CHAPTER 1

POS TAGGERS: AN OVERVIEW

1.0. INTRODUCTION

Parts-of-speech tagging or better known as POS tagging has a significant role in the research area of Natural Language Processing (NLP) as it provides a large amount of information of a lexical word in a sentence as well as its neighboring words. A POS tagger automatically annotates each word in a natural language sentence of a corpus with appropriate parts-of-speech marker. In other words, a tagger plays a crucial role in the research work in natural language processing areas of Machine Translation (MT), Information Retrieval and Extraction, Speech Synthesis and Recognition, Word Sense Disambiguation (WSD) and other related areas. Furthermore, it is the primary prerequisite to develop a parser which serves as a pre-processor for syntactic parsers and has major contributions in their efficiency. So, for all these applications, a tagger with maximum possible accuracy is needed as POS tagging forms the first step in natural language understanding.

The main aim of the chapter is two-fold: (i) to provide required background to the present research and (ii) to describe the methodology and the tools used for the current research to provide the background to the thesis. The chapter has been organized into the following sections. The first section elaborates on the various techniques and approaches used to build a POS tagger. The next two sections provide a brief review on the tagging
approaches used in POS tagging for the corpora of different Indian languages in general and Bangla in specific. The fourth section states the proposed hypothesis for the present work followed by the objective of the dissertation. The next section deals with methodology and tools. It provides an insight into the entire process of the creation of the Hybrid model and elaborates on the various programming tools along with the corpus and the tagset used in developing the Hybrid POS tagger in Bangla. The concluding section presents an overview of the following chapters of the dissertation.

1.1. APPROACHES AND METHODS TO BUILD POS TAGGERS

Several approaches and methods have been proposed and implemented over the years to develop POS taggers for different languages. Out of them, the linguistic approach or the technique of writing hand-crafted rules for disambiguation forms the standard approach explored by Green and Rubin, 1971, Klein and Simmons, 1963 between mid sixties to mid seventies (Cutting et al, 1992; Dandapat, 2009). As it has been mentioned before, TAGGIT, a rule-based tagger developed by Greene and Rubin in 1971, which has been used to tag the Brown corpus (Dandapat, 2009), and ENGTWOL, a constrained grammar based tagger in English, built by Karlson et al. in 1995 (Jurafsky &Martin, 2000) represents the classical approach of manual tagging. The machine learning approaches like the Hidden Markov Models (Brants, 2000; Collins, 2002), CRF (Lafferty et al, 2001), decision tree model (Màrquez et al, 1998), Maximum Entropy Model (Ratnaparkhi, 1996), Support Vector Machines (M`arquez and Gim`enez, 2004) adopt annotated corpus to learn the advanced and complex features of different languages for POS tagging. These complex features are encoded by making use of probabilistic distribution, disambiguation rules, decision trees and so on. However, the machine
learning algorithms need a good amount of annotated training data to achieve high accuracy levels. But, most of the languages lack the availability of annotated corpus. Hence, the unsupervised and semi-supervised learning techniques have been adopted for POS tagging which enable the researchers to work on unannotated corpus as well.

There are three primary methods of tagging, namely, manual, automatic and hybrid (Guilder, 1995). Manual tagging involves tagging each word of a corpus by hand. A corpus is annotated manually by two or more people based on a particular tagset decided after mutual discussions. This is a laborious, time consuming and an infeasible job in case of a large corpus.

Guilder (1995) points out that an automated POS tagging employs different approaches to develop a tagger. The two primary approaches of automated POS tagging are:

a. Supervised approach

b. Unsupervised approach

The supervised method of tagging involves working with a pre-tagged corpora which serves as the basis for developing the tools to be used in the tagging process like, the tag dictionary, word or tag frequencies and the rule set (Guilder, 1995). The performance of such a tagger generally increases with the increase in the corpus size. On the other hand, the unsupervised tagging techniques do not require a pre-existing corpus but use computational methods, like the Baum-Welch algorithm to automatically induce tagsets (Guilder, 1995). These techniques utilize an untagged corpus for their training
data. The unsupervised techniques induce a dictionary, word frequencies, transformation rules, affix frequencies, and tag sequence probabilities to assign tags to the test data. The main advantage of using an automated approach is that it is very portable and has the potential of delivering good performance when trained and tested on the same genre of text. But, unfortunately, pre-tagged corpora are not readily available for many languages and genres. Both supervised and unsupervised methods of tagging are further classified into the **Rule-based approach**, **Stochastic or probability approach** and the **transformation-based** approach (Guilder, 1995). We shall look briefly at each of them.

**1.1.1. RULE-BASED APPROACH**

Rule-based POS tagging is the oldest and earliest approach to tag each word in a sentence using hand-written rules based on the dictionary to get possible tags for each word to be tagged. This framework of tagging has been introduced by Eric Brill in 1992 (Brill, 1992). The rule-based taggers implement linguistic rules to assign correct tags to all the words in a sentence or a text depending on the context. In case of ambiguity, the linguistic features of the preceding word and following word are used. The hand-crafted rules are devised with linguistic information and are used to select the correct tag. These manually written rules are also called as *context frame* rules, as they employ contextual information for the task of disambiguation (Dandapat et al, 2004), (Kumawat & Jain, 2015). So, the basic idea of rule-based tagging is to assign a list of potential tags to each word and then using a large list of hand-crafted disambiguation rules to narrow down this list of possible tags to a single POS tag for each word. TAGGIT is the first ever built rule-based tagger based on context-pattern rules and used a set of 71 tags and 3300 disambiguation rules (Brill, 1992). As mentioned in Brill (1992), several rule-based
taggers have been built by Klein and Simmons (1963), Greene and Rubin (1971). Brill (1992) developed a simple and robust rule-based tagger based on probabilistic models.

Brill’s tagger, as it is known as, has several advantages over stochastic taggers. It requires a limited amount of information or lexicon and a small set of meaningful rules as opposed to the large number of statistics required for stochastic taggers. The tagger automatically identifies its weaknesses and works upon them thereby improving its performance and it also has a better portability from one corpus genre or tagset to another.

Jurafsky and Martin (2000) states that ENGTWOL developed by Voutilainen in 1995, an English rule-based tagger is considered to be one of the most important taggers of the models based on constraint grammar architecture. It is based on the same two-stage architecture of the rule-based approach, but more sophisticated and advanced than the early algorithms. The ENGTWOL lexicon has two-level morphology which contains 56,000 entries for English word stems (Jurafsky & Martin, 2000). As described by Mohanty (2005), in the 1st stage, each word is run through a lexical transducer where the entries for the potential POS are returned and in the second stage, around 1100 constraints are applied to the input sentences to eliminate the incorrect parts of speech tags.

1.1.2. TRANSFORMATION BASED TAGGING

Transformation-based tagging, also known as Transformation-based Error-driven Learning (TEL) is a rule-based algorithm for automatic POS tagging of a given corpus (Brill, 1995). Eric Brill introduced this algorithm and developed the tagger known
as Brill’s tagger which showed that this approach can also be used for POS tagging with a high accuracy like the other machine learning approaches (Brill, 1995). This approach transforms one state to another implementing the transformation rules to find a suitable tag for each word. It extracts linguistic information automatically from the corpus, thus, allowing us to have the linguistic knowledge in a readable form. The transformation-based approach is based on the rules that specify the tags according to the context of the words and these rules are automatically induced from the training corpus using machine learning techniques (Brill, 1994).

Megyesi (1999) in her paper on Brill’s POS tagger describes how a transformation-based tagger works. A transformation-based tagger requires an initial state annotator, a set of transformation rules and an objective function for learning. The grammar is induced straight away from the training data without any human interference. A small, manually and correctly annotated training corpus serves as the input to the tagger. The initial annotator is a program that assigns each and every word its most probable tag according to the training corpus using a set of tagging rules. So, the program derives the contextual, morphological and lexical information from the training corpus and learns how to assign the most probable POS tag for a word. In the second stage, the transformation rules are examined and the one which gives the most accurate tagging gets selected and the data is retagged again according to the rules. Now, all these three steps are repeated and run through a cycle and the algorithm stops when it comes across a stopping criterion. After training the tagger, the tagger is first run by tagging an unannotated corpus with initial-state annotator, based on the tagset of the training corpus, and then, the rewrite rules are applied wherever they are required. But, one of the main
disadvantages of the Brill’s tagger is that it takes a lot of time to train a corpus and does not provide tag probabilities. However, the tagger’s biggest strength lies in its simplicity and robustness. Since, the set of learned rules are easy to understand, debugging and development becomes easier to make. A manual monitoring of the model of a statistical tagger is error-prone, tedious, and less useful, but a set of transformation rules is not only understandable and can be modified manually, but this tagger can be even used by people with no previous experience in NLP.

1.1.3. THE PROBABILISTIC APPROACH

The statistical or the machine learning frameworks are the most popular approaches for POS tagging. The probabilistic or stochastic approach is used to develop language models to disambiguate a word sequence by assigning the most frequent tag sequence. A simple stochastic tagger assumes that each word is known and has a finite set of potential tags which can be taken from a dictionary or through morphological analysis (Jurafsky & Martin, 2000). Nugues (2006) has described the stochastic tagger as,

When a word has more than one possible tag, statistical methods are used to determine the optimal sequence of POS tags like $T= t_1, t_2, t_3,... t_n$ in a given sequence of words $W= w_1, w_2, w_3, ..., w_n$.

(Nugues, 2006: 163)

So, according to Jurafsky and Martin (2000), the stochastic taggers resolve the tagging ambiguities by using a training corpus to calculate the probability of a given tagged word in a given context. But, the disadvantage of such tagger is that it may
generate a valid tag for a word as well as unacceptable tag sequences. However, the stochastic models have been widely used for POS tagging because of their simplicity.

Apart from using this word frequency approach, another alternative approach is also used to calculate the probability of the given sequence of tags occurring in a given sequence of words. This is known as the n-gram approach where the best tag for a given word is determined by the probability by which it occurs with the n previous tags. The Viterbi algorithm is one of the most common algorithms that implement the n-gram approach for tagging. According to Dandapat (2009), the stochastic tagger needs large amount of annotated text for its development. Stochastic taggers with more than 95% word-level accuracy have been developed for German, English and other European languages, for which large amount of labeled data is available.

1.1.3.1. HIDDEN MARKOV MODEL (HMM)

Among the stochastic models, the bi-gram and tri-gram Hidden Markov Models (HMM) are the most popular statistical tools for modeling generative sequences that can be described by an underlying process generating an observable sequence (Manning &Schütze, 1999; Collins, 2002). The HMM models combine both the approaches of word frequency measurements and tag sequence probabilities of stochastic tagging. The HMM taggers choose the most likely tag for a word given the previous tag. For instance, consider the following two sentences,

a. The/DET race/NN will/AUX start/VB in/PP an/DET hour/NN.

b. The/DET horse/NN is/VB expected/VB to/PP race/VB tomorrow/ADV.
As we can see from the above sentences, the word “race” needs to be tagged appropriately. Here, in the 1st sentence “race” is tagged as noun whereas, in the second one it is tagged as verb. Now, in order to choose between noun and verb, the probability between the two needs to be computed according to the preceding tag, i.e. the probability of “to” occurring with “race” is calculated. This is calculated from a corpus by counting and normalizing. So, it is seen that a verb has more probabilities to follow “to” than a noun. For instance, to eat, to sleep, to play are more common to find than ran to home, walk to school. Hence, based on this probability count, “race” is assigned as verb.

TNT (Trigrams and Tags) (Brants, 2000) is an efficient tri-gram statistical HMM tagger which uses a suffix analysis technique to predict the lexical probabilities for unknown tokens based on properties of the words in the training corpus which share the same suffix (Brants, 2000). The HMMs are significantly used for signal processing, speech processing, extracting target information from documents, POS tagging, and phrase chunking (Dandapat et al, 2004). However, Dandapat (2009) points out that the simple HMM models do not function efficiently when small amounts of labeled data corpus are used to estimate the model parameters. The bi-gram, and tri-gram HMM models like TNT, face difficulties in predicting the probabilities from a limited size of training data because they have a large number of parameters.

1.1.3.2. MAXIMUM ENTROPY MODEL (MEM)

Maximum Entropy Model (MEM) is another statistical model designed by Ratnaparkhi in 1996, for the tagging task which trains from an annotated corpus, simultaneously uses many contextual “features” to predict the POS tag and assigns them
to previously unseen corpus with an accuracy of almost 96%, which is termed as the “state-of-the-art accuracy” (Ratnaparkhi, 1996). So, this algorithm falls under the category of feature based classification algorithms for POS disambiguation, the other being the Conditional Random field (CRF) model. Mohnot et al (2014) points out that this model represents several features of a word and effectively deals with long term dependency, unlike the HMM model. This tagging algorithm is grounded on the axiom of maximum entropy modeling which predicts observations from a training corpus and selects the model with the maximum uniform distribution (Curran, 2004). The MEM model has been used in tagging the Wall Street Journal corpus (WSJC) from the Penn Treebank Project and it got trained from a largely annotated corpus with POS tags. Curran (2004) also reasoned that the Maximum Entropy Model is suitable for the tagging task of the Wall Street Journal corpus because it combines diverse forms of contextual information in an organized manner, and does not impose any distributional assumptions on the training data. Apart from tagging, the model has also been used in language modeling, prepositional phrase attachment and word morphology (Ratnaparkhi, 1996). One of the main drawbacks of this model is the “label bias problem”, in which the model is prone to prefer the states with less number of outgoing transitions (Mohnot et al, 2014).

1.1.3.3. Conditional Random Field (CRF)

Conditional Random field (CRF) or better known as CRF, designed by Lafferty et al in 2001 is a popular probabilistic framework for segmenting and labeling strings of sequential data. In other words, CRF is a probabilistic model commonly used for structured predictions (Sutton & McCallum, 2010). Dandapat (2009) defines a CRF algorithm as
“A CRF is an undirected graphical model that defines a single exponential model
over label sequence given the particular observation sequence”.

(Dandapat, 2009: 78)

In contrast to simple tagging models of HMM framework, the CRF model can
handle complex and overlapping features of words by computing the conditional
probability of the values on specified output nodes based on the values of the specified
input nodes (Ekbal et al, 2007). According to PVS (2007) and Kumar et al, (2010), it is a
form of undirected graphical model that defines a single log-linear distribution over label
sequences given a particular observation sequence. CRFs define conditional probability
distributions $P(Y|X)$ of label sequences given input sequences. As mentioned earlier, this
model is also classified as the feature-based algorithms like the Maximum Entropy
Model. Chen (2012) describes the functioning of a CRF model and states that in a CRF,
each feature function is a function that takes in a sentence $s$, the position $i$ of a word in
the sentence, the label $l_i$ of the current word, the label $l_{i-1}$ of the previous word as inputs
and generates the output in the form of a real-valued number of either 0 or 1. He further
illustrates the point that one possible feature function could calculate how much we
presume that the current word should be labeled as an adjective given that the previous
word is “very”. The CRF framework has quite a few advantages over the HMMs and
stochastic grammars (Lafferty et al, 2001). It includes the positive features of the
Maximum Entropy Markov models and also solves the „label bias problem” of the
MEMMs in a principled manner. The CRF English tagger was implemented based on
the theoretic model presented in Lafferty et al (2001). The toolkit uses L-BFGS - an
advanced convex optimization procedure to train CRF models. The model was trained on
sections 01.24 of WSJ corpus and the accuracy of the system was 97% (Phan et al, 2005). This tagger tagged 500 sentences in a second.

1.1.3.4. SUPPORT VECTOR MACHINE (SVM)

The Support Vector Machine or the SVM approach is a machine learning algorithm which is used for binary classification, and has been applied effectively to perform the task of tagging (Marquez et al, 2004; Giménez et al, 2004). The SVM learning algorithm facilitates in building a highly efficient, portable, simple, robust and flexible POS tagger which is easy to configure and train a corpus by tuning with very few parameters. The algorithm is language independent which makes it easy to apply to any language by learning it from a supervised data. Moreover, it can also learn from unsupervised corpus in case of data scarcity in any language. Giménez and Marquez (2004) designed a SVM tagger which includes the features mentioned. The tagger was trained and tested on both English and Spanish corpus which successfully generated state-of-the-art accuracy of 97.16 % and 96.89% respectively. For evaluation in English, the Wall street Journal corpus of 1,17,300 words was applied and Spanish used the LEXESP corpus containing 10600 words. However, the SVM algorithm needs to be modified to make it competent enough to handle the tagging of unknown words.

1.1.4. HYBRID APPROACH

The Hybrid approach of tagging refers to the combination and implementation of more than one tagging approach. For example, a tagger which uses the rule-based and the statistical algorithms is a Hybrid tagger. Both the rule-based and the stochastic tagging models have their own advantages and disadvantages. The main drawback of a rule-based
approach is that it requires prior linguistic knowledge to design the disambiguation rules in order to analyze each token and drafting these rules manually becomes a tedious job in case of large sized corpora. On the other hand, in stochastic systems, the tagging models assign tag sequence based on the probability, frequency and statistics which are not correct in accordance with the grammatical rules of a language (Kumawat & Jain, 2015). In a Hybrid model, the tagger can incorporate the constructive features of both the rule-based and machine learning techniques, and generates a highly efficient tagger. An example of a Hybrid tagger is one developed by Jurish (2003). He has used the Hybrid approach of tagging by implementing the HMM tagging algorithm and the rule based system to design a *dwdst* POS tagger which comprises of the *preprocessor*, the *lexical classifier*, a *morphological analyzer*, the *analysis restrictor* and the *HMM disambiguator*.

The following figure gives an illustration of the different models and techniques of POS tagging.
1.2. POS TAGGERS FOR INDIAN LANGUAGES – APPROACHES USED

POS tagging in Indian languages remains a challenging and interesting research area in the recent years. Quite a lot of research work has been done to develop and improve the task of POS tagging and the taggers in Indian languages. Building good and efficient POS taggers for Indian languages becomes a challenging task due to the non-availability of pre-tagged corpora for most of the languages. Even though, there is a lack of readily available annotated corpus, the morphological rich features of Indian languages facilitate the researchers to build the taggers even with inadequate resources. The linguistic analysis of words and affixes helps in annotating the corpus and incorporate them into various machine learning approaches to build the taggers. This section presents
a brief overview of the various approaches used in building the POS taggers for Indian languages.

1.2.1. POS TAGGERS BASED ON RULE-BASED FRAMEWORK

Dandapat (2009) points out that the oldest work on Indian language POS tagging was in Hindi by Bharati et al. in 1995, where a framework for implicit POS tagging based on the computational Paninian parser was presented. The Paninian grammar framework has been formalized by Bharati et al in 1995, which works on the free word order languages and makes use of the vibhakti, case endings and karaka relations for tagging and parsing sentences (Bharati et al, 1995). Makwana et al (2015) reported in their survey that this framework applies the notation of karaka relation between nouns and verbs in a sentence. Initially, this grammar framework has been applied to Sanskrit. Now, this framework has been adopted for other languages like, Hindi, Odia to build tagsets and taggers in the respective languages.

Most of the taggers for Indian languages incorporate the morphological analysis of the words and affixes combined with other machine learning techniques to build the taggers. Rule-based framework is the most commonly used algorithm in developing the POS taggers in languages like, Hindi, Tamil, Telugu, Kannada, Punjabi and more. The insufficient amount of statistical information and the shortage of annotated corpus drive the researchers to adopt the technique of linguistic analysis for tagging. In addition, various statistical and machine learning approaches to tagging like, the HMM, CRF, Maximum Entropy, SVM, and decision tree algorithms have also been used to create POS taggers in Hindi, Tamil, Telugu, Kannada, Odia, Malayalam and so on.
Garg et al (2012) developed a rule-based tagger in Hindi using a tagset of 30 standard tags on a news corpus consisting of different genres of news. The tagger was trained on 26,149 tokens and tested on three different sets of test data belonging from the genres of news, essays and short stories consisting of 17233, 5039, 3877 tokens respectively. By applying the recall and f-measure technique of evaluation, the system achieved an overall accuracy of 87.55%. The system failed to tag the unknown words and also fell short in assigning the appropriate tags according to the disambiguation rules.

For Tamil, Arulmozhi et al. (2004) developed a rule-based POS tagger combining both lexical and context sensitive rules. The lexical rules referred to a combination of rules and suffixes that have been used to assign tags to each word in the corpus without considering the context information. Further, manually written context sensitive rules have been applied to tag the correct POS tags for unknown words and erroneously tagged words. They have applied a coarse-grained tagset comprising 12 tags. The tagger generated an accuracy of 83.6% with the lexical rules and the accuracy rate was improved to 88.6% after the addition of the context sensitive rules.

For Telugu, Ganesh (2006) developed a rule-based POS tagger in three stages. In the first stage he has developed a Telugu tagset by manually tagging different texts using 53 Telugu tags which cover all the parts-of-speech of the language. In the next stage, a corpus or text has been taken as input and fed to the Telugu Morphological Analyzer which generates the corresponding output. Finally, the morpho-syntactic rules have been applied to the original corpus to disambiguate the POS ambiguities.
Govilkar et al (2015) proposed a rule-based Marathi tagger which extracts the root words using a morphological analyzer and then compare them with the corpus to assign the correct tag. If a word is found ambiguous with more than one tag, then disambiguation rules have been added to generate the appropriate tag.

The only available tagger for Punjabi language has been created by Gill et al (2009) who have implemented the rule-based approach to tag. The tagging system is based entirely on the grammatical categories of different types of agreement in Punjabi sentences and has used hand-crafted linguistic rules to disambiguate the parts-of-speech information for a given word according to the context of occurrence. A fine-grained tagset of 630 tags has been developed as well. To calculate the accuracy of the tagger, a manual evaluation has been done based on random sampling of 25,006 words of an 8 million word corpus. So, using 40 hand-coded disambiguation rules and 630 tags on an 8 million words Punjabi corpus, the tagger attained an accuracy rate of 80.29% (with unknown words) and 88.86% (without unknown words).

1.2.2. POS TAGGERS BASED ON MACHINE LEARNING APPROACHES

Among the machine learning methods, HMM is the most frequently used technique followed by Entropy model and CRF. Dandapat (2009), Ekbal et al (2009) in their papers stated that, two machine learning contests took place in the year 2006 on chunking and parts-of-speech tagging for Indian languages to provide a platform for researchers to work on a common problem. The contests were carried out for Hindi, Bengali and Telugu. All these three languages applied a common tagset comprising 27 tags, known as the IIIT-H tagset. The first contest was organized by NLP Association of
India (NLPAI) and IIIT-Hyderabad in 2006. The competing teams worked on POS tagging in Bangla, Hindi and Telugu using these three learning algorithms and compared the taggers in terms of accuracy.

Dalal et al (2006) used the Maximum entropy model to build a POS tagger for Hindi and experimented their system over a corpus of approximately 35,000 words annotated with 29 different POS tags with an accuracy of 89% in the NLPAI-ML 2006 contest (Dalal et al, 2006). Further, Singh et al (2006) described a CRF based Hindi statistical tagger where they have used 24 different types of lexical and spelling features to produce the model parameters. They tested their tagger on a corpus of around 12,000 words which was annotated with a tagset of 23 tags and the reported accuracy was 88.95% using a 4-fold cross validation accuracy test. Smriti et al. (2006) have also introduced a POS tagger for Hindi which uses an annotated Hindi news corpus of 15,562 words taken from BBC Hindi News site, a tagset of 23 POS tags, and a comprehensive morphological analysis with a reasonable good amount of lexicon and a learning algorithm based on decision tree algorithm. They obtained an accuracy of 93.45% with their tagger. Srivastava et al, (2008) have developed a simple POS tagger based on HMM, which employs a naïve (longest suffix matching) stemmer that functions as a pre-processor to achieve a fairly accurate accuracy of 93.12%.

For Malayalam, Manju et al (2009) implemented a stochastic HMM based POS tagger where the HMM algorithm and a morphological analyzer have been used respectively to train and test a tagged corpus of about 1,400 words. This HMM based POS tagger reported an accuracy of about 90%.
Shambavi et al (2012) presented a Maximum Entropy based POS Tagger in Kannada which have a manual annotated training set of 51267 words using a tagset of 25 tags. After systematic experiments, the best suited feature set for the language has been selected. For evaluation, a corpus of 2892 words has been downloaded from Kannada websites. The tagger generated an accuracy of 81.6%. Shambavi et al (2012) have also proposed two more taggers based on the HMM and CRF models. For training and testing these models, the data has been taken from EMILLE corpus. The training data comprises 51,269 words and test data includes around 2932 tokens. The HMM and the CRF tagger reported an accuracy of 79.9% and 84.58% respectively.

For Marathi, Singh et al (2013) developed three POS taggers based on statistical approach using Unigram, Bigram, Trigram and HMM algorithms. They have also developed a Marathi tagset and have compared the accuracy of all the taggers’ output. So, the accuracy rate of the three Marathi POS taggers in Unigram, Bigram, Trigram and HMM techniques turned out to be of 77.38%, 90.30%, 91.46% and 93.82% respectively.

In addition to these frameworks, the Support Vector Machine algorithm, and the decision tree models have also been implemented in many Indian languages to build the taggers. Das et al (2015) have built a SVM based Odia POS tagger using a small tagset of only five tags. The tagger has been trained and tested on a corpus of 10,000 words resulting in an accuracy of 82%.

Antony et al (2010) developed a POS tagger for Malayalam, where the training, testing, and evaluation have been done using the SVM algorithms. The tagger generated an accuracy of 94% and displayed a higher and improved accuracy than the HMM based
Antony et al (2010) also presented a Kannada POS tagger based on SVM approach to analyze and annotate the Kannada corpus. The corpus, extracted from Kannada newspapers and books, has been morphologically analyzed and tagged manually using a proposed tagset of 30 tags. Then, the tagger has been tested on a corpus of 56,000 words which garnered an accuracy of 86%.

Dhanalakshmi et al (2009) have also developed a Tamil tagger based on the SVM algorithms. They have tagged a raw corpus of size of about 2,25,000 words using their „AMRITA“ tagset and trained it with the machine learning based SVM Tool. Further, a raw corpus has been tested using the SVM Tool and an overall accuracy of 95.64% has been obtained.

1.2.3. POS TAGGERS BASED ON HYBRID APPROACH

Quite a good number of research works have also been done implementing the Hybrid approach to create a tagger. However, it has been observed that most of the hybrid taggers for Indian languages use the combination of rule-based and HMM based algorithm in their hybrid framework. Arulmozhi et al. (2006) have developed a Hybrid Tamil POS tagger combining both the rule-based and HMM framework. First, a HMM based statistical tagger has been used to annotate the raw corpus. But, it was observed that some sentences and words have not been tagged due to the limitation of the algorithm or the amount of training corpus. Then, the untagged sentences and words have been processed through the rule-based system. Although the HMM tagger produces an extremely low accuracy rate of 66%, the hybrid system works efficiently with 97.3% accuracy. Mohnot et al (2014) have also proposed a Hindi POS tagger based on the
combination of rule-based and HMM based techniques. In the NLPAI-2006 contest, most of the competing teams applied the HMM framework as one of the approaches in their hybrid learning algorithms, the others being, CRF and Maximum Entropy Model to perform POS tagging in Bangla, Hindi and Telugu. (Dandapat, 2009).

1.3. POS TAGGERS FOR BANGLA- APPROACHES AND ISSUES

Similar to the attempts made in developing POS taggers for other Indian languages, there have been attempts for developing POS taggers for Bangla in various frameworks. Dandapat (2009) has done a significant work on developing the Bangla POS tagger. He has applied all the three statistical approaches, the HMM, Maximum Entropy and CRF models for automatic POS tagging and disambiguation of the Bangla text using the CIIL corpus. The training data included manually annotated 3625 sentences, which comprised of approximately 40,000 words for all the models. A fixed set of 11,000 unlabeled sentences of roughly 100,000 words obtained from the CIIL corpus has been used to re-estimate the model parameter during semi-supervised HMM learning (Dandapat, 2009). Before building this tagger, Dandapat et al (2004) have also used the HMM framework to build the tagger, using a corpus based semi-supervised learning algorithm. Their system has used a small tagged corpus of 500 sentences and an unannotated corpus of 50,000 words along with a Bangla morphological analyzer. They experimented their tagger on a corpus of 100 sentences, i.e., 1003 words resulting in an accuracy of 95%.

Further, Bandhopadhyay et al (2006) have also incorporated the HMM algorithm to build a Bangla tagger. The tagger has been trained on the training sets, named as,
ANNOT-A and ANNOT-B consisting of 40956 tokens in total and it has been tested on the development test set, named, ANNOT-D consisting of 5967 tokens and generating 85.42% accuracy (Bandhopadhyay et al, 2006). Furthermore, it has been tested on the unannotated test set of 5129 tokens and resulting in 79.12% accuracy. Ekbal and Bandhopadhyay (2008) have developed a Bangla news corpus containing approximately 34 million word forms. From this corpus they have extracted a data set of 72,341 tokens to build the POS taggers implementing the HMM and SVM algorithms. They have used the tagset of 26 POS tags, based on the tagset defined for the Indian languages for training purpose. Out of the 72,341 words, around fifteen thousand tokens were extracted for development set and the remaining tokens were used as the training corpus. Further, the taggers were tested on a golden test corpus of 35,000 tokens. The taggers have achieved an accuracy rate of 85.56% and 91.23% for HMM and SVM respectively. Dandapat et al (2006) have also designed a Hybrid model for POS tagging in Bangla by incorporating both the unsupervised and the supervised techniques to train the corpus in HMM model and then, applying a morphological analyzer to tag the Bangla corpus. They have implemented different frameworks of POS tagging in their Hybrid model, which includes the supervised learning and the partially supervised learning along with the morphological analyzer to decode the best possible tag sequence and evaluated the tagging performance on 20 files each containing 25 sentences.

Ekbal et al (2007) have devised a Bangla POS tagger using the CRF framework. They have also used the tagset of 26 tags. The model made use of contextual information and variety of features of the words which are used to predict the tags. The tagger has been trained and tested with the 72,341 and 20,000 tokens, respectively. This tagger has
the capability of handling the unknown word problems effectively and improves the accuracy of the tagger significantly till 90.3%. Ekbal et al (2008) have also employed the statistical Maximum Entropy model to build a tagger which applies a range of contextual information along with a variety of features that are needed for the prediction of the different lexical tags. Sarkar and Ghosh (2013) proposed a memory based POS tagger in Bangla, which is based on the Memory-Based Learning (MBL) algorithm. Their tagger has been tested on the NLTK data set which consists of 8905 Bangla sentences, applying the tagset of 26 POS tags. Further, the competence of the tagger has been computed by comparing it with the HMM based tagger developed by Dandapat et al (2006). So, using a 10-cross fold validation process of assessment, the memory-based tagger and the HMM based tagger scored an accuracy rate of 80.77% and 78.68% respectively, thereby proving that the memory-based tagger performs more efficiently than an HMM based tagger.

Furthermore, Hasan et al (2007) attempted to compare the performance of the various POS tagging techniques, like, the statistical approach and the transformation-based approach, for Bangla. In statistical approach, N-gram and HMM techniques have been implemented and the Brill’s tagger has been used for transformation-based approach. They have experimented and compared both Bangla and English taggers using unigram, bigram, HMM and Brill tagging modules, taken from NLTK. For Bangla, a tagset of 41 tags has been used and for English, they have used the Brown tagset. Similarly, for training and test set, they have used the tagged Brown corpus from NLTK for English. But, for Bangla, a small corpus of 5000 words collected from a newspaper
has been used. However, the English POS taggers recorded a higher accuracy of more than 96%, than the Bangla taggers, which reported an accuracy of 90%.

Finally, Ekbal et al (2009) developed three individual POS taggers in Maximum Entropy (ME), Conditional Random Field (CRF) and SVM frameworks and compared the performance of the taggers of all the three frameworks to determine the most effective approach. They have used the IIIT-H tagset of 27 POS tags and trained and tested all the three taggers on the same training and testing set of 57,341 and 35,000 words respectively. The results certify that the SVM based tagger is the most competent approach which exhibits the highest accuracy rate of 85.83% followed by CRF and Maximum Entropy with 84.11% and 81.75% respectively. However, they have included different techniques like Named entity recognition, lexicon features to deal with the tagging of the unknown words to enhance the accuracy of the taggers.

Based on the analysis of the Bangla POS taggers discussed above, we have noted down a few issues related to the development of taggers for Bangla. Firstly, most of the taggers have been tested on sets of annotated or unannotated corpus not exceeding the size of one lac words, which has affected the performance of the taggers. Secondly, the ambiguity has featured as the primary issue in almost all the approaches to Bangla POS tagging. Even after applying different disambiguation techniques, most of which are statistically motivated, the ambiguity issue still persisted in the error analysis. Since, most of the POS taggers have used a fine-grained tagset; the most common error which featured in the list of error analysis of most of the approaches is the confusion in tagging between the proper and common noun, and between adjectives and common nouns. Furthermore, the approaches faced difficulty in tagging the unknown words in the tagged
output. Moreover, the statistical algorithm HMM has been one of the most preferred frameworks in developing the taggers. Lastly, it is seen that POS tagging in Hybrid framework refers to building two different POS taggers adopting different statistical algorithms and comparing the accuracy to decide upon the potent and dynamic approach. Besides this, developing a Bangla tagger also involved in combining the stochastic methods, like HMM and morphological analyzer for disambiguation.

1.4. HYPOTHESIS

The central proposition of the thesis is to ascertain the fact that a Hybrid model, which is a combination of two or more different tagging techniques, generates a better accuracy rate than a single POS tagging framework. In other words, the dissertation aims to prove that a POS tagger in a hybrid model performs better than the concept based taggers in terms of generating a good percentage of accuracy.

1.5. OBJECTIVE OF THE THESIS

The primary objective of the dissertation is to design and develop a Hybrid model for Bangla POS tagging by merging the statistical and linguistically-motivated algorithms, namely, the Conditional Random Field (CRF) algorithm, and two linguistically motivated tagging techniques, Rule-based algorithm and list-based algorithm, also termed as Look-up approach. In order to achieve this objective, we wish to

- explore and investigate the three different tagging frameworks, namely CRF, Rule-based, and the dictionary based Look-up frameworks.
• develop an algorithm for Hybrid model by incorporating all these three techniques together.

• manually annotate news corpora of Bangla based on a tagset designed by us to train the CRF tagger.

• collect a large size of unannotated test corpus, which will be used both for tagging and evaluation.

• do the linguistic analysis of the inflected lexicon of the word classes for developing the rule-based tagging module.

• compare and evaluate the accuracy of the tagged output of the CRF and the Hybrid model to compute the difference in the accuracy percentage.

• code programs in Perl language to develop the tagger.

1.6. METHODOLOGY

Several stages are involved to build the Hybrid modeled POS tagger in Bangla. The following points give a road map to the methodology to develop the proposed tagger.

• The first stage is the preprocessing stage which includes in preparing the Bangla corpus apt for the task of tagging. In order to accomplish this objective, the Bangla news corpus will be merged (cf. section 1.6.1), cleaned and tokenized to form a list of tokens which will be divided into training set and test set. The training set will be manually annotated using a tagset, which will be designed for the present research.

• The second stage entails on building the tagger through four modules. In the first module, the training set will be fed to the CRF tagger to enable the tagger to learn
the linguistic features of the language and assign appropriate tags to an untagged test corpus based on its training.

- The second module, that is, the Look-Up based tagging requires a dictionary of words that belong to the closed category of POS tags and a Perl program to build a Look-up tagging module. The dictionary will be formulated using the linguistic knowledge. The output of the CRF tagger, i.e., the tagged corpus will serve as the input to the look-up system. The first step in executing this module will involve in listing the unambiguous set of words of different word classes of closed class category in separate text files. The next step requires in encoding a program that will compare the tagged input file with different lists of the corpus, match the words and will tag them accordingly.

- The Rule-based tagging will constitute the third step in building the POS tagger, where the unambiguous suffixes will be decided by linguistically analyzing the morphological features of all the word classes. Subsequently, a program will be coded in Perl whose objective will be to match the inflectional endings with the corpus file and assign the corresponding tags to each matched occurrence. The combined tagged output of CRF and Look-up based module will act as the input to this tagging program.

- In the fourth module, a Perl program will be created that will encompass the tags of all the three frameworks along with the corpus. This program will compare the tags of all the three algorithms and assign the correct tag to the corpus in a fresh text file, thereby developing the POS tagger in the Hybrid model.
Finally, an assessment on the performance of the tagger will also be executed by employing the TnT tagger, to count the number of matched and unmatched tags in both the CRF and the Hybrid POS tagger to establish the main hypothesis of the dissertation. To do so, two separate evaluation text files will be created manually for both the tagging models for computation of the accuracy.

The following sub-sections illustrate the creation of the Bangla news corpus and the programming language along with the various programming tools and the tagset required in constructing the prospective tagger.

1.6.1. CORPUS

For the present work, a Bangla news corpus has been collected from two sources: (i) the TDIL (Technology Development for Indian Languages) corpus and (ii) online newspapers. The TDIL corpus consists of Unicode compliant Bangla articles collected from the printed documents of different disciplines and sources. The size of the total corpus is 48.6 Mb containing twelve hundred and seventy text files. The text documents are a collection of various articles taken from books, magazines, journals and newspapers of different genres that range from social science, agriculture, anthropology, commerce, banking, economics, botany, chemistry, history, law, literature; articles like biographies, child literature, book reviews and criticism, fine arts, folk lore, travelogues, mass media articles in magazines, and newspaper articles of sports news, crime, political news, editorials, national and international news collected from two leading newspapers, „Anandabazar Patrika” and „Bartaman Patrika”. Out of these text documents of different genres and disciplines, we have extracted only the newspaper articles of a particular
newspaper, that is, „Anandabazar Patrika”. Moreover, the newspaper articles have been restricted only to national, political, and sports news. The corpus has been restricted to a particular newspaper and specific genres of news in order to generate a good percentage of accuracy in the tagged output. So, a total number of one hundred and eighty seven files of news articles have been extracted from the TDIL corpus. Further, a part of the corpus has been developed by manually visiting the online webpage of the „Anandabazar Patrika” (http://anandabazar-unicode.appspot.com), copying the news articles and saving them in UTF-8 format in separate files.

1.6.1.1. COLLECTING AND SAVING THE CORPUS

After collecting and saving the text documents of the news articles in separate files, they have been merged together into a single Unicode compliant text file. The merging of the files has been done by using the concatenation command in the LINUX operating system. So, the final corpus is the output of the concatenated files as a single text document in UTF-8 format. This file acts as the main input for the tagger to be developed. The total size of the raw corpus is 5,388 Kb.

1.6.1.2. CLEANING AND TOKENIZING THE CORPUS

After creating the corpus, a punctuation split program is used to separate the punctuation markers from the words. The output of the punctuation split program is used as the input for tokenization. A tokenizer program is used to tokenize the input corpus and produce a list of tokens. The list of tokens is created to utilize it as the input file for manual tagging for the purpose of developing the training and test corpus in CRF. Therefore, the statistics of the final input corpus for tagging is as follows:
Table 1: Statistics of the corpus

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Size of the raw corpus</td>
<td>5,388 Kb</td>
</tr>
<tr>
<td>Size of the tokenized corpus</td>
<td>5,987 Kb</td>
</tr>
<tr>
<td>Total no. of tokens</td>
<td>#491503</td>
</tr>
</tbody>
</table>

1.6.1.3. PROGRAMMING TOOLS

In order to make a fitting and proper input list from the raw corpus for the tagging purpose, several programming tools have been built and generated in both the operating systems, Windows and Linux. It is to be noted that, there are a few tools, which have a better and effective execution in Linux than Windows and vice-versa. These tools have been developed using the Perl\textsuperscript{x} programming language on a programming platform of Edit Plus\textsuperscript{x}. The description, development, and the advantages of building these tools are discussed below.

a) **File merger:** As discussed above, the newspaper articles have been collected and saved in separate text files in Unicode format. It is, therefore, necessary to combine and merge all these files into one single file to obtain a large, unified and consolidated set of systematically structured corpus. So, a „file merger” is required for this job. This task of merging the text files have been carried out in Linux OS, because the Windows OS does not have the capability to support large files. The
concatenation command is used in the Linux interface which merges multiple text files into a single set of raw corpus which is further processed using the following programs.

b) **Punctuation split**: The „punctuation split“ is a Perl program which splits the words from the punctuation markers in a given corpus. The punctuation marks are considered to be individual tokens or strings which are not supposed to be attached to a word or a token. But, in the corpus, it is seen that words are generally attached to punctuation markers. So, when the corpus is tokenized, the words with attached punctuation markers are also counted as separate and individual tokens, resulting in the repetition of same words, and hence, inaccurate output. So, in order to avoid repetition of tokens and incorrect accuracy, the punctuation split program is used on the raw corpus which separates the punctuation markers from the words by a white space. Given below is the sample of the transliterated raw text:

1. সে বলল: “আমি এই কাজ আর করব না”। কিন্তু খুব ভালভাবে বোঝা যায় যে, এটা তার মনের কথা নয়; অন্য কেউ তাকে দিয়ে বলাচ্ছে।

[se bollo: “ami ei kaj ar korbo na” l kintu, khub bhalobhabe bojha jai, je eta tar moner kotha noi; onno keo take diye bolacche]

[He told, that, “I will not do this work anymore”. But, it was understood that, this was not his own decision; someone else made him say this.]

As we can see, the Bangla text consists of words where the punctuation markers are attached to them. Now, the punctuation split program takes this text as the input and generates the output as follows,
2. সে বলল: “আমি এই কাজ আর করব না”। কিন্তু খুব ভালোভাবে বোঝা যায় যে, এটা তার মনের কথা নয়; অন্য কেউ তাকে দিয়ে বলাচ্ছে।

[se bollo: “ami ei kaj ar korbo na” | kintu, khub bhalobhabe bojha jai, je eta tar moner kotha noi; onno keo take diye bolacche |]

[He told, that, “I will not do this work anymore”. But, it was understood that, this was not his own decision; someone else made him say this.]

The above data illustrates that the punctuation strings like “:”, and “,” which were attached to the words like, “bollo” and “kintu” get separated by a white space, thus, generating a clean text with individual tokens.

3. **Tokenizer**: A “tokenizer” performs the task of tokenization by breaking up the sequence of strings in a text and determining boundaries for individual tokens (Mondal, 2011). As it has been mentioned before, the split file is fed as the input for tokenization. The tokenized output is redirected to a text file in the following format,
In addition to the tools used to make a suitable set of text data required for manual annotation, a few more programming applications have also been employed in the present research.
applying which all the repeated tokens are deleted thereby resulting in a tokenized file of unique tokens, also known as *types*. However, for the present work, sorting is required to remove the tagged duplicates from the final list of tagged output as well as the CRF output, to get a unique list of words and their corresponding tokens. These two files have been sorted to make a better comparison in the TnT tagger to evaluate the efficiency of the hybrid tagger. So, the sorting is done by devising a Perl program where each word and its corresponding tag has been considered as a single token. The program compares both the words and tags and removes all the common words and their corresponding tags.

- **TnT Tagger:** The TnT tagger is a very competent and robust statistical tagger developed by Thorsten Brants in 1999 (Brants, 2000). Dandapat (2009) describes TNT as the stochastic trigram model based on HMM framework which adopts a technique of suffix analysis to estimate the lexical probabilities for unknown words based on the features of the words which share identical suffix in the training data. The primary feature of this tagger is that it can train on different languages using any tagset. Basically, it is a language independent tagger which means that it is not optimized for any particular language. Instead, it is effectively used to train a large variety of corpora owing to its adaptability feature to adapt to any language, domain or tagset (Brants, 2000). For the present work, the TnT model has been employed to evaluate the performance of the proposed tagger. To evaluate a tagged file in this model, two separate text files are required, an evaluation text file comprising of accurate tags created by manual editing and a statistically tagged file. The *tnt-diff* application is used for the assessment which
compares the two tagged files and exhibits the difference in the number of matched and unmatched tokens, thus, giving the accuracy rate as well as the error percentage of the hybrid tagger.

**1.6.2. TAGSET**

Tagsets or lexical tags have an essential role in NLP because they provide significant information about a word and its neighbors in a corpus. A set of standard tags forms an indispensable part in the POS tagging task as it imparts the morphosyntactic information of the words and their contexts in a language by allotting a specific tag to each parts-of-speech. The tagsets for a language can be categorized into two major groups, namely, *coarse-grained* tagset and *fine-grained* tagset. A coarse-grained tagset uses a small number of tags and consists of the basic tags such as Noun (N), Verb (V), Adjective (ADJ), Preposition (PREP) and so on. On the other hand, the fine-grained tagsets have more specific tags such as Personal pronoun (PRP), Plural noun (NNS), Comparative adjective (JJR) and so forth.

Baskaran et al (2008) have traced the history of POS tagset design. The early efforts to design POS tagset started in 1970s that resulted in the *UPenn, Brown* and *C5* tagsets that mainly focused on English using flat structure and were mostly simple lists of coarse-grained tagsets. The *Penn Treebank* used a tagset of 45 tags and 61 tags were used for *C5* tagset. However, the *CLAWS2* tagset brought a change in the structure of the tagsets from a flat structure with unitary tags and introduced a hierarchical structure for decomposing tags.
According to Baskaran et al (2008), a POS tagset design should take into consideration all the probable morphosyntactic categories that can exist in a specific language or a group of languages. Research work in POS tagset design for European and East Asian languages started with the basic listing of important morphosyntactic features in one language which has evolved in later years towards hierarchical tagsets, decomposable tags, and common framework for multiple languages (EAGLES) etc. The tagset for English follows the Penn Treebank tagset, but for languages like Catalan, Spanish, Russian, Italian, EAGLES tagset is used. According to them, the publication of EAGLES guidelines for morphosyntactic annotation of corpora was an earliest attempt to develop a common tagset guideline for several European languages. The objective of EAGLES1 was to standardize the tagsets used in different projects and different languages to achieve flexibility, reusability, cross-linguistic compatibility, and interchangeability. (Baskaran et al, 2008).

However, it has been observed that several tagsets have been developed for European languages along with large amount of annotated data. But, the research work in tagset design for Indian Languages (IL) presents a contradictory picture. Comparatively, very less work has been done in designing tagsets for Indian languages than the European languages. One of the main reasons of the lack of research lies in the fact that most of the tagsets for ILs are language specific and cannot be used for tagging data in other language owing to their complex morphosyntactic characteristics. This inconsistency causes a hindrance to the interoperability and reusability of annotated corpora which further affects the NLP research in ILs, where already the non-availability of tagged data is a serious issue (Baskaran et al.,
Despite the limitations and drawbacks, a number of tagsets like, the BIS tagset, AUKBC tagset, LDC-IL tagset, Tamil tagset, MSRI Sanskrit tagset (IL-POSTS), JNU-Sanskrit tagset (JPOS), IIIT (ILMT) tagset, Sanskrit consortium tagset (CPOS) and CIIL Mysore tagset have been formulated by several research groups working on different projects in Indian languages. (Chandra et al., 2014).

Baskaran et al (2008) have attempted to design a common POS-tagset framework for ILs, by providing a detailed analysis of eight languages from two major families, Indo-Aryan and Dravidian. They have developed the framework that follows the hierarchical tagset layout similar to the EAGLES guidelines, but with significant changes according to the ILs requirements. According to them, both the Indo-Aryan and Dravidian Languages share noteworthy similarities in morphology and syntax which makes it possible to devise a common tagset framework that can exploit the similar features to facilitate the mapping of different tagsets to each other. So, the hierarchy of their IL POSTS framework has been set in three levels. The first level is the *Obligatory* level which consists of the *categories* that have the highest level parts-of-speech classes. These tags are universal for all languages and must be included in any morphosyntactic tagset derived from the framework. The second level is the *Recommended* level which includes the *Types*, the sub-classes of categories. Some *types* may also be *Optional* for certain languages. The third level consists of *Attributes* which are the morphosyntactic features of Types. The attributes are optional in nature, though in some cases they may be recommended. So, there are 11 categories (including the punctuations and residual categories) that have been identified as universal categories for all ILs that are obligatory for any tagset derived
from IL-POSTS. The table below summarizes the POS tags developed by Baskaran et al. (2008).

<table>
<thead>
<tr>
<th>CATEGORIES</th>
<th>TYPES</th>
</tr>
</thead>
<tbody>
<tr>
<td>Noun (N)</td>
<td>Common (C)</td>
</tr>
<tr>
<td></td>
<td>Proper (P)</td>
</tr>
<tr>
<td></td>
<td>Verbal (V)</td>
</tr>
<tr>
<td></td>
<td>Spatiotemporal (ST)</td>
</tr>
<tr>
<td>Verb (V)</td>
<td>Main (M)</td>
</tr>
<tr>
<td></td>
<td>Auxiliary (A)</td>
</tr>
<tr>
<td>Pronoun (P)</td>
<td>Pronominal (PR)</td>
</tr>
<tr>
<td></td>
<td>Reflexive (RF)</td>
</tr>
<tr>
<td></td>
<td>Reciprocal (RC)</td>
</tr>
<tr>
<td></td>
<td>Relative (RL)</td>
</tr>
<tr>
<td></td>
<td>Wh (WH)</td>
</tr>
<tr>
<td>Postposition (PP)</td>
<td></td>
</tr>
<tr>
<td>Adverb (A)</td>
<td>Manner (MN)</td>
</tr>
<tr>
<td></td>
<td>Location (LC)</td>
</tr>
<tr>
<td>Nominal Modifiers (J)</td>
<td>Adjectives (J)</td>
</tr>
<tr>
<td></td>
<td>Quantifiers (Q)</td>
</tr>
<tr>
<td>Demonstratives (D)</td>
<td>Absolute (AB)</td>
</tr>
<tr>
<td></td>
<td>Relative (RL)</td>
</tr>
</tbody>
</table>
Table 2: IL-POS Tagset (Baskaran et al., 2008)

Developing a tagset depends on three main features, which are, the granularity of the tagset, the choice of the category of the tags and the choice of new tagset versus standard tagsets. Firstly, the granularity of the tagset should be decided, that is, whether the tagset should be a coarse-grained or a fine-grained. Dandapat (2009) argues that the granularity of the tagset directly affects the effectiveness in the accuracy of the tagger. The tagger’s accuracy will be higher if the tagset is too coarse because it will only consider the key lexical tags. However, a few important
distinctions might get missed out in the coarse-grained tagset. Alternatively, in case of a fine-grained tagset, if there are excessive fine distinctions, it results in the reduced performance of the tagger as it becomes difficult for the tagger to learn the highly enriched features of a word. Secondly, the categorical features of the tags should be considered while designing a tagset. A word can be tagged either according to its syntactic category or the lexical category (Chandra et al., 2014). Now, tagging the words according to the syntactic category has an advantage over the lexical category, since the syntactic category incorporates the context while tagging a word, which is absent in lexical category. So, even though a tagset based on a lexical category can efficiently train a tagger due to its consistency, it results in the loss of precision and accuracy. Finally, while determining the tags for a particular tagset, one can either introduce a completely new set of tags or can use a standard tagset as a reference and modify it according to the requirement of the tagger (Dandapat, 2009), (Chandra et al., 2014). In addition to these issues, Dandapat (2009) explains that issues like, type of applications (in some tagging task, only the category information is sufficient whereas in others, complex information about the tags is needed to perform the tagging), tagging approaches (rule-based, stochastic techniques can handle large size tagsets as compared to other algorithms), size of the corpus (a fine-grained tagset is difficult to handle in case of a large annotated data) and availability of resources should also be considered while designing a tagset.

1.6.2.1. THE TAGSET FOR HYBRID POS TAGGER

Based on the features mentioned above, we have devised a small list of tagset comprising of 14 tags which encompasses the entire parts-of-speech of Bangla. The
The tagset includes both the major categories of parts-of-speech and the sub-classes of the categories as well. So, it can be said that it incorporates the features of both the coarse-grained as well as the fine-grained tagsets. We have adopted the IL-POS tag (Baskaran et al, 2008) as the reference framework of tagset to design the current tagset. The tagset consists of 14 tags which are given below in the table format,

<table>
<thead>
<tr>
<th>TAG</th>
<th>CATEGORY</th>
<th>EXAMPLE</th>
</tr>
</thead>
<tbody>
<tr>
<td>N</td>
<td>Noun</td>
<td>[ram b'at k'au/] Ram rice eat 'Ram eats rice'.</td>
</tr>
<tr>
<td>VM</td>
<td>Main Verb</td>
<td>[fe k'elefe] He play 'He is playing'.</td>
</tr>
<tr>
<td>VAUX</td>
<td>Auxiliary Verb</td>
<td>[fe kuto korte parto] He work do could have 'He could have done the work'.</td>
</tr>
<tr>
<td>NV</td>
<td>Negative Verb</td>
<td>[fe kau kore na] He work do not 'He does not work'.</td>
</tr>
<tr>
<td>ADJ</td>
<td>Adjective</td>
<td>[mota c'ele] Fat boy 'Fat boy'.</td>
</tr>
<tr>
<td>ADV</td>
<td>Adverb</td>
<td>[fe azube] He today go 'He will go today'.</td>
</tr>
<tr>
<td>PP</td>
<td>Postposition</td>
<td>[ram b'at jenun gelam] I her because go 'I went because of her'.</td>
</tr>
<tr>
<td>CONJ</td>
<td>Conjunction</td>
<td>[ram b'at cima ndu vhej] Ram or Jodu go 'Ram or Jodu will go'.</td>
</tr>
<tr>
<td>NEG</td>
<td>Negation</td>
<td>[boto coko kom na] Book like this not 'The book is not like this'</td>
</tr>
<tr>
<td>DET</td>
<td>Determiner</td>
<td>[az boto b'ala] This book good 'This book is good'.</td>
</tr>
<tr>
<td>CL</td>
<td>Classifier</td>
<td>[bato-son c'elefe nda] Twelve boy go 'Twelve boys will go'</td>
</tr>
<tr>
<td>COMP</td>
<td>Complementizer</td>
<td>[ram bolo se fe doko] Ram tell that he go 'Ram told that he will go'.</td>
</tr>
<tr>
<td>NUM</td>
<td>Numeral</td>
<td>[5-2 cimoc b'at dno] 12 spoon rice give 'Give 12 spoons of rice'.</td>
</tr>
<tr>
<td>SYM</td>
<td>Symbol</td>
<td>[ram b'at k'au 1] Ram rice eat 'Ram eats rice'</td>
</tr>
</tbody>
</table>

Table 3: Proposed POS tagset
Now, according to the IL POS tagset framework, there are 11 category tags which belong to the major parts of speech category like, Noun, Verb and so on that have been mentioned in table 2. These categories include the sub classes of Types like, Main Verbs, Auxiliary verbs that are further associated with a number of attributes related to each of them like, Gender (Gen), Number (Num), Person (Per), Tense (Tns), negation (Neg) are associated with for the types, Main and Auxiliary verb (Baskaran et al, 2008). Based on this classification, we have modified our tagset according to our requirements. Since the components of a nominal phrase, like nouns and pronouns share similar morphological and syntactic features, we have included the noun and pronoun along with their subclasses like proper, common, abstract, relative pronoun, and interrogative pronoun under the main category of „Noun” (N) tag. Likewise, the Adjective (ADJ) tag covers the quantifiers and modifiers. However, the Classifier (CL) denotes only the numerals with classifier markers, like one person, two people. The Interjections are grouped with the Conjunction (CONJ) tag. The Numbers (NUM) denote the numerals without any attached markers while the punctuations are tagged as Symbols (SYM).

From the above table, we have observed that in case of Verbs, the verb tag has been categorized into the sub-classes, viz, Main Verb (VM), Auxiliary Verb (VAUX), and Negative Verb (NV). The verbs have been grouped into different categories owing to their complex morpho-syntactic structures. Chatterji et al. (2012) have grouped the Bangla verb chunk into two parts, namely, main verb (VM), a finite or infinite verb which is an essential part carrying the main meaning of the verb chunk and the auxiliary verb (VAUX), which is the finite verb carrying the
inflectional features of tense, aspect with it. Now, a main verb can occur individually in a verb chunk or followed by the auxiliary verb. For instance, in a sequence of Bangla verb chunk, [kore pʰelecʰɪ] (have done), [kore] is the non-finite verb which acts as the VM followed by the finite verb [pʰelecʰɪ] as the VAUX, whereas in the verb chunk, [korecʰɪ] (have done), the finite verb itself acts as the VM. Moreover, a verb chunk can also be a complex predicate where the noun is related to the main verb by part-of dependency relation (Chatterji et al., 2012). For example, in the complex predicate [bæbohɑr korecʰe] (use), the noun, [bæbohɑr] is linked to the main verb [korecʰe] to depict the meaning of the verb. Similarly, the negative verbs can occur both as a distinct verb followed by the main verb in the verb chunk as well as an inflectional ending attached to the auxiliary verb. For example, in [korte pərnt] (could not do), [pərnt] is the auxiliary verb to which the negative inflectional ending -nt is attached preceded by the main verb [korte]. On the other hand, in the verb chunk, [həbe nɑ] (not happen), [nɑ] denotes the negation preceded by the main verb [həbe]. Hence, due to these kinds of morpho-syntactic intricacies, we have divided the verb into three different tags. Therefore, it can be rightly said that the current tagset combines the features of both the coarse-grained and fine-grained tagsets. Now, using this tagset, 35,219 words have been manually tagged to develop the training set in CRF. The following figure demonstrates the combination of the granularity of the tagset,
1.7. ORGANIZATION OF THE THESIS

The dissertation has been organized into four chapters which are as follows.

- **Chapter 1**: The first chapter introduces the field of the research work, that is, POS tagging, states the objective of the dissertation, offers a proposed hypothesis and provides a brief overview of the different approaches and techniques used for POS tagging and the prior work done in POS tagging in Indian languages in general and Bangla in specific. In addition, it explores the various programming tools and the programming language employed for this work. It, thus, discusses the methodology and gives a concise sketch of the road map to develop the proposed tagger. It also talks about the tagset and the creation of the corpus required for tagging.

- **Chapter 2**: The second chapter presents the linguistic background as it illustrates the linguistic analysis of the word classes of Bangla, and the
morphological analysis of the inflections of verb, adverbs, nouns, adjectives and pronouns for the purpose of creating the rule-based algorithm.

- **Chapter 3**: The third chapter offers to give an overall description of the proposed tagger and elucidates the processing stages to create the tagger in the Hybrid approach along with the evaluation procedure to compute the accuracy of the tagger.

- **Chapter 4**: The concluding chapter elaborates on the various tagging issues and challenges faced, analyze the errors generated by both the CRF and Hybrid models. It also lists the limitations of the tagging model and outlines the scope of future work to improve its performance.
NOTES

1 Machine Translation: The accuracy of a POS tagging is a vital pre-processing step for a high quality, efficient machine translation. The probability of translating a word from a source language to the word of a target language is effectively dependant on the POS category of the source language word (Dandapat, 2009). For instance, the word [d̂ije] will be translated as either postposition (by) or a verb (to give) depending on its POS category of the source language.

ii In information extraction and retrieval systems, if a query is given to the retrieval system along with all the POS information of a word, then it generates a more refined extraction of information of the word. For example, if a person wants to look for documents containing the word „right” as an adjective, then, including the POS information will remove the irrelevant documents with „right” as noun or an adverb.

iii The assignment of the POS tags to each word of a sentence forms one of the fundamental building blocks of speech synthesis and recognition. It is a main requisite in the tasks of text-to-speech processing. Dandapat (2009) claims that the POS tags give significant information about a word and its neighbors which can be useful in a language model for speech processing. He further explains that, a parts-of-speech of a word tell us about how a word should be pronounced according to the grammatical category. For example, in Bangla, the word [kore] is tagged as a postposition when it is pronounced as [kore] whereas it becomes a verb when it is pronounce as [kɔre]. So, the POS tagging for text-to-speech processing tasks relies heavily on the prosodic features of the words which determine the tags.
*Word Sense Disambiguation* refers to the process of choosing the right word based on the context (Jurafsky & Martin, 2000). A training corpus comprising of tagged words according to their sense of occurrence, a tagset, and a set of features extracted from the tagged corpus is required for the purpose. Therefore, a POS tagger forms the compulsory requisite to perform the task of Word Sense Disambiguation (WSD).

It is useful in parsing sentences as, the unique tags to each word reduce the number of parses making the disambiguation task easier and more effective.

The CRF English tagger is a Java based English tagger built on FlexCRFs and developed by Phan et al. (2005). FlexCRFs is a conditional random field toolkit for segmenting and labeling sequence data written in C/C++ using STL library.

File merger has been discussed in section 1.6.1.3.

Punctuation split has been discussed in section 1.6.1.3.

Tokenizer has been explained in section 1.6.1.3.

PERL, also known as Practical Extraction and Report Language is an Unicode compliant, portable, general purpose, open source programming language designed by Larry Wall in 1987 (Schwartz et al, 2011). We have used the Perl version 5.10.0 for our present work, which is the latest version of the programming language currently used globally.

Edit Plus is a text editor which is used in the Windows Operating system and it provides a platform to write different programs in Perl.

The Sort program has been discussed in details in Chapter 3.

The morphological and syntactic synthesis of nouns and pronouns share few common features like, classifier markers, number, case markers and emphatic markers.