Part-of-Speech Tagging of Bhojpuri Data

Mayank, Indian Institute of Technology, Banaras Hindu University, Varanasi
Deevashwer, Indian Institute of Technology, Banaras Hindu University, Varanasi
Janvijay Singh, Indian Institute of Technology, Banaras Hindu University, Varanasi
Dr. Anil Kumar Singh, Indian Institute of Technology, Banaras Hindu University, Varanasi

Part-of-Speech Tagging is considered a major initial step in building a language model and accuracy of a lot of other subsequent steps rely on how well the part-of-speech tags have been assigned to the words. Bhojpuri being a relatively low-resource language needs a good analysis of POS Tagging to build a good language model for it. This article shows the part-of-speech tagging of Bhojpuri data using four state-of-the-art POS taggers namely TnT(Trigrams and Tags), CRF++(Conditional Random Fields), SVMStruct(Structured Support Vector Machine) and MaxEnt(Maximum Entropy). We attempt to use these Part-of-Speech Taggers on a Bhojpuri trained corpus consisting of one-fifth of a million words. The performance of each tagger on Bhojpuri corpus has been compared to that on a Hindi trained corpus consisting of one-tenth of a million words and observations have been made on the basis of these results. We hence move to the conclusion of selecting one of these taggers to be the best for our training corpus, do a tag specific analysis over the Bhojpuri data, highlight some of the problems associated with POS tagging of Bhojpuri language and propose a few solutions for these problems. We were able to achieve an accuracy of 91.82% on Bhojpuri compared to an accuracy of 97.14% on Hindi.

CCS Concepts:•**Computing methodologies** → *Machine translation; Language resources;*

General Terms: Natural Language Processing, Algorithms

Additional Key Words and Phrases: NLP, SVM, Markov Models, CRF, Maximum Entropy

1. INTRODUCTION

Part-of-Speech Tagging refers to the process of assigning part of speech tags like noun, verb, etc. to the words present in corpus. It is primarily done as an attempt to better understand the sentences and hence, build promising models that are more capable in generalising the structure of language. It is the first step to building a language model, and the performance of a lot of other steps like chunking, parsing, etc. depends on how well these part of speech tags have been assigned. Bhojpuri is a low resource language spoken in East India. Unavailability of a proper analysis for its part-of-speech tagging has always constrained the development of language models for it. We present our analysis of POS Tagging of Bhojpuri corpus based on four state-of-the-art taggers namely, Maximum Entropy(MaxEnt), Trigrams and Tags(TnT), Conditional Random Fields(CRF++) and Structured Support Vector Machines(SVMStruct).

1.1. Initial Work

Being a low-resource language, availability of a properly tagged corpus poses a big problem. The initial work involved collecting a decently tagged corpus and pruning it to remove most of the incomplete and noisy elements. We have used available implementations of the above-mentioned taggers. CRF++ and SVMStruct have their implementations available as a software package. For the remaining two taggers, we used edu.stanford.nlp.tagger.maxent.MaxentTagger class of Stanford POS tagger for MaxEnt and TnT class of the NLTK library to implement our own program to find the correct POS tags for the test data. The same implementations were used for POS Tagging of the Hindi tagged corpus, which was used for performance comparison with Bhojpuri, available to us.

2. PART-OF-SPEECH TAGGING

2.1. POS Tagging using MaxEnt

We used the maxent class of the Stanford POS tagger to implement our program. MaxEnt is a state-of-the-art POS tagger which is easy and fast at training a model as well as tagging the test data [Ratnaparkhi 1996]. We trained our model using this API. We first generated a propsFile for use in model building. This file is a bank for all the flags that are provided by the API for specifying the algorithm to run in a fashion more specific to your needs. We used a generic architecture for building our model. A generic architecture is a feature extractor that is language independent and works well for the UTF-8 encoded data. As our Bhojpuri corpus was reasonably large, we created 100 different pairs, each one spanning the complete data set, of training and testing data, maintaining a ratio of 3:1 respectively. We ran our program for all of these 100 pairs, the results of which are shown in Fig.1. We used the same generic architecture to build a Hindi model based on 75% of our Hindi data. The testing was done on the rest of Hindi data and finally, the mean accuracies of Hindi and Bhojpuri were compared. The mean accuracy of Bhojpuri turned out to be 91.82% whereas, that of Hindi turned out to be 97.08%.

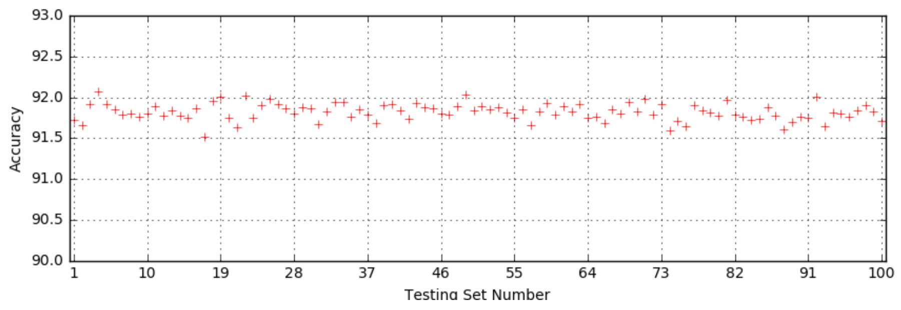

Fig. 1. Accuracy(fMeasure) of MaxEnt for 100 different test data sets

2.2. POS tagging using TnT Tagger

TnT is a Statistical Part-of-Speech tagger which stands for Trigrams and Tags. It can be trained on a variety of languages and virtually any tagset [Brants 2000]. Its algorithm already has many smoothing techniques built in.

NLTK has a module nltk.tag.tnt which implements the TnT() class. We used this class to construct our own POS tagging program based on TnT. We faced a difficulty in storing the model generated by TnT().train(). The API assumes that the model generation and subsequent tagging on the test data will be done on-the-fly. This posed a big problem to what we were thinking of building a model and then storing it and testing it on the test data. We came across a library called Pickle. This library stores any python object in the form of binary bit strings onto a file, which can then be loaded into the memory as and when required. Although it affects the speed of the tagger, but provides it with greater convenience. We built 10 pairs of training and testing files from the Bhojpuri corpus in a similar way as we did for MaxEnt. We did not have good computational resources as we ran the whole experiment on a laptop with 5th gen Intel i7 processor. This constrained us from making 100 randomly generated test and training files as we did in MaxEnt because the speed of using MaxEnt for tagging and training proved out to be at least 30-40 times better than any other tagger that we are considering in this report. Once the models were built for all the ten training data files, we tagged the corresponding 10 test data files and recorded the results. The results are illustrated in Fig.2.

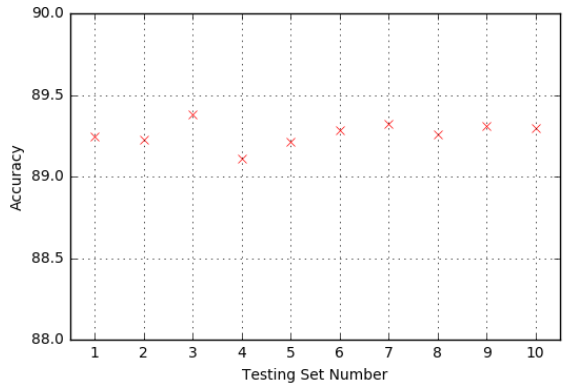

Fig. 2. Accuracy(fMeasure) of TnT for different test data sets

We did a similar procedure on Hindi tagged corpus and the mean accuracies of Bhojpuri and Hindi were compared. The mean accuracy of Bhojpuri turned out to be 89.27%, and that of Hindi turned out to be 94.56%.

2.3. POS tagging using CRF++ Tagger

CRF stands for Conditional Random Fields. CRF++ is an open source implementation of CRF algorithm for sequence labelling. CRF has an advantage over other taggers that it can be used for a variety of NLP operations like Named Entity Recognition, Text Chunking and Information Extraction along with POS-tagging. We conducted tests on a smaller fraction of corpus, extracted training and testing data out of it and compared the accuracy while increasing the complexity of the template file, the file that defines the features to be used by the tagger, with every iteration. The results of this test are shown in Fig.3.

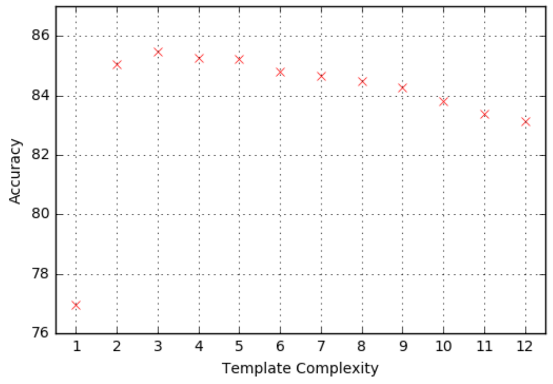

Fig. 3. Accuracy(fMeasure) of CRF++ with increasing template complexity

It is evident from the figure that template number 3 gives the maximum accuracy, moving left or right to it reduces the accuracy, and hence, we selected this template for the rest of the tagging process. Then, ten pairs of training and testing data files were made as earlier and models were trained and tested for each of the pairs. The results are shown in Fig.4. For Bhojpuri, we got a mean accuracy of 90.67% compared to 96.58% for Hindi.

Table I. Template Used for CRF++

n-gram Feature	**Feature Name**	**Components**
1.	U00	%x[-1,0]
2.	U01	%x[0,0]
3.	U02	%x[1,0]

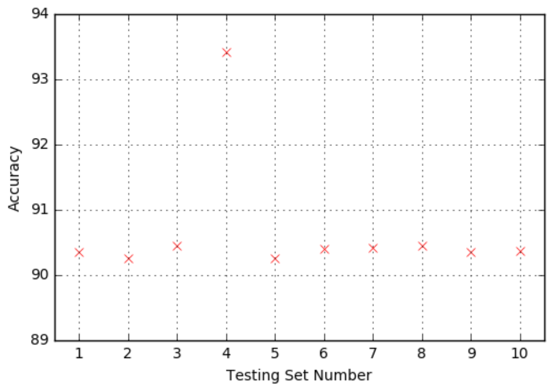

Fig. 4. Accuracy(fMeasure) of CRF++ for different test data sets

2.4. POS tagging using SVMStruct Tagger

SVM stands for Support Vector Machine, which is a widely used algorithm in Machine Learning. But, SVM can not be used for sequence labelling [Jesus Gimenez 2004]. This requires for the use of Structured Support Vector Machine(SVMStruct). It belongs to the category of supervised learning algorithms. It approximates the output values using an approximation hypothesis trained by labelled training data. It can predict trees, sets and sequences unlike SVM, which can only predict regression and classification. It is used widely in parsing and Markov models in POS-tagging. Ten pairs of training and testing data files were created similar to MaxEnt and other taggers. The following feature table was used for building models for all these training data files.

ambiguity classes	a_0, a_1, a_2
may_be's	m_0, m_1, m_2
PoS features	p_{-2}, p_{-1}
PoS bigrams	$(p_{-2}, p_{-1}), (p_{-1}, a_{+1}), (a_{+1}, a_{+2})$
PoS trigrams	$(p_{-2}, p_{-1}, a_{+0}), (p_{-2}, p_{-1}, a_{+1}),$ $(p_{-1}, a_0, a_{+1}), (p_{-1}, a_{+1}, a_{+2})$
single characters	$ca(1), cz(1)$
prefixes	$a(2), a(3), a(4)$
suffixes	$z(2), z(3), z(4)$
lexicalized features	SA, CAA, AA, SN, CP, CN, CC, MW, L
sentence_info	punctuation ('.', '?', '!')

Fig. 5. Feature Table Used for SVMStruct

The test files were then tagged using the SVMStruct software package for all the ten pairs, the results of which are shown in Fig.6. Similarly, accuracy for Hindi data was also calculated which turned out to be 97.14 compared to the mean accuracy of 90.29 for Bhojpuri.

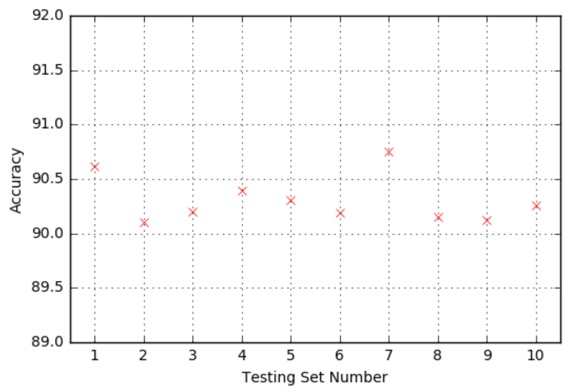

Fig. 6. Accuracy(fMeasure) of SVMStruct for different test data sets

3. RESULTS

All the four taggers were run for Hindi as well as Bhojpuri and the recorded results are shown in Fig.7.

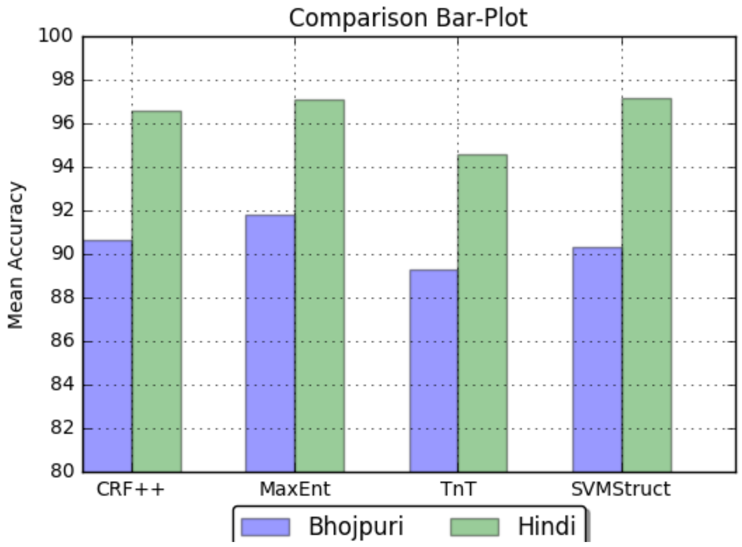

Fig. 7. Final Results

It can be clearly seen that for Bhojpuri the max accuracy was given by MaxEnt - 91.82% and for Hindi, it was given by SVMStruct - 97.14%.

Table II. Results of Tagging

Tagger	**Average Accuracy of Hindi**	**Average Accuracy of Bhojpuri (No. of Models)**
CRF++	96.58	90.67 (10)
SVMStruct	97.14	90.29 (10)
MaxEnt	97.08	91.82 (10)
TnT	94.56	89.27 (10)

4. ERROR ANALYSIS

The error scan on the tagging results shows us some interesting facts about our Bhojpuri tagged corpus. We found that around 250 words were already tagged as ”Unk” in the Bhojpuri corpus compared to half of this number(130) in the Hindi corpus. This reduces the recall of tagging our Bhojpuri test data more than that it does in tagging the Hindi test data and removing them from the corpus directly will disturb the concerned sentence structure. This is one of the factors contributing to the low tagging accuracy for Bhojpuri as compared to Hindi. We did another experiment to determine whether on increasing the test data size, how much does accuracy change. We selected 50%, 75%, 85% and 95% of the total corpus as our training data. The experiment was conducted using MaxEnt tagger running on default options set. The results of this experiment showed that accuracy keeps on improving and reach to 90.88% with 95% corpus as train data and remaining 5% as test data.

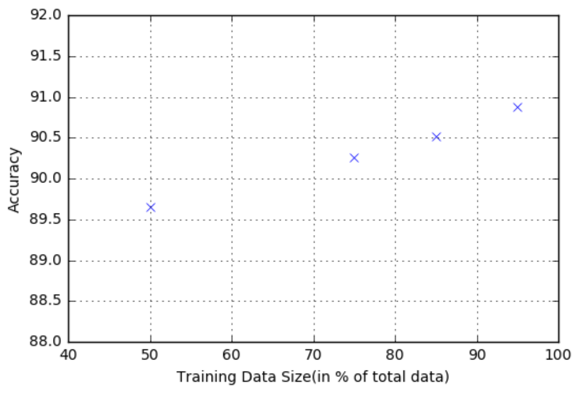

Fig. 8. Accuracy Vs Training Data Size

Table III. Training Data Size Vs Accuracy

Training Data Size(%)	**Test Data Size(%)**	**Accuracy % using MaxEnt**
50	5	89.65
75	5	90.26
85	5	90.52
95	5	90.88

It may be argued that the accuracy of MaxEnt here, even after taking 95% of total corpus as training data is less than the MaxEnt accuracies for all the 100 test data sets as shown in Fig.1, where we just took 75% of total corpus for training. It may be so that random sentences that we are considering to make the testing data file containing 5% of total corpus, might have resulted in a not-so-good set for the tagger and as this testing data contains just one-fifth of the sentences than the ones in the 25% of total corpus test data, an incorrectly tagged word can affect the accuracy more than it could do in the 100 testing files considered in calculating average accuracy of MaxEnt. This is similar to a case where let us say we have just two testing words for the tagger. Now if the tagger outputs a wrong tag the accuracy falls down to 50%. Still, this plot shows a good picture that increasing the training data size is indeed increasing the tagging accuracy. Hence, if we increase our corpus size, hence increasing the training data size as well, we can achieve a greater overall accuracy.

5. CONCLUSION

Having described the experiments we conducted on the tagged corpus of Bhojpuri, we conclude that MaxEnt tagger performs best for this language and corpus giving the maximum accuracy of 91.82%. CRF++ and SVMStruct also perform well on the corpus giving a comparable accuracy of 90.67% and 90.29% respectively. We would also like to point it out that if the 250(approx.) words that were found in the Bhojpuri corpus which were already tagged as ”Unk” can be reduced by using a better tagged corpus, then the accuracy will increase for sure. Given below is a plot showing the tagging accuracies for some common tags taken from a random MaxEnt tagging iteration. This shows us some of the tags that are more prone to errors.

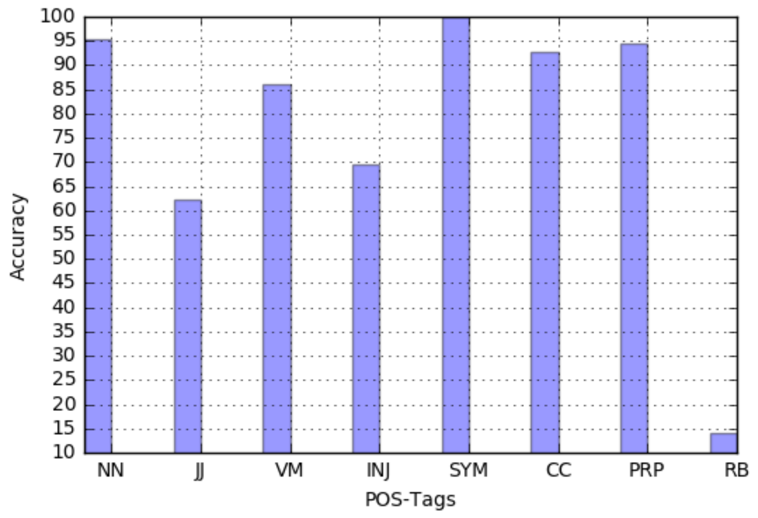

Fig. 9. Accuracy of some common tags

Here NN: Noun, JJ: Adjective, VM: Main Verb, INJ: Preposition, CC: Coordinating Conjunction, PRP: Personal Pronoun and RB: Adverb. We can increase the accuracy to 93+% by taking a bigger training corpus which will count for lesser unseen words in the test data. As the bar chart shows, adjectives (JJ), adverbs (RB) and prepositions (INJ) perform badly compared to other tags, with adverbs performing worst. A bigger and more accurately labelled corpus can improve accuracies for these tags. We can also use an external dynamic dictionary to check for unseen words from and assign them their most likely tag. We can also use a morph analyser to give suitable tags based on word morphology in case on unseen words.

6. FUTURE RESEARCH

We would like to propose the need for building more accurately tagged Bhojpuri corpus which is larger than at least a million words. This would require linguists showing interest in working in the development of better NLP tools for the Bhojpuri language. Special focus needs to be given to the tags more prone to mistakes like JJ, INJ and RB as we have discussed above.
We will also work on taking more specific algorithmic parameters than the general parameters we took in writing this paper. As shown in the paper taking different templates for CRF++ results in decent accuracy changes.
As Nouns have a very high existence percentage among other POS tags, so we propose that a good Hindi corpus can also be used in the training file to make the trained model better. This idea comes from the fact that Proper Nouns like names, are similar in Hindi and Bhojpuri.

REFERENCES

Thorsten Brants. 2000. TnT – A Statistical Part-of-Speech Tagger. In *Proceedings of the 6th Applied Natural Language Processing Conference and the 1st North American Chapter of Association for Computational Linguistics*. 224–231.

Lluis Marquez Jesus Gimenez. 2004. SVMTool: A general POS tagger generator based on Support Vector Machines.. In *Proceedings of the 4th. Language Resources and Evaluation Conference (LREC)*. http://www.lrec-conf.org/proceedings/lrec2004/pdf/597.pdf

Adwait Ratnaparkhi. 1996. A maximum entropy model for part-of-speech tagging.. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing, University of Pennsylvania*. 133–142.