Implementing Probabilistic Models on Hindi

Aatish Varma
Department of Computer Science
New York University

December 18, 2018

1 Introduction

An important area of study in Natural Language Processing is Probabilistic Parts of Speech (POS) tagging. Two interrelated classes of methods used to POS tag texts are Hidden Markov Models and Maximum Entropy Models, both of which predict future states based on probabilities generated from previous states. In class, we learned about one specific Hidden Markov Model, specifically the Bigram Viterbi Algorithm, which computes a given word's POS by using emission and transition probabilities from the current and previous word, based on data from a training corpus. We then learned about the Maximum Entropy Markov Model (MEMM) [4] which uses a Maximum Entropy Distribution (least biased) to classify words, based on a set of features.

After applying these algorithms on the English language, I was curious about the robustness of such algorithms, and more specifically, the consistency of the results produced by the algorithms across different types of data sets. I wanted to see to what extent the accuracies of these algorithms were dependent on the size of the data as well as the nature of the language that the data was written in.

Two attributes that describe the mechanisms of a language are Morphology and Syntax. After trying the algorithm on English, a syntax heavy language, I wanted to conduct experiments on a more morphologically rich language. I settled on Hindi, my native language, which is morphologically far richer than English. It is important to note that when I refer to a language being morphologically rich, I ultimately mean that much of the sentence structure can be interpreted through the individual words themselves, since they have been modified to express different grammar. Such languages are also known as Inflectional languages.

In the first set of experiments, I use the Viterbi Algorithm to tag Hindi's Categorical and Type tags, to understand the similarities English and Hindi have in syntax. Then, to tag Hindi's Morphological Attributes, I take a unique approach using MEMMs.

2 Data

There are some significant differences between English and Hindi, which affect the methodologies for POS tagging. Hindi is a morphologically rich language, meaning that much of the grammar is encoded in individual words and is less present in phrase structure. As a result, POS tags are far more descriptive. This is the opposite case for English, in which most words can be assigned to 1 out of 36 Parts of Speech [As listed in the Penn Treebank].

This task was only possible because of the availability of a Hindi corpora, on which I could train, develop, and test my algorithm on. Through a bit of research, Professor Meyers and I were able to find an annotated POS set created by Microsoft Research India. The data contained 18 annotated files, each with approximately 290-300 sentences.

The data provided by Microsoft Research India (MRI) is annotated using the IL-Post Framework. This framework represents Hindi's morphologically rich nature by dividing parts of speech into three layers: Category, Type, and Attribute. MRI describes the Category layer as the "primary grammatical class" to which words belong to (examples are Noun, Verb, Adjective). Types are more exact specifications of these Categories (Common Noun, Proper Noun, Verbal Noun, etc). Attributes represent the "morpho-syntactic" features of Hindi and it's denominations, and differ across Category-Type combinations. For example, Common Nouns (NC) can have Gender, Number, Case, and an optional Case-Marker, while a Verbal Noun only has Case and an optional Case-Marker. The data set grouped the Category and Type layers into one layer, which allowed me to tag Category and Type together.

3 Preparing Data

Creating training, development, and testing corpus' proved to be a pedantic task, especially since the data was laced with tiny errors, which made parsing difficult. Given that the MRI data set had 16 text files, I allocated 12 text files for training, 1 file for development, and 3 for test. I trained my algorithms on the training file, and tested the algorithms multiple times on my development file, until finally testing on the three test files.

4 Tagging Category and Type

While doing research, I saw that similar experiments done on other languages with fine-grained POS tags used some sort of decision tree [2]. Because decision trees and similar structures use Dynamic Bayesian Networks [9], I inferred that any algorithm that used similar mechanisms would generate apt results. I had previously implemented the Viterbi Algorithm, a Hidden Markov Model (HMM) which by definition also used a Dynamic Bayesian Networks. Because the Category and Type in the Hindi data set were in the same format as the data set I used for tagging English words, I decided to use the same algorithm.

Implementing the algorithm can generally be broken down into these very simple steps:

- Generating Transition and Emission Probabilities by iterating through Training File.
- Scanning individual sentences, computing emission probability and transition probability products, finding max probabilities.
- Handing OOV cases.

See Results in 'Evaluation and Results'

5 Tagging Attributes

Attributes are heavily dependent on the Category, Type, and other morphological features. Because the annotated data set had Category, Type, and Attribute in one string (separated by a period), I first wanted to see how the Viterbi Algorithm would fare in tagging the entire POS tags from the test corpus. However, as I expected, it produced poor results (7% accuracy). This was due to two specific reasons:

1. The number of POS possibilities with the IL-Post framework are huge(typically 1 category/type, and 4-5 attributes per POS = 27 possible Category/Type possibilities * 4 tags (at average) * 3 choices per attribute (at average) = 324 distinct possibilities), so transition and emission probabilities generated were not significant enough to be able to predict patterns in POS.

2. Attributes are given based on morphology, so determining a word's attributes based on patterns of attributes from the training file (in cases where that specific combination wasn't present in the training file) would not produce accurate results.

I looked at similar studies to see how fine-grained tags (like above) have been accurately tagged. In [1], words were stemmed and were then passed to a Morphological Analyzer (MA) which contained suffix replacement rules (SRRs), lexicon rules, group identification rules and morpheme analysis rules like below.

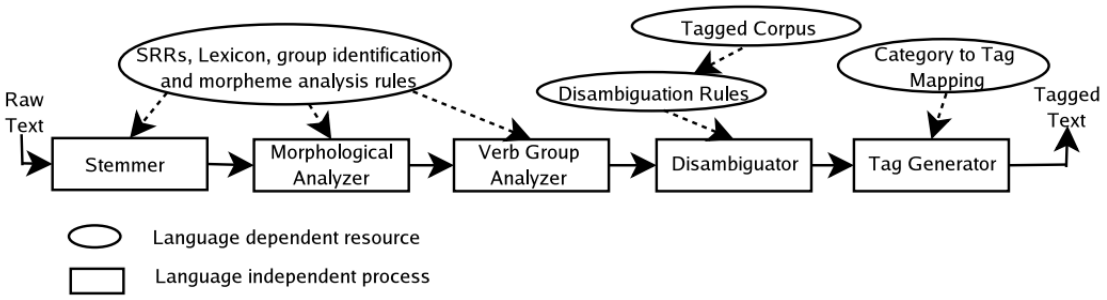

Figure 1: Taken from [1]

Though it is plausible, it is extremely difficult to create an attribute tagging rule set which handles every morphological rule for Hindi because morphology is dependent on many aspects, such as who is saying the sentence, the meaning of the sentence, and word order. Again, it can still be done but only if the writer of the rule set has a very high linguistic understanding of the language. This is, for a large part, why most authors of studies on POS tagging for Hindi were native Indians or Hindi Linguists.

6 Maximum Entropy Markov Model For Attribute Tagging

After realizing that implementing a Morphological Analyzer was not plausible with my current linguistic understanding level of Hindi, I wanted to analyze different methods of tagging morphological attributes.

In previous studies, Maximum Entropy Markov Models, along with a Morphological Analyzer [5], were used in tagging both morphologically-based tags and syntactically-based tags in morphologically rich languages. What makes MEMM's stand out are their accuracy in identifying OOV words. Instead of relying on the word alone, the model chains user-added features, which often are far better predictors of an unknown word than the transition probabilities of the previous word.

Because Morphological Attributes also rely on features of a word, like the letters (characters) of the word and the Categorial/Type POS of the current, previous, and next words, I wanted to see how MEMM's would fare when predicting individual morphological attributes.

From previous studies read, I could not find any experiments which solely used a MEMM to identify morphological attributes for Hindi. Most used morphological analyzers and dictionaries [5][6], while others used morphological features like suffixes, digits, and special characters [7] to tag Categorial and Type tags. I wanted to try the opposite (using Categorial and Type tags along with other features to detect morph attributes), which made experimentation interesting. Though Hindi is a morphologically rich language,

doing such experiments can truly reveal if some attributes are more syntactically rich than others, and even more specifically, on what specific features these features are dependent.

7 Experiments on Identifying Morphological Attributes

I focused my experimentation on tagging a couple of specific morphological attributes that were more present in different categorial POS tags than others.

The morphological attributes to be tagged by the MEMM were:

- **Gender**: Male, Female, 0 (Not Identifiable), None (Not Applicable)
- **Number:** Singular, Plural, 0 (Not Identifiable), None (Not Applicable)
- **Case:** Direct, Oblique, None (Not Applicable)

The features that the MEMM used in tagging were:

- **Words**: Previous Word, Previous Previous Word,
- **Category and Type**: Previous Category and Type, Previous Previous Category and Type, Next Category and Type, Next Category and Type

8 Evaluation and Results

I tested the Viterbi Algorithm on the three test files and produced the results below. I have also included my algorithms results on an English file with roughly the same number of words in the training file (59K words).

Category and Type Tagging	
File	Accuracy
Test File 1	75.00 %
Test File 2	77.12%
Test File 3	79.78 %
English File	81.65%

My experiments in tagging Gender, Number, and Case using an MEMM produced the results below.

Attribute Tagging			
File Number	Gender	Number	Case
File 3	85.46%	89.04%	93.97%
File 4	87.58%	89.48%	94.30%
File 5	88.79%	88.75%	91.49%

Because I could not find any other experiment which separately tagged Category/Type tags and Attribute tags, it is difficult to separately compare the two sets of experiments above to other models. However, all models that I read about used somewhat similar methods to mine and produced state of the art results. Their accuracies are shown below.

Results From Similar Studies			
Algorithm	Author	Training File Size (words)	Accuracy
MEMM with word features	Dalal [7]	15.5K	94.81%
CN^2	Singh [1]	15.5K	93.45%
Rule Based	Garg [8]	26.1K	87.55%
Supervised HMM with MA	Dandapat [5]	40K	86.64%
MEMM	Dandapat [5]	40K	84.65%
Supervised HMM	Dandapat [5]	40K	77.29%

Though my results seem very high for tagging morphological tags without a Morphological Analyzer, it is important to note that most published experiments which used a Morphological Analyzer were identifying the entire POS, which is formed of chains of multiple different morphological attributes and category/type. My simple model tagged a single chosen attribute (gender, case, number) in an experiment and used the correct category and type (which was taken from the key) as a feature input. As a result, there there was far less variables my model had to solve for.

9 Error Analysis

In these experiments, any error was largely based on a lack of data, the probabilistic nature of the algorithm, or the language. For both Hindi and English experiments, the training file only had around 59,000 tokens, which is generally considered too low for statistical-based models like HMM's to produce extremely high results. This resulted in a higher occurrence of out of vocabulary (OOV) words (The algorithm detected 14% of words to be OOV in Test 3). The Viterbi Algorithm I implemented handled these OOV cases by choosing the POS that the directly preceding POS pointed to most of the time. This method is simply a "best-case guess" and frequently chooses the wrong POS, which is why implementing a more functional OOV strategy is also a future area of work. MEMM's handle OOV cases far better by assuming "unknown values to be learnt are connected in a Markov chain rather than being conditionally independent of each other" [10], which is a reason why accuracies for OOV cases were higher when using this model.

Similarly, with a lack of training data, it is difficult to generate probabilities for most, if not all, possible POS to POS combinations. By this I mean that a given POS can point to many other POS', so for the Viterbi algorithm to choose the right one for a given scenario, it should know most possible POS to POS combinations that exist in a language. However, with little data, the Viterbi algorithm only knows a subset of those possible POS' and makes a misinformed decision.

For the MEMM, better results could have potentially been achieved with a more extensive list of features. In my experiments, I only used at most 6 features, most of which were nearby word features (previous and next words). Some additional features I intend on using in future experimentation, which have also worked well in the past, are word stem and word letter.

Also, with all languages, there are edge cases that are very hard to account for. This is especially true for Hindi because of language inflection.

Lastly, because there were many errors in the files, it is possible that all errors were not caught, which produced flawed results. For example, when testing my algorithm on the development corpus, I noticed that there was a POS denoted "VAUXmas" in my training corpus. Upon further inspection, I realized that the POS tag in the files was missing a "." between the CT and Attribute, so when it came to parsing for just category, the "VAUX" category and "mas" attribute (denoting "masculine") had been joined. Although I was able to remove or rectify most small errors, there still may be some in the files.

10 Future Work

Though the most immediate area for future work is creating a detailed Morphological Analyzer which can tag Attribute labels, there are other areas of future work that can better the results found in this study, and lead to new discoveries in others.

In the set of experiments done, I hampered with the issue of not having a considerable amount of data to train my model with (60,000 tokens). Other studies I have read which focused on statistical POS tagging for Hindi echoed the same issue ("English has annotated corpora in abundance, enabling usage of powerful data driven machine learning methods. But very few languages in the world have [the] resource advantage that English enjoys" [1]). Because the amount and variability of the data drives the accuracy of POS tagging, more annotated data can improve the way the models perform.

Furthermore, while these probabilistic models do generate good results, neural architectures (using neural networks and deep learning) in NLP have been gaining popularity, and have generated "State of the Art" results [3], even with noisy data. Labeau, Loser, and Allauzen in "Non-lexical neural architecture for fine-grained POS Tagging" were able to generate morphologically rich POS tag from character streams using two layer Convolutional Neural Networks. However, models like these also require more data (this particular experiment used the TIGER corpus that contains 900,000 tokens), which reaffirms the first problem.

11 Conclusion

Though Hindi is a morphologically rich language, part of its POS can be determined using probabilistic method, which is traditionally used for POS tagging more syntax-heavy languages. I was able to use the same Viterbi algorithm on two languages, which have completely different structures (not only morphological, but the basic grammar is different: Subject-Object-Verb vs. Subject-Verb-Object) and the results were very similar (only 2% difference). Furthermore, the MEMM used was very simple and didn't take many features, but was still able to attain 85%+ accuracy. However, the main question still needs to be explored: to what degree of accuracy can a hard-coded rule set/morphological analyzer tag morphological attributes?

This is an area I plan on exploring in the near future, and in doing so, I hope to gain an understanding of not only how morphologically rich languages really function, but also, how computers can be trained to perceive POS' the way humans do.

12 References

[1] 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 the COLING/ACL 2006 Main Conference Poster Sessions, pages 779-786.

[2] Helmut Schmid and Florian Laws. 2008. Estimation of Conditional Probabilities with Decision Trees and an Application to Fine-Grained POS Tagging. In COLING 2008, Manchester, Great Britain.

[3] Matthieu Labeau, Kevin Löser, Alexandre Allauzen, and Rue John von Neumann. 2015. Non-lexical neural architecture for fine-grained pos tagging. In Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP), pages 232–237.

[4] Adwait Ratnaparkhi. 1996. A maximum entropy model for part-of-speech tagging. In Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP).

[5] 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 Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions, pages 221-224. Association for Computational Linguistics.

[6] Georgiev, G., Zhikov, V., Simov, K., Osenova, P., and Nakov, P. (2012). Feature-Rich Partof-speech Tagging for Morphologically Complex Languages: Application to Bulgarian. In Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics, EACL, Avignon, France.

[7] Aniket Dalal, Kumar Nagaraj, Uma Sawant and Sandeep Shelke. (2006). Hindi Part-ofSpeech Tagging and Chunking: A Maximum Entropy Approach, In Proceeding of the NLPAI Machine Learning Competition, 2006.

[8] Garg, N., Goyal, V., & Preet, S. (2012). Rule Based Hindi Part of Speech Tagger. COLING.

[9] Kevin A. Murphy. (2002). A Tutorial on Dynamic Bayesian Networks. MIT AI Lab.

[10] Maximum-entropy Markov model. (2018, January 23). Retrieved from https://en.wikipedia.org/wiki/Maximum-entropy_Markov_model.