowledge base gathered from the training of the bilingual corpus and the different heuristics of the linguistics.

3.3 Parallel Bilingual corpus

In this section, we describe the data that we used to test and train the model described above. The Monolingual tagged Nepali corpus, provided from Madan Puraskar Pustakalaya, is used to train the TnT tagger for Nepali text. The corpus includes 111,264 words with the annotation of different tags associated with each word of the corpus and it includes 43 different number of POS tags. The tag set used in this corpus is used as a tag set for our model described above. The Wall Street Journal (WSJ) corpus included with TnT tagger is used to train the tagger for English text. It uses Penn Treebank tagset (45 tags) and included about 1.2 millions of tokens. The description of the tag sets used will be described on the next chapter.

3.4 Tagging

Tagging is the process of association of word category (tag) to the word of a sentence. The TnT POS tagger is used for tagging the text in both languages with different tagset.

3.4.1 TnT POS Tagger

TnT (Trigrams'n'Tags) [38] is a very efficient statistical part-of-speech tagger that is trainable on different languages and virtually any tag set. The component for parameter generation gets trained on the POS tagged corpora. The system incorporates several methods for smoothing and of handling unknown words. TnT is not incorporated for particular language. Instead, it is optimized for training on a large variety of corpora.

The tagger is implementation of the Viterbi algorithm for second order Markov model for part-of-speech tagging. The states of the model represents tags, output represents the words. Transition probabilities depend on the states, thus pair of tags. Output probabilities only depend on the most recent category. For calculating the tags of i^{th} word, the tagger calculates based on the following formula:

$$P(t_i \mid w_i) = \arg\max_t P(t_i \mid t_{i-1,} t_{i-2}) P(w_i \mid t_i)$$

Here, argmax operation maximizes the product of the probability of a tag pattern and the probability of a word getting a particular tag.

This tool is suitable for tagging any language which uses white spaces to separate words, like Nepali, Hindi, English, and French. In Nepali and English languages too, words are separated by white spaces, which makes TnT the best tools for the tagging of the Nepali and English language text. Besides the permission to use, copy, and modify this software and its documentation is granted to non-commercial entities for free. This is an important reason behind choosing this tool.

3.4.2 Specification and description of the tagset used for Nepali and English language

One of the crucial issues that needs to be subtly addressed while designing a POS tagset is its' size. Generally, the assumption is –"the smaller the tagset, the greater the accuracy". However, in saying so, the compulsory categories evident in the language would not be missed and at the same time also not necessarily increase the size of the tagset whenever economy can be maintained. Hence, a middle ground has been adopted while designing a POS tagset of any language.

The tagset for Nepali [34] currently includes 43 tags and covers almost all the grammatical categories in the Nepali language. By the reference of Penn Treebank tagset, the tagset of the Nepali is designed. The short description of tag set used here is given follow:

Category	**POS Tag ID No**	**POS Name**	**POS Tag**
Noun	1	Common Noun	NN
	2	Proper Noun	NNP
Pronoun	3	Personal Pronoun	PP
	4	Possessive Pronoun	PP$
	5	Reflexive Pronoun	PPR
	6	Marked Demonstrative	DM
	7	Unmarked Demonstrative	DUM
Verb	8	Finite Verb	VBF
	9	Auxiliary Verb	VBX
	10	Verb Infinitive	VBI
	11	Prospective Participle	VBNE
	12	Aspectual Participle	VBKO
	13	Other Participle Verb	VBO
Adjective	14	Normal/Unmarked	JJ
	15	Marked Adjective	JJM
	16	Degree Adjective	JJD
Adverb	17	Manner Adverb	RBM
	18	Other Adverb	RBO

Intensifier	19	Intensifier	INTF
Postpositions	20	Le-Postposition	PLE
	21	Lai-Postposition	PLAI
	22	Ko-Postposition	PKO
	23	Other Postpositions	POP
Conjunction	24	Coordinating	CC
	25	Subordinating Conjunction	CS
Interjection	26	Interjection	UH
Number	27	Cardinal Number	CD
	28	Ordinal Number	OD
Plural Marker	29	Plural Marker हरु	HRU
Question Word	30	Question Word	QW
Classifier	31	Classifier	CL
Particle	32	Particle	RP
Determiner	33	Determiner	DT
Unknown Word	34	Unknown Word	UNW
Foreign Word	35	Foreign Word	FW
Punctuation	36	sentence Final	YF
	37	sentence Medieval	YM
	38	Quotation	YQ
	39	Brackets	YB
Header List	41	Header List	ALPH

Symbol	42	Symbol	SYM
Abbreviation	43	Abbreviation	FB

Table 3.1: List of Part-of-Speech tags for Nepali Langauge

The detail description of Nepali tagset used here is found in [34].

The short description of the English POS tagset (Penn Treebank tagset) is given below:

POS tag ID No.	POS Tag Name	POS Category
1	CC	Coordinating Conjunction
2	CD	Cardinal Number
3	DT	Determiner
4	EX	Existential there
5	FW	Foreign Word
6	IN	Preposition or subordinating conjunction
7	JJ	Adjective
8	JJR	Adjective, Comparative
9	JJS	Adjective, Superlative
10	LS	List item marker
11	MD	Modal
12	NN	Noun, singular or mass
13	NNS	Noun, plural
14	NP	Proper noun singular

15	NPS	Proper noun plural
16	PDT	Predeterminer
17	POS	Possessive ending
18	PP	Personal pronoun
19	PP$	Possessive pronoun
20	RB	Adverb
21	RBR	Adverb, comparative
22	RBS	Adverb, superlative
23	RP	Particle
24	SYM	Symbol
25	TO	To
26	UH	Interjection
27	VB	Verb, base form
28	VBD	Verb, past tense
29	VBG	Verb, gerund or present participle
30	VBN	Verb, past participle
31	VBP	Verb, non-3rd person singular present
32	VBZ	Verb, 3rd person singular present
33	WDT	Wh-determiner
34	WP	Wh-pronoun
35	WP$	Possessive wh-pronoun
36	WRB	Wh-adverb

Table 3.2: List of Part-of-Speech tags for English Language

3.5 Chunking

Chunking involves finding the group of related words in a sentence and provides them a chunk category. A chunk can be defined as a collection of contiguous words such that the words inside a chunk have dependency relations among them, but only the head of the chunk has the dependency relation with the outside chunk. Hence a chunker is tool to identify such chunks in the given sentences. For the chunker we have defined a set of linguistic rules for chunking the Nepali as well as English phrases. The rules are on the form of regular expressions and the basic units of the chunk rule are the POS tag categories. Besides, we have devised a simple algorithm to find the chunks in the Nepali or English sentences on the basis of chunk rules. Chunk rules here are on the form of regular expression of POS tags to capture the matches of chunks on the given patterns. Bal et. al. in [34] give the chunking algorithm and the sample of chunk rules for Nepali language. The rules defined here for our model are somewhat different than defined in [34]. The following is the description of the algorithm applied for identifying the chunks in a sentence.

Let us say we have following n rules for defining n chunks:

Pattern 1→ chunk 1

Pattern 2→ chunk 2

…

…

…

Pattern n → chunk n

Now if a given sentence has a following pattern of POS tags of its respective lexemes:

T1 T2 T3 T4 T5…TK

Then we proceed the chunking as follow:

T1 T2 T3 T4 T5…TK-1 TK

Start End

Initially we mark the first token with the start pointer and the last token as the end pointer. Now if the pattern is between the start and end inclusive (i.e. T1 T2 T3 T4 T5 … TK) is there in the chunk rule then we will tag the whole pattern as a single chunk. If it is not there then we will decrease the end pointer to one token left and continue doing this until and unless the pattern between start and end is there in the chunk rule.

T1 T2 T3 T4 T5…TK-1 TK

Start End

Similarly by continuing the same process if the end pointer reached the T3 token

T1 T2 T3 T4 T5…TK-1 TK

Start End

and the pattern between start and end (i.e. T1 T2 T3) is there in the chunker rule then we chunk tokens between start and end pointer as a single chunk and shift the start pointer to one token right of end pointer and end pointer to the rightmost end tokens as follow

T1 T2 T3 T4 T5…TK-1 TK

Start End

We will continue this doing until the start pointer reaches the last token i.e. TK.

Following is the pseudo code for the chunking algorithm described above.

```
Flag=false;
Start = points to first token in the list;
End = points to the last token in the list;
While(start <=end )
{
        While(flag !=true)
        {
                pat = pattern between Start and End inclusive;
                For all patterns present in the rule
                {
                        If( pat is found there in the rule)
                        {
                                Flag=true;
                                Start=end+1;
                                End=points to the last token in the list;
                                Mark the pattern between Start and End as a single chunk;
                        }
                        else {
                                End=End - 1;
```

```
                                    }
                        }
                }
}
```

3.5.1 Chunk types and chunk rules

According to [34] there are 11 chunk types defined on the Nepali language. But here only 8 chunk types are defined and used for the purpose of simplicity and efficient implementation. The less the number of chunk category more the accuracy can be achieved on statistical approaches. The different types of verb chunk defined on [34] are merged into the single verb chunk category. The following is the short description of them.

NCH	Noun Chunk
VCH	Verb Chunk
AJCH	Adjective Chunk
AVCH	Adverb Chunk
ADVCH	Adverbial modifier Chunk
CNCH	Conjunction Chunk
CLCH	Classifier Chunk
PNCH	Punctuation Chunk

Table 3.3: List of Chunk categories used for both language

The chunking rules in the form of regular expression are as follows:

English Chunk Rules:

NCH:((IN)((PRP)|(PRP$)))|((IN)?(TO)?(PRP#)?((JJ)|(JJS))?((NN)|(NP)|(NNS)))

NCH:(IN)?(DT)|((TO)?(PRP))|((JJS)(IN)(DT)(NN))

NCH:(IN)?(DT)?(((CD)?((NN)|(NNS)))|(NP)|(NNPS))

NCH:(IN)(DT)(JJ)?((CD)|(NN)|(NNS))|((IN)(CD))

NCH:((CD)+(CC)(CD)+)|(CD)+

NCH:((TO)?(NN)(CD)?)|(NP)

NCH:(NN)(IN)(NP)

NCH:(DT)?(JJ)?((NP)|(NN))

VCH:((VV)|(VB)|(VVD)|(VBD)|(VVN)|(VHP)|(VHZ)|(VVG))|((VV)(RP)?)

VCH:((VBD)(VVN)|(VVD))|((MD)(RB)?((VB)|(VV)))|((VHZ)(VBN)?(VVN))

VCH:(((VHP)|(VHZ))(RB)?(VVN))|((VBP)|(VBZ)|(VVP)|(VVZ))

VCH:(((VBZ)|(VBP))(RB)?(((VBG)|(VVG))|((VBN)|(VVN))))

VCH:(TO)((VV)|(VB))

PNCH:(SENT)|(SYM)

AJCH:(JJ)|((RB)(JJ))

AVCH:((RB)|(RBR)|(RBS))|((IN)(CD)?(RB))|(EX)

CNCH:(CC)

Nepali Chunk Rules:

NCH:(DUM)|(PP)|((PP)(POP))|(((DUM)|(DT))(NN))

NCH:(((DUM)|(DT))(NN))((PKO)|(PLE)|(PLAI)|(POP))?

NCH:((DUM)|(DM))|(NN)((PKO)|(PLE)|(PLAI)|(POP))?

NCH:((DM)(POP))

NCH:((NN)|(NNP))((PKO)|(PLE)|(PLAI)|(POP))?

NCH:((CD)|(OD))((NN)|(NNP))

NCH:((NN)(HRU))|((NN)(POP))

NCH:(PP$)(NN)

NCH:((JJM)|(JJ)|(JJD))(NN)

VCH:(VBX)|(VBI)|(VBNE)|(VBF)

VCH:(VBO)|(VBKO)|((VBI)(VBF))

VCH:(VBO)|((VBKO)?(VBX))|((VBO)(VBKO))

AJCH:((INTF)?(JJ))|((JJ)(RBM))

AVCH:(RBM)?(INTF)?((RBM)|((VBO)?))

AVCH:(NN)?((POP)|((VBI)(POP))|((VBKO)(POP)))

CNCH:(CC)|(CS)

CLCH:(CL)(POP)?

PNCH:(YF)

3.6 Alignment

Based on the bilingual sentence aligned training corpus, the chunk annotated sentence pair as test data can be aligned at chunk level. Among the different alignment between the chunks of the given sentence pair, the best alignment will be obtained. The word translation probability table is found from running the training corpus by EM algorithm.

3.6.1 Expectation Maximization (EM) Algorithm for Training

For calculating the parameters mentioned above (parameters of chunk alignment), we use a generative algorithm called Expectation Maximization (EM) for training. The EM algorithm guarantees an increase in likelihood of the model in each iteration, i.e., it is guaranteed to converge to a maximum likelihood estimate.

A set of sentence aligned parallel corpus is used as the training data. Let the number of sentence pairs in the training data be ***N*** and the lengths of the source and target sentences be ***l*** and ***m***, respectively. The translation parameter ***T*** is learned during training using expected translation counts ***tc***. Let the number of iterations during training be ***n***. Then, the iterative EM algorithm corresponding to the translation problem can be described as:

Step-1: Collect all word types from the source and target corpora. For each source word ***e*** collect all target words ***n*** that co-occur at least once with ***e***.

Step-2: Initialize the translation parameter uniformly (uniform probability distribution), i.e., any target word probably can be the translation of a source word ***e***.

T(n | e) = 1/(number of co-occurring target words)

Step-3: Iteratively refine the translation probabilities until values are good enough

for n iterations do

Initialize the expected translation count tc to 0

for each sentence pair (e, n) of lengths l, m do

update the expected translation count

for j=1 to m do

set total to 0

for i=1 to l do

total += T(nj|ei)

for i=1 to l do

tc(nj|ei) += T(nj|ei)/total

end for

end for

end for

end for

re-estimate the translation parameter values

for each source word e do

set total to 0

for each target word f do

total += tc(nj|ei)

for each target word f do

calculate T(nj|ei)= tc(nj|ei)/total

end for

end for

end for

end for

After the training we will have translation probability values for source and target words. Since, in IBM model theory, T***(nj | ei)*** is assumed to be independent from T***(nj'|ei')***, we can find the best alignment by looking at the individual translation probability values. The best alignment can be calculated in a quadratic number of steps equal to ***l ×m***.

3.6.2 Alignment Algorithm

The translation probability can be decomposed as:

$$P(e \mid n) = \sum_{a} P(e, a \mid n)$$

$$P(e_1^m \mid n_1^n) = \sum_{a_1^m} P(e_1^m, a_1^m \mid n_1^n)$$

or,

For given English sentence (e) and Nepali sentence (n), the most probable alignment is

$$\hat{a} = \arg\max_{a}(P(a \mid e, n)$$)

where $P(a \mid e, n) = \frac{P(a, e \mid n)}{\sum_{a} P(a, e \mid n)}$

The term $\sum_{a} P(a, e \mid n)$ will be same for each alignment so this can be removed

IBM model-1 is used to formulize the parameter of the model.

$$P(a, e \mid n) = P(a \mid n) * P(e \mid a, n)$$

The assumption of the IBM model-1 is: each words/chunks can be chosen with uniform probability $P(a \mid n) = \frac{1}{n^m}$, since there are n^m possible alignments. All alignments are equally likely; hence the $P(a \mid n)$ will be constant for every alignment. P (a,e|n) is now only depends on the lexicon probability P(e|a,n).

Suppose we have already know the length of the English sentence 'm' and the alignment 'a' as well as the Nepali sentence 'n', the probability of the English sentence would be,

$$P(e \mid a, n) = \prod_{j=1}^{m} P(e_j \mid n_{a_j})$$

Where e_j and n_{a_j} are the English and Nepali Chunk respectively.

Hence the best alignment can be defined as:

$$\hat{a} = \arg\max_{a} \quad P(a \mid e, n)$$
$$= \arg\max_{a} \quad P(a, e \mid n)$$
$$= \arg\max_{a} \quad P(a \mid n) * P(e \mid a, n)$$
$$= \arg\max_{a} \quad \frac{1}{n^m} * \prod_{j=1}^{m} P(e_j \mid n_{a_j})$$
$$= \arg\max_{a_j} \quad P(e_j \mid n_{a_j}), \qquad 1 \le j \le m$$

The advantage of IBM Model-1 is it does not need to iterate over all alignments. It is easy to figure out what the best alignment is for a pair of sentences (the one with the highest $P(a \mid e, n)$ or $P(a, e \mid n)$). This can be done without iterating over all alignments.

Here, $P(a \mid e, n)$ is computed in terms of $P(a, e \mid n)$. In model-1 $P(a, e \mid n)$ have 'm' factors, one of each English word/chunk. Each factor looks like $P(e_j \mid n_{a_j})$. Suppose that the English word/chunk e_2 would rather connect to n_3 that n_4, i.e. $P(e_2|n_3) > P(e_2|n_4)$. That means if we are given two alignments which are identical except for the choice of connection for e_2, then we should prefer the one that connects e_2 to n_3; no matter what else is going on in the alignments.

For 1<= j <= m

$$a_j = \arg\max_{i} P(e_j \mid n_i)$$

That's the best alignment, and it can be computed in a quadratic number of operations (n*m).

The lexicon probability (here in our model chunk translation probability) can be calculated in terms of lower level word translation probability.

$$P(e_c \mid n_c) = \arg\max_{a} \prod_{p=1}^{K} t(e'_p \mid n'_{a_p})$$

Where, K is the length of the English chunk in terms of number of words.

The word translation table is generated from the training of bilingual corpus by EM algorithm.

The alignment algorithm described above can be summarized as:

Let $e = e_{c_1} ... e_{c_i} ... e_{c_m}$ and $n = n_{c_1} ... n_{c_j} ... n_{c_n}$, where e_{c_i} is English Chunk and n_{c_j} is Nepali chunk and assume,

$e_{c_i} = e_{w_1} ... e_{w_p} ... e_{w_K}$,
$n_{c_j} = n_{w_1} ... n_{w_q} ... n_{w_l}$

For each English chunk e_{c_i} of English sentence

For each Nepali chunk n_{c_j} of Nepali sentence

For each word e_{w_p} in English Chunk

For each word n_{w_q} in Nepali chunk

Find the most probable word pair with greatest probability
$t(e_{w_p} \mid n_{w_q})$

End For

End For

Find the most probable chunk pair with highest probability

$$P(e_{c_i} \mid n_{c_j}) = \prod_{p=1}^{k} t(e_{w_p} \mid n_{w_q})$$

End For

End For

The detail description of the different phases of implementation can be described by the following diagram:

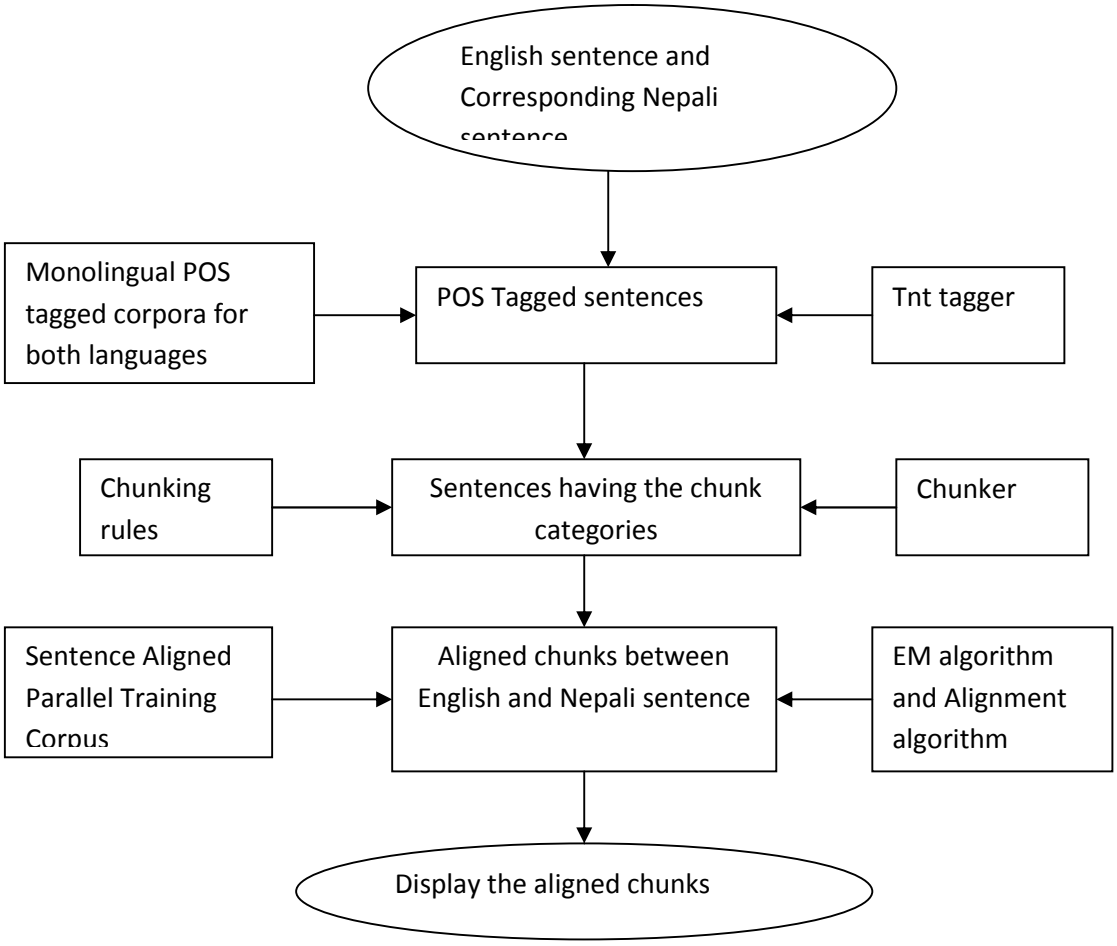


Figure 3.2: Phases of Implementation

CHAPTER IV

TESTINGH AND ANALYSIS

In this chapter we will measure the accuracy of the model in terms of different statistical tools. The alignment accuracy not only depends on the performance of aligner of the model but also it depends on performance of other former tools chunker and tagger. The evaluation of the alignment given by the model is done by comparing the alignments prepared by the human annotators. Here we use some sentences as a test data for the evaluation of the model. And a bilingual sentence aligned corpus is used as a training data.

We use the same evaluation criterion as described by Och and Ney in [31]. In this paper the alignment evaluation is done in word level but here we use same criterion in the chunk level alignment. The generated chunk alignment is compared to a reference alignment which is produced by human experts.

The result is evaluated in terms of alignment precision and recall, as defined in the following:

$$Alignment\ \mathrm{Pr}ecision = \frac{\#chunks\ correctly\ aligned}{\#chunks\ aligned}$$

$$Alignment\ \mathrm{Re}call = \frac{\#chunks\ correctly\ aligned}{\#chunks\ should\ be\ aligned}$$

Precision is the number of correct results divided by the number of all returned results and *Recall* is the number of correct results divided by the number of results that should have been returned. The F_1 score can be interpreted as a weighted average of the precision and recall, where an F_1 score reaches its best value at 1 and worst score at 0.

The F-measure (**F_1 score**) is the harmonic mean of precision and recall:

$$F-measure = \frac{2 \times recall \times precision}{recall + precsion}$$

$$AER = 1 - F-measure$$

The analysis of the alignment is done in terms of above metrics and again these measurements are also dependent on the tagging and chunking process.

3.1 Training and Test corpus

The sentence aligned parallel corpora used as a training corpus for training the model can be shown in Appendix A. It consists of simple sentences and very small in size.

The test corpus has been generated from the words defined on the training corpus to reduce the problem of unknown words. This can be shown in Appendix B.

The given sentence pair is first tagged by the TnT tagger and then chunk categories are defined for the group of words based on the rule defined on the model. And finally the alignment is done between the chunks of the both sentence pair. The output of the program for some test cases is shown below:

3.2 Input/output of the Program

English Sentence: This book is on the table.

Nepali Sentence: यो किताब टेबुल मा छ ।

English Chunked Sentence: (this/DT book/NN)/NCH (is/VBZ)/VCH (on/IN the/DT table/NN)/NCH (./SYM)/PNCH

Nepali Chunked Sentence: (यो/DUM किताब/NN)/NCH (टेबुल/NN मा/POP)/NCH (छ/VBX)/VCH (।/YF)/PNCH

Alignment:

this book यो किताब

on the table टेबुल मा

is छ

English Sentence: boys are going to school.

Nepali Sentence: केटा हरु स्कुल गईरहेका छन्।

English Chunked Sentence: (boys/NNS)/NCH (are/VBP going/VBG)/VCH (to/TO school/NN)/NCH (./SYM)/PNCH

Nepali Chunked Sentence: (केटा/NN हरु/HRU)/NCH (स्कुल/NN)/NCH (गईरहेका/VBKO छन्/VBX)/VCH (।/YF)/PNCH

Alignment:

to school स्कुल

boys केटा हरु

are going गईरहेका छन्

English Sentence: this school has a good ground.

Nepali Sentence: यो स्कुल सँग रामो चउर छ ।

English Chunked Sentence: (this/DT school/NN)/NCH (has/VBZ)/VCH (a/DT good/JJ ground/NN)/NCH (./SYM)/PNCH

Nepali Chunked Sentence: (यो/DUM स्कुल/NN सँग/POP)/NCH (रामो/JJM चउर/NN)/NCH (छ/VBX)/VCH (I/YF)/PNCH

Alignment:

has छ

a good ground रामो चउर

this school यो स्कुल सँग

English Sentence: a boy is sitting on the table.

Nepali Sentence: एक केटा टेबुल मा बसिरहेको छ।

English Chunked Sentence: (a/DT boy/NN)/NCH (is/VBZ sitting/VBG)/VCH (on/IN the/DT table/NN)/NCH (./SYM)/PNCH

Nepali Chunked Sentence: (एक/CD केटा/NN)/NCH (टेबुल/NN मा/POP)/NCH (बसिरहेको/VBKO छ/VBX)/VCH **(** I/YF)/PNCH

Alignment:

is sitting बसिरहेको छ

on the table टेबुल मा

a boy एक केटा

English Sentence: she is a good girl.

Nepali Sentence: उनी रामो केटि हुन्।

English Chunked Sentence: (she/PRP)/NCH (is/VBZ)/VCH (a/DT good/JJ girl/NN)/NCH (./SYM)/PNCH

Nepali Chunked Sentence: (उनी/PP)/NCH (रामो/JJM केटि/NN)/NCH (हुन्/VBF)/VCH (।/YF)/PNCH

Alignment:

she उनी

a good girl रामो केटि

is हुन्

English Sentence: a teacher is singing a song on the ground.

Nepali Sentence: एक िसछक एक गीत चउर मा गाईरहेको छ ।

English Chunked Sentence: (a/DT teacher/NN)/NCH (is/VBZ singing/VBG)/VCH (a/DT song/NN)/NCH (on/IN the/DT ground/NN)/NCH (./SYM)/PNCH

Nepali Chunked Sentence: (एक/CD)/CD (िसछक/CD)/CD (एक/CD गीत/NNP)/NCH (चउर/NN मा/POP)/NCH (गाईरहेको/VBKO छ/VBX)/VCH (।/YF)/PNCH

Alignment:

on the ground चउर मा

a song एक गीत

a teacher एक गीत

is singing गाईरहेको छ

English Sentence: he is singing a very popular song.

Nepali Sentence: ऊ एक धेरै प्रख्यात गीत गाईरहेको छ ।

English Chunked Sentence: (he/PRP)/NCH (is/VBZ singing/VBG)/VCH (a/DT)/NCH (very/RB popular/JJ)/AJCH (song/NN)/NCH (./SYM)/PNCH

Nepali Chunked Sentence: (ऊ/PP)/NCH (एक/CD)/NCH (धेरै/JJ)/AJCH (प्रख्यात/JJ गीत/NN)/NCH (गाईरहेको/VBKO छ/VBX)/VCH (I/YF)/PNCH

Alignment:

song प्रख्यात गीत

he ऊ

a गाईरहेको छ

is singing गाईरहेको छ

very popular प्रख्यात गीत

English Sentence: a girl is sitting on the ground.

Nepali Sentence: एक केटि चउर मा बसिरहेको छ।

English Chunked Sentence: (a/DT girl/NN)/NCH (is/VBZ sitting/VBG)/VCH (on/IN the/DT ground/NN)/NCH (./SYM)/PNCH

Nepali Chunked Sentence: (एक/CD केटि/NNP)/NCH (चउर/NN मा/POP)/NCH (बसिरहेको/VBKO छ/VBX)/VCH (I/YF)/PNCH

Alignment:

on the ground चउर मा

is sitting बसिरहेको छ

a girl एक केटि

English Sentence: a teacher teaches this lesson.

Nepali Sentence: एक शिछक यो पाठ पढाउँछ ।

English Chunked Sentence: (a/DT teacher/NN)/NCH (teaches/VBZ)/VCH (this/DT lesson/NN)/NCH (./SYM)/PNCH

Nepali Chunked Sentence: (एक/CD)/CD (शिछक/POP)/AVCH (यो/DUM पाठ/NN)/NCH (पढाउँछ/VBF)/VCH (।/YF)/PNCH

Alignment:

this lesson यो पाठ

Teaches पढाउँछ

a teacher एक

English Sentence: he teaches in the school.

Nepali Sentence: ऊ स्कुल मा पढाउँछ ।

English Chunked Sentence: (he/PRP)/NCH (teaches/VBZ)/VCH (in/IN the/DT school/NN)/NCH (./SYM)/PNCH

Nepali Chunked Sentence: (ऊ/PP)/NCH (स्कुल/NN मा/POP)/NCH (पढाउँछ/VBF)/VCH (।/YF)/PNCH

Alignment:

teaches पढाउँछ

he ऊ

in the school स्कुल मा

English Sentence: this book has a good lesson.

Nepali Sentence: यो किताब सँग राम्रो पाठ छ ।

English Chunked Sentence: (this/DT book/NN)/NCH (has/VBZ)/VCH (a/DT good/JJ lesson/NN)/NCH (./SYM)/PNCH

Nepali Chunked Sentence: (यो/DUM किताब/NN सँग/POP)/NCH (राम्रो/JJM पाठ/NN)/NCH (छ/VBX)/VCH (।/YF)/PNCH

Alignment:

a good lesson राम्रो पाठ

has छ

this book यो किताब सँग

3.3 Analysis

The average precision and recall measure of the above given sentences is found by calculating the individual precision and recall measure of the given sentences and finding the average of them.

Precision= 75%

Recall= 80%

F-score = 77.4 %

AER=22.6%

The Precision rate gives the alignment accuracy when it is compared to the total alignment given by the model for the particular sentence where the Recall rate gives the accuracy of the alignment when it is compared to the actual alignment given by the human annotator. Here in our alignment model the Recall rate has been found greater than the Precision rate. i.e The model gives the better alignment accuracy against the human annotated alignment than the total alignment given by the model. The Alignment Error Rate (AER) is found to be 22.6%.

The model accuracy is highly dependent on the accuracy of tagging, chunking and the alignment process. The tagging accuracy depends on the performance of TnT tagger and the chunking accuracy depends on the chunking rules defined on the previous chapter. The training data plays the significant role to define the translation probability which will be used on the alignment process. The EM algorithm described in the previous chapter is responsible to find the probabilities of the word pair of the source and target language. And it runs over the training corpus we defined. The alignment model depends on the quality of the training corpus. Here the words in the sentences of the test corpus are taken from the training corpus so accuracy of the model is somewhat good. But if the sentences having the words not defined on the training corpus are used as testing data then the accuracy of the model is dramatically decreased. Hence the problem of appearing the unknown words in a test data has been decreased the accuracy of the model. So to obtain the better accuracy of the model it is necessary to define the training corpus having the large quantity of high quality bilingual data aligned at sentence level.

CHAPTER V

CONCLUSION AND FUTURE WORK

5.1 Conclusion

In this dissertation, the alignment model for aligning the sentences at the chunk level has been developed. The alignment model first performs tagging by using the TnT tagger and then does chunking of the both tagged sentences using the grammatical rules of bracketing the chunks. And finally it gives the alignment between the source chunks and target chunks using the statistical information gathered from bilingual sentence aligned corpus. The IBM model-1 has been used for the alignment algorithm in which the total alignment probability only depends on the lexicon probability for the given sentence pair. The knowledge-base for statistical information in the alignment is generated by training the bilingual corpus using Expectation Maximization (EM) algorithm. Since the model uses a set of grammatical rules to perform chunking, it needs a POS tagged corpus and the set of rules for bracketing the chunks. The famous TnT tagger is used for obtaining the POS tagged corpus. The model performs chunking using a rule based technique and does the alignment by the help of statistics generated from bilingual sentence aligned training corpus; hence the model is of hybrid nature. The system aligns English and Nepali sentences on its chunk level which further help on Statistical Machine Translation as well as helpful for translational memory system. The important factor on going the chunk level alignment is the complexity of the model. Since the length of the sentence will be decreased when the sentence is analyzed at the chunk level rather than at word level, hence the time complexity of the alignment algorithm will be decreased.

Our alignment model does rely on the information from tagging and chunking process of the given sentence pair. Hence the model accuracy not only depends on the alignment algorithm and the training corpus but also depends on the former two models tagger and the chunker. It has been observed that the model accuracy seems high for the sentences defined on the training corpus and the accuracy has been decreased for the sentences which are unknown to training corpus. So to obtain the better accuracy of the model it is necessary to define training corpus having the large

quantity of high quality bilingual data and the sufficient chunking rules for bracketing the chunks of the given sentence pair.

5.2 Further Recommendations

The alignment model, we devised, is highly influenced by linguistic rules for defining the chunks of a sentence and the statistical information gathered from the training corpus. So, in future, the sufficient rules can be devised by the help of linguists so that the chunking accuracy will be increased. The large quantity of high quality training corpus can be defined to obtain the sufficient amount of the knowledge base for the alignment as well as tagging process. The problem of unknown words can be handled by defining the training corpus having the large quantity of distinct words. The chunking algorithm in our model is totally rule based so it is some what difficult to define the sufficient rules for bracketing the chunks in both languages specially those students who do not have the knowledge of linguistics. So it will be better for the students having the computing background if the statistical methods can be devised for the chunking process instead of using rule based techniques.

References

[1] S. Abney, *Parsing by Chunks. In: Robert Berwick, Steven Abney and Carol Tenny (eds.), Principle-Based Parsing*. Kluwer Academic Publishers, Dordrecht. (1991).

[2] I. Arad, *A quasi-statistical approach to automatic generation of linguistic knowledge (*Ph D thesis, UMIST, Manchester, 1991).

[3] C. Bannard and C. Callison-Burch, *Paraphrasing with bilingual parallel corpora,* In Proceedings of the 43th Annual Meeting of the Association for Computational Linguistics, (2005), 597-604.

[4] A. Bharati, V. Sriram, K. A. Vamshi, R. Sangal and S. Bendre, *An Algorithm for Aligning Sentences in Bilingual Corpora Using Lexical Information*, International Institute of Information Technology, Hyderabad.

[5] A. Bharati, D. M. Sharma, L. Bai, R. Sangal, AnnCorra*: Annotating Corpora Guidelines for POS and Chunk Annotation for Indian Languages,* Technical Report (TR-LTRC-31), Language Technologies Research Centre IIIT, Hyderabad. http://ltrc.iiit.ac.in/MachineTrans/publications/technicalReports/tr031/posguidelines.pdf

[6] P. F. Brown, S. A. Della Pietra, V. J. Della Pietra, and R. L. Mercer, *The mathematics of statistical machine translation: Parameter estimation*, Computational Linguistics, 19(2) (1993), 263 - 311.

[7] P. Brown, J. Cocke, S. Della Pietra, V. Della Pietra, F. Jelinek, J.D. La_erty, R. Mercer, and P.S. Roossin, *A statistical approach to machine translation*, Computational Linguistics, Vol. 16, no 2, (1990), 79–85.

[8] P. Brown, J. Lai and R. Mercer, *Aligning Sentences in Parallel Corpora,* 47th Annual meeting for the Association of Computational Linguistics, (1991).

[9] P. F. Brown, S. A. Della Pietra, V. J. Della Pietra, and R. L. Mercer. *Analysis, statistical transfer, and synthesis in machine translation,* In Proceedings of the Fourth International Conference on Theoretical and Methodological Issues in Machine Translation, (1992), 83-100, Montreal.

[10] R. D. Brown, J. G. Carbonell, and Y. Yang, *Parallel text Processing: Alignment and use of Translation Corpora, chapter Automatic dictionary extraction for cross-language information retrieval*, (2000), 275-298, Kluwer Academic Publishers.

[11] G. Chinnapa and A. K. Singh, *A Java Implementation of an Extended Word Alignment Algorithm Based on the IBM Models*, Language Technology Research Center IIIT Hyderabad, India.

[12] K. W. Church, *Char align: A program for aligning parallel texts at the character level*. In Proceedings of the 31st Annual meeting of the association for Computational Linguistics, (1993), 1-8, Columbus, Ohio.

[13] A.P. Dempster, N. M. Laird, and D.B. Rubin, *Maximum likelihood from incomplete data via the EM algorithm (with discussion)*, Journal of the royal statistical society*, 39, (1997).*

[14] M. Diab and P. Resnik, *An unsupervised method for word sense tagging using parallel corpora,* In Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics, (2002), 255-262.

[15] P. Fung and K. W. Church, *K-vec: A new approach for aligning parallel texts,* In Proceedings of the 15th International Conference on Computational Linguistics, (1994), 1096-1102, Kyoto, Japan.

[16] W. A Gale and K. W Church, *A Program for Aligning Sentences in Bilingual Corpora,* Computational Linguistics, also presented at ACL-91, (1994).

[17] W. Gale and K. Church, *Identifying word correspondences in parallel texts,* In Processing of the Fourth DARPA Speech and Natural Language workshop, (1991), 152-157, Pacific Grove, CA.

[18] D. Hiemstra, *Using Statistical Methods to Create a Bilingual Dictionary* (Master Thesis, University of Twente, 1996).

[19] W. J. Hutchins and H. L. Somers, An *introduction to machine translation*, (1992).

[20] Y.S. Hwang, K. Paik, and Y. Sasaki, *Bilingual Knowledge Extraction Using Chunk Alignment*, PACLIC 18, (December 8th -10th, 2004), Waseda University, Tokyo.

[21] D. Jurafsky, J. H. Martin, *Speech and Language Processing: An Introduction to Speech Recognition Natural Language Processing and Computational Linguistic*, (2006).

[22] P. Koehn, F. J. Och, and D. Marcu, *Statistical phrase based translation,* In Proceedings of HLT-NAACL, (2003), Edmonton, Canada, 81-88.

[23] P. Koehn, *Pharaoh: A beam search decoder for phrase-based statistical machine translation models,* In Proceedings of COLING- 96, the 16th International Conference on Computational Linguistics, volume 1, (2004) 115-124, Washington, DC, USA.

[24] J. M. Kupiec, *An algorithm for finding noun phrase correspondences in bilingual corpora*, In Proceedings of the 31st Annual Meeting of the association for Computational Linguistics, (1993), Columbus, Ohio.

[25] D. Marcu and W. Wong, *A Phrase-based, joint probability model for statistical machine translation*, In Proc. Of EMNLP, (2002).

[26] I. D. Melamed, *Automatic discovery of non-compositional compounds in parallel data,* In Proceedings of the 2nd Conference on Empirical Methods in Natural Language Processing (EMNLP), (1997), Providence, RI.

[27] I.D. Melamed, *Models of translational equivalence among words,* Computational Linguistics, Vol. 26, No. 2, (2000), 221–249.

[28] I. D. Melamed, *Empirical Methods for Exploiting Parallel Texts*, MIT Press, (2001).

[29] J. Nerbonne, *Parallel text Processing: Alignment and use of Translation Corpora*, *chapter Parallel texts in computer-assisted language learning*, (2000), 299-311. Kluwer Academic Publishers.

[30] F. J. Och, C. Tilhnalm, H. Ney, *Improved alignment models for statistical machine translation*. *In Proc. of the Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora,* (1999), 20-28, University of Maryland, College Park, MD, USA, June.

[31] F. J. Och and H. Ney, *A systematic comparison of various statistical alignment models*. Computational Linguistics, 29 (1), (2003), 19-52.

[32] F. Och and H. Ney, *The alignment template approach to statistical machine translation,* Computational Linguistics, 30(4), (2004), 417-449.

[33] G. Pohl and M. Mihaltz, *Exploiting parallel corpora for supervised word-sense disambiguation in english-hungarian machine translation*, In Proceedings of the 5th International Conference on Language Resources & Evaluation (LREC'06), (2006), Genoa, Italy.

[34] P. Rupakheti, L. P. Khatiwada and B. K. Bal, *Report on Nepali Computational Grammar,* Madan Puraskar Pustakalaya Lalitpur, PatanDhoka,Nepal.

[35] S. Sato and M. Nagao, *Toward memory-based translation*. In Proceedings of the I5th International Conference on Computational Linguistics (COLING-90), volume 1, (1990), Helsinki, Finland.

[36] B. Schrader, *ATLAS – A new text alignment architecture*, Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, (July 2006), 715–722,Sydney, Association for Computational Linguistics.

[37] E. Sumita, H. Iida, and H. Kohyama, *Translating with examples: a new approach to machine translation,* In Proceedings of the Third International Conference on Theoretical and Methodological Issues in Machine Translation of Natural Languages (TMI'90), (1990), Austin, Texas.

[38] B. Thorsten, *TnT- Statistical Part-of-Speech Tagger*.

http://www.coli.uni-saarland.de/~thorsten/tnt/

[39] E. F. Tjong, K. Sang, S. Buchholz, *Introduction to the CoNLL-2000 Shared Task: Chunking,* In: Proceedings of CoNLL-2000 and LLL-2000*,* (2000), 127-132, Lisbon, Portugal.

[40] K. Toutanova, H. T. Ilhan, and C. manning, *Extensions to HMM based statistical word alignment models, In Proc. of EMNLP, (2002).*

[41] S. Vogel, H. Ney, and C. Tillman, *HMM-Based Word Alignment in Statistical Translation*, (1999).

[42] W. Wang and M. Zhou, *Structure Alignment Using Bilingual Chunking, (* 2002).

[43] H. Watanabe, S. Kurohashi, and E. Aramaki, *Finding structural correspondences from Bilingual Parsed Corpus for Corpus-based Translation*, in COLING 2000, (2000), 906-912.

[44] T. Watanabe, E. Sumita, and H. G. Okuno, *Chunk-based Statistical Translation, in proceeding of the 41st Annual Meeting of the Association for Computational linguistics*, (July 2003), 303-310.

[45] W. Weaver, *Translation In: Machine Translation of Languages*, MIT Press, Cambridge, MA (1949).

[46] D. Wu, *Stochastic inversion transduction grammars and bilingual parsing of parallel corpora,* In *Computational Linguistics*, volume 23(3), (1997), 377–403.

[47] D. Wu and X. Xia, *Large-scale automatic extraction of an english-chinese translation lexicon.* In Proceedings of the First Conference of the Association for Machine Translation in the Americas, (1994), 206-213, Columbia,Maryland.

[48] K. Yamada and K. Knight, *Syntax-based Statistical Translation Model,* In Proceedings of the Conference of the Association for Computational Linguistics, *(ACL-2001)*.

[49] Y. Zhang and S. Vogel, *An efficient phrase-to-phrase alignment model for arbitrarily long phrase and large corpora*, In proceedings of the Tenth Conference of the European Association for Machine Translation, (2005).

[50] Y. Zhou, C. Zong, and B. Xu, *Bilingual Chunk Alignment in Statistical Machine Translation*, IEEE, International Conference on Systems, Man and Cybernetics, (2004).

Appendix A

Sentenced aligned parallel corpora used for training

English Corpus

this book is on the table

this is a book

boys are going to school

a boy is going to school

a boy is sitting on the table

a boy is singing a song

this is a good song

she is a good girl

i go to school

boys are playing on the ground

a girl is sitting on the ground

she reads book

i am a teacher

a teacher teaches this lesson

this school has a good ground

this song is very popular

he teaches in the school

he has a book

he is popular

this lesson is on this book

Nepali Corpus

यो किताब टेबुल मा छ

यो एक किताब हो

केटा हरु स्कुल गईरहेको छन्

एक केटा स्कुल गईरहेको छ

एक केटा टेबुल मा बसिरहेको छ

एक केटा गीत गाईरहेको छ

यो रामो गीत हो

उनी रामो केटि हुन्

म स्कुल जान्छु

केटा हरु चउर मा खेलिरहेका छन्

एक केटि चउर मा बसिरहेको छ

उनी किताब पढ्छिन्

म सिछक हु

एक सिछक यो पाठ पढाउँछ

यो स्कुल सँग रामो चउर छ

यो गीत धेरै प्रख्यात छ

ऊ स्कुल मा पढाउँछ

ऊ सँग किताब छ

ऊ प्रख्यात छ

यो पाठ यो किताब मा छ

Appendix B

Test Corpus

Source Sentence (English)	*Target Sentence (Nepali)*
this book is on the table .	यो किताब टेबुल मा छ ।
this teacher is very popular .	यो सिछक धेरै प्रख्यात छ ।
this song is very popular .	यो गीत धेरै प्रख्यात छ ।
a teacher is sitting on the ground .	एक सिछक चउर मा बसिरहेको छ ।
boys are singing on the ground .	केटाहरु चउर मा छन् गाईरहेको ।
a teacher is going to school .	एक सिछक स्कुल गईरहेको छ ।
a boy is sitting on the table .	एक केटा टेबुल मा बसिरहेको छ ।
this book has a good lesson .	यो किताब सँग राम्रो पाठ छ ।
he is a popular teacher .	ऊ एक सिछक प्रख्यात छ ।
she teaches in a popular school .	उनी एक प्रख्यात स्कुल मा पढाउँछ ।
this ground is very popular .	यो चउर धेरै प्रख्यात छ ।
a teacher is singing a song on the ground .	एक सिछक एक गीत चउर मा गाईरहेको छ ।
she is a popular girl .	उनी एक केटि प्रख्यात छ ।
he is singing a very popular song .	ऊ एक धेरै प्रख्यात गीत गाईरहेको छ ।
she is a teacher .	उनी एक सिछक छ ।

he is a good boy .	ऊ एक केटा रामो छ ।
this boy is a good teacher .	यो केटा एक सिछक रामो छ ।
this is a popular school .	यो स्कुल प्रख्यात एक छ ।
boys are sitting on the table .	केटाहरु टेबुल मा बसिरहेको छन् ।
he reads this lesson .	ऊ यो पाठ पढ्छन् ।
i am singing a song .	म एक गीत हु गाईरहेको ।
she is going to school .	उनी स्कुल गईरहेको छ ।
she is singing a song .	उनी एक गीत गाईरहेको छ ।
this school has a good teacher .	यो स्कुल सँग एक सिछक रामो ।
i am a good boy .	म एक केटा रामो हु ।
a teacher reads this lesson .	एक सिछक यो पाठ पढ्छन् ।
boys are popular .	केटाहरु प्रख्यात छन् ।
this girl is going to school .	यो केटि स्कुल गईरहेको छ ।
a teacher is sitting on the ground .	एक सिछक चउर मा बसिरहेको छ ।
he is popular .	ऊ प्रख्यात छ ।
school is very good .	स्कुल धेरै रामो छ ।
this book is very good .	यो किताब धेरै रामो छ ।

Appendix C

Code of Implementation

Source code to tag the sentences

```
Process p = Runtime.getRuntime().exec("c:\\smt\\running.bat");
```

Where ruuning.bat contains

```
c:\\smt\\tnt.exe c:\\smt\\wsj c:\\smt\\eng.txt > c:\\smt\\tageng.txt
```

Source code to make the chunks

```
String line;
        rfr.readTokenizePopulate();
        while(ifr.hasMore()){
            line=ifr.getLine();
            this.isp=new InputSentenceParser(line);
            this.isp.tokenizePopulate();
            this.cfr.setWordPosList(this.isp.getWordPosList());
            this.cfr.startChunking();
            this.chunkList=this.cfr.getChunkList();
            for(int i=0;i<this.chunkList.size();i++){

this.out.print("("+this.chunkList.get(i).getSubSentence()+")"+"/"+this.chunkList.get(i).getChu
nkSet()+" ");
            }
            this.out.print("\n");
        }
```

Source code to make parallel corpora

```
Corpus c = new Corpus();

        String englishSentences[];
        String nepaliSentences[];
        englishSentences = c.getEnglishcorpus();
        nepaliSentences = c.getNepaliCorpus();
        parallelSentence = new ArrayList<SentencePair>();

        for(int i=0;i<englishSentences.length;i++)
        {
            SentencePair snt;
            String sTokens[] = tokenPopulate(englishSentences[i]);
            String tTokens[] = tokenPopulate(nepaliSentences[i]);
            snt = new SentencePair(sTokens,tTokens);
            this.parallelSentence.add(snt);
        }
```

Source code to make the word pair

```
wordtable = new Hashtable<String,SourceTargetWordPairs>();
        for(int i=0;i<parallelSentence.size();i++)
        {
            SentencePair snt = parallelSentence.get(i);
            String[] sourceSentence = snt.getSourceSentence();
            for(int j=0; j<sourceSentence.length; j++)
```

```
                {
                        HashSet<String> targetWords =
findTargetWords(sourceSentence[j]);
                        SourceTargetWordPairs stwpairs = new
SourceTargetWordPairs(sourceSentence[j],targetWords);
          this.wordtable.put(sourceSentence[j],stwpairs);
                }
        }
```

Source Code to EM Algorithm

```
    public void EMalgorithm()
    {
        String sword;
        String tword;
        //Hashtable<WordPair,Double> translationCount = new
Hashtable<WordPair,Double>();
    Hashtable<String,Double> translationCount = new Hashtable<String,Double>();
        translationProbability = new Hashtable<String,Double>();
        SourceTargetWordPairs[] st = wordtable.values().toArray(new
SourceTargetWordPairs[0]);

        //Set uniform probability
        for(int i=0; i<st.length; i++)
        {
            sword = st[i].getSourceWord();
```

```
                    Object[] targetWords = st[i].getTargetWords().toArray(new Object[0]);


                    //Calculate the word pair probability
                    double tProb = (double) 1/st[i].getTargetWords().size();


                    for(int j=0; j<targetWords.length; j++)
                    {
                            tword = targetWords[j].toString();
                            WordPair wp = new WordPair(sword,tword);
                            //Initialize the translation probability to the HashTable
                            translationProbability.put(wp.toString(),tProb);
                    }
            }
            for(int n=0;n<3;n++)//run the EM Algorithm for n=2 times
            {
                    //double tc = 0;
                    //WordPair[] WordPairObjects = translationProbability.keySet().toArray(new
WordPair[0]);
        WordPair[] WordPairObjects = new WordPair[translationProbability.size()];
        for(int i=0; i<translationProbability.size(); i++)
           WordPairObjects[i] = new WordPair();
                    for(int i=0;i<WordPairObjects.length;i++)
                    {
                            //for each wordPair initialize translation count by 0
```

```
                translationCount.put(WordPairObjects[i].toString(),0.0);
            }//end for

            for(int s=0;s<parallelSentence.size();s++)//for each sentence pair
            {
                SentencePair snt =parallelSentence.get(s);
                String[] targetSentence = snt.getTargetSentence();
                String[] sourceSentence = snt.getSourceSentence();
                for(int j=0;j<targetSentence.length;j++)//for each word of the
targetSentence
                {
                    double total = 0.0;
                    tword = targetSentence[j];

                    for(int i=0;i<sourceSentence.length;i++)//for each word of
the sourceSentence
                    {
                        double d = 0;
                        sword = sourceSentence[i];

                        WordPair wp = new WordPair(sword,tword);

        //change here
                        if(translationProbability.get(wp.toString()) == null)
```

```
                d = 0;
            else
                d = translationProbability.get(wp.toString());
                                        //total += translationProbability.get(wp.toString());
            total += d;
                                        for(int k=0;k<sourceSentence.length;k++)//for each
word of the sourceSentence
                                        {
                double tpValue = 0;
                double tcValue = 0;
                                            String sourceWord = sourceSentence[k];
                                            WordPair wpObject = new
WordPair(sourceWord,tword);
                //change here
                if(translationProbability.get(wpObject.toString()) != null)
                  tpValue = translationProbability.get(wpObject.toString());
                                            //double tpValue =
translationProbability.get(wpObject.toString());
                if(translationCount.get(wpObject.toString()) != null)
                  tcValue = translationCount.get(wpObject.toString());
                                            //double tcValue =
translationCount.get(wpObject.toString());
                                            tcValue += (tpValue/total);

        translationCount.put(wpObject.toString(),tcValue);
```

```
                    }//end for
                }//end for
            }//end for
        }//end for
        //re-estimate the translation parameter(translationProbability) values
        for(int s=0;s<parallelSentence.size();s++)//for each sentence pair
        {
            SentencePair snt =parallelSentence.get(s);
            String[] sourceSentence = snt.getSourceSentence();
            String[] targetSentence = snt.getTargetSentence();
            for(int i=0;i<sourceSentence.length;i++)//for each word of the
sourceSentence
            {
                double total = 0;
                sword = sourceSentence[i];

                for(int j=0;j<targetSentence.length;j++)//for each word of
the targetSentence
                {
                    tword = targetSentence[j];
                    WordPair wp = new WordPair(sword,tword);
                    total += translationCount.get(wp.toString());
                    for(int k=0;k<targetSentence.length;k++)
                    {
```

```
                double tcValue = 0;
                double tpValue = 0;
                                                String targetWord = targetSentence[k];
                                                WordPair wpObject = new
WordPair(sword,targetWord);
                //change here
                if(translationCount.get(wpObject.toString()) != null)
                  tcValue = translationCount.get(wpObject.toString());
                                                //double tcValue =
translationCount.get(wpObject.toString());
                if(translationProbability.get(wpObject.toString()) != null)
                  tpValue = translationProbability.get(wpObject.toString());
                                                //double tpValue =
translationProbability.get(wpObject.toString());
                                                tpValue = tcValue/total;

        translationProbability.put(wpObject.toString(),tpValue);
                                                }//end for
                                        }//end for
                                }//end for
                        }//end for
                }
```

Source code to align the chunk

```
double tProb;
    int ii = 0;
                final WordPair[] wp = new WordPair[translationProbability.size()];
                for(int i=0; i<translationProbability.size(); i++)
        {
           wp[i] = new WordPair();
        }
        System.out.println("error aayyekop thau");
           for(String str : translationProbability.keySet())
           {
              String a[] = str.split(" ");
              wp[ii].setSourceWord(a[0]);
              wp[ii].setTargetWord(a[1]);
              ii++;

           }
                System.out.println("error aayyekop thau");
                System.out.println(wp[0].getSourceWord()+","+wp[0].getTargetWord());
                WordPair test = new WordPair(wp[0].getSourceWord(),wp[0].getTargetWord());
                System.out.println(test.getSourceWord()+","+test.getTargetWord());
                System.out.println(wp[0].toString().equals(test.toString()));

    System.out.println("yeha bata suru ho");
                for(int i=0; i<wp.length; i++)
                {
                        tProb = translationProbability.get(wp[i].toString());
```

```
    System.out.println("("+wp[i].getSourceWord()+","+wp[i].getTargetWord()+"):\t"+tProb);
                }

    //implementing in chunk level
int count = 0;
String inputEng[] = new String[tblForEnglishChunk.size()-1];
String inputNep[] = new String[tblForNepaliChunk.size()];
for(String str : tblForEnglishChunk.keySet())
{
   if(str.equals(". "))
      continue;
   else
   {
      inputEng[count] = str;
      count++;
   }
}

count = 0;
for(String str : tblForNepaliChunk.keySet())
{
   inputNep[count] = str;
   count ++;
}
      //array to put the obiatned tarined value and find the maximum from it
double[] maxVal = new double[tblForNepaliChunk.size()];
for(int i=0; i<maxVal.length; i++)
```

```
    maxVal[i] = 0.0;


for(int i=0; i<inputEng.length; i++)
{
    count = 0;
    for(int i1=0; i1<maxVal.length; i1++)
        maxVal[i1] = 0.0;


    double val = 0.0;
    double mul = 0.0;
    for(int j=0; j<inputNep.length; j++)
    {
        String[] sEng = inputEng[i].split(" ");
        String[] sNep = inputNep[j].split(" ");
        for(String ss : sEng )
        {
            for(int k=0; k<sNep.length; k++)
            {
                WordPair checkWP = new WordPair(ss.toLowerCase(),sNep[k]);
                if(translationProbability.get(checkWP.toString()) != null)
                    val = translationProbability.get(checkWP.toString());
                else
                    val = 0.0001;
                //mul = mul * val;
                if(val > mul)
                    mul = val;
            }
```

```
            maxVal[count] = maxVal[count] + mul;
            mul = 0.0;
          }
          tblToFindMostProbableChunk.put(maxVal[count], inputNep[j]);
          count ++;
        }
        //now find the maximum valued aligned chunk
        Arrays.sort(maxVal);

listHoldingTheAlignedChunk.add(inputEng[i]+tblToFindMostProbableChunk.get(maxVal[tblForNepali
Chunk.size()-1]));


         JFrame frm = new JFrame();
        frm.setTitle("Chunk Alignment");
        JTextArea a = new JTextArea(10,10);
        frm.add("Center", new JScrollPane(a));
        for(String s : listHoldingTheAlignedChunk)
        {
          a.append(s);
        }
      frm.setSize(250,250);
      frm.setVisible(true);
      frm.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);


      }


      //repeating code
      //test the result
```

```
JFrame frm = new JFrame();
   frm.setTitle("Chunk Alignment");
   JTextArea a = new JTextArea(10,10);
   frm.add("Center", new JScrollPane(a));
   for(String s : listHoldingTheAlignedChunk)
      a.append(s+"\n");

frm.setSize(250,250);
frm.setVisible(true);
frm.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

//make the matrix
JFrame frmMatrix = new JFrame();
frmMatrix.setTitle("Matrix Form");
JTable tblMatrix;
DefaultTableModel model;
Object colNames[] = new Object[tblForEnglishChunk.size()];
//colNames[0] = null;
int count1 = 0;

for(String s : tblForEnglishChunk.keySet())
{
   if(s.equals(". "))
   {
      colNames[count1]=" ";
      count1++;
      continue;
```

```
    }
    colNames[count1]=s;
    count1++;
}
//cheks the space geree
for(int i=0; i<colNames.length; i++)
{
    if(colNames[i].equals(" "))
    {
        colNames[i] = colNames[0];
        colNames[0]= " ";

        break;
    }
}
                Object colValues [][] = null;
model = new DefaultTableModel(colValues, colNames);

for(String s : tblForNepaliChunk.keySet())
{
    int cc = 1;
    Object[] rowValues = new Object[tblForEnglishChunk.size()];
    rowValues[0]= s;
    for(String ss : listHoldingTheAlignedChunk)
    {
        String temp = colNames[cc] + s;
        if(temp.equals(ss))
```

```
        {
            rowValues[cc] = "*";
        }
        cc++;
    }
    model.addRow(rowValues);
}
```