CHAPTER I

INTRODUCTION

1.1 Dictionary

A 'dictionary' as Oxford English Dictionary defines is a "a book dealing with the individual words of a language (or certain specific classes of them) so as to set forth their orthography, their pronunciation, signification and use, their synonyms, derivations and history, or at least some of these facts: for convenience of reference the words are arranged in some stated order, now, in most languages, alphabetical and in some dictionaries the information given is illustrated by quotations from literature".

This definition points out a book that we consult to know the information of general words or general information of words in a language, which is also known as a lexicon or a wordbook. The examples of such dictionaries are Oxford English Dictionary, Webster's English Dictionary, Random House English Dictionary, etc.

The word dictionary also refers to

(i) a book in which words in one language are listed alphabetically and are followed by words which have the same meaning in another language. For example, English-Nepali or Nepali-English Dictionary.

(ii) any alphabetically ordered reference book on one particular subject or limited group of subjects e.g. Dictionary of Linguistics, Dictionary of Physics, Dictionary of National Biography .

(iii) a book that lists the words in alphabetical order and includes only one aspect of the word information, for example, Dictionary of Etymology, English Pronouncing Dictionary of Slang or Dictionary of Abbreviation,

(iv) a list of words of a language in electronic form, for example, stored in a computer's spellchecker. Meaning is the 'core' of a dictionary. It may be given in terms of easier (another) word (e.g. invoice -bill) or it may be in terms of definition (e.g. student - A person who is studying at a school college or university) or in terms

of translation in other language (e.g. man - मानिस, मानिस - man). However, a dictionary is not only meant for the meaning but also the information like spelling, pronunciation, syllabification, stress, word-class, etymology, history, collocation, use and usage depending upon its type[3] .

1.2 Corpus

Corpus is a finite collection of text. In modern linguistics the term is used to refer to large collections of texts, in electronic form, selected to represent as more as possible a language or a verity of language for the purpose of linguistic research. If the collection of texts contains documents in more than one language it is referred to as multilingual corpora.

1.2.1 Parallel Corpus

A Parallel corpus is a collection of texts in different language where one of them is the original text and the other is their translations. A bilingual corpus is collection of texts in two different languages where each of one is translation of other. Parallel corpora hold a huge amount of linguistic information and this is the reason why they have many applications in the field of natural language processing. The type of corpus that is going to be used in this thesis is parallel corpus.

Parallel corpora are very important resources for tasks in the translation field like linguistic studies, information retrieval system development or natural language processing [6].In order to be useful; those 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.

1.2.2 Bilingual Parallel Corpus

A bilingual corpus is a collection of texts in two different languages where each of one is translation of other. Aligned bilingual corpora have been proved useful in many ways including machine translation, sense disambiguation and bilingual lexicography.

1.2.3 Application of parallel corpora

Parallel corpora are turned out to be a powerful tool in the hands of scientists, translators and linguists. For the last two decades researchers in the field of natural language processing and the general applied linguistics have been working with parallel corpora. Nowadays parallel corpora are in electronic form and they have become an important resource in language engineering while they are used widely in multilingual lexicography and terminology, human and Machine Translation (MT), Multilingual Information Retrieval, language learning and so on. In language learning parallel corpora can be used by extracting basic linguistic information from texts for teaching and learning of the language pairs. They can be used by students in order to find translation pairs and learn translation techniques. It is considered as a challenge for the student to understand the translated sentences and built concepts and structure, based on the original one, supplementing in this way the teaching process. Parallel corpora can be found useful in multilingual terminology. As the technology evolves, new terms are introduced in new subject areas that are not included in existing dictionaries. Analysis of parallel corpora at a word alignment level is a useful mean in the extraction of multilingual terminology which is used by terminologists and translators. In the field of Multilingual Information Retrieval, the query written in one language must be translated in to the target languages of the documents under demand. The difficulty occurs when multi terms of the query form a phrase, unable to be identified by bilingual dictionaries. Parallel corpora can be used then for a word to word translation based on translation probability using larger blocks of aligned text Parallel corpora are turned out to be a powerful tool for automated translation. They are utilized in statistical methods in order to automatically extract word translation equivalents with minimal or without the use of linguistic information .

1.3 Identifying words and sentences

Identifying sentences is not as easy as it might appear. It would be easy if periods always were used to mark sentence boundaries but unfortunately many periods have others purpose. They may appear for example in numerical expression and abbreviations. A simple set of heuristics can be used to identify sentences boundaries for some language like Chinese even the identification of words is not a trivial task

because written Chinese consists of a character steam with no space separator between words [35].

1.4 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.

1.4.1 Document Alignment

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

1.4.2 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.

Aligned paragraph are further segmented into sentences. 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 orders of sentences are changed in the translation.

1.4.3 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 [12] 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 [7] 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 [37], 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 words for both the languages. The algorithm first breaks the sentences of both the languages into small units called words. 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 words that match between the two sentences. The algorithm then carries out the alignment of sentences using these scores.

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.

1.4.4 Word Alignment

Word alignment is the natural language processing task of identifying translation relationships among the words in a bitext, resulting in a bipartite graph between the two sides of the bitext, with an arc between two words if and only if they are translations of one another. 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 language word. The probability of some target language word to be a translation of source language 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 algorithm and a Viterbi search is carried out to compute the most probable sequence of word translation pairs.

In 1991, Gale and Church [14] 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 [14] works only for parallel corpus and makes use of the word position and frequency feature to find word correspondences.

Most current SMT systems [18, 26] use a generative model for word alignment such as the freely available tool GIZA++ [25], 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 [9] 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 [9]. Considerable effort has also been made to align English-Chinese translation texts [14, 37].

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 falls between word and sentence alignments, but it is usually resolved subsequent to word alignment.

1.5 Stage in automatic dictionary construction

The automatic creation of bilingual dictionary generally takes four steps[11].These steps are shown in the following block diagram along with their description in the following sections.

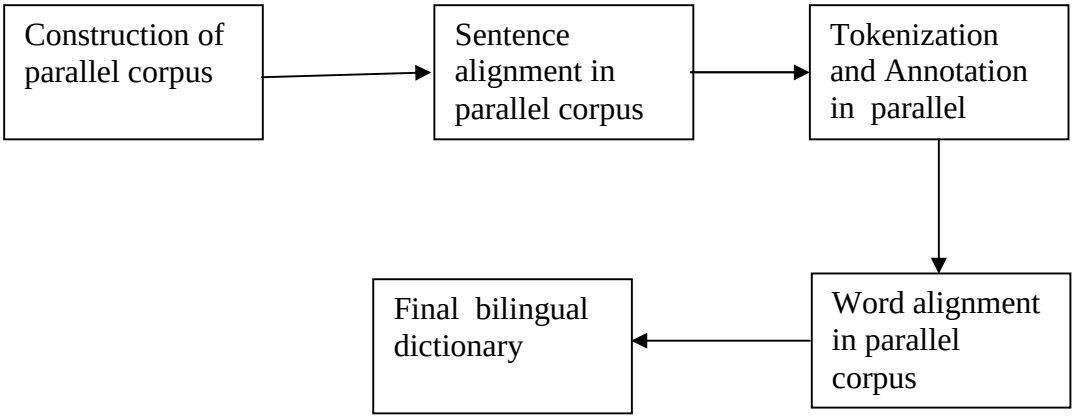


Fig 1.1: Stage in Automatic Dictionary Construction

1.5.1 Construction of parallel corpus

The documents in both languages are collected from the different sources to make the parallel corpus. One of the most used resources for the construction of parallel corpus is the web for the collection of documents in multiple languages.

1.5.2 Sentence alignment in Parallel Corpus

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

1.5.3 Tokenization

Tokenization is a very important task in corpus processing. It refers to the isolation of word units called tokens from text and further separation of punctuation marks, numbers and so on.

1.5.4 Annotation and Categorization

Corpus annotation is the process of attaching special codes (Tags) to words in order to indicate their special features. Tagging may incorporate linguistic information and depending on the linguistic information that is employed, different methods can be used. The most popular and common annotation methods used is Part-of-Speech (POS) annotation.

Category refers to the broad word class that consists of those words that fall in same classes. Less number of categories on the corpus gives the more accurate result in statistical models. But the case is opposite in linguistic model or rule based approaches. Our dissertation mainly focuses on the statistical method so we use categorization of word at higher level of abstraction.

1.5.5 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.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.

1.5.5.1 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 contains instructions for an operation to be performed, and a context describing where that rule should be applied. The operation to be performed alters the list of tags associated with an ambiguously-tagged word in such a way that one or more potential tags are eliminated from consideration, reducing the ambiguity. For instance, a rule for an English tagger might state that where one of the potential tags for a word is

infinitive verb, that reading should be removed *if* the preceding word is not tagged as 1) a modal verb, 2) the primary verb “do”, or 3) the infinitive marker “to”. This rule takes advantage of the known restriction of the infinitive form of the verb in English to these contexts. Different system used different computational implementations, which in turn allow different types of rules to be incorporated into the system.

As can be inferred from the foregoing brief description, taking a “rule-based” approach to disambiguation in tagging does not imply using grammar rules as traditionally formulated by linguists. Disambiguation, rule-based or otherwise, typically makes use of short-range information, as mentioned above.

1.5.5.2 Probabilistic Approach

The basic principle of probabilistic approaches is that statistical information concerning the frequency with which sequences of tags occur is gathered from long stretches of running text. This data is used to deduce which of the optional analyses of an ambiguously tagged word is the more likely to be correct. For instance, acquiring frequency statistics on a tagged corpus of English, a system might discover that the tag for a subject pronoun is followed by the tag for a verb 70% of the time, the tag for an adverb 29% of the time, and the tag for a noun 1% of the time. If that system, during the course of tagging, then encounters a word following a subject pronoun that was ambiguously tagged as either noun or verb, it can use its statistical knowledge to deduce that the verb tag is most likely to be correct

In practice, a model as primitive as the example here would be incapable of handling long sequences of ambiguous tokens and would be unlikely to perform particularly well. Thus, modern stochastic taggers utilize a mathematically more sophisticated approach known as a Markov model. Markov models allow the calculation of the probabilities of different tag sequences by combining different tag transition probabilities. The mathematics of Markov models are discussed in detail by Charniak et al. [8].

The most immediate advantage of a stochastic system over rule-based systems is that the linguist does not have to write an effective set of rules to produce an effective system. As Brill [5] puts it “the appeal of stochastic techniques over traditional rule-

based techniques comes from the ease with which the necessary statistics can be automatically acquired and the fact that very little handcrafted knowledge need to be built into the system". Probabilistic systems also represented a step forward in accuracy over early rule-based taggers. A further advantage is that they were in general more widely applicable.

Since in this dissertation, a probabilistic approach based on Hidden Markov model is used. Therefore, in this section, some examples of Markov model taggers will be considered in length, some which require tagged training data and some which do not.

1.6 Alignment of words

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

1.7 Application overview of bilingual dictionary

1.7.1 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. 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.

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.

Since core dictionary has the entries in both source and target language word, the disambiguaty can be removed in some extent.

1.7.2 Cross 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.

1.7.3 Multilingual Query Translation

In contrast query translation translates the query into all target document languages and then monolingual retrieval is performed separately for each document language. This approach is most commonly used as it is much easier to implement it and the only requirement is a tool for the translation of the query text, usually a machine readable bilingual dictionary.

Machine translation based approach uses existing machine translation techniques to perform automatic translation of the queries. The application of this approach is

simple but the quality of the results is not very satisfying. The reason for that appears to be the fact that queries usually do not contain enough contextual information that is necessary to machine translation in order to achieve word sense disambiguation. The main idea in a dictionary based approach is to replace each term of the query with the equivalent term or set of terms in the desired language. The equivalent terms are looked up into a bilingual dictionary. This is the most popular approach because of the simplicity of its application and the existence of a variety of machine readable bilingual dictionaries.

CHAPTER II

BACKGROUND AND PROBLEM DEFINITION

2.1 Background

According to the state of the art there are no methods that could enable the wholly automatic production of dictionaries. The method we propose is based on statistical word alignment on sentence aligned parallel corpora. Although this approach has been widely used by the machine translation community for at least 16 years [36] to improve the quality of dictionaries for machine translation purposes.

Dictionaries are an important part of natural language processing tasks and linguistic work. Domain-specific dictionaries can for example be used in cross-language web and intranet search engines. Creating dictionaries manually is labor intensive and time consuming, and many methods to make this process automatic have been proposed [10]. Word alignment tools are often used for the creation of bilingual word lists [16]. Many assumptions about the characteristics of words and their translations for extracting bilingual vocabulary underlie the algorithms in such tools, and parallel or comparable corpora are needed as input. However, finding such corpora is often a difficult and arduous task, especially for small languages. The Internet is a useful resource for finding corpora in different languages, and many large corporations and organizations have abundant information in multilingual web sites. However, these text sets are often noisy, containing a lot of non-parallel parts which need to be removed in order to create useful parallel corpora.

Dictionaries are very useful in most of the natural language processing task such as word sense disambiguation, language translation, and search engines.

The task of writing a bilingual dictionary might be conceived of as assigning the relevant language units of the target language (TL) to the relevant language units of the source language (SL). These language units ideally can be characterized as form-meaning pairs, and are usually referred to as *lexical units*. According to [2].

A headword in one of its senses is a lexical unit (or LU) [...]. LUs are the core building blocks of dictionary entries." Thus the dictionary building process includes the characterization of the LUs to be included in the dictionary, and the selection of the most appropriate pairings between the source language and target language LUs. In some cases source language and target language LUs are described fully independently [22], in other cases only the source language LU list is built and the target language equivalents are produced by the means of translation afterwards. In either case the relation of *translational equivalence* has to hold between the corresponding entries. However, finding the ideal translations is not at all obvious, as[2]: *„The perfect translation – where an SL word exactly matches a TL word – is rare in general language, except for the names of objects in the real world (natural kind terms, artefacts, places, etc.)"*. Moreover, in the case of *encoding dictionaries* (i.e. dictionaries providing speakers of the SL with information on how to express themselves in a foreign language) relevant contextual information of the TL also has to be included in the dictionary to give hints to users on how a TL expression should be used correctly.

2.2 Problem Definition

The automations of bilingual dictionary extraction from parallel corpus mostly depends upon the word alignment approaches.

Formally, the following definition of alignment at word 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 word 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 word e_i in position i to a word n_j in position j=a_i. The alignment $a_1^m = a_1 \ldots a_i \ldots a_m$ may contain

alignment $a_i = 0$ with the empty word n_0 to account for source words that are not aligned with any target word.

After finding the best alignment between the words in two languages, the problem of dictionary creation is to list the all aligned target word to the source word with the highest probability taken from the training corpus.

2.3 Approaches on dictionary creation

Word alignment methods enable the unsupervised learning of word pairs from sentence-aligned corpora. As stated above, one of the main advantages of using word alignment for the purpose of dictionary creation is that it helps to eliminate human intuition during dictionary building. On the other hand, it exploits parallel corpora, that is, as opposed to other techniques it does not presume the existence of refined resources (e.g. monolingual explanatory dictionaries, sense-inventories characterized on the basis of monolingual corpora.

Word alignment aims at finding alignment links between words in a parallel corpus. Bilingual lexicon extraction goes further: its goal is to identify the lexical word type links based on alignment between word tokens. Thus, dictionary extraction might be decomposed into two basic steps:

(1) The text/sentence alignment of the parallel corpus is extended to a word alignment.

(2) Some criterion is used (e. g. frequency) to select the aligned pairs for which there is enough evidence to include them in a bilingual dictionary.

The word alignment algorithms can be classified into two broad categories: ***association approaches*** **and** ***estimation approaches***

2.3.1 Association approaches

Methods following this approach employ heuristics that most of times are based either on the co-occurrence measures or on string similarity measures of words in the two languages.

2.3.1.1 Co-occurrence measures

Co-occurrence measures presuppose that the texts are sentence aligned and they are based on the idea of counting the frequency of word pairs that co-occurred in the aligned sentences. This frequency is then used in association measures for the identification of word correspondences.

One statistical association measure of co occurrence is to test if co-occurrence of pair of words appears considerably more then it word be expected, based on chances.

Another method of co-occurrence measure is by using the Dice coefficient which is used to measure the correction between discrete events. In this case the occurrence of two words in one text and its translation. The Dice coefficient takes a value between 0 and 1(0, 1) with 1 representing the highest probability of one word being a translation of the other.

A third statistical association measure is mutual information derived from information theory and is a quality that measures the mutual dependence of two random variables. In the case of word alignment it measures the amount of common information between two words. The idea behind it is that words that are assumed to have a lot of information in common are likely to be translations of one another.

2.3.1.2 String Similarity Measures

Another method for alignment is using string similarity measure. String similarity algorithms can be used to compare the number of common characters of two words.

One algorithm that employs this idea of character comparison is the longest common subsequence algorithm. By using this algorithm, a longest common substance ratio can be calculated and therefore a comparison between a pair of language with different characters of both languages is employed in parallel.

Another method utilized for string similarity measure is the N-grams approach is the grouping of words that contain many common substrings of N subsequent characters. In this way the character structure of the word is compared and used to find pairs or words and word variants.

2.3.2 Estimation approach

Estimation approaches to word alignment are inspired by statistical machine translation. Statistical machine translation is an application of the noisy channel model from information theory [32] to the task of machine translation. In what follows, the detail description of how the noisy channel model can be used for the purpose of word alignment can be found in [16,20].

Estimation approach makes use of parallel corpora to estimate probabilistic alignment models. This approach has been influenced by statistical approaches in machine translation [9] and it is used to handle words that do not have an equivalent correspondence in the other language .In estimation approach, alignment is modeled as hidden connections in statistical translation model, where each word in a target language string is connected to not more than one word in the source language.

2.4 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:

$$\begin{aligned}\hat{n} &= \arg\max_n \{P(n \mid e)\} \\ &= \arg\max_n \{P(n) * P(e \mid n)\}\end{aligned}$$

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 [12] 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.

2.4.1 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:

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

Model-2: Distortion or Alignment Parameter

Given source and target sentence lengths J and I, probability that i^{th} target word is connected to j^{th} source word, the distortion probability is given as $P(i \mid j, I, J)$. Now, the probability of an alignment $\boldsymbol{a}$, given the target sentence and the lengths of the source and target sentences is:

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

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 $\boldsymbol{i}$.

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 [2] 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 French words) is modeled. The full description of Model 4 of [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].

2.5 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:

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.

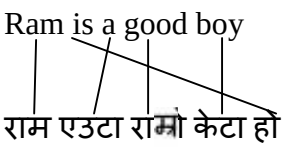
Unlike European languages, most of the Indian languages are morphologically rich and have the feature of compounding, thereby making the problem different in terms of SMT. When we are trying to align two European languages, we are much more likely to get one-to-one alignments, but when at least one of 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.

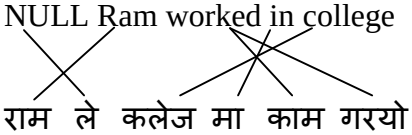
Apart from compounding, tense, aspect and modality (TAM) of Indian language verbs also are a cause of errors in alignment. This is because the TAM information is distributed over several words, which causes problems for the fertility model. This is, in fact, one of major factors in reducing the alignment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

Example -1. Translation (one to one alignment):



Example -2. Distortion (word order) and NULL insertion (‘spurious’ words):



Example -3. Fertility:(one to many alignment)

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

Chapter III

IMPLEMENTATION

This chapter describes the implementation detail of the Model described in the previous chapter. The implementation of the model is done using java programming language and several APIs created to represent the data structures and tagging, and alignment operations. These are the constructs that are used for the implementation of the Model.

3.1 Specification of the model

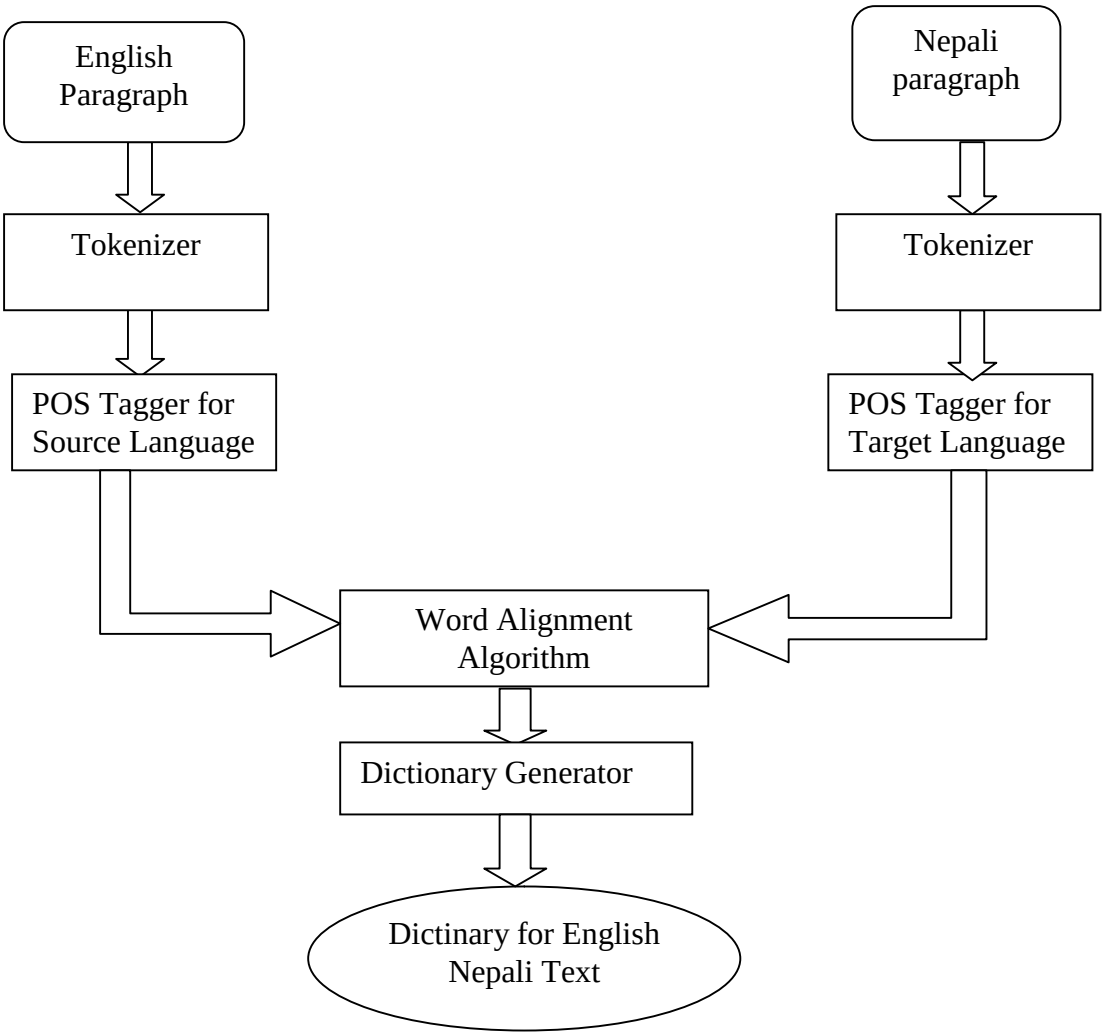


Fig 3.1: work flow model of the dictionary construction

3.2 Description of the model

The model works mainly in four steps. They are Tokenizing, tagging word alignment and dictionary generation from aligned word list. The model first takes a paragraphs pair on both languages and feeds into the tokenizer. Tokenizer separates the words in each paragraph in each language. Then this annoted paragraphs feeds to the separate taggers for each language used by the model. Here TnT tagger [33] is used as a POS tagger for providing the part-of-speech category to the every words of the paragraph. The aligner will then align the words of both paragraphs based on the knowledge base gathered from the training of the bilingual corpus and the different heuristics of the linguistics. Finally the dictionary generator searches the source word in the alignment table and finds out all its correspondening target words in the target language and lists them as a form of dictionary.

3.2.1 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.

3.2.2. Tokenization

Here we built tokenization of words in given parallel paragraph of English and nepali with the regular pattern matching class of Java programming language. The pattern described in regular expression is matched with built in the regular expression class in java.

3.2.3 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.

3.2.4. 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.

3.2.5 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 [30] 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 Language

The detail description of Nepali tagset used here is found in [30].

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 Langauge

3.3 Alignment

Based on the bilingual sentence aligned training corpus, the word annotated sentence pair as test data can be aligned at word level. Among the different alignment between the word 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.

3.3.1 Expectation Maximization: The Intuition

The key intuition behind EM is this: If we know the number of times a word aligns with another in the corpus, we can calculate the word translation probabilities easily. Conversely, if we know the word translation probabilities, it should be possible to find the probability of various alignments. Apparently we are faced with a chicken-and-egg problem! However, if we start with some uniform word translation probabilities and calculate alignment probabilities, and then use these alignment probabilities to get (hopefully) better translation probabilities, and keep on doing this, we should converge on some good values. This iterative procedure, which is called the *Expectation-Maximization* algorithm, works because words that are actually translations of each other, co-occur in the sentence-aligned corpus [28].

3.3.2 Expectation Maximization (EM) Algorithm for Training

For calculating the parameters of word algorithm, 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

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***.

3.3.3 Alignment Algorithm

The translation probability can be decomposed as:

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

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

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 word respectively.

Hence the best alignment can be defined as:

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

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. Each factor looks like $P(e_j \mid n_{a_j})$. Suppose that the English word 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 word translation table is generated from the training of bilingual corpus by EM algorithm.

The alignment algorithm described above can be summarized as:

Let $e = e_{w_1} ... e_{w_i} .....e_{w_m}$ and $n = n_{w_1} ... n_{wj} .....n_{w_n}$ where e_{w_r} is English word and n_{w_r} is Nepali word

For each sentence pair in sentence aligned paragraphs

For each word e_{w_i} in English sentence

For each word n_{w_j} in Nepali sentence

If (category of e_{w_i} = = category of n_{w_j})

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

End For

End For

End for

3.4 Word class and category

Words that function similarly with respect to what can occur nearby or with respect to the affixes they take are grouped into classes. For example the noun category consists of both proper noun and common noun. The word categories used in the above alignment algorithm are described following table.

Word category	English Tag	Nepali Tag
Noun	NN NNS NP NPS	NN, NNP
Pronoun	PP PP$ WP WP$	PP, PP$,PPR, DM, DUM
Verb	VB VBD VBG VBN VBP VBZ	VBF, VBX, VBI, VBNE VBKO, VBO
Adjective	JJ JJR JJS	JJ ,JJM, JJD
Adverb	RB RBR RBS WRB	RBM ,RBO
Interjection	UH	UH
Determiner	DT	DT
Conjunction	IN CC	CC ,CS
Foreign Word	FW	FW
others	EX, LS, MD, PDT, POS TO , RP, , WDT	FB, ALPH, YB, YQ, YM, YF, UNW, RP, CL, QW, HRU, POP, PKO, PLAI, PLE, INTF
Number	CD	CD ,OD
Symbol	SYM	SYM

Table 3.3: List of word categories used for both language

3.5 Dictionary generation algorithm:

From the table of source word and target word generated by above alignment algorithm, dictionary generation algorithm extracts the word pair from both languages. The idea is that the algorithms takes first the source word from the table and list its corresponding pair in target language. If the source word is already listed in the dictionary then it will add its meaning in the list of target word i.e the complexity of this algorithm will be linear. The pseudo code of this procedure can be summarized as follows

For each source words of table

    If(source word is already present in the dictionary)

        Add target word as its meaning in the list

    Else

        Add source word in the dictionary and add target word as its meaning

    End if

End for

CHAPTER IV

TESTING AND ANALYSIS

In this chapter we will measure the accuracy of the model. The alignment accuracy not only depends on the performance of aligner of the model but also it depends on performance of other former tools tokenizer and tagger. The evaluation of the dictionary given by the model is done by comparing the dictionary prepared by the human annotators.

There are different ways to evaluate extracted dictionaries. Some of the most common ways are the use of gold standards. The gold standard method is based on recall and precision evaluation metrics.

4.1 Gold standards

The evaluation of alignment output can be performed by comparing it to gold standards (also called reference data) which is constructed before the alignment process takes place. Gold standards are consisted of sample text and its equivalent in the target languages that is pre-linked by the reviewers and then it is used to test the alignment results automatically. There are two approaches used with gold standards. The first approach of performing a complete alignment of the sample, breaks down to segments the sentences in the source and target languages and then the translation equivalences are marked. The second approach is using the “translation spotting” method. In this method a number of words or phrases are extracted from the source text and then all the sentences of the target text that contain these words or phrases are presented to the reviewer in order to choose the corresponding target word or phrase and compare the equivalences.

Here we use the method of statistical measure to evaluate the dictionary performance. This measure are very similar to those used by Och and Ney in [26].They used the performance of word aligner. we extend the same idea to the dictionary since dictionary also depends upon the performance of word alignment.

Evaluation of the output can be performed by expert who performs the evaluation after alignment. Matrices for evaluation are recall and precision.

Precision is defined as the ratio of the correct translation over the sum of all translations.

$$\Pr ecision = \frac{Number\ of\ \ correctly\ alignd\ \ words}{Number\ of\ \ possible\ correct\ \ words}$$

Recall is defined as the ratio of the correct translations to the possible correct translation

$$\mathrm{Re} call = \frac{Number\ of\ \ correctly\ alignd\ \ words}{Number\ of\ \ obtained\ \ words}$$

Precision is the number of correct results divided by the number of all results returned by aligner and *Recall* is the number of correct results divided by the number of results that should have been aligned.

The F-measure can be interpreted as a harmonic mean of precision and recall. It is defined as

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

The average error rate(AER) is calculated as

$$AER\ \ = 1 -\ \ F\ \ -\ measure$$

4.2 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 word categories are defined for the group of words based on the rule defined on the model. And finally the alignment is done between the words of the both paragraph pair. The output of the program for some test cases is shown below:

4.3 Input /Output of program

English Paragraph

A teacher is sitting on the ground. He teaches in the school. He has a book. This book has a good song. A teacher teaches this lesson. he is singing a song on the ground.

Nepali Paragraph

एक शिक्षक चउर मा बसिरहेको छ । उ स्कुल मा पठाउछ । उ सग किताब छ। यो किताब सग एउटा राम्रो गित छ । एक शिक्षक ले यो पाठ पठाउछ । उ एउटा गित चउरमा बसेर गाईरहेको छ ।

Dictionary as output

Source word	Nepali Meaning generated
a	एक
teacher	शिक्षक
is	छ
Sitting	बसिरहेको
on	मा
the	छ
ground	चउर
he	उ
school	स्कुल
teaches	पठाउछ
has	सग
book	किताब
this	यो
good	राम्रो

song	**गत**
lesson	**पाठ**
singing	**गाईरहेको**

English Paragraph

I have a small house. This house has small garden. This garden is green. Garden has small lake.

Nepali Paragraph

म सग सानो घर छ । यो घर सग सानो बगैचा छ । बगैचा हरियो छ। बगैचा सग सानो तलाउ छ ।

Dictionary as output

Source Words	**Nepali Meaning Generated**
I	म
has	सग
small	सानो
house	घर
is	छ
this	यो
garden	बगैचा
green	हरियो

English Paragraph

A boy is going to school. He is carrying a bag. He is in school dress. He is carrying books.

Nepali paragraph

एउटा केटो स्कुल गइरहेको छ । उसले झोला बोकेको छ । उ स्कुल पोशाक मा छ । उसले किताब बोकेको छ ।

Dictionary Generated as output

Source Words	**Nepali Meaning Generated**
a	**एउटा**
boy	**केटो**
school	**स्कुल**
going	**गइरहेको**
is	**छ**
He	**उसले, उ**
Bag	**झोला**
Carrying	**बोकेको**
dress	**पोशाक**
in	**मा**
book	**किताब**

English Paragraph

My collage has fifty teachers. Five teacher teaches English. Ten teacher teaches Maths. Ten teachers teaches computer science.

Nepali Paragraph

मेरो कलेज मा पचास जना शिक्षक छन । पाच जना ले इगलिस पढाउछन । दश जना ले गणित पढाउछन। दश जना ले कम्प्युटर बिग्यान पढाउछन ।

Dictionary generated as output

Source Words	Nepali Meanings generated
my	मेरो
collage	कलेज

has	मा, छन
fifty	पचास
teacher	शिक्षक
Five	पाच
teacher	शिक्षक
teaches	पढाउछन
english	इगलिस
Ten	दश, जना
Maths	गणित
computer	कम्प्युटर
science	बिग्यान

English Paragraph

Nepal has five development regions. Nepal has fourteen zones. Nepal has seventy five districts. Kathmandu is the capital of Nepal.

Nepali Paragraph

नेपाल मा पाच विकास छेत्र छन। नेपाल मा चोध अञ्चल छन । नेपाल मा पचहतर जिल्ला छन । नेपाल को राजधानि काठमाडौ हो ।

Dictionary generated as output

Source Words	**Nepali Meaning generated**
Nepal	नेपाल
has	मा, छन
five	पाच
development	विकास
region	छेत्र
fourteen	चोध
zones	अञ्चल
seventy five	पचहतर
district	जिल्ला
Kathmandu	काठमाडौ
is	हो
capital	राजधानि

4.4 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 paragraph and finding the average of them.

Precision= 70%

Recall= 73%

F-score = 71.46 %

AER=28.54%

This result shows that the proposed model works well for the given input. In our analysis the recall is shown greater than precision it means that precision gives the comparison of alignment accuracy when it is compared to the result given by the model and Recall gives the accuracy of the alignment when compared with the actual alignment given by the human annotator.

4.5 Limitation of proposed model

Since our model uses the small manually formed training corpus which has limited number of words, some time the word in the input paragraph will not found in training corpus and hence they will have no translation probability and does not get aligned with target word. These words are called Unknown word and hence they should be handled by providing sufficient large and quality training corpus and other approach which are useful for handling the unknown words.

Another limitation is that our model depend upon the performance of tagger. If the tagger tags the word incorrectly then they will fall in wrong category and hence ultimately they will not be aligned with any target words. These word are classified as Null aligned word. These can be removed by using good quality tagger with sufficiently large tagged training corpus to train the tagger.

CHAPTER V

CONCLUSION AND FUTURE WORK

5.1 Conclusions

In this dissertation , a corpus-driven technique for the automatic creation of bilingual English –Nepali dictionary is proposed. The proposed automatic method uses the statistical methods of word alignment. The model first does the tokenization of given paragraph in both English Nepali text. Then these paragraphs are tagged with the famous TnT tagger. After the tagging the categorization procedure is done to classify the common word class in a category. Then the alignment model uses these information to align the corresponding source and target words. For this propose The model gather the parameter from the bilingual training corpus. The dictionary is thus generated by listing these source word and target words. So its efficiency mostly relay on the size of training corpus as well as its quality. This model uses the tagging and categorization of words to disambiguate the word sense so tagger performance should also considered when evaluating the accuracy of the model. The developed dictionary may be useful for the number of Natural language processing task such as cross-information retrieval, multilingual document retrieval and multilingual web searching. The proposed method renders the generation of reversed dictionary more straightforward, since the word alignment has to be re-applied in the opposite direction.

However, one principle bottleneck of the approach is that the availability of parallel corpora is not easy since it is tedious task to create the parallel corpora. Further refinement for the parameters also has to be carried out to increase the coverage of proposed dictionary. The other limitation to the model arises due to the performance of other tools that are used in the model.

5.2 Further Recommendation

We propose the statistical method to create bilingual dictionary that mostly rely on the word alignment model. The word alignment model efficiency is based on the estimated parameter for the translation probabilities so the further work can increase the efficiency of the model by considering the large size of corpus. The model we described has not included the sufficient method of hardly the unknown word. so it can be improved on further research more. Here we proposed another variation on the categorization of word classes. This technique can be extended further to get the much reliable and efficient dictionary.

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] B.T.S Atkins, M Rundell, *The Oxford Guide to Practical Lexicography*. Oxford: Oxford University Press PP 162-163

[3] [BM Bhandari] *A survey of Dictionary making* English Journal of NELTA, vol s1 no 1-2.

[4] 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

[5] E Brill, *A simple rule-based part of speech tagger*. In: Proceedings of the Third Conference on Applied Natural Language Processing *(ANLP'92),* Trento, 1992.

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

[8] E. Charniak, , C . Hendrickson, N. Jacobson, M. Perkowitz,., *Equations for part of speech tagging*. In: Proceedings of the Eleventh National Conference on Artificial Intelligence. Menlo Park: AAAI Press/MIT Press, 1993.

[9] 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

[10] H. Dalianis, S. Vulupillai *Automatic construction of Dictionaries on sparse parallel corpora in the Nordic languages* ,proceeding of the workshop on Multi-source multilingual information Extraction and summarization, pages 10-16, Manchester August 2008.

[11] H. Dalianis, H.xing, Zhang *creating a Reusable English-Chinese parallel corpus for Bilingual Dictionary construction.*

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

[14] 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.

[15] 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.

[16] Hiemstra, D. *Using statistical methods to create a bilingual dictionary*. Master's Thesis,(1996). University of Twente.

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

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

[19] 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

[20] Manning D. C. and Schütze, H. *Foundations of Statistical Natural Language Processing*, (1999). MIT Press, Massachusetts, US.

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

[22] Martin, W. (2007) Government Policy and the Planning and Production of Bilingual Dictionaries : The 'Dutch' Approach as a Case in Point, International Journal of Lexicography, September 1, 20(3): 221—237

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

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

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

[26] 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

[27] A..Ramanathan, *Stastistical Machine Tanslation, Ph.D. S*iminar Report, Dapartment of Computer Science and Engineering IIT, Bombay

[28] Ribeiro, A., Pereira Lopes, J., G., Mexia, J. . *Extracting Equivalents from Aligned Parallel Texts: Comparison of Measures of Similarity*. (2000) IBERAMIA-SBIA: 339--349

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

[30] 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.

[31] Shannon, C., E. *A mathematical theory of communication. Bell Systems Technical Journal* (1948) 27. PP 379--423.

[32] B. Thorsten, *TnT- Statistical Part-of-Speech Tagger*. http://www.coli.uni-saarland.de/~thorsten/tnt/

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

[34] Varma, N. (2002). *Identifying Word Translations in Parallel Corpora Using Measures of Association.* Master's Thesis, University of Minnesota.

[35][wu 1994] Wu, D. (1994), *Learning an English-Chinese Lexicon from a Parallel Corpus. In: Proceedings of AMTA'94*. 206--213.

[36] 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

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

[38] 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).

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
a teacher is sitting on the ground
he teaches in the school
he has a book
this book has a good song
he is singing a song on the ground
nepal has five development regions
nepal has fourteen zones
nepal has seventy five districts

kathmandu is the capital of Nepal
my collage has fifty teachers
five teacher teaches english
ten teacher teaches Maths
ten teachers teaches computer science
he is carrying a bag.
he is in school dress.

Nepali Corpus

यो किताब टेबुल मा छ

यो एक किताब हो

केटाहरु स्कुल गईरहेको छन्

एक केटा स्कुल गईरहेको छ

एक केटा टेबुल मा बसिरहेको छ

एक केटा गीत गाईरहेको छ

यो राम्रो 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 शिक्षक हु

एक शिक्षक 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

एक शिक्षक चउर मा बसिरहेको छ
उ स्कुल मा पठाउछ

उ सग किताब छ

यो किताब सग एक राम्रो गित छ

एक शिक्षक यो पाठ पठाउछ

उ एक गित चउरमा गाईरहेको छ

नेपाल मा पाच विकास छेत्र छन

नेपाल मा चोध अञ्चल छन

नेपाल मा पचहतर जिल्ला छन

नेपाल को राजधानि काठमाडौ हो

मेरो कलेज मा पचास जना शिक्षक छन

पाच जना ले इगलिस पढाउछन

दश जना ले गणित पढाउछन

दश जना ले कम्प्युटर बिग्यान पढाउछन

उसले झोला बोकेको छ

उ स्कुल पोशाक मा छ

उसले किताब बोकेको 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 शिक्षक धेरै प्रख्यात छ ।
this song is very popular .	यो गीत धेरै प्रख्यात छ ।
a teacher is sitting on the ground .	एक शिक्षक चउर मा बसिरहेको छ ।
boys are singing on the ground .	केटाहरु चउर मा छन् गाईरहेको ।
a teacher is going to school .	एक शिक्षक 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 शिक्षक 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 शिक्षक 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 शिक्षक छ ।
he is a good boy .	ऊ एक केटा राम्रो छ ।
this boy is a good teacher .	यो केटा एक शिक्षक राम्रो 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 पढ्छ ।
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 शिक्षक राम्रो ।
i am a good boy .	म एक केटा राम्रो हु ।
a teacher reads this lesson .	एक शिक्षक यो पाठ पढ्छ ।
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 शिक्षक चउर मा बसिरहेको छ ।
he is popular .	ऊ प्रख्यात छ ।
school is very good .	स्कुल धेरै राम्रो छ ।
this book is very good.	यो किताब धेरै राम्रो छ ।
nepal has five development regions .	नेपाल मा पाच विकास छेत्र छन।
nepal has fourteen zones .	नेपाल मा चोध अञ्चल छन।
nepal has seventy five districts.	नेपाल मा पचहतर जिल्ला छन।
kathmandu is the capital of Nepal .	नेपाल को राजधानि काठमाडौ हो ।
my collage has fifty teachers.	मेरो कलेज मा पचास जना शिक्षक छन ।
five teacher teaches English.	पाच जना ले इगलिस पढाउछन ।

ten teacher teaches Maths.	दश जना ले गणित पढाउछन ।
ten teachers teaches computer science.	दश जना ले कम्प्युटर बिग्यान पढाउछ ।
I have a small house.	म सग सानो घर छ ।
this house has small garden.	यो घर सग सानो बगैचा छ ।
this garden is green.	***यो बगैचा हरयो छ।***
garden has small lake.	बगैचा सग सानो तलाउ छ ।
a boy is going to school	**एक केटो स्कुल गइरहेको छ ।**
he is carrying a bag.	उसले झोला बोकेको छ ।
he is in school dress.	**उ स्कुल पोशाक मा छ ।**
he is carrying books.	**उसले किताब बोकेको छ ।**

Appendix C

Code of Implementation

Appendix 1 Source Code to make parallel corpus

```
public void makeParallelCorpus() throws IOException
      {
            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);
            }
      }
```

Appendix 2 Source code to make the word pair

```
public void makeWordPair()//collect the word types from source and
related words
                                                //from target corpora
      {
            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);
                  }
            }
      }
```

Appendix 3 Source code of 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<WordPair,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];

                              //commented heere
                    //System.out.println(word);

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

                                    //commented here
                        //System.out.println(sword);

                                    WordPair wp = new
WordPair(sword,tword);

                        //commneted here

      /*System.out.println(wp.getSourceWord());

      System.out.println(wp.getTargetWord());

      System.out.println(translationProbability.get(wp.toString()));

      System.out.println(translationProbability.containsKey(wp.toStri
ng()));
                                                                  */
                        //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
               }

      }
```

Appendix 4 Source code to make the word Dictionary

```
public void displayTranslationProbability()
      {
            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].getTargetWor
d());
            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].getTarge
tWord()+"):\t"+tProb);
            }


              //implementing in chunk level
        int count = 0;
        //String inputEng[] = new String[tblForEnglishChunk.size()-
1];
```

```
        String inputEng[] = arg1[0].split(" ");
        //String inputNep[] = new String[tblForNepaliChunk.size()];
        String inputNep[] = arg1[1].split(" ");



        //array to put the obiatned tarined value and find the
maximum from it
        double[] maxVal = new
double[inputNep.length];//[tblForNepaliChunk.size()];
        for(int i=0; i<maxVal.length; i++)
            maxVal[i] = 0.0;



        //starts the new code here
        String testAns[] = new String[inputEng.length];
        count = 0;
        for(int i=0; i<inputEng.length; i++)
        {
            double val = 0.0;
            double mul = 0.0;
            for(int j=0; j<inputNep.length; j++)
            {
                WordPair checkWP = new
WordPair(inputEng[i].toLowerCase(), inputNep[j]);
                if(translationProbability.get(checkWP.toString()) !=
null)
                    val =
translationProbability.get(checkWP.toString());
                else
                    val = 0.0001;
                if(val > mul)
                {
                    mul = val;
                    testAns[count] = inputEng[i] +"/" + inputNep[j];
                }
            }
            count ++;
        }
        //ends the new code here
System.out.println("heloo ");



        //make the matrix
        JFrame frmMatrix = new JFrame();
        frmMatrix.setTitle("Matrix Form");
        JTable tblMatrix;
        DefaultTableModel model;
        Object colNames[] = new Object[inputEng.length];//
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);
String strEng[] = new String[inputEng.length];
String strNep[] = new String[inputEng.length];
for(int i=0; i<testAns.length; i++)
{
    String[] t = testAns[i].split("/");
    strEng[i] = t[0];
    strNep[i] = t[1];
}
```