TAGGING MALAYALAM TEXT WITH PARTS OF SPEECH - TNN Tagger and SVM Tagger Comparison

Rajeev R R¹, Jisha P Jayan¹, and Dr. Elizabeth Serly³

¹,³ Indian Institute of Information Technology and Management, Trivandrum, India
Email: {rajeev, sherly}@iiitmk.ac.in
¹ Sree Sankaracharya University of Sanskrit, Kalady, India
Email: jishajayan@gmail.com

Abstract— Parts of Speech Tagger (POS) also called, as grammatical tagging or word category disambiguation, is the task of assigning to each word of a text the proper POS tag in its context of appearance in sentences. The importance of the problem focuses from the fact that the POS is one of the first stages in the process performed by various natural language related process. There are different approaches to the problem of assigning a part of speech (POS) tag to each word of a natural language sentence. Here we present a tag set for the Indian language Malayalam, a relatively free word order morphologically productive and agglutinative language. The paper is about performing Part of speech tagging for the Indian language Malayalam using SVM Tool, which was implemented using support vector machines and TnT tagger, which was built using Hidden Markov Model. The SVM Tool is giving accuracy of 87.5% and TnT tagger is giving an accuracy of about 80%.

Keywords— Supervised and unsupervised, statistical method, disambiguation and prepositional phrase, TNT Tagger, SVM Tagger, pos tagging, Malayalam Computing.

I. INTRODUCTION

Most NLP applications demand at some step some part–of–speech (POS) information, usually at initial stages prior to elaborate further complex data analyses. Parts of Speech Tagging, a grammatical tagging, is a process of marking the words in a text as corresponding to a particular part of speech, based on its definition and context. This is the first step towards understanding any languages. Annotated corpora find its major application in text speech and NLP like Speech Recognition, Speech Synthesis, Information retrieval etc. It proves to be a basic building block for constructing statistical models for automatic processing of natural languages. Many such corpora are available for language across the world and have proved to be a useful step towards natural language processing. A lot of work has been done relating to this in NLP field.

The definition of parts of speech has been based on morphological and syntactic function; words that function similarly with respect to the affixes they take (their morphological properties) or with respect to what can occur nearby (their 'distributional properties') are grouped into classes. Assigning a POS tag to each word of an un–annotated text by hand is very time consuming. And that is why POS Tagging has become one of the well–studied problems in the field of NLP. The taggers vary in accuracy and also in their implementation. A lot of techniques have also been explored to make tagging more and more accurate. These techniques vary from being purely rule based in their approach to being completely stochastic.

Some of these taggers achieve good accuracy for certain languages. But unfortunately, not much work has been done with regard to Indian languages especially Malayalam.

II. MALAYALAM LANGUAGE

Malayalam belongs to the Dravidian family of languages and is one of the four major languages of this family with a rich literary tradition, inflectionally mainly adding of suffixes with the root or the stem word forms rich in morphology. The origin of Malayalam as a distinct language may be traced to the last quarter of 9th Century A.D. Malayalam is a language registering a heavy amount of agglutination. Throughout its gradual evolution Malayalam has been influenced by the various circumstances prevailed on different periods. The important influence among these is the influence of Sanskrit and Prakrit brought into Kerala by Brahmins. In modern Malayalam also a good part of vocabulary is of Sanskrit origin. Influence of Sanskrit is evident in the alphabet, phonology and vocabulary and to a lesser extent in morphology also. The language must be certainly being older, but linguistic research is yet to be discovering unmistakable evidence to prove its antiquity. There are different spoken forms in Malayalam even though the literary dialect through out Kerala is almost uniform.

III. PARTS OF SPEECH TAGGING

The importance of the problem focuses from the fact that the POS is one of the first stages in the process performed by various natural language related process. POS tagging is the commonest form of the corpus annotation. POS is mainly for information retrieval, text to speech, information extraction and linguistic research for corpora and for higher level NLP tasks like parsing, semantics, machine translation and many more. Pos tag gives some information about the sense of the word in the context of use. It is a non–trivial task:

- Some words (at least in a sense of this word) that occur in the lexicon or dictionary have more than one possible Part of Speech.
- Some words are unknown.
- Tags are not well-defined.

Part-of-speech analysis usually consists of (i) introduction of ambiguity (lexical analysis) and (ii) of-speech ambiguities. The main problem with disambiguation (elimination of illegitimate alternatives). While introducing ambiguity is regarded as relatively
straightforward, disambiguation is known to be a difficult and controversial problem. There are two main methodologies: the linguistic and the data-driven. The linguistic approach may seem an obvious choice. A part-of-speech tagger’s task is often illustrated with a noun-verb ambiguous word directly preceded by an unambiguous determiner (e.g. table in the table). This ambiguity can reliably be resolved with a simple and obvious grammar rule that disallows verbs after determiners.

There are different models for POS tagging - the supervised and unsupervised POS tagging models. In supervised POS tagging models require a pre-annotated corpus which is used for training to learn information about the tag set. The performance of the model mainly increases with increase in the size of the corpus. The unsupervised model does not require a pre-annotated corpus. Different approaches have been proposed for POS tagging. Many Machine-learning methods have been applied for the parts of speech tagging. Some among them are Hidden Markov Model [1], the Transformation Based Error Driven [2], Statistical Based [3], Maximum Entropy Model [4], Decision Tree [5], Rule Based [6], Memory Based [7]. The last approach to this is Neural Networks [8].

Tagging can be seen as a prototypical problem in lexical ambiguity; advances in part-of-speech tagging could readily translate to progress in other areas of lexical, and perhaps structural, ambiguity, such as word sense disambiguation and prepositional phrase attachment disambiguation. Also, it is possible to cast a number of other useful problems as part-of-speech tagging problems, such as letter-to-sound translation and building pronunciation networks for speech recognition.

However, in part-of-speech tagging, we frequently encounter words that do not exist in training data. Such unknown words are usually handled by an exceptional processing, because the statistical information or rules for those words are unknown. Many machine learning methods have been applied for part-of-speech tagging, such as the hidden Markov model (HMM), the transformation-based error driven system, the decision tree and the maximum entropy model. Though these methods have good performance, the accuracy for unknown words is much lower than that for known words, and this is a non-negligible problem where training data is limited.

A. Principle

Part-of-speech tagging is harder than just having a list of words and their parts of speech, because some words can represent more than one part of speech at different times. This is not rare—in natural languages (as opposed to many artificial languages), a large percentage of word-forms are ambiguous. For example, even "dogs", which is usually thought of as a just a plural noun, can also be a verb:

The sailor dogs the hatch.

"Dogged", on the other hand, can be either an adjective or a past-tense verb. Just which parts of speech a word can represent varies greatly.

Commonly there are 8 parts of speech in English: noun, verb, adjective, preposition, pronoun, adverb, conjunction, and interjection. However, there are clearly many more categories and sub-categories. For nouns, plural, possessive, and singular forms can be distinguished. In many languages words are also marked for their "case" (role as subject, object, etc.), grammatical gender, and so on; while verbs are marked for tense, aspect, and other things.

In part-of-speech tagging by computer, it is typical to distinguish from 50 to 150 separate parts of speech for English, for example, NN for singular common nouns, NNS for plural common nouns, NP for singular proper nouns. Work on stochastic methods for tagging Koine Greek has used over 1,000 parts of speech, and found that about as many words were ambiguous there as in English. A morphosyntax descriptor in the case of morphologically rich languages can be expressed like Ncnsan, which means Category=Noun, Type = common, Gender = masculine, Number = singular, Case = accusative, Animate = no.

A lot of work has been done in part of speech tagging of Western languages. These taggers vary in accuracy and also in their implementation. A lot of techniques have also been explored to make tagging more and more accurate. These techniques vary from being purely rule based in their approach to being completely stochastic. Some of these taggers achieve good accuracy for certain languages. But unfortunately, not much work has been done with regard to Indian languages especially Malayalam. The existing taggers cannot be used for Indian languages. The reasons for this are:

- The rule based taggers would not work because the structure of Indian languages differs vastly from the Western languages
- The stochastic taggers can be used in a very crude form. But it has been observed that the taggers give best results when there is some knowledge about the structure of the language.

IV. STATISTICAL PARTS OF SPEECH TAGGING

Statistical techniques model probability distributions over tags by using transition probabilities between tags or words and tags, and the lexical probabilities of tags for words. The process tries to find the sequence of tags that has the highest probability, given a sequence f words. Most prevalent techniques are Hidden Markov Models and Maximum entropy models.

The statistical (stochastic) approaches select the most likely interpretation based on the estimation of statistics from unambiguously tagged text. Either word frequencies or n-gram probabilities can be used as the criterion to be maximized. The most common algorithm for implementing an n-gram approach is the Viterbi Algorithm, which avoids the polynomial expansion of a breadth first search by trimming the search tree. The next level of complexity that can be introduced into a stochastic tagger combines the previous two approaches, using both tag sequence probabilities and word frequency measurements. These approaches use a Markov model [9], a hidden Markov model, or a perceptron model [10].
A Hidden Markov Model (HMM) is a statistical model in which the system modeled is thought to be a Markov process with the unknown parameters. In this model, the assumptions on which it works are the probability of the word in a sequence may depend on its immediate word preceding it and both the observed and hidden words must be in a sequence. This model can represent the observable situations and in POS tagging, the words can be seen themselves, but the tags cannot. So HMM are used as it allows observed words in input sentence and hidden tags to be build into a model, each of the hidden tag state produces a word in a sentence.

![Fig. 1 State transitions in a hidden Markov model](https://example.com/f1.png)

- **x** — hidden states
- **y** — observable outputs
- **a** — transition probabilities
- **b** — output probabilities

Markov Models can only represent observable situations, and in part of speech tagging, although the words themselves can be seen, the tags themselves cannot. For this reason, a HMM is used, as it allows observed events (the words in the input sentence) and hidden events (the tags) to be built in to the model. Each hidden tag state produces a word in the sentence. HMMs are both probabilistic and non-deterministic. Knowing an observation (word) sequence does not imply that a unique state (tag) sequence can be inferred.

The figure below shows the general architecture of an HMM. Each oval shape represents a random variable that can adopt a number of values. The random variable \( x(t) \) is the value of the hidden variable at time \( t \). The random variable \( y(t) \) is the value of the observed variable at time \( t \).

The arrows in the diagram denote conditional dependencies.

![Fig. 2 HMM Architecture](https://example.com/f2.png)

The Hidden Markov Model is a finite set of states, each of which is associated with a (generally multidimensional) probability distribution. Transitions among the states are governed by a set of probabilities called transition probabilities. In a particular state an outcome or observation can be generated, according to the associated probability distribution. It is only the outcome, not the state visible to an external observer and therefore states are “hidden” to the outside; hence the name Hidden Markov Model. The most common algorithm for implementing an n-gram approach is known as the Viterbi Algorithm; an efficient dynamic search algorithm that avoids the polynomial expansion of a breadth first search by trimming. The search tree at each level using the best N Maximum Likelihood Estimates (where N represents the number of tags of the following word).

With HMM, Viterbi algorithm, a search algorithm is used for various lexical calculations. The Viterbi algorithm operates on a state machine assumption. It is a dynamic programming algorithm that is mainly used to find the most likely of the hidden states, results in a sequence of the observed words. This is one of the most common algorithms that implement the n-grams approach. This algorithm mainly work based on the number of assumptions it makes.

The algorithm assumes that both the observed and the hidden word must be in a sequence, which corresponds to the time. It also assumes that the sequence must be aligned. One of the basic views behind this algorithm is to compute most likely tag sequence occurring with the unambiguous tag until the correct tag is obtained. At each level most appropriate sequence and the probability including these are calculated. The Viterbi algorithm is a computationally simple method of finding the maximum a posteriori estimate of a sequence of observations.

V. TAGSET DESIGN

Words are assigned parts of speech in order to capture generalizations about grammatically well-formed sentences, such as “The noun is adjective”. Determining the parts of speech of the words in a sentence can help us to identify the syntactic structure of the sentence, and in some cases determine the pronunciation or meaning of individual words. Good tagset design is particularly important for highly inflected languages. If all of the syntactic variations that are realized in the inflectional system were represented in the tagset, there would be a huge number of tags, and it would be practically impossible to implement or train a tagger. A POS tagset design should take into consideration all possible morphosyntactic categories that can occur in a particular language or group of languages.

A. Tagset for POS

Malayalam belongs to the Dravidian family of languages, inflectionally mainly adding of suffixes with the root or the stem word forms rich in the morphology. The Tagset given below was developed by IIIT Hyderabad.
Since words are formed by the suffix addition with root; the word can take the POS tag based on the root / stem. Hence it can be stated that the suffixes play major role in deciding the POS of the word.

VI. TAGGERS

A. Trigrams N Tag (Tnt)

TnT tagger is proposed by Thorsten Brants and in literature its efficiency is reported as one of the best and fastest on different languages such as German, English, Slovene and Spanish. TnT is a statistical approach, based on a Hidden Markov Model that uses the Viterbi algorithm with beam search for fast processing. A Markov based tagger aims to find a tag sequence which maximizes

\[ P(\text{word}|\text{tag}) \propto P(\text{tag}|\text{tag}_1...\text{tag}_{n-1}), \]

where the first factor is the emit (or lexical) probability, the likelihood of a word given certain tag, and the second factor is the state transition (or contextual) probability, the likelihood of a tag given a sequence of preceding tags. TnT is trained with different smoothing methods and suffix analysis. The parameter generation component trains on tagged corpora. The system uses several techniques for smoothing and handling of unknown words. TnT can be used for any language, adapting the tagger to a new language; new domain or new tagset is very easy. The tagger is implemented using Viterbi algorithm for second order Markov models. Linear interpolation is the main paradigm used for smoothing and the weights are determined by deleted interpolation. To handle the unknown words, suffix trie and successive abstraction are used. There are two types of file formats used in TnT, untagged input for tagger and the tagged input for tagger.

Trigrams N Tags (TNT) is a stochastic HMM tagger based on trigram analysis, which uses a suffix analysis technique based on properties of words like, suffixes in the training corpora, to estimate lexical probabilities for unknown words that have the same suffixes. Its greatest advantage is its speed, important both for fast tuning cycle and when dealing with large corpora. The strong side of TnT is its suffix guessing algorithm that is triggered by unseen words. From the training set TnT builds a trie from the endings of words appearing less than n times in the corpus, memorizes the tag distribution for each matrix. A clear advantage of this approach is the probabilistic weighting of each label, however, under default settings the algorithm proposes a lot more possible tags than a morphological analyzer would.

B. SVM

Svmtool is a simple, flexible, and effective generator of sequential taggers based on Support Vector Machines and how it is being applied to the problem of part-of-speech tagging. This SVM-based tagger is robust and flexible for feature modeling (including lexicalization), trains efficiently with almost no parameters to tune, and is able to tag thousands of words per second, which makes it really practical for real NLP applications. Regarding accuracy, the SVM-based tagger significantly outperforms the TnT tagger exactly under the same conditions.

The svmtool is intended to comply with all the requirements of modern NLP technology, by combining simplicity, flexibility, robustness, portability and efficiency with state-of-the-art accuracy. This is achieved by working in the Support Vector Machines (SVM) learning framework [11], and by offering NLP researchers a highly customizable POS-tagger. SVM is a machine learning algorithm for binary classification, which has been successfully applied to a number of practical problems, including NLP. This learning bias has proved to have good properties in terms of generalization bounds for the induced classifiers.

Previous to the tagging, SVM models (weight vectors and biases) are learned from a training corpus using the SVMTrain component. Different models are learned for the different strategies. Then, at tagging time, using the SVMTagger component, one may choose the tagging strategy that is most suitable for the purpose of the tagging. Finally, given a correctly annotated corpus, and the

<table>
<thead>
<tr>
<th>Main Tags</th>
<th>Representation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Noun</td>
<td>NN</td>
</tr>
<tr>
<td>Noun Location</td>
<td>NST</td>
</tr>
<tr>
<td>Proper Noun</td>
<td>NNP</td>
</tr>
<tr>
<td>Pronoun</td>
<td>PRP</td>
</tr>
<tr>
<td>Compound Words</td>
<td>XC</td>
</tr>
<tr>
<td>Demonstration</td>
<td>DEM</td>
</tr>
<tr>
<td>Post Position</td>
<td>PSP</td>
</tr>
<tr>
<td>Conjunctions</td>
<td>CC</td>
</tr>
<tr>
<td>Verb</td>
<td>VM</td>
</tr>
<tr>
<td>Adverb</td>
<td>RB</td>
</tr>
<tr>
<td>Particles</td>
<td>RP</td>
</tr>
<tr>
<td>Adjectives</td>
<td>JJ</td>
</tr>
<tr>
<td>Auxiliary Verb</td>
<td>VAUX</td>
</tr>
<tr>
<td>Negation</td>
<td>NEG</td>
</tr>
<tr>
<td>Quantifiers</td>
<td>QF</td>
</tr>
<tr>
<td>Cardinal</td>
<td>QC</td>
</tr>
<tr>
<td>Ordinal</td>
<td>QO</td>
</tr>
<tr>
<td>Question Words</td>
<td>WQ</td>
</tr>
<tr>
<td>Intensifiers</td>
<td>INTF</td>
</tr>
<tr>
<td>Interjection</td>
<td>INJ</td>
</tr>
<tr>
<td>Reduplication</td>
<td>RDP</td>
</tr>
<tr>
<td>Unknown Words</td>
<td>UNK</td>
</tr>
<tr>
<td>Symbol</td>
<td>SYM</td>
</tr>
</tbody>
</table>
corresponding SVMTool predicted annotation, the SVMTeval component displays tagging results. Although SVMs have good properties and performance, their computational cost is large. It is difficult to train for a large amount of training data, and testing time increases in more complex models. When compared to SVM, TnT was the fastest for both training and tagging. POS tagging involves many difficult problems, such as insufficient amounts of training data, inherent POS ambiguities, and most seriously, many types of unknown words which are ubiquitous in any application and cause major tagging failures in many cases.

VII. TESTING AND RESULTS

The words and tags are taken from the training file to build a suffix tree data structure. In this tree structure the word and tag frequency are stored and the letter tree is build taking the word and its frequency as the argument. While training, the transition and emission property matrix are calculated and the models of the language are building. The lexicon file created during the generation of the parameter contains the frequencies of the words and its tags, which occurred in the training corpus. A hash of the tag sequence and its frequency is build. This is used for determining the lexical probability. The n-gram file that is also generated during the parameter generation contains the contextual frequencies for the unigrams, bigrams, and trigrams. While testing Viterbi algorithm is applied to find best tag sequence for a sentence. If tag sequence is not present smoothing techniques are applied according to runtime arguments of the postagger. Whenever the word is not found in the matrices suffix smoothing technique are applied.

A Malayalam corpus was tagged using the tagset for Malayalam. About 2 lakhs words were tagged and this tagged corpus is used. Then the corpus was learned using both TNT and SVM tool. When learned, the dictionary file was created for the corpus. After learning was performed, input text file was given to both the tools and tagging was performed. When tagging was done and the output file was verified, it was seen that SVM tool tags the text more accurately than TNT tool.

Ex: Input text.
innu mazha peyyaan saadhyata uNTenn avan paRanjnju.
[Which means “He told that there is is a possibility to rain today.”]

When we compare both the outputs, we can see that the word innu which means today is tagged as a verb in TnT but as a noun in SVM. So we can say that SVM gives us a better result than TnT.

CONCLUSIONS

Part-of-speech tagging now is a relatively mature field. Many techniques have been explored. Taggers originally intended as a pre-processing step for chunking and parsing but today it issued for named entity recognition, in message extraction systems, for text-based information for speech recognition, for generating intonation in speech production systems and as a component in many other applications.

<table>
<thead>
<tr>
<th>Tagged output from SVM</th>
<th>Tagged output from TnT</th>
</tr>
</thead>
<tbody>
<tr>
<td>innu</td>
<td>innu</td>
</tr>
<tr>
<td>mazha</td>
<td>mazha</td>
</tr>
<tr>
<td>peyyaan</td>
<td>peyyaan</td>
</tr>
<tr>
<td>saadhyata</td>
<td>saadhyata</td>
</tr>
<tr>
<td>uNTenn</td>
<td>uNTenn</td>
</tr>
<tr>
<td>avan</td>
<td>avan</td>
</tr>
<tr>
<td>paRanjnju</td>
<td>paRanjnju</td>
</tr>
<tr>
<td>.</td>
<td>.</td>
</tr>
</tbody>
</table>

It has aided in many linguistic research on language usage. The Parts of Speech Tagging for Malayalam using the statistical approach has been discussed. The system works fine with the Unicode data. The POS were able to assign tags to all the words in the test case. These also focus on the point that a statistical approach can also work well with highly morphologically and inflectionally rich languages like Malayalam

ACKNOWLEDGMENT

We would like to thank Ministry of Communication and Information Technology, Department of Information Technology, Government of India, for promoting a project on Indian Language to Indian Language Machine Translation System, where this part of the work was carried out. Large annotated corpora are the need for training and tagging the parts of speech taggers. Therefore we would also like to thank all the people from the ILMT Lab who were involved in the collecting and tagging of the Malayalam Corpus.
REFERENCES


