Shallow Parsing of Marathi

M.Tech. Dissertation

Submitted in partial fulfillment of the requirements
for the degree of

Master of Technology

by

Harshada Anil Gune

Roll No: 07305904

under the guidance of

Prof. Pushpak Bhattacharyya

Department of Computer Science and Engineering
Indian Institute of Technology, Bombay
Mumbai

Abstract

Shallow parsing is used to provide only a limited amount of syntactic information and is useful in many natural language processing tasks. The work presented in this report aims to develop a high accuracy shallow parser for morphologically rich language-Marathi. We aim to reduce the data requirement of machine learning techniques by appropriate feature engineering based on the characteristics of the language. The crux of the approach is to use a powerful morphological analyzer to generate rich features for a CRF based sequence classifier. Accuracy figures of 95% for Part of Speech Tagging and 98% for Chunking are obtained for Marathi. We further show that by harnessing features of a morphologically rich language, one can hope to build a high accuracy sequence classifier even with the modestly sized training data.

Acknowledgements

I would like to take this opportunity to express my gratitude towards my guide **Prof. Pushpak Bhattacharyya**, without whose support this work was impossible. I would like to mention special thanks to Mitesh Khapra, a research scholar, for his continuous technical assistance.

I would also like to thank Vidyadhar Kulkarni for helping me on the linguistic front. Persuing this work has been an enlighting experience for me and I have enjoyed it.

Contents

Chapter 1

Introduction

Shallow Parsing, which is also called as partial parsing, is the Natural Language Processing task which tries to provide some understanding of the structure of the sentence without generating a complete parse tree. It is the task of recovering only a limited amount of syntactic information from natural language sentence. [[Berwick et al., 1991]] proposed this as a useful and relatively tractable precursor to full parsing, since it provides a foundation for further levels of analysis, while still allowing more complex attachment decisions to be postponed to a later phase. Shallow Parsing involves two primary tasks; POS tagging and chunking.

1.1 POS Tagging

Part-of-speech (POS) tagging is the task of labeling each word in a sentence with its appropriate syntactic category called part of speech (e.g., noun, verb, adverb etc.). There are two factors determining the syntactic category of a word (a) the words lexical probability (e.g. without context, bank is more probably a noun than a verb) (b) the words contextual probability (e.g. after a noun or a pronoun, bank is more a verb than a noun, as in "I bank with SBI"). Hence it disambiguates the part of speech of a word when it occurs in different contexts. The number of part of speech tags in a tagger may vary depending on the information one wants to capture. The morphosyntactic features of the language and the degree of desire to

represent the granularity of these morpho-syntactic features etc., decide the tags in the tag set.
Example:
Shallow_**JJ** parsing_**NN** is_**VM** an_**DT** interesting_**JJ** activity_**NN**.

1.2 Chunking

Chunking is the task of identifying and labeling different types of phrases such as Noun phrase (NP), Verb phrase (VG), Adjectival phrase (JJP) *etc* in a sentence. Chunking is a process of dividing a text's sentences into series of words that together consist a grammatical unit called as a phrase or chunk. A chunk is a minimal, non recursive phrase consisting of correlated, inseparable words/entities. In other words, a chunk is a group of one or more words that acts as a single unit in the syntax of a sentence. A typical chunk consists of a single content word surrounded by function words. Chunking is done after POS tagging.
Example:
[**NP** Shallow parsing] [**VP** is] [**NP** an interesting activity].

Both POS tagging and chunking are important for all language processing activities. These two tasks are considered as important preprocessing activities. This helps in doing deep parsing of text.

1.3 Applications of Shallow Parsing

Not all NLP applications require a complete syntactic analysis. A full parse often provides more information than needed. Shallow parsing can be seen as alternative to full parsing in such cases.

- **Summary Generation and Question Answering:** In these applications, we are generally interested in information about specific syntactic-semantic relations such as agent, object, location, time *etc* (basically, who did what to whom, when, where and why), rather than elaborate configurational syntactic analyses.

Example: Consider a sentence, "I brought a big ball". Words in this sentence will be chunked as below:

$$[\text{I}]_{\mathbf{NP}}\ [\text{brought}]_{\mathbf{VP}}\ [\text{a big ball}]_{\mathbf{NP}}.$$

Now, ask the questions like: what did I bring or who brought a big ball. Answers to these questions are nothing but the chunks shown above.

- **Information Extraction and Retrieval:** Noun phrases can be used in information retrieval systems. In this application, shallow parsing can be used to retrieve the data from the documents depending on the chunks rather than words. In particular noun phrases are more useful for retrieval and extraction purposes.

- **Named Entity Recognition:** Named entity recognition is meant for identifying the proper names of persons, locations, organizations *etc*. These names will be either part of noun chunks or complete noun chunks themselves. As a preliminary to named entity recognition, it would be useful to just pick out noun phrases from a piece of text.

- **Machine Translation:** Shallow parsing is also useful in machine translation systems where structural transformation is required from one language to another. Normally in language processing, sentences are parsed to identify the syntactic structure of the sentence. Shallow parse is sufficient to identify the specific constituents in the sentence that has to undergo transformation.

To carry these out in a complete way, we would probably want to use a proper syntactic parser. But shallow parsing can provide relevant information without attempting a complete sentence parse.

It can be argued that shallow parsing actually reflects aspects in human language processing. For example, when a person speed-reads through a text, he or she does not fully parse the sentence but instead look for "key phrases". These key phrases are actually phrases, which mean that shallow parsing is the machine analogy for speed reading.

1.4 Approaches to Shallow Parsing

A lot of work has been done on shallow parsing, with methods varying from simple rule-based to data-driven approaches or a combination of both. In the following sections, we present issues related with statistical and rule based approaches along with their pros and cons.

1.4.1 Statistical Methods

The task of shallow parsing is ideally suited for machine learning techniques because of relative ease of training. The main advantage with data-driven approach to shallow parsing is that they are language and tag set independent and thereby easily applicable to new languages and domains. The problem of shallow parsing can be seen as sequence labeling task, in which task is to assign pos/chunk labels to every word of a sentence. Then a variety of probabilistic models can be seen as basic framework to solve the labelling problem of shallow parsing.

HMMs are widely used probabilistic models for such problems. However HMMs are generative models, assigning a joint probability to paired observation and label sequences; the parameters are trained to maximize the joint likelihood of training examples. To define a joint probability over observation and label sequences, such model needs to generate all possible observation sequences. Moreover, it is not practical to represent multiple interacting features of observations in such models. Maximum entropy models are conditional probabilistic sequence models that specify the probability of possible label sequence given an observation sequence as opposed to joint probability of HMMs. Moreover, there is no need to make independence assumptions unlike in HMM. However, maximum entropy models exhibit a weakness called *label bias problem*, biasing towards states with few successor states. These difficulties are the main motivations for looking at CRF models as an alternative. Conditional random fields overcome all of the above problems making CRFs attractive for the labeling applications.

1.4.2 Rule Based Methods

Shallow parsing is frequently assumed to be a mature technology with machine learning systems and these systems are considered to provide adequate accuracies. However, portability and reusability are questionable when porting to new domains. Rule based approach to shallow parsing consists of coding the necessary knowledge in a set of rules written by linguists. Rules are usually universal i.e. they are not domain dependent. Moreover, machine learning based methods demand high quality tagged corpus. So in absence of such data, one has no other option than rule based methods. If inconsistently and inaccurately annotated data is available, a machine learning system will learn to model the inconsistencies and inaccuracies and will achieve high score on test data that has same properties as the training material.

NLP research around the world has taken giant leaps in the last decade with the advent of efficient machine learning algorithms and the creation of large annotated corpora for various languages. However, NLP research in Indian languages has mainly focused on the development of rule based systems due to the lack of annotated corpora.

Statistical Methods	**Rule Based Methods**
(+) easy process of training	(-) rule generation is quite cumbersome process
(+) language and tag set independent	(-) dependent on language and tag set
(-) need large training data	(+) doesn't need training data
(-) not reusable to new domains	(+) usable with new domains
(-) data sparsity needs to be handled carefully	(+) no special attention is required
(-) quality of corpus matters	(-) quality of linguistic rules matters

Table 1.1: Comparison between statistical and rule based methods for shallow parsing

1.5 Challenges Faced

Statistical or rule based methods, both have their own challenges that need to be addressed. Table 1.1 lists the pros and cons of both the approaches. Following challenges are normally faced while building a good shallow parser in either of the way:

- Statistical methods require a good training data. Training data requires human parsing or manual tagging and usually this data is expensive, because it has to be produced by trained linguists and in any case cannot be produced automatically by definition.

- Even though a large training data exists, it can be noisy: contains interruptions, corrections, typographical errors etc. Moreover some data is inherently ambiguous and cannot at all be tagged correctly without semantic analysis.

- Training data can be inconsistent, with one instance of an identical or similar text being tagged one way and with another instance being tagged in a different way. It is difficult for a machine learning algorithm to produce a correct set of rules from such training data since it is unclear what the correct output is.

- Shallow parsing needs many context words; it may require three to five context words to reach their best results. This can have impact on the efficiency on the algorithms and on their resilience to noise.

- The issues of reusability are retained when porting the trained models to new domains.

- Even though large amount of data is available for training, unknown word handling is an issue of major concern. Data sparsity needs to be handled carefully while implementing shallow parsers for morphologically rich language like Marathi.

- Indian languages like Marathi are morphologically rich in nature, developing rule based taggers and chunkers which is a cumbersome process do not perform well for POS tagging and chunking for Indian languages.
- Marathi is a free word order language, implying that no fixed order is imposed on the word sequence. This creates difficulties for a statistical tagger as many permutations of the same sentence are possible.
- In case of rule based approach, the accuracy of shallow parser highly depends upon the quality of regular expression rules which involves many costs in formulating rules. Moreover it is easy to introduce inconsistencies in a rule based system when number of rules increases.
- Many POS taggers use morphological analyzer as a module in their tagger. But building morphological analyzer to a particular Indian language itself is a very difficult task.

1.6 Outline of the Report

Rest of the report is organized as follows: Chapter 2 is devoted to related work done in shallow parsing for Indian languages. Chapter 3 gives brief introduction to Marathi morphology, chapter 4 describes the architecture of Marathi shallow parser. The experiments and results are discussed in chapter 5. Chapter 6 concludes the report with the possible directions for future work.

Chapter 2

Related Work

In this chapter, we discuss the previous work done on shallow parsing. We primarily focus on work done on POS tagging and chunking of Indian languages.

2.1 Previous Work on Foreign Languages

2.1.1 POS Tagging

For English, there are many POS taggers, employing machine learning techniques such as Hidden Markov Models [Brants, 2000], transformation based error driven learning [Brill, 1995], decision trees [Black et al., 1992], maximum entropy methods [Ratnaparkhi, 1996], conditional random fields [Lafferty et al., 2001]. There are also taggers which are hybrid using both stochastic and rule-based approaches, such as CLAWS [Garside and Smith, 1997]. Morphology-based POS tagging of some languages like Turkish [Oflazer and Kuruöz, 1994], Arabic [Tlili-Guiassa, 2006], Czech [Hajic et al., 2001], Modern Greek [Giorgos et al., 1999] and Hungarian [Megyesi, 1999] have been tried out using a combination of hand-crafted rules and statistical learning. These systems use large amount of corpora along with morphological analysis to POS tag the texts.

2.1.2 Chunking

The earliest work on chunking based on machine learning goes to [[Church, 1988]] for English. [[Ramshaw and Marcus, 1995]] used transformation based learning using a large annotated corpus for English. [[Skut and Brants, 1998]] modified Church's approach and used standard HMM based tagging methods to model the chunking process. [[Zhou et al., 2000]] continued using the same methods, and achieved an accuracy of 91.99% precision and 92.25% recall using a contextual lexicon. [[Kudo and Matsumoto, 2001]] use support vector machines for chunking with 93.48% accuracy for English. [[Veenstra and van den Bosch, 2000]] use memory based phrase chunking with accuracy of 91.05% precision and 92.03% recall for English. Some other techniques in English are based on Generalized Winnow [[Zhang et al., 2002]], Conditional Random Field [[Sha and Pereira, 2003]], maximum entropy models [[Koeling, 2002]], [[Molina and Pla, 2002]] using specialized HMM. These chunkers have reached accuracy of approximately 94%. These accuracies are aided by the availability of large annotated corpus for English.

2.2 Previous Work on Indian Languages

Due to the lack of annotated corpora, previous research in POS tagging and chunking in Indian languages has mainly focused on rule based systems utilizing the morphological analysis of word-forms. [[Bharati et al., 1995]] in their work on computational Paninian POS parser, described a technique where POS tagging is implicit and is merged with parsing phase. [[Ray et al., 2003]] proposed an algorithm that identifies Hindi word groups on the basis of the lexical tags of the individual words.

2.2.1 Hindi POS Tagger Using Naive Stemming

[[Shrivastava and Bhattacharya, 2008]] presented HMM based POS tagger for Hindi which employs a naive (longest suffix matching) stemmer as a pre-processor to achieve reasonably good accuracy of 93.12%. Their work does not require

any linguistic resource apart from a list of possible suffixes for the language and hence proved that even without employing tools like morphological analyzer or resources like a pre-compiled structured lexicon, it is possible to harness the morphological richness of Indian Languages. The aim is to utilize the morphological features of Hindi without resorting to complex and expensive linguistic analysis.

Exploded Input Model

The heart of the approach is to "explode" the input in order to increase the length of the input and to reduce the number of unique types while learning. The input words are splitted into stem and suffix, thus doubling or " exploding" the size of input data. A simple list of all possible suffixes in the language is prepared and is used for splitting input words, resulting in a crude and not very linguistically sound stemming. However, this "longest suffix splitting" rules out the need for rule based stemmer system which would have made one to rely on extensive linguistic resources. After exploding the input in this way, newly generated suffix tokens need to be given appropriate tags. Different schemes are proposed on how to tag these suffixes and comparative study is presented with each scheme.

Training HMM

Now the new input training set S_E where $S_E = R \cup M$; R is the stem set and M is the set of suffixes and tag set T is replaced with $T_E = T \cup T_s$; T_S is the set of suffix tags (i.e. suffix list) and T is the original POS tag set.

The HMM remains the same as the standard HMM as all the required changes are made to the training and testing data at a pre-processing stage. Results are evaluated against this exploded input HMM and significant increase in overall accuracy is noted down as compared to standard HMM (one without exploding the input).

2.2.2 Feature Rich POS Tagger for Hindi

[[Aniket Dalal, 2007]] employed Maximum Entropy Markov Model (MEMM) with a rich set of features capturing the lexical and morphological characteristics

of language. The features are based on information retrieved from a lexicon generated from the corpus, a dictionary and a stemmer. Average accuracy of 94.38% is reported with MEMM model. Following features are defined for training:

- *Contextual features:* The size of context window is determined based on empirical observations. The best result of 85.59% is obtained with the context window consisting of POS tag of previous word, current word and next word. The model with only contextual features is assumed to be baseline model.

- *Morphological features:* Morphological features are adapted to handle unseen words. Suffix information provided by stemmer is considered as morphological feature.

- *Lexical features:* Features that detect anomalies in the text and features that deal with special symbols and punctuation symbols are added.

- *Categorial Features:* The more restricted set of POS tags is used as categorial feature. This feature helps in tagging of unseen words when there is no explicit bias for a word in learnt model.

- *Compound Features:* These are added to handle proper nouns.

Preprocessing of Data

Preprocessing of data is done in order to generate the list of unique words in training corpus called 'lexicon' and a restricted dictionary called 'TinyDict'. The lexicon also stores a boolean flag 'Yes/No' indicating if word has occurred as proper noun or not. Lexicon serves 2 purposes: 1.As a list of seen proper nouns and 2.As an indicator for seen words so that the information from the restricted dictionary can be utilized for unseen words. For every word in the corpus, TinyDict stores information about the list of possible POS tags according to dictionary. One more resource containing list of suffixes for all words is generated in preprocessing step.

The various experiments are carried out to measure the influence of various feature functions described above. Addition of restricted set of POS tags improved

the accuracy significantly. This approach can be extended to other languages just by building language appropriate resources like lexicon and stemmer.

2.2.3 Stochastic Models for Bengali POS Tagging

⟦Dandapat et al., 2007⟧ tried HMM and Maximum Entropy (ME) based stochastic taggers for Bengali. Supervised and semi supervised bi-gram HMM and ME based models are tried. Three taggers are implemented as follows: 1.HMM-S: It is based on supervised HMM model parameters 2.HMM-SS: Based on the semi-supervised model parameters 3.ME: Based on Maximum Entropy approach.

Feature Selection

POS tags produced by MA is integrated with these models to further increase accuracy. Suffix information is used during smoothing of emission probabilities in case of HMM and it is used as feature in ME model. The combination of contextual features (current word and previous tags), prefixes and suffixes of length ≤ 4 are used as features for ME model. The model parameters for supervised HMM and ME models are estimated from the annotated text group. For semi supervised learning, the HMM learned through supervised training is considered as initial model. Further, a larger unlabeled training data is used for re-estimating parameters of semi supervised HMM.

Observations

The use of POS tags gave better results than the use of suffix information. With combination of both the results are even better. The percentage of improvement is higher when amount of training data is less. The ME model does better than HMM model for smaller training data. But with higher amount of training data performance of HMM and ME model are comparable. Semi supervised HMM performs when very little tagged data is available, however they are slightly poorer than supervised HMM for moderate size training data and use of suffix information.

2.2.4 Assamese POS Tagger

[Navanath et al., 2009] reported first work on the POS tagger for Assamese, which is a morphologically rich, agglutinative and relatively free word order language similar to other Indian languages. They reported accuracy of 87% with the use of HMM. More accurate results are expected with larger training corpora. In their work, they use a large POS tag set containing 172 POS tags.

Approach

Four database tables are built using probabilities extracted from the manually tagged corpus: word-probability table, previous-tag-probability table, starting-tag-probability table and affix-probability table. Each word of a sentence is stored in an array. After that, each word is searched in the word-probability table. If the word is unknown, its possible affixes are extracted and searched in the affix-probability table. From this search, the probable tags and their corresponding probabilities for each word are obtained. All these probable tags and the corresponding probabilities are stored in a two dimensional array which is called as the lattice of the sentence. If probable tags and probabilities for a certain word are not obtained from these two tables, a tag CN (Common Noun) and probability 1 is assigned to the word since occurrence of CN is highest in the manually tagged corpus. After forming the lattice, the Viterbi algorithm is applied to the lattice that yields the most probable tag sequence for that sentence.

2.2.5 HMM Based Hindi Chunker

[Singh et al., 2005] proposed a first chunker for Hindi using HMM with an accuracy of 91.7%. The chunking task is broken up into two subtasks: first, identifying the chunk boundaries and second, labeling the chunks with their syntactic categories.

Chunk Boundary Identification

They used HMMs trained on different tag schemes (STRT, CNT, STP, STRT_STP)

to mark the chunk boundaries, where STRT stands for start of the chunk, CNT is continuation of the chunk, STP is the end of the chunk and STRT_STP is used when chunk contains single word. Three different schemes were proposed using these tags: 2-tag scheme {STRT, CNT}, 3-tag scheme {STRT, CNT, STP} and 4-tag scheme {STRT, CNT, STP, STRT_STP}. Comparative study is done for these schemes. These tag schemes are applied to different formats of training data where different formats of training data consisted of either word only or POS tags only or word and POS both. All the tag schemes are finally converted to 2-tag scheme {STRT, CNT} while evaluating the results. Highest accuracy is reported with 4-tag scheme. Detailed error analysis showed that for some tags (PRP, QF, QW, RB, RP etc.) a combination of word and POS performes better, while for other tags only POS information performs better.

Chunk Labeling

Once the chunk boundaries are marked, chunk label identification is done as a next step. Two methods are presented for chunk label marking: HMM based chunk labeling and rule based chunk labeling. In case of machine learning approach, chunk boundary tags are augmented with the chunk labels for learning purpose. However, a rule based system is claimed to be effective over machine learning method. This is because there are only five types of chunks and hence application of rules to find out chunk type is very effective.

2.3 Shallow Parsing Contest for South Asian Languages

Apart from all these efforts, a contest was held as a part of IJCAI (International Joint Conference on Artificial Intelligence) workshop [[SPSAL, 2007]] on "Shallow Parsing for South Asian Languages" in 2007, in which participants built shallow parsers for three Indian languages Hindi, Bengali and Telugu.

A wide range of learning techniques starting from Navie Bayes, HMMs, Decision Trees to Maximum Entropy Model and Conditional Random Fields are tried

out in the submitted systems of shallow parsers. In the rest of this chapter, we give a brief overview of different implementations of shallow parsers submitted in this workshop.

2.3.1 Data Set

All the implementations are designed for three languages Hindi, Bengali and Telugu with 20000 words of training data, 5000 words of development data and 5000 words of test data in all the three languages. The test data contained approximately one third of unseen words in Hindi and Bengali whereas for Telugu this number is alarmingly high. It is due to the agglutinative nature of Telugu where a root can combine with other word or a suffix to form a larger word-form.

2.3.2 Statistical POS Taggers

1. [[Rao and Yarowsky, 2007]] used HMM for POS tagging and Out Of Vocabulary (OOV) words are assigned a tag that maximizes the likelihood of the tag given a fixed length suffix of the word. That is, if the word is OOV then predicted $tag = argmax_{tag} P(tag/suffix)$. In absence of morphological taggers, a fixed length suffix can serve as an approximation to the morphological model. Hence suffix information is used to predict OOV tags. However, length of suffix chosen was different for Hindi, Bengali and Telugu (2,3 and 3 respectively). [[Rao and Yarowsky, 2007]] also showed that TnT [[Brants, 2000]] performs better than QTag [[Tufis and Mason, 1998]], eventhough both are Hidden Markov Model tagger implementations. Accuracies of 69.07%, 73.90% and 72.38% are obtained for Bengali, Hindi and Telugu respectively.

2. [[Sastry et al., 2007]] proposed HMM based POS tagger. The Trigrams 'n' Tags i.e. the TnT tagger model [[Brants, 2000]] is used to mark the POS tags. The tagger implements the Viterbi algorithm for second order Markov models. As mentioned in [[Sastry et al., 2007]], TnT is an efficient statistical part-of-speech tagger that can be trained on different languages and

on virtually any tag set. The component for parameter generation trains on tagged corpora. The system incorporates several methods of smoothing and of handling unknown words. TnT is not optimized for a particular language. Instead, it is optimized for training on a large variety of corpora. In this model, smoothing is carried out using linear interpolation, and the respective weights are determined by deleted interpolation. Accuracies of 74.58%, 78.35% and 75.27% are obtained for Bengali, Hindi and Telugu respectively.

3. [[Ekbal et al., 2007]] also based their POS tagger on HMM for Bengali, Hindi and Telugu with accuracies as 73.17%, 76.87% and 67.69% respectively. However, they used different approach for handling unknown words. Following 3 approaches are adopted to handle unknown words in case of Bengali:

   - *Suffix information:* The list of suffixes for Bengali is used which contains the suffixes appearing at the end of verb, noun and adjective. A null suffix is also kept for those words that have none of the suffixes in the list. The probability distribution of particular suffix with respect to specific POS tags is generated from all words in the corpus that share the same suffix.
   - *Named Entity Recognition:* NER system in Bengali is used to handle unknown words. A pattern based bootstrapping method is used to develop this NER system. The Bengali training data is searched for named entities by looking at POS tags, either NNPC or NNP. Then NER algorithm generates a lexical pattern p using a context window of width 4 around each occurance of named entities in the training data. The new patterns are then generated from these patterns from bootstrapping. All these patterns form a set P of potential patterns. Every pattern p in the 'P' is matched against the test data, to find the NEs. If identified NEs are found to be unknown words, then NNPC and NNp tags are assigned to them accordingly.

- *Lexicon Features:* To handle the unknown words further, a lexicon consisting of approximately 20000 entries has been used. The lexicon has been developed in a semi-supervised way from a tagged Bengali news corpus, developed from the web. Lexicon contains the root words and their basic part of speech information such as: noun, verb, adjective, adverb, pronoun and indeclinable. If an unknown word is found in the lexicon, then the corresponding part of speech information is extracted.

4. ⟦Dandapat, 2007⟧ adopted a maximum entropy (ME) based POS tagger. The tagger makes use of morphological and contextual information of words. It reported accuracy of 77.61%, 75.69% and 74.47% for Bengali, Hindi and Telugu respectively. Their tagger and chunker are based on ⟦Ratnaparkhi, 1996⟧ POS tagger.

Feature Selection

The feature set used for POS tagging consists of contextual window of size 3, prefix and suffix information(Here, the prefix/suffix is a sequence of first/last few characters of a word, which does not mean a linguistically meaningful prefix/suffix.). Morphological information is further integrated with the ME model. The morphological analyzer outputs the possible POS tags for a word, thus restricting the set of possible tags.

Absence of Sound Morphological Information

In reality it is possible that the morphological analyzer is neither complete nor sound. ⟦Dandapat, 2007⟧ carried out two different experiments to understand the relative performance of the Bengali tagger under two situations: 1.When morphological information is neither sound nor complete (ME+IMA) 2.When morphological restriction is sound and complete (ME+CMA). (ME+CMA) gave an increment of 9%, while (ME+IMA) gave an increment of 4% which is still a promising result.

2.3.3 Combined Methods for POS Tagging

1. [[PVS and G, 2007]] used CRFs and TBL for building POS tagger and achieved accuracy of about 77.37%, 78.66% and 76.08% for Telugu, Bengali and Hindi respectively. They showed that improved training methods based on morphological information, contextual and the lexical rules help in achieving good results. The baseline template consisted of current word and window size of 4.

   Improving Features

   The baseline template is further enriched by adding morphological information like root word, suffix information, prefix information. The last two letters, last three and last four letters of the word are added as suffix information to the corpus. Root information of the word and probable tags of the word are extracted from morphological analyser. The size of the word also played the major role in tagging with threshold on size as three. This marked 1% improvement in accuracy with the reason being average length of non-functional words in Indian languages is around 3. Finally a set of transformation rule is applied to correct the errors produced by CRFs.

2. [[Rao et al., 2007]] also used combination of statistical approach (HMM) and rule based approach for POS tagging. The claim is, in POS tagging, when a hybrid system is used, disadvantages of one component can be reduced by using other component. When a word is unknown it can not be tagged by statistical model, but gets correctly tagged by rule based method.

   Data Sparsity

   In a language, there are always new words which may not be there in training corpus however large, the training corpus is. This is called as lexical item sparseness. In such cases, value of emission probability of HMM becomes zero. This can be handled by considering transition probabilities of HMM. But due to sparseness of data, it is found in some instances values of transition probabilities also to be zero. This may be called as structural

sparseness.

Rule Based System

Whenever, both probability distributions (emission and transition) give value of zero, the linguistic rules are used to tag the words. This rule based system consisted of context sensitive rules which can be applied across Indian languages ang lexical rules specific to Dravidian languages. Thus [[Rao et al., 2007]] used linguistic rules for smoothing instead of using statistical smoothing methods. Accuracy obtained is 72.17%, 76.34% and 53.17% for Bengali, Hindi and Telugu respectively.

Approach	**Resources Used**	**Features**	**Average Accuracy**
HMM	NER, Lexicon and Suffix list for Bengali	Lexical Features	72.57%
HMM	-	Lexical Features	71.78%
HMM	-	Suffixes for unseen words	76.06%
HMM + Rule Based	-	Lexical Features	67.22%
MEMM	Morphological Analyzer	Lexical Features + Suffix and Prefix	75.92%
CRF + TBL	Morphological Analyzer	Root, POS, Suffix, Prefix, Length of word	77.37%

Table 2.1: Summary of Approaches for POS Tagging

2.3.4 Statistical Chunkers

1. For chunking [[PVS and G, 2007]] tried out HMMs to mark chunk lables. Then the system is trained for predicting the chunk bounderies using CRF with the basic features such as words, POS tags and combination of both.

Finally, the chunk lables and chunk boundary names are merged to obtain chunk tag. It is basically HMM+CRF model for chunking. Chunker performed at 79.15%, 80.97% and 82.74% for Telugu, Hindi and Bengali respectively.

2. ⟦Rao and Yarowsky, 2007⟧ used Conditional Random Fields (CRF) and showed how to improve the performance of machine learning system by the feature enhancement process. They used following feature enhancement techniques:

   - *Relabeling*: Standard BIO notation which is used to mark chunk bounderies was extended to new notation called BIES where - Begin(B), Inside(I), End(E) and Singleton(S). BIES notation is functionally equivalent to BIO notation and lossless bidirectional conversion from one to other is possible. But representation in BIES form allows better discrimination of the chunk bounderies.
   - *Handling punctuations*: Trailing punctuation symbols are purged out of the chunk and separate chunks are created for them. The claim was, punctuation hinders learning of patterns in the data and performance can be improved by separating punctuations into separate stand alone chunks.
   - *Effect of composite classifiers*: The chunking task is treated as a composite classification task where two separate classifiers were learnt for the boundery prediction and the chunk labeling task respectively. However, lower performance was reported with CRF model.

3. ⟦Sastry et al., 2007⟧ used the Cocke-Kasami-Young (CKY) algorithm for chunking and obtained the accuracy as 70.99%, 69.92% and 74.74% for Bengali, Hindi and Telugu respectively. The training files are used to extract chunk templates and following parameters are estimated:

   - *Chunk-POS combinations:* When a chunk template C is discovered with the POS tag of the word just outside the chunk being p, the count

of "chunk-POS" combination C-p is increased.

- *Chunk structure:* The label of a chunk indicates the kind of chunk it is. *e.g.* If the chunk template is ((NP UH SYM)), then the label of the chunk is NP, and it consists of words with POS tags UH and SYM.

Then a dynamic programming algorithm is used to find a chunk ordering for each sentence such that the probability of occurrence of the chunk sequence is maximized.

4. [[Dandapat, 2007]] based his chunker on maximum entropy model(ME) and reported the accuracy figures as 80.59%, 74.92% and 68.59% for Bengali, Hindi and Telugu respectively. Features used for chunking are current word, POS tag of current word, POS tags of previous 2 words, POS tags of next 2 words and chunk tags of previous two words. As the chunker is heavily loaded with the POS tags of the context words, experiments were done to understand the performance of the chunker under the situations: 1.When POS labels are gold standard and 2.When POS labels are outputted by his own POS tagger which is also based on ME model. The performance is better in first case as expected.

2.3.5 Rule Based Chunkers

1. [[Rao et al., 2007]] preferred Transformation Based Learning (TBL) approach for chunking, where a single base rule (or a few base rules) is provided to the system and other rules are learned by system itself during the training phase for recognization of the chunks. Accuracies of 65.28%, 73.80% and 50.38% are reported for Bengali, Hindi and Telugu respectively.

2. [[Ekbal et al., 2007]] also used rule based approach with the rules designed for Bengali. Chunking algorithm is divided into two steps: chunk boundary identification and chunk labeling. The chunk boundaries are identified by some handcrafted linguistic rules that check whether two neighboring POS

tags belong to the same chunk or not. Once the boundaries are identified, the components within a chunk help to assign the label on the chunk. They obtained accuracies of 80.63%, 71.65% and 53.15 for Bengali, Hindi and Telugu respectively.

Approach	**Resources Used**	**Features**	**Average Accuracy**
Rule Based	Rules for Bengali	-	68.47%
Transformation Based	-	Lexical Features	63.15%
ME	-	current word + context window of 1 containing only POS	74.70%
CRF	-	Word + POS and BIES notation	88.51%
HMM + CRF	-	Word + POS	80.95%

Table 2.2: Summary of Approaches for Chunking

2.4 Moral of the Story

All this literature survey gives emphasis on usage of machine learning techniques for POS tagging and chunking as both can been seen as sequence labeling tasks. However performance of these techniques might vary depending upon the corpora available.

It can be seen from table 2.1, higher accuracies are accompanied by use of some of the morphological features, *e.g.*, suffix, prefix or POS tags *etc*. This reveals the fact that, even though the same data set is used, accuracies can be improved with the help of linguistic analysis.

The work on Indian languages flashes an important message: Most of the Indian languages are resource poor languages, but a high accuracy classifier can be

built for them by harnessing their morphological richness. To be specific, detailed linguistic analysis of morphosyntactic phenomena and adroit handling of suffixes can offset the need for a large corpus. Thus *statistical methods coupled with careful morphological analysis can be used to build shallow parser for highly inflectional languages like Marathi.*

Chapter 3

Marathi Morphology

Morphology is the study of forms of words in the language and it deals with the morphemes, where morpheme is a term which refers to the smallest component of a word that (a) seems to contribute some sort of meaning, or a grammatical function to the word to which it belongs, and (b) cannot be decomposed into smaller morphemes.

Morphological analysis of Marathi plays significant role in natural language processing because Marathi, a pan Indian Language, is rich in morphology. Marathi is an Indo-Aryan language spoken by the people of western and central India. It is the official language of the state of Maharashtra and it is estimated to be more than 1300 years old, evolving from Sanskrit.

3.1 Morphological Richness of Marathi

Morphologically rich languages are characterized by a large number of morphemes in a single word, where morpheme boundaries are difficult to detect because they are fused together. Marathi is highly inflectional and agglutinative language. The possible inflections of a single word are huge in number. For example, table 3.1 shows some of the possible inflections of a word झाड {zaad} (tree). The different forms of a single word झाड are formed here.

Marathi is agglutinative language meaning words are formed by fusing differ-

Inflected Form	Meaning
झाडावर	on the tree
झाडाचा	of the tree
झाडाला	to the tree
झाडाने	by the tree

Table 3.1: Inflections of a Single Word

ent morphemes together. For example, consider a word झाडासमोरच्याने (zadasamorchyane) (by the one in front of the tree).

$$\text{झाडासमोरच्याने} = \text{झाड} + \text{समोर} + \text{च} + \text{ने}$$

Here the morphemes झाड, समोर, च and ने are fused together to form a final word.

Not all these inflected or derived Marathi words can be stored in the dictionary. The statistics given in [[Dixit et al., 2006]] says that for a single noun in Marathi, over 200 forms that are either adjectives or adverbs may be possible. Similarly, a verb may exhibit over 450 forms. At the same time, the language is expected to include over 10,000 nouns and over 1,900 verbs. Over 175 postpositions can be attached to nominal and verbal entities. That is why, dictionaries can not cover all possible inflections, derivations and compounds obtainable from all root words. The morphological richness of the language suggests that implementation of shallow parser should be supported by morphological features of the language. Hence we give an overview of Marathi morphology in the next section.

3.2 Word Categories

Morphological analysis of Marathi is done category wise where 8 categories are defined for Marathi. Parameters for changes in the root word of every category are identified. Following are word categories defined in Marathi:

3.2.1 Nouns

Nouns cover the most of the part of the language. Nouns in Marathi exhibit highly inflectional and derivational behavior. Marathi nouns inflect for gender, number and case. When they inflect for case, in most of the cases, a single stem is formed from the root and then case markers and the postpositions can get attached to this stem. This makes inflectional Marathi noun morphology absolutely economic in nature. Traditionally, the process of stem formation from a root word is called as *saamaanyaroopa* formation, where *saamaanyaroopa* (oblique form) is the stem form of a noun, to which the case markers or the postpositions are attached. Table 3.2.1 illustrates the *saamaanyaroopa* formation of some nouns. Saamaanyaroopa

Noun	**Saamaanyaroop**	**Steps**	**Words from same paradigm**
देव	देवा	ultimate insertion of आ	दुकानदार, विषय
कपडा	कपड्या	ultimate deletion of आ ultimate insertion of या	माळा, हिवाळा
रहिवासी	रहिवाशा	ultimate deletion of ई penultimate deletion of स penultimate insertion of श ultimate insertion of आ	आदिवासी, निवासी

Table 3.2: Saamaanyaroop Formation

formation involves the insertions and deletions at ultimate and penultimate positions of root word. Depending upon these insertions and deletions as well as some other factors like gender, number etc., nouns are classified into various paradigms. The last column in table 3.2.1 lists some of the nouns which fall under same paradigm of the noun given in the first column. That is, they will follow the same steps to form the stem from the root word.

There are no generalized rules for inflections of nouns with gender or number. However, the process of saamaanyaroop formation is applicable to plural form of nouns (Table 3.2.1 gives saamaanyaroops of singular forms of nouns). That is, a common stem is formed for plural form of a noun and suffixes are attached to it.

For example, saamaanyaroop of plural form of word देव is देवां. Marathi nouns also show highly derivational behavior. Consider again the word झाडासमोरच्याने. Table 3.3 illustrates how the noun झाडcombines with different morphemes to form a new word, every time belonging to different category.

Word	**Category**	**Meaning**
झाड	Noun	tree
झाडासमोर	Adverb	in front of the tree
झाडासमोरचा	Adjective	the one in front of the tree
झाडासमोरच्याने	Noun	by the one in front of the tree

Table 3.3: Deriving words from a noun

3.2.2 Pronouns

Number of pronouns and their respective inflected forms are finite and less when compared to verbs and nouns. Number of inflected forms of a pronoun and the rules describing such inflection are almost equal in number. Hence all inflected forms of the pronouns are stored in the lexicon instead of deriving rules for them.

3.2.3 Adjectives

Adjectives are mainly, inflectional and non-inflectional. For example, सुन्दर (beautiful), नाजूक (fragile) are the example of non-inflectional adjectives, while चांगला and कोरडा are the inflectional adjectives. Adjectives don't have their own existence. They agree in gender and number with the nouns they modify. Inflectional adjectives can act as nouns in the sentence. Table 3.2.3 lists inflections of some adjectives according to number and person of nouns they modify. Adjectives are also derived from nouns, pronouns and verbs. When genitive case markers or some Shabdayogi avyays are attached to nouns, it produces adjectives. These forms are automatically covered in noun, pronoun and verb morphology.

	Masculine	Feminine	Neuter
Singular	शहाणा	शहाणी	शहाणं
Plural	शहाणी	शहाण्या	शहाणी

Table 3.4: Inflection of an Adjective

3.2.4 Verbs

Marathi verbs inflect for gender, number and person of the subject and the direct object in a sentence. They also inflect for tense and aspect of the action as well as mood of the speaker. Postpositions can be attached to verb stems also. For example, करणे (do) and केल्यानंतर (after done). Inflectional Verb Morphology covers inflection of verb root for gender, number, person, tense, aspect and mood. Derivational Verb Morphology covers derivation of words of other parts of speech obtained either by an attachment of suffixes or postpositions to a verb stem. Inflectional morphology of Marathi verbs is based on *aakhyaata* theory while derivational morphology of Marathi verbs is based on *krudant* theory of [[Damale, 1970]].

Aakhyaata

Aakhyaata refers to a group of suffixes. These suffixes are closing suffixes and they agree with person. Aakhyata attaches to a verb root to form a finite verb form.

कर + तो = करतो

verb root + aakhyaata = verb form

The phonemic shape of the Aakhyaata pratyaya is the basis of naming the Aakhyaatas. Classification of aakhyaata prtyayaas is done on the basis of phonemic shape as well as the possibility of inflection for gender. Table 3.5 gives the classification based on phonemic shape and illustrates some examples of verb root बस { bas} (sit)

Aakhyaata	Inflection for Gender	Examples
प्रथम ताख्यात(Prathama taakhyaata)	Yes	बसतो, बसते, बसतात
द्वितीय ताख्यात (Dwitiya taakhyaata)	Yes	बसता, बसत्या
लाख्यात (Laakhyaata)	Yes	बसलो, बसल्या, बसले
वाख्यात (Vaakhyaata)	Yes	बसावा, बसावीस, बसाव्या
ई-आख्यात ('I:' aakhyaata)	No	बसे, बसा, बसत
ऊ-आख्यात ('U:' aakhyaata)	No	बसो, बसोत
ईलाख्यात ('I:laa'khyaata)	No	बसशील, बसतील
चाख्यात (Chaakhyaata)	Yes	बसायचो, बसायची, बसायचे

Table 3.5: Classification of Aakhyaatas

Krudanta

As mentioned above, Krudanta theory forms the base of derivational morphology of Marathi verbs. Krudant refers to a group of suffixes when attached to a verb root, derives some other categories of language, specifically nouns, adjectives and adverbs. Krudant generates non-finite verb forms in Marathi. Table 3.6 [Valambe, 2007] lists the krudants along with examples.

3.2.5 Adverbs

Adverbs in Marathi can be inflectional and non-inflectional. For example, हळूहळू (slowly) and भरभर (fast) do not inflect. When adverbs inflect, they generally inflect for attachment of postpositions. For example, खाली (under) and खालपासून (from the underneath). Non-inflectional adverbs are stored in lexicon while doing morphological analysis. Attachment of postpositions to nouns, adjectives, verbs and pronouns is one of the strategies of adverb formation. The set of these derived adverbs is automatically covered at the level of morphology of postpositions, nouns, verbs, adjectives and pronouns.

Krudant Type	**Example**			
	Krudant Suffix	**Root**	**Example Sentence**	**Category of Derived Form**
णे-krudant	णे	वाग	हे **वागणे** ठीक नाही	Noun
त-krudant	त	खा	ती नेहमी **खात** असते	Aderb
ता-krudant	ता	दा	तूच आमचा **दाता** आहेस	Noun
	ता	चाल	तो घरी **चालता** झाला	Adjective
	ता	वाज़	मी चार **वाजता** जाईन	Adverb
ताना-krudant	ताना	धाव	तो **धावताना** पडला	Adverb
ऊ-krudant	ऊ	खा	मला **खाऊ** दे	Noun
ऊन-krudant	ऊन	बस	खाली **बसून** जेव	Adverb
ला-krudant	ला	कर	**केल्याने** होत आहे रे	Noun
	ली	झाक	त्याची **झाकली** मूठ माहीत आहे	Adjective
वे-krudant	वे	जा	घरी **जावयास** उशीर होईल	Noun
	वे	फ़स	ते लोक **फ़सवे** आहेत	Adjective

Table 3.6: Types of Krudants

3.2.6 Postpositions

Postposition is the morpheme that follows the words and shows the relation between the word that is followed and other word in the sentences. Case markers (vibhakti prtyaya) and *shabdayogi avyaya* are classified as postpositions in Marathi because they show same behavior. *shabdayogi avyayas* are not inflected by gender, number or person. In Marathi, postpositions are attached to all classes of words except interjection.

3.2.7 Conjunctions and Interjections

All conjunctions and interjections in Marathi are non-inflectional.

3.3 Harnessing Marathi Morphology

We repeat here the saying that Marathi exhibit highly inflectional and derivational behavior. Also it is agglutinative and free word order language. These morphological features of Marathi challenge the task of shallow parsing, if not harnessed correctly.

Inflectional Behavior

As mentioned in section 3.1, not all the words can be seen in training data because of highly inflectional nature of Marathi words. This may lead to incorrect labeling of unseen words.

Derivational Behavior

Consider following two sentences,
धाकटा मुलगा गुणी आहे.
धाकटा गुणी आहे.
Here the word धाकटा is adjective, its category is changed to noun in second sentence. This hinders the process of learning the model parameters.

Agglutinative Behavior

There can be more words like झाडासमोरच्याने which are formed by fusing more than two morphemes. A large number of permutations are possible of various morphemes. All these can not be present in training data, and hence model can not learn them leading to failure of correct labeling of unseen words.

Free Word Ordering

Consider a sentence, "I like mango". Table 3.7 lists all valid ordering of this sentence in Marathi. Thus many permutations of a single sentence are possible

Marathi Sentence	**POS Sequence**
मला आंबा आवडतो	PRP NN VB
मला आवडतो आंबा	PRP VB NN
आंबा मला आवडतो	NN PRP VB
आंबा आवडतो मला	NN VB PRP
आवडतो मला आंबा	VB PRP NN
आवडतो आंबा मला	VB NN PRP

Table 3.7: Free Word Ordering in Marathi

which imposes difficulty while learning the model.

These problems can be overcome by integrating morphological information, specifically suffix information and categories produced by Morphological Analyzer (MA). In the next section, we present three motivating examples which show that if morphology is harnessable then above mentioned problems are not debilitating.

3.3.1 Appreciating Suffixes

First, consider the following Marathi sentence,

हा रस्ता दोन गावांना **जोडणारा** आहे.

*haan raasta don gavaanna **jodnara_VM** aahe.*
*this road two villages **connecting** is*
*this is the road **connecting_VM** two villages.*

The word जोडणारा {jodnara} (connecting) in the above sentence is a verb and can be categorized as such by simply looking at the suffix णारा {naaraa} as this suffix does not appear with any other POS category. A statistical POS tagger which does not use suffix information fails to identify the correct POS tag for this word when the word does not appear in the training data. On the other hand, when suffix information is used as a feature the same classifier is able to identify the correct POS tag of जोडणारा {jodnaara} even when it does not appear in the training data. Hence, using suffix information ensures that a classifier is able to learn meaningful patterns even in the absence of large training data. Next, we consider two examples for chunking,

- **VGNN (Gerund Verb Chunk)**
  माणसाने उडण्याचा प्रयत्न केला.
  *maanasane **udnyacha_VGNN** prayatna kela.*
  *man **fly** try do*
  *man tried **flying_VGNN**.*

- **VGINF (Infinitival Verb Chunk)**
  त्याने चालायला सुरुवात केली.
  *tyaane **chalayla_VGNF** suruvaat keli.*
  *he **walk** start did*
  *he started **to walk_VGINF**.*

Here, we are dealing with the case of two specific verb chunks, *viz.*, VGNN (gerund verb chunk) and VGINF (infinitival verb chunk). A chunk having a gerund always gets annotated as VGNN and a chunk having an infinitival verb always gets annotated as VGINF. Thus, the correct identification of these verb chunks boils down to the correct identification of gerunds and infinitival verb forms in the sentence which in turn depend on the careful analysis of suffix information. For example, in Marathi, the attachment of the verbal suffix "ण्याचा"

{nyAchA} to a verb root always results in a gerund. Similarly, the attachment of the verbal suffix "आयला" {AyalA} to a verb root always results in an infinitival verb form. The use of such suffix information as features can thus lead to better generalization for handling unseen words and thereby reduce the need for additional training data. For instance, in the first sentence, even when the word "उडण्याचा" {udnyacha} does not appear in the training data, a classifier which uses suffix information is able to label it correctly based on its experience of previous words having suffix "ण्याचा" {nyacha} whereas a classifier which does not use suffix information fails to classify it correctly.

3.3.2 Restricting Categories

It is known that POS for a word is restricted to a limited set of tags. For example, a word पकड {pakad} has one of two possible categories noun (tongs) and verb (hold). Assume that the POS tag of a word w can take values from the set $T_{MA}(w)$, where $T_{MA}(w)$ is computed by Morphological Analyzer (MA). If T is the set of all POS tags, then note that, the size of $T_{MA}(w)$ is much smaller than T. Thus we can restrict choice of tags as well as tag sequences for a given sentence which in turn helps in learning the model. As stated in [[Aniket Dalal, 2007]], this feature is crucial for unseen words as there is no explicit bias for a word in built in model and an artificial bias is created with the help of limited tag set. A special case of this feature is when the restricted set has only one tag and implies that a word is tagged with that particular POS category with very high probability. Consider again same example sentence:

मला आंबा आवडतो.
malA AmbA Avadato
I mango like
I like mango.

Category list of the word आंबा {AmbA} (mango) produced by MA will have only one entry as Noun. Thus word आंबा is tagged correctly with a high confidence even if it is not seen/seen with different ordering of the sentence in the training.

On the other hand, if category features by MA are not used, classifier might fail to identify it correctly.

Chapter 4

Architecture of Marathi Shallow Parser

In this chapter, we describe the architecture of Marathi shallow parser. The work on Hindi POS tagging [[Singh et al., 2006]] comes closest to our approach where accuracy of (93-94%) is reported for Hindi. The methodology uses exhaustive morphological analysis backed by highcoverage lexicon, locally annotated corpora and a CRF based learning algorithm. The heart of the system is the detailed linguistic analysis of morphosyntactic phenomena of Marathi, efficient use of suffix information and feature engineering to generate rich features for CRF based classifier.

This approach can be used for other inflectional languages by providing the language specific resources in the form of suffix replacementrules (SRRs), lexicon and morpheme analysis rules *etc.* and keeping the processes the same as shown in Figure 4.1 and 4.2.

4.1 Ambiguity Schemes

The word is said to have an ambiguity when word can have multiple POS categories depending upon its context. For example, in the following sentence, word *back* can have POS tags either verb or noun or adjective depending on its context.

*I get **back_[Verb]** to the **back_[Adjective]** seat to give rest to my **back_[Noun]***

As mentioned in [[Singh et al., 2006]], the criterion to decide whether the tag of a word is a noun or a verb is entirely different from that of whether a word is an adjective or an adverb. Therefore Ambiguity Schemes are defined. For example, some word can occur as conjunction, post-position or a noun, then it falls in an Ambiguity Scheme 'Conjunction-Noun-Postposition' and if some word can occur as a Noun or a Verb, then it falls in an Ambiguity Scheme 'Noun-Verb'. Similarly all the ambiguous words are grouped into sets according to the Ambiguity Schemes that are possible in Marathi, *e.g.*, Adjective-Noun, Adjective-Adverb, Noun-Verb, etc. This idea was first proposed by [[Giorgos et al., 1999]] for Modern Greek POS tagging.

For example, the word जात {jaat} meaning cast or go (cast- noun, go- VM/VAUX) can appear as a noun or a main verb or an auxiliary verb. Hence it falls in the ambiguity scheme <NN-VM-VAUX>.

4.2 Resources

1. **Lexicon:** The most important resource is the lexicon. Lexicon stores the words in the language in their uninflected form along with their paradigm and category information. All the words that inflect in the similar way are grouped together in one paradigm. Example entries in the lexicon:
   <word>, <paradigm>, <category>
   <दोरी>, <परी>, <noun>
   <आहेर>, <भक्त>, <noun>
   If words can have multiple POS categories, then word will have that many entries in the lexicon with respective paradigm and category information. We have the lexicon containing 23,615 words.

2. **Suffix Replacement Rule Files:** Another important resource is the Suffix Replacement Rule (SRR) file, used by Morphological Analyzer. The suffix replacement rule encodes the information needed to get the root word from

the inflected word form. Suffix replacement rules are different for different paradigms and one paradigm may have more than one rules. Format of the rule is as follows:

Paradigm, Ultimate Deletion, Penultimate Deletion, Penultimate Insertion, Ultimate Insertion

Here ultimate insertion and deletion specify what is to be inserted or deleted from the ultimate position of the word form, while penultimate insertion and deletion specify what is to be inserted or deleted from the penultimate position of the word form. Example entry in the SRR file is given below:

<रास> <स,,,शी>

3. **Training Data:** Besides the above mentioned resources, POS tagged and chunk tagged data are required for training purpose.

4.3 POS Tagger

The proposed Marathi POS tagger contains 2 important components. First, a morphological analyzer which provides ambiguity schemes and suffix information for generating a rich set of features. These features are then fed to a CRF based engine which couples them with other elementary features (previous/next words and bigram tags) for learning a sequence labeler. Each of these components is described in detail in the following sub-sections.

4.3.1 Morphological Analyzer

We use the the Morphological Analyzer (MA) developed at IITB [Pawar, 2008, Bapat, 2010] which is used in generating morphological features for training. MA is a completely rule based system comprising of a lexicon and a list of suffix replacement rules. As mentioned above, lexicon contains root form and paradigm of the words along with their Ambiguity Scheme. All the words that inflect in the similar way are grouped together in one paradigm. Ambiguity Scheme refers to the list of possible POS categories a word can take. This can add valuable infor-

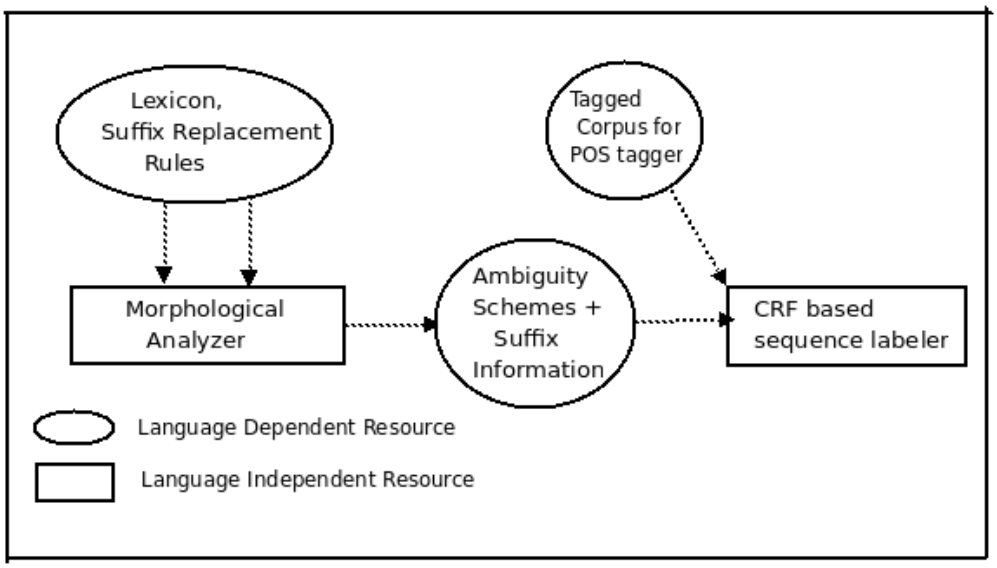

Figure 4.1: Architecture of POS Tagger

mation to a sequence classifier by restricting the set of possible POS categories for a word.

During the analysis process, stemmer is used to segment the word into stem and suffixes. These suffixes are used directly as features for training a sequence labeler. Next, we need the root form of the word so that the Ambiguity Scheme of the word can be identified using a lexicon lookup. The root form is obtained from the stem using suffix replacement rules. In the case of nouns, these suffix replacement rules are developed by looking at the morphological patterns of the words in a paradigm. We have a list of around 16,187 nouns categorized into 78 paradigms. For verbs, the identification of root form is simple as it just involves scanning the list from right to left and stripping of suffixes by looking up a list of valid suffixes. Once the root form is obtained, the Ambiguity Scheme can be looked-up using the lexicon described above.

4.3.2 CRF

Conditional Random Fields [[Lafferty et al., 2001]] are undirected graphical models used for labeling sequential data. Under this model, the conditional probability distribution of a tag given the observed word sequence is given by,

$$P(Y|X;\lambda) = \frac{1}{Z(X)} \cdot e^{\sum_{t=1}^{T}\sum_{k=1}^{K}\lambda_k f_k(Y_{t-1},Y_t,X,t)} \tag{4.1}$$

where,

$$X = source\ word$$
$$Y = target\ word$$
$$T = length\ of\ source\ word$$
$$K = number\ of\ features$$
$$\lambda_k = feature\ weight$$
$$Z(X) = normalization\ constant$$

We used CRF++[1], an open source implementation of CRF, for training and further decoding the tag sequence. We used the following features for training the sequence labeler (here, w_i is the *i-th* word, t_i is the *i-th* pos tag).

Features used for POS tagger

- t_i and w_j such that $i-2 < j < i+2$
- $t_i\ t_{i-1}$ and w_j such that $i-2 < j < i+2$
- t_i and suffix information of w_i
- $t_i\ t_{i-1}$ and suffix information of w_i
- t_i and ambiguity scheme of w_i
- $t_i\ t_{i-1}$ and ambiguity scheme of w_i

Here, the first two features are ***weak features*** which depend only on the previous/next words and previous tags. The last four features are rich ***morphological features*** which make use of the output of the morphological analyzer.

[1] http://crfpp.sourceforge.net/

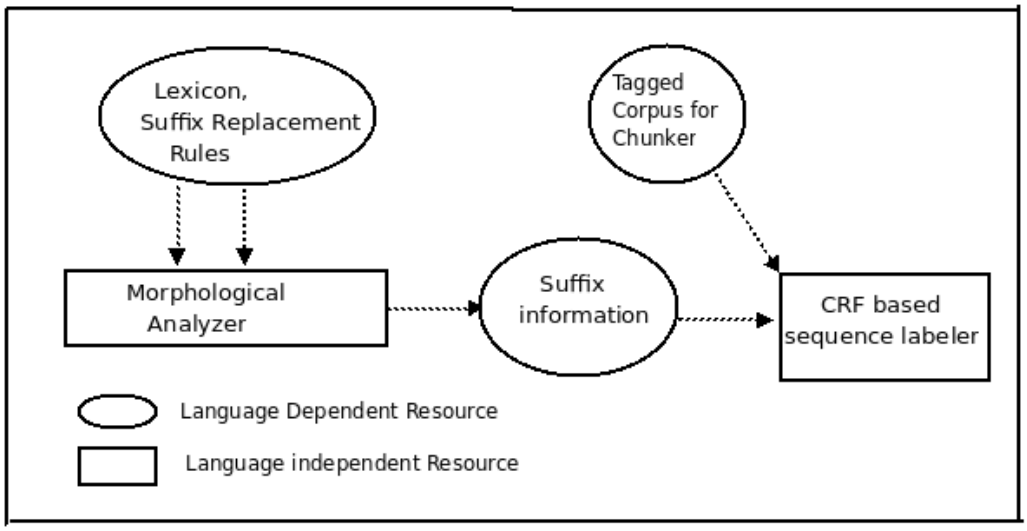

Figure 4.2: Architecture of Chunker

4.4 Chunker

Morphological analyzer and CRF based training algorithm are the basic components of Marathi chunker as shown in figure 4.2

4.4.1 Morphological Analyzer

As described in previous section, a rule based morphological analyzer is used to generate morphological features like suffix information for the training of a chunker. The architecture of a morphological analyzer is same as described in section 4.3.1. It used to provide rich features for CRF based engine.

4.4.2 CRF

Again we have used CRF++, an open source implementation of CRF, for training the chunker. We used the following features while training the chunker (here, w_i is the *i-th* word, t_i is the *i-th* pos tag and c_i is the *i-th* chunk tag).

Features used for Chunker

- c_i and t_j, w_j such that $i-1 < j < i+1$
- c_i c_{i-1} and t_j, w_j such that $i-1 < j < i+1$
- c_i and suffix information of w_i
- c_i c_{i-1} and suffix information of w_i

Here again, the first two features are ***weak features*** and the last two features are ***rich morphological features***.

Chapter 5

Experiments and Results

In this chapter, we discuss the various experiments done on POS tagging and chunking. We divide the experiments into two steps. Initially we describe how the various feature sets are tried to obtain the best accuracy figures for POS tagger and chunker. Next we do the comparative analysis of these feature sets against the data size to show that appropriate feature enhancement can go a long way in improving the accuracy of a state of the art sequence classifier and simply throwing in a large amount of annotated corpora does not serve the purpose. We show that even smaller size corpus coupled with a wise linguistic insight can beat a large size data in terms of accuracies.

5.1 Data Set

We used the training data from TOURISM and NEWSPAPER domain for all our experiments. This data was hand annotated by two lexicographers adept in Marathi. The total size of the corpus was kept large (106273 POS tagged words and 63033 chunks) to study the impact of the size of training data versus the amount of linguistic information used. In each of the experiment, we first divided the data into four folds (75% for training and 25% for testing). The statistics about each POS tag and chunk tag are summarized in Table 5.1 and Table 5.2

POS Tag	Frequency in Corpus	POS Tag	Frequency in Corpus
NN	51047	RP	359
NST	578	CC	3735
PRP	8770	QW	630
DEM	3241	QF	1928
VM	17716	QC	2787
VAUX	6295	QO	277
JJ	7311	INTF	158
RB	1060	INJ	22
UT	97	RDP	39
PSP	69	NEG	154

Table 5.1: POS Tags in Training Data

Chunk Tag	Frequency in Corpus	Chunk Tag	Frequency in Corpus
NP	40254	JJP	2680
VGF	7425	VGNF	3553
VGNN	1105	VGINF	58
RBP	782	BLK	2337
CCP	4796	NEGP	43

Table 5.2: Chunk Tags in Training Data

5.2 Accuracy Measures

The POS tagger and chunker are able to assign tags to all the words in test sets and hence the precision and recall figures of the POS tagger and chunker are same and have been considered as the accuracy of the models. In case of chunker, tagwise accuracy and chunkwise accuracies are defined. Tagwise accuracy measures the total number of correctly identified tokens. We have considered chunkwise accuracy, which gives the credit to identifying correct chunk boundaries as well as identifying correct chunk labels within the chunk boundaries. Tagwise accuracy gives credit to partially identified chunks also, whereas chunkwise doesn't give credit to them.

5.3 Feature Variations for POS Tagging

We performed the step by step feature engineering to achieve the best accuracy figures for the POS tagger. The experiments are divided into two different settings:

5.3.1 Weak Features (WF)

Here we use the basic CRF classifier with elementary word features (*i.e.*, words appearing in a context window of 2[1]) and bigram tag features. Overall accuracy of 85% is obtained with this set. Figure 5.1 shows the accuracies of individual POS tags. Confusion matrix is drawn as shown in table 5.4. It can be seen from the confusion matrix that VM and VAUX are major source of errors. Also all the POS tags are getting confused as NN. Careful analysis of this result showed that most of the unknown words are tagged as NN. Table 5.3.1 gives the idea about the data sparsity present in the some of the POS tags. Second column gives the number of unknown words present in corresponding POS tag, while column 3 gives the percentage of *wrongly marked unknown words in each tag*.

[1]Some experiments were done with the context size and then window size 2 is fixed for further experiments.

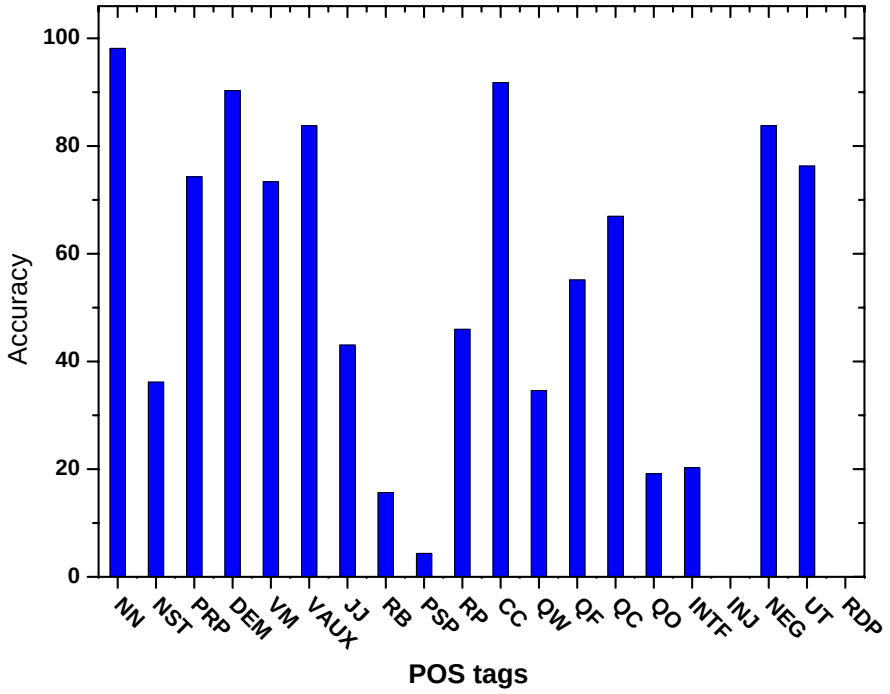

Figure 5.1: Per-POS Accuracy Distribution Using WF

POS Tag	Tagwise Unseen Word %	Error % of unseen words
NST	16	100
PRP	6	100
VM	16	63
VAUX	4	77
JJ	22	98
RB	24	100
QW	10	100
QF	9	100

Table 5.3: Unseen Words Statistics with WF

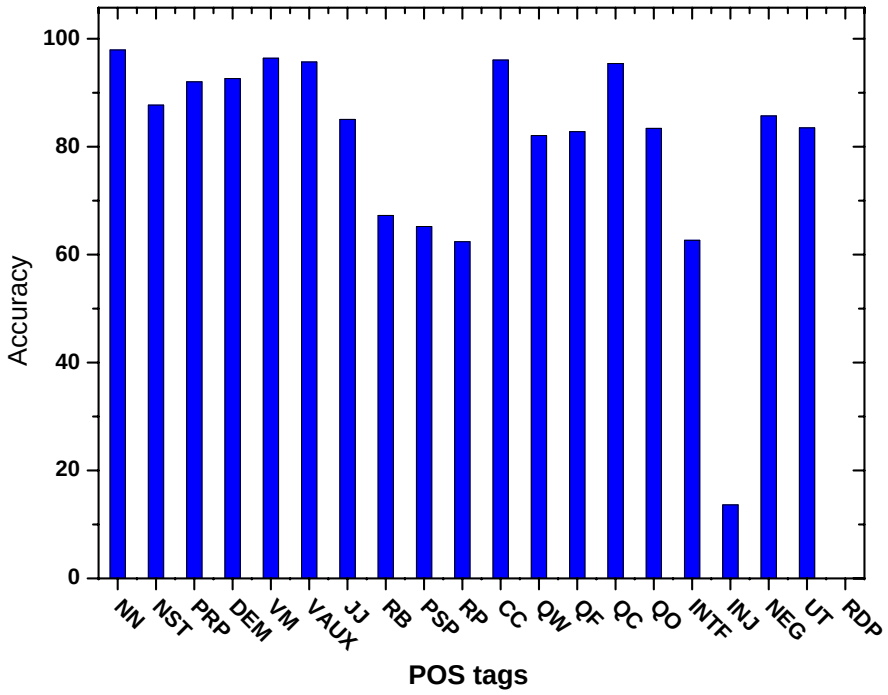

Figure 5.2: Per-POS Accuracy Distribution Using MF

5.3.2 Morphological Features (MF):

In addition to the elementary features, we use the ambiguity schemes and suffix information provided by the morphological analyzer. Accuracy is boosted to 95% when either of these information is used. Accuracies of individual POS tags are shown in Figure 5.2. Note that, using morphological features has affected the performance of verb tags significantly. Table 5.5 shows the confusion matrix with MF. Note that, inclusion of morphological information has reduced the verb errors as well as the confusions being a noun.

Addition of morphological features has also improved the performance of unseen words in some tags. Table 5.3.2 gives the idea about percentage of errors of unseen words after morphological features are applied. Comparing this with 5.3.1, we can see that number of errors in unseen word tagging are reduced. Thus

	NN	NST	PRP	DEM	VM	VAUX	JJ	RB	PSP	RP	CC	QW	QF	QC	QO	INTF	INJ	NEG	UT	RDP
NN	50092	0	63	1	621	23	142	23	0	0	17	3	35	18	2	0	0	1	0	0
NST	337	209	8	0	21	0	1	0	0	0	1	0	0	0	1	0	0	0	0	0
PRP	1756	0	6515	341	68	16	3	10	0	0	29	15	13	3	0	0	0	0	0	0
DEM	99	0	207	2926	4	1	2	0	0	0	1	0	0	0	0	0	0	0	0	0
VM	3876	0	3	8	12995	807	20	0	0	0	6	0	0	0	0	0	0	0	0	0
VAUX	271	0	1	1	748	5273	0	0	0	0	0	0	0	0	0	0	0	0	0	0
JJ	3824	2	2	0	244	9	3147	4	0	0	2	0	75	0	0	2	0	0	0	0
RB	808	2	4	0	24	1	31	166	0	0	4	0	10	9	1	0	0	0	0	0
PSP	62	0	0	0	1	0	1	0	3	0	0	1	0	0	0	0	0	0	0	0
RP	35	0	4	3	6	4	0	0	0	165	139	2	1	0	0	0	0	0	0	0
CC	158	0	98	0	6	1	3	0	0	38	3428	0	0	0	0	0	0	0	3	0
QW	309	0	12	0	45	6	0	0	0	0	0	218	40	0	0	0	0	0	0	0
QF	721	1	3	0	29	3	95	2	0	0	0	0	1063	9	0	2	0	0	0	0
QC	879	0	0	0	28	1	1	3	0	0	0	0	5	1866	4	0	0	0	0	0
QO	210	0	0	0	7	1	0	1	0	0	0	0	0	5	53	0	0	0	0	0
INTF	84	0	0	0	3	0	10	2	0	0	0	0	27	0	0	32	0	0	0	0
INJ	20	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
NEG	12	0	1	0	11	1	0	0	0	0	0	0	0	0	0	0	0	129	0	0
UT	10	0	0	0	2	0	1	0	0	1	9	0	0	0	0	0	0	0	74	0
RDP	22	0	0	3	6	0	0	3	0	0	0	2	2	1	0	0	0	0	0	0

Table 5.4: Confusion Matrix for POS tagging with WF

	NN	NST	PRP	DEM	VM	VAUX	JJ	RB	PSP	RP	CC	QW	QF	QC	QO	INTF	INJ	NEG	UT	RDP
NN	49988	18	92	2	167	4	548	75	0	0	24	4	61	47	9	0	0	1	0	0
NST	33	507	9	0	3	0	6	13	3	0	1	0	2	0	1	0	0	0	0	0
PRP	145	3	8071	312	8	5	6	13	0	0	44	114	42	4	1	0	1	0	0	0
DEM	3	0	231	3002	2	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
VM	225	1	4	9	17078	347	44	1	0	0	4	0	0	0	0	0	0	2	0	0
VAUX	10	0	1	1	257	6025	0	0	0	0	0	0	0	0	0	0	0	0	0	0
JJ	816	8	3	0	45	1	6218	80	0	0	1	0	131	1	1	6	0	0	0	0
RB	167	16	10	0	10	1	106	713	0	0	5	0	15	11	2	3	0	0	0	1
PSP	18	1	0	0	0	0	1	0	45	1	1	1	0	0	0	0	0	0	0	0
RP	6	1	4	1	1	1	1	1	1	224	114	2	1	0	0	0	0	1	0	0
CC	18	2	72	0	1	0	3	3	0	43	3588	0	0	0	0	0	0	1	4	0
QW	8	0	58	0	0	0	0	0	0	0	0	517	47	0	0	0	0	0	0	0
QF	90	5	38	0	2	1	138	18	0	0	0	15	1596	11	0	14	0	0	0	0
QC	78	0	0	0	1	0	12	10	0	0	0	0	14	2659	13	0	0	0	0	0
QO	24	0	0	0	2	0	5	7	0	0	0	0	1	7	231	0	0	0	0	0
INTF	3	0	1	0	0	0	19	8	0	0	0	0	28	0	0	99	0	0	0	0
INJ	15	0	0	0	3	0	0	0	0	0	0	0	0	0	0	0	3	0	1	0
NEG	3	0	0	0	12	1	2	0	0	3	1	0	0	0	0	0	0	132	0	0
UT	2	0	0	0	1	0	0	0	0	2	11	0	0	0	0	0	0	0	81	0
RDP	4	2	5	3	9	2	2	5	0	0	0	3	3	1	0	0	0	0	0	0

Table 5.5: Confusion Matrix for POS tagging with MF

without integrating additional training data, we could handle the words which were unknown in the corpus. As discussed in the section 3.3, morphological features assist in tagging unseen words by making use of restricted POS tags and suffix information.

POS Tag	**Tagwise Unseen Word %**	**Error % of unseen words**
NST	16	52
PRP	6	32
VM	16	8
VAUX	4	31
JJ	22	38
RB	24	61
QW	10	46
QF	9	67

Table 5.6: Unseen Words Statistics with MF

5.4 Feature Variations for Chunking

5.4.1 Weak Features

Again we divided all the features sets into two sets. As with the POS tagger, weak features consist of the elementary word features, POS tags and bigram tag features. Figure 5.3 shows the individual chunkwise accuracy using WF (Word, POS tag and contextual window of size 1). The detailed error analysis of all the feature sets in WF is given below. It helps to understand how could we derive the best features in WF that give the best accuracy measures. Table 5.7 presents the confusion matrix calculated with these features in WF set. It can be seen from the table, major errors are saturated in verb chunks and hence they need special attention.

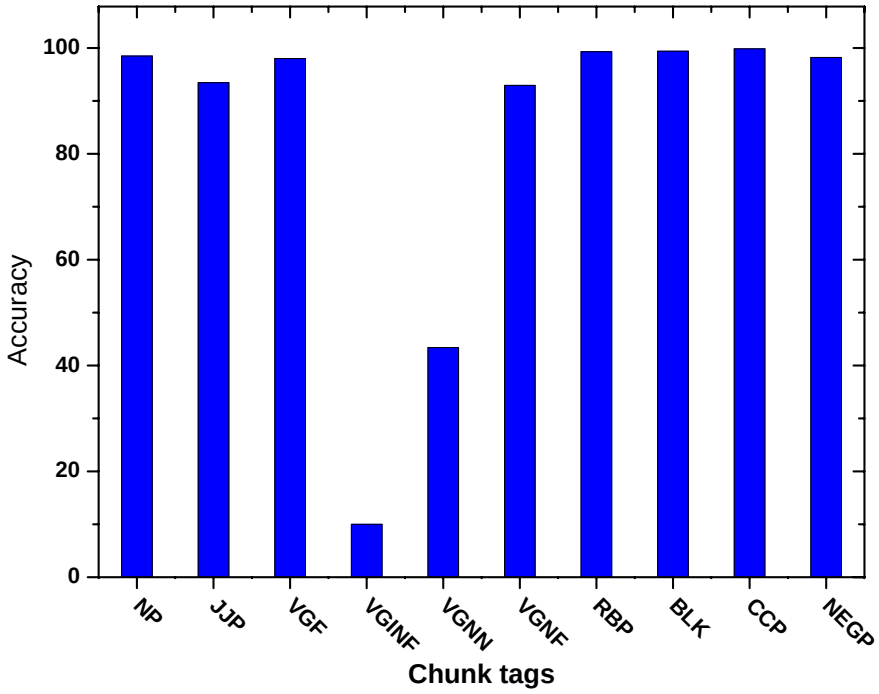

Figure 5.3: Per-Chunk Accuracy Distribution Using WF

	NP	JJP	VGF	VGINF	VGNN	VGNF	RBP	BLK	CCP	NEGP
NP	88346	124	79	0	0	6	7	77	10	0
JJP	266	4300	1	0	3	2	3	0	0	0
VGF	13	0	20783	0	23	242	1	1	0	0
VGINF	0	0	13	9	16	59	0	0	0	0
VGNN	4	0	280	0	797	850	0	0	0	0
VGNF	8	0	350	5	99	5241	0	0	0	0
RBP	6	3	1	0	0	0	1112	0	0	0
BLK	22	0	1	0	0	0	0	3536	0	0
CCP	9	1	2	0	0	0	0	0	7213	0
NEGP	1	0	0	0	0	0	0	0	0	55

Table 5.7: Confusion Matrix for Chunking with WF

WF1

Features used: Word, its POS tag and window size = 0
We started with the simplest feature set. As we did not consider any contextual information in this feature set, only atomic features, that is word and its POS tag are the possible features.
Results:

- Tagwise Accuracy: 84%
- Chunkwise Accuracy: 80.09%

Chunk Type	Accuracy in %
NP	72.02
JJP	90.44
VGNN	49.94
VGINF	15.5
VGF	98.13
VGNF	93.23
CCP	99.85
RBP	99.16
NEGP	98.21
BLK	99.38

Analysis: As we can see from results, accuracies of chunk types CCP, RBP, NEGP and BLK are good compared to other chunks. The reason is the length of these chunks in most of the cases is one only. Their chunks tags do not depend upon surrounding words or surrounding chunks. Moreover, their POS tags (CC, RB, NEG) uniquely determine the chunk tags as, CCP, RBP and NEGP only. For the POS tags CC, RB and NEG, there are no other possible chunk tags. This in turn reduces the scope of making wrong choice, hence increases the accuracy. While in case of all types of verb chunks, the accuracy is very low except VGF (No of occurrences of VGF in training data are large.). All forms of verbs are marked as either VM (verb main) or VAUX(auxiliary verb) as their POS tags. But it is hard to predict the VGF, VGNF, VGNN or VGINF chunks just by seeing local information of a word. For example, consider a following sentence:

मी पळायला गेले होते
PRP VM VM VAUX
B-NP B-VGINF B-VGF I-VGF

Consider, chunking of a word, पळायला. When model sees its POS tag as VM and word itself, the probability of tagging it as VGF is high, just by seeing POS tag. Similar reason holds for NP. Noun phrases usually consist of more than one

word. Hence they are related to their neighboring words. Absence of contextual information affects their accuracy. For example:

छान पुस्तक आहे
JJ NN VM
B-NP I-NP B-VGF

POS tag of a word छान is JJ. With no contextual information available, it can not be easily determined if छान forms JJP chunk or it is part of some NP chunk. Consider, one more example sentence:

लोकमान्य टिळकांचा विजय असो
NN NN NN VM
B-NP I-NP B-NP B-VGF

Here 3 words form two noun chunks. It is not possible to identify chunk boundaries unless surrounding words are known. Thus noun chunks are heavily context dependent.

WF2

Features used: Word, its POS tag and window size = 1
Now we added some contextual information to previous feature set by changing window size to 1.
Results:

- Tagwise Accuracy: 97.43
- Chunkwise Accuracy: 96.91

Chunk Type	Accuracy in %
NP	98.49
JJP	93.45
VGNN	43.4
VGINF	10
VGF	98
VGNF	92.93
CCP	99.85
RBP	99.3
NEGP	98.21
BLK	99.4

Analysis: Overall accuracy is significantly increased. Accuracy of NP is increased drastically for the reason stated above.

WF3

Features used: Word, its POS tag and window size = 2

Results:

- Tagwise Accuracy: 97.30
- Chunkwise Accuracy: 96.78

Chunk Type	Accuracy in %
NP	98.54
JJP	93.42
VGNN	40.60
VGINF	4
VGF	97.86
VGNF	91.45
CCP	99.84
RBP	99.06
NEGP	98.21
BLK	99.38

Analysis: We tried one more experiment by increasing window size to 2. However we didn't find any noticeable difference in the overall accuracies (accuracies decreased a bit). We could not make any conclusion about window size and hence decided to do one more experiment with window size = 3.

WF4

Features used: Word, its POS tag and window size = 3
Results:

- Tagwise Accuracy: 97.14
- Chunkwise Accuracy: 96.6

Chunk Type	**Accuracy in %**
NP	98.51
JJP	93.34
VGNN	39.71
VGINF	4.44
VGF	97.53
VGNF	90.09
CCP	99.84
RBP	98.97
NEGP	98.21
BLK	99.38

Analysis: Increasing window size to 3 reduced accuracies a bit. It is observed that accuracies are dropping with increasing size of window. The experiments with window size were performed to find out optimal size. Just increasing window size is not affecting accuracy significantly, rather there is decrease in both the accuracies. Moreover, increasing window size leads to increase in number of features generated by CRF and hence the training overhead. So it is necessary to think of some other features.

WF6

Before adding some new features to above experiments, let's check the accuracy with only POS tag information.
Features used: POS tag and window size = 1
Results:

- Tagwise Accuracy: 95.41
- Chunkwise Accuracy:95.00

Chunk Type	**Accuracy in %**
NP	97.41
JJP	93.53
VGNN	10.10
VGINF	0
VGF	97.17
VGNF	84.13
CCP	99.80
RBP	99.25
NEGP	98.21
BLK	99.52

Analysis: There is a decrease in both the accuracies. The decrease in verb chunks. The reason is very clear, it is hard to tag chunks, specifically verb chunks, just by seeing sequence of POS tags. It is the word only which can make difference between VGF or VGNF or VGNN or VGINF chunk tag. Note that, accuracies of the chunks like CCP, RBP, NEGP are not affected at all because they can be tagged by looking at the POS information.

WF7

Features used: POS tag and window size = 2
Results:

- Tagwise Accuracy: 95.48
- Chunkwise Accuracy: 94.93

Chunk Type	**Accuracy in %**
NP	97.45
JJP	93.56
VGNN	15.21
VGINF	2.22
VGF	95.98
VGNF	83.41
CCP	99.74
RBP	99.16
NEGP	98.21
BLK	99.29

Analysis: Similar conclusions are obtained and we state again words are required to distinguish between verb chunks. But chunks like CCP, NEGP and RBP perform very well even with only POS information.

Here, we would like to mention the purpose of experimenting with POS information only. Indian languages have similar grammar structure with some minor differences. If some model is trained on one language and it could perform well on another language having similar grammatical structure, then one doesn't need to worry about the training corpus for other language which comes at a very high manual cost. So if the model performs well with POS information only, we could test it on some other language. The results are very promising and further experiments can be done on this.

5.4.2 Morphological Features

In addition to the elementary features, suffix information provided by the morphological analyzer is included in this set. So features in final MF set for chunker includes word, its POS tag, suffix information of word and contextual window

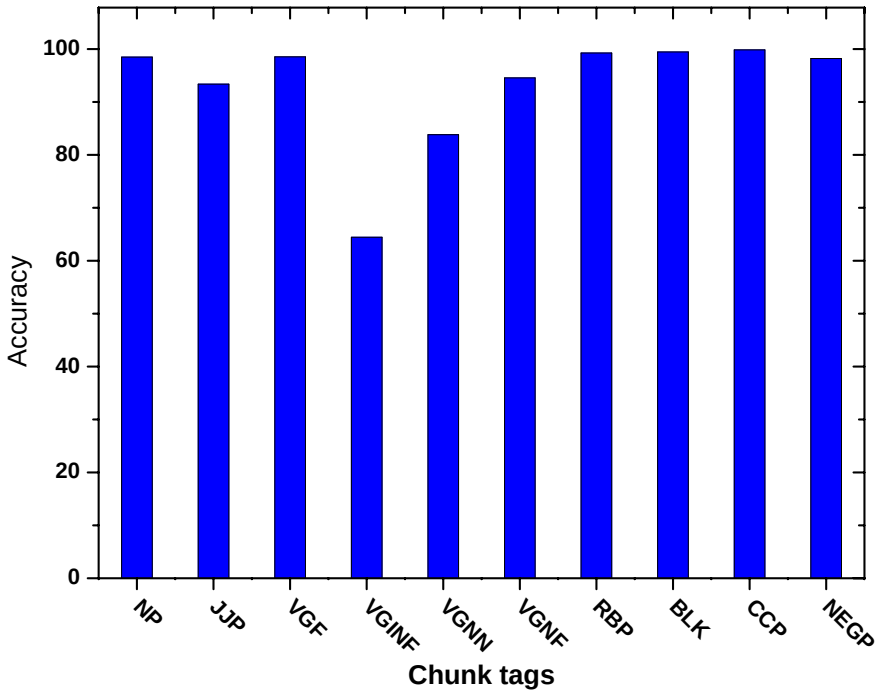

Figure 5.4: Per-Chunk Accuracy Distribution Using MF

of size 1. Chunkwise accuracies for this set can be seen from figure 5.4. Suffix information obtained from morphological analyzer helps to correct the confusion between various verb chunks. Table 5.8 gives the confusion matrix calculated with these MFs and shows that no of errors in verb chunks are reduced with inclusion of suffix information. Before arriving at this set, we tried some experiments on window size as detailed below.

MF1

Features used: Word, its POS tag, suffix and window size = 1
Results:

- Tagwise Accuracy: 98.18
- Chunkwise Accuracy: 97.87

	NP	JJP	VGF	VGINF	VGNN	VGNF	RBP	BLK	CCP	NEGP
NP	88346	126	80	0	0	4	7	77	9	0
JJP	267	4299	1	0	4	1	3	0	0	0
VGF	15	0	20857	0	39	150	1	1	0	0
VGINF	0	0	11	58	7	21	0	0	0	0
VGNN	4	0	163	0	1570	194	0	0	0	0
VGNF	7	0	229	14	106	5347	0	0	0	0
RBP	7	3	1	0	0	0	1111	0	0	0
BLK	21	0	1	0	0	0	0	3537	0	0
CCP	10	1	2	0	0	0	0	0	7212	0
NEGP	1	0	0	0	0	0	0	0	0	55

Table 5.8: Confusion Matrix for Chunking with MF

Chunk Type	**Accuracy in %**
NP	98.49
JJP	93.37
VGNN	83.82
VGINF	64.44
VGF	98.53
VGNF	94.55
CCP	99.84
RBP	99.25
NEGP	98.21
BLK	99.46

Analysis: We added suffix information to the feature set. Now our training data consists of word, its POS tag, suffix of a word and chunk tag. For a particular word, if suffix is not present, then we added a tag "NoSuff" in the suffix position of that word. Accuracies of verb chunks increased significantly and hence overall accuracy. (Accuracies of CCP, NEGP and other tags were already good). The

reason is verb chunks VGNN and VGINF are ended with some specific suffixes. VGNN are the verb forms which take suffixes of nouns, while VGINF verb forms end with suffix "आयला". Examples of VGINF are "करायला", "जेवायला" etc. Hence, addition of suffix information helps in making correct choice of VGNN and VGINF. Most of the noun chunks are also ended with suffix, hence suffix information helps in NP chunking also.

MF2

Features used: Word, its POS tag, suffix and window size = 2
Results:

- Tagwise Accuracy: 98.08
- Chunkwise Accuracy: 97.80

Chunk Type	**Accuracy in %**
NP	98.52
JJP	93.45
VGNN	80.85
VGINF	55.55
VGF	98.48
VGNF	94.21
CCP	99.84
RBP	99.06
NEGP	98.21
BLK	99.41

Analysis: We tried window size experiments again with above feature set MF1. No significant changes found in the result.

MF3

Features used: Word, its POS tag, suffix and window size = 3
Results:

- Tagwise Accuracy: 98.01
- Chunkwise Accuracy: 97.72

Chunk Type	**Accuracy in %**
NP	96.18
JJP	97.2
VGNN	94.95
VGINF	80.85
VGF	98.52
VGNF	95.78
CCP	99.78
RBP	98.51
NEGP	98.73
BLK	57.14

Analysis: With window size increased, slight drop in accuracies is being observed. Thus we go with best one that is window size = 1. So the feature set that gave us best accuracy contains word, its POS tag, suffix information and contextual window of size 1.

5.5 Linguistic Insight Vs Statistical Might

Next, we try to compare the linguistic wisdom versus statistical brawn. For that, we varied the training data in increments of 10K and calculated the accuracy of WF and MF models of POS tagger and chunker to see the impact of size of training data. The x-axis represents the size of the training data and the y-axis represents the precision of the tagger/chunker.

Figure 3 plots the average precision of the POS tagger across all categories using WF and MF for varying sizes of the training data. Figure 6 plots the average precision of the chunker across all categories using WF and MF. Next, to show that the impact of morphological analysis is felt more for verbs than other POS categories we plot the accuracies of verb pos tags (Figure 4) and verb chunk tags

(Figure 7) using WF and MF for varying sizes of the training data. We also plot the accuracies of non-verb POS tags and non-verb chunk tags in Figure 5 and 8.

5.5.1 Linguistic attention abandons a large size corpora

Figure 3 shows that using a large amount of annotated corpus(91k words), the best accuracy one can hope for is around 85% if morphological information is not harnessed. i.e., if only weak features are used adding more data will definitely not be of much use as the curve is already close to saturation. On the other hand, if morphological information is harnessed then accuracy as high as 94% can be obtained by using data as small as 18k words. Figure 6 says a similar story. In the absence of morphological features a large amount of annotated corpus(60k words) is needed to reach an accuracy of 96% whereas if suffix information is used then the same accuracy can be reached using a much smaller training corpus(20k words). This clearly shows that while dealing with morphologicall rich languages, time and effort should be invested in building powerful morphological analyzer.

5.5.2 Importance of verbs

Figure 4 shows that in case of POS tagging using suffixes as features results in a significant increase in accuracy of verbs. Specifically accuracy increases from 40% to 95% using a very small amount annotated corpus (18K words). Comparing this with figure 5 we see that while using morphological information definitely helps other POS categories, the impact is not as high as that felt for verbs. Figures 7 and 8 for chunking show a similar pattern i.e., the accuracy of verb chunks is affected more by morphology as compared to other chunk tags. These figures claim that verbs is where all the action lies and they indeed need special treatment in terms of morphological analysis.

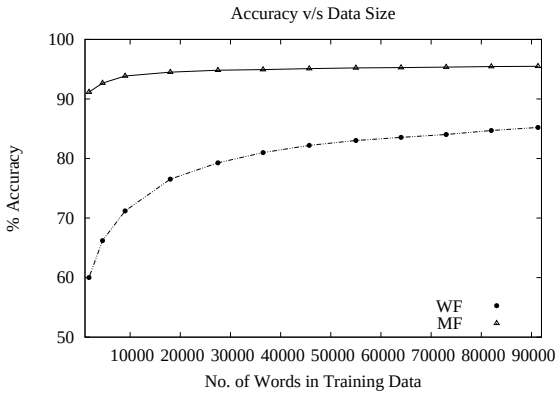

Figure 3: Average Accuracy of all POS Tags

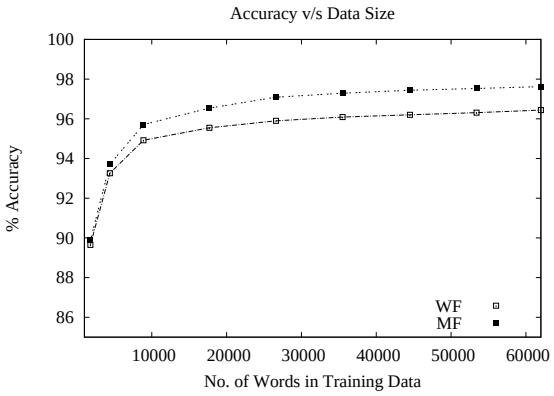

Figure 6: Average Accuracy of all Chunk Tags

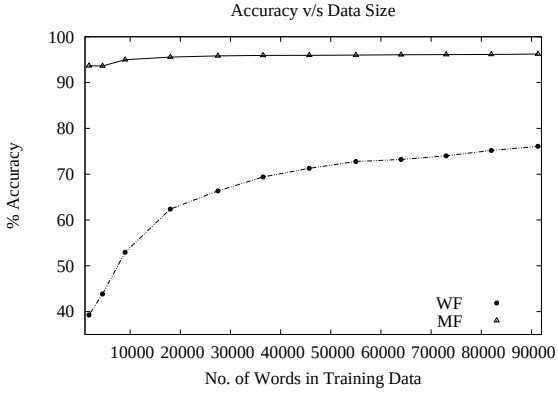

Figure 4: Average Accuracy of Verb POS Tags

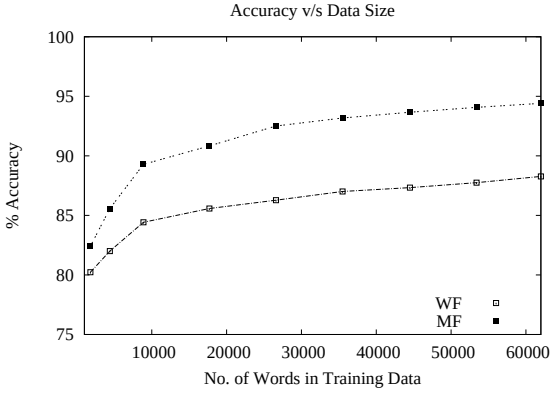

Figure 7: Average Accuracy of Verb Chunks

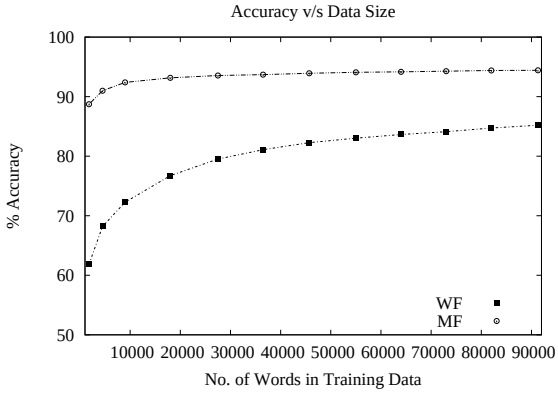

Figure 5: Average Accuracy of Non Verb POS Tags

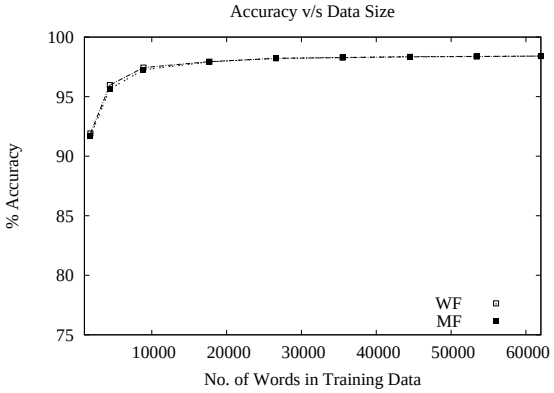

Figure 8: Average Accuracy of Non Verb Chunks

Chapter 6

Conclusion and Future Work

Shallow parsing, a Natural Language Processing (NLP) task provides the partial syntactic information about the sentence. It involves two basic NLP tasks, POS tagging and chunking. Shallow parsing is useful in the areas like information extraction, information retrieval, named entity recognition, machine translation *etc*. Methods ranging from rule based shallow parsers to statistical based shallow parsers are seen in the literature.

Shallow parsing of Indian languages benefit from the morphological features when coupled with statistical methods as they exhibit morphological richness. Marathi also belongs to this set of morphologically rich languages.

Different experiments on Marathi shallow parser showed that if the features of language are not harnessed properly, one may not be able to build high quality shallow parser even in the presence of a large corpora. Instead, a wise linguistic study can achieve the goal with a very decent sized training data. Adroit handling of suffixes in Marathi introduced the significant performance gain for verb POS and verb chunk tags and hence the gain in overall accuracy. Finally, we could build POS tagger and chunker for Marathi with accuracies of 95% and 98%.

There is still a scope of improvement, specifically in POS tagging. The confu-

sion matrix shows that NN-JJ confusions can be targeted as the next step. Detailed error analysis can help to decide if some type of rule based post processing is required or various morphological features can reduce the errors.

The experiments that are performed to test significance of linguistic analysis vs data size can be tested on some other Indian languages. The experiments done in chunking with only POS information as feature set can be further investigated. If an accuracy of 95% is obtained with only POS information, the learned model can be tested on other similar Indian languages. Since Indian languages share similar syntactic structure, only the POS sequence might be able to label the chunk sequence correctly. This might help in resource poor scenario where a language can make use of the models learned on resources of another language.

Bibliography

[**Aniket Dalal, 2007**] Aniket Dalal, Kumar Nagaraj, U. S. S. S. (2007). Hindi Part-of-Speech Tagging and Chunking : A Maximum Entropy Approach. In *Proceedings of the ICON*.

[**Bapat, 2010**] Bapat, M. (2010). Marathi morphological analysis. Master's thesis, Computer Science Department, IIT Bombay.

[**Berwick et al., 1991**] Berwick, I. R., Abney, S., (eds, C. T., and Abney, S. P. (1991). Parsing By Chunks.

[**Bharati et al., 1995**] Bharati, A., Chaitanya, V., and Sangal, R. (1995). *Natural Language Processing : A Paninian Perspective*. Prentice Hall India.

[**Black et al., 1992**] Black, E., Jelinek, F., Lafferty, J., Mercer, R., and Roukos, S. (1992). Decision tree models applied to the labeling of text with parts-of-speech. In *HLT '91: Proceedings of the workshop on Speech and Natural Language*, pages 117–121, Morristown, NJ, USA. Association for Computational Linguistics.

[**Brants, 2000**] Brants, T. (2000). TnT - A Statistical Part-of-Speech Tagger. In *6th Applied Natural Language Processing (ANLP '00), April 29 - May 4*, pages 224–231, Seattle, USA. Association for Computational Lingusitics.

[**Brill, 1995**] Brill, E. (1995). Transformation-Based Error-Driven Learning and Natural Language Processing: A Case Study in Part-of-Speech Tagging. *Computational Linguistics*, 21(4):543–565.

[**Church, 1988**] Church, K. W. (1988). A Stochastic Parts Program and Noun Phrase Parser for Unrestricted Text. In *ANLP*, pages 136–143. Proceedings of Second Conference on Applied Natural Language Processing.

[**Damale, 1970**] Damale, M. K. (1970). *Shastriya Marathi Vyaakarana*. Pune Deshmukh and Company.

[**Dandapat, 2007**] Dandapat, S. (2007). Part of Speech Tagging and Chunking with Maximum Entropy Model. In *Proceedings of IJCAI Workshop on Shallow Parsing for South Asian Language*.

[**Dandapat et al., 2007**] Dandapat, S., Sarkar, S., and Basu, A. (2007). Automatic Part-of-Speech Tagging for Bengali: An Approach for Morphologically Rich Languages in a Poor Resource Scenario. In *ACL*. The Association for Computer Linguistics.

[**Dixit et al., 2006**] Dixit, V., Dethe, S., and Joshi, R. K. (2006). Design and Implementation of a Morphology-based Spellchecker for Marathi, an Indian Language. In *Special issue on Human Language Technologies as a challenge for Computer Science and Linguistics. Part I. 15*, pages 309–316. Archives of Control Sciences.

[**Ekbal et al., 2007**] Ekbal, A., Mandal, S., and Bandyopadhyay, S. (2007). POS Tagging Using HMM and Rule Based Chunking. In *Proceedings of IJCAI Workshop on Shallow Parsing for South Asian Language*.

[**Garside and Smith, 1997**] Garside, R. and Smith, N. (1997). A Hybrid Grammatical Tagger: CLAWS. In Garside, R., Leech, G., and McEnery, T., editors, *Corpus Annotation*, pages 102–121. Longman, London.

[**Giorgos et al., 1999**] Giorgos, O., Dimitris, K., Thanasis, P., and Dimitris, C. (1999). Decision Trees and NLP: A case study in POS Tagging.

[**Hajic et al., 2001**] Hajic, J., Krbec, P., Kveton, P., Oliva, K., and Petkevic, V. (2001). Serial Combination of Rules and Statistics: A Case Study in Czech Tagging. In *ACL*, pages 260–267.

[**Koeling, 2002**] Koeling, R. (2002). Chunking with maximum entropy models. In *In: Proceedings of CoNLL-2000 and LLL-2000, pages 139-141, Lisbon, Portugal, 2000.*

[**Kudo and Matsumoto, 2001**] Kudo, T. and Matsumoto, Y. (2001). Chunking with Support Vector Machines. In *NAACL*.

[**Lafferty et al., 2001**] Lafferty, J., McCallum, A., and Pereira, F. (2001). Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. In *Proc. 18th International Conf. on Machine Learning*, pages 282–289. Morgan Kaufmann, San Francisco, CA.

[**Megyesi, 1999**] Megyesi, B. (1999). Improving Brill's POS Tagger For An Agglutinative Language.

[**Molina and Pla, 2002**] Molina, A. and Pla, F. (2002). Shallow Parsing using Specialized HMMs. *Journal of Machine Learning Research*, 2:595–613.

[**Navanath et al., 2009**] Navanath, S., Dhrubajyoti, D., Utpal, S., and Jugal, K. (2009). Part of Speech Tagger for Assamese Text. In *Proceedings of the ACL-IJCNLP 2009 Conference Short Papers*, pages 33–36, Suntec, Singapore. Association for Computational Linguistics.

[**Oflazer and Kuruöz, 1994**] Oflazer, K. and Kuruöz, I. (1994). Tagging and Morphological Disambiguation of Turkish Text. In *ANLP*, pages 144–149.

[**Pawar, 2008**] Pawar, S. S. (2008). Part of Speech Tagging of Indian Languages. Master's thesis, Computer Science Department, IIT Bombay.

[**PVS and G, 2007**] PVS, A. and G, K. (2007). Part-Of-Speech Tagging and Chunking Using Conditional Random Fields and Transformation Based Learning. In *Proceedings of IJCAI Workshop on Shallow Parsing for South Asian Language*.

[**Ramshaw and Marcus, 1995**] Ramshaw, L. and Marcus, M. (1995). Text Chunking Using Transformation-Based Learning. In Yarovsky, D. and Church, K., editors, *Proceedings of the Third Workshop on Very Large Corpora*, pages 82–94, Somerset, New Jersey. Association for Computational Linguistics, Association for Computational Linguistics.

[**Rao and Yarowsky, 2007**] Rao, D. and Yarowsky, D. (2007). Part of Speech tagging and Shallow Parsing for Indian Languages. In *Proceedings of IJCAI Workshop on Shallow Parsing for South Asian Language*.

[**Rao et al., 2007**] Rao, P. R. K., R, V. S. R., R, V., and L, S. (2007). A Text Chunker and Hybrid POS Tagger for Indian Languages. In *Proceedings of IJCAI Workshop on Shallow Parsing for South Asian Language*.

[**Ratnaparkhi, 1996**] Ratnaparkhi, A. (1996). A Maximum Entropy Model for Part-of-Speech Tagging. In Brill, E. and Church, K., editors, *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 133–142. Association for Computational Linguistics, Somerset, New Jersey.

[**Ray et al., 2003**] Ray, P. R., V., H., Sarkar, S., and Basu, A. (2003). Part of Speech Tagging and Local Word Grouping Techniques for Natural Language Parsing in Hindi. In *Proceedings of the ICON*.

[**Sastry et al., 2007**] Sastry, G. M. R., Chaudhari, S., and Reddy, P. N. (2007). A Hmm Based Part-Of-Speech Tagger and Statistical Chunker for 3 Indian Languages. In *Proceedings of IJCAI Workshop on Shallow Parsing for South Asian Language*.

[**Sha and Pereira, 2003**] Sha, F. and Pereira, F. C. N. (2003). Shallow Parsing with Conditional Random Fields. In *HLT-NAACL*.

[**Shrivastava and Bhattacharya, 2008**] Shrivastava, M. and Bhattacharya, P. (2008). Hindi POS Tagger Using Naive Stemming: Harnessing Morphological Information Without Extensive Linguistic Knowledge. In *Proceedings of the ICON*.

[**Singh et al., 2005**] Singh, A., Bendre, S., and Sangal, R. (2005). HMM Based Chunker for Hindi. In *Proceedings of International Joint Conference on NLP*.

[**Singh et al., 2006**] Singh, S., Gupta, K., Shrivastava, M., and Bhattacharyya, P. (2006). Morphological Richness Offsets Resource Demand - Experiences in Constructing a POS Tagger for Hindi. In *Proceedings of ACL-2006*.

[**Skut and Brants, 1998**] Skut, W. and Brants, T. (1998). Chunk Tagger - Statistical Recognition of Noun Phrases. *CoRR*, cmp-lg/9807007. informal publication.

[**SPSAL, 2007**] SPSAL (2007). Workshop On Shallow Parsing for South Asian Languages (SPSAL).

[**Tlili-Guiassa, 2006**] Tlili-Guiassa, Y. (2006). Hybrid Method for Tagging Arabic Text.

⟦ **Tufis and Mason, 1998** ⟧ Tufis, D. and Mason, O. (1998). Tagging Romanian Texts: a Case Study for QTAG, a Language Independent Probabilistic Tagger.

⟦ **Valambe, 2007** ⟧ Valambe, M. R. (2007). *Sugam Marathi Vyaakarana Lekhan*. Nitin Prakashan, Pune.

⟦ **Veenstra and van den Bosch, 2000** ⟧ Veenstra, J. and van den Bosch, A. (2000). Single-Classifier Memory-Based Phrase Chunking. In Cardie, C., Daelemans, W., Nédellec, C., and Sang, E. T. K., editors, *Proceedings of the Fourth Conference on Computational Natural Language Learning and of the Second Learning Language in Logic Workshop, Lisbon, 2000*, pages 157–159. Association for Computational Linguistics, Somerset, New Jersey.

⟦ **Zhang et al., 2002** ⟧ Zhang, T., Damerau, F., and Johnson, D. (2002). Text Chunking based on a Generalization of Winnow. *Journal of Machine Learning Research*, 2:615–637.

⟦ **Zhou et al., 2000** ⟧ Zhou, G., Su, J., and Tey, T. (2000). Hybrid Text Chunking. In Cardie, C., Daelemans, W., Nédellec, C., and Sang, E. T. K., editors, *Proceedings of the Fourth Conference on Computational Natural Language Learning and of the Second Learning Language in Logic Workshop, Lisbon, 2000*, pages 163–165. Association for Computational Linguistics, Somerset, New Jersey.

Appendix A

POS Tags

Category	**Tag Name**	**Example**
Noun	NN	नदी, माणूस
NLoc	NST	पुढे, मागे, नंतर, आधी
Proper Noun	NNP	राम, सीता
Pronoun	PRP	मी, तू
Demonstrative	DEM	हे* घर
Verb-finite	VM	करणे, करतो
Verb Aux	VAUX	आहे
Adjective	JJ	सुंदर, जाडा
Adverb	RB	सावकाश
Post position	PSP	पुढे, वर
Particles	RP	जणू, मात्र, काही
Conjuncts	CC	आणि, व
Question Words	WQ	काय, कोण
Quantifiers	QF	खूप* सुन्दर
Cardinal	QC	१, २
Ordinal	QO	पहिला, दुसरा
Interjection	INJ	अय्या, वाह
Negation	NEG	न
Quotative	UT	म्हणजे
Reduplicative	RDP	चालत चालत

Appendix B

Chunk Tags

Chunk Type	**Tag Name**	**Example**
Noun Chunk	NP	(हे_DEM **नवीन** _JJ **पुस्तक** _NN)_NP
Adjectival Chunk	JJP	दिवस_NN (**मस्त** _JJ)_JJP गेला_VM
Finite Verb Chunk	VGF	मी_PRP घरी_NN (**जेवले**_VM)_VGF
Non Finite Verb Chunk	VGNF	तो_PRP (**खेळताना**_VM)_VGNF पडला_VM
Infinitival Verb Chunk	VGINF	मला_PRP (**गायला**_VM)_VGINF आवडते_VM
Gerund Verb Chunk	VGNN	(**लिहायच्या**_VM)_VGNN त्रासातून_NN सुटका_NN
Adverb Chunk	RBP	तो_PRP (**हळूहळू**_RB)_RBP चालतो_VM
Conjunct Chunk	CCP	राम_NNP (**आणि** _CC)_CCP श्याम_NNP खेळतात_VM
Miscellaneous	BLK	नदी_NN (**जणू**_UT)_BLK आमची_PRP आईच_NN