Improving statistical POS tagging using Linguistic feature for Hindi and Telugu

by

Phani Gadde, Meher Vijay Yeleti

in

ICON-2008: International Conference on Natural Language Processing
(*ICON-2008*)

Report No: IIIT/TR/2008/189

Centre for Language Technologies Research Centre
International Institute of Information Technology
Hyderabad - 500 032, INDIA
December 2008

Improving statistical POS tagging using linguistic features for Hindi and Telugu

S Phani Kumar Gadde, Meher Vijay Yeleti
LTRC, International Institute of Information Technology,
Hyderabad, India.
{phani_gadde,meher_vijay}@students.iiit.ac.in

Abstract

In this paper we describe some strategies for improving statistical POS tagging using Hidden Markov Models (HMM) for Hindi and Telugu. We describe how adding features to HMM improves its accuracy. We also describe a method for effective handling of compound words in Hindi. Experiments show that GNP[1] and category information of a word are crucial in achieving better results. The maximum accuracy achieved with HMM based approach is 92.36% for Hindi and 91.23% for Telugu. We achieved an improvement of 1.85% in Hindi and 0.72% in Telugu over the previous methods.

1 Introduction

POS tagging is the task of assigning a part of speech tag to each word of a natural language text based on both its definition and its context. Identifying the POS tags in a given text is an important prerequisite for many Natural language applications such as chunking, parsing, speech recognition and word sense disambiguation. POS tagging is very often used for partial parsing of texts for example to quickly finding names or other phrases for the information extraction applications. POS taggers have been developed using statistical techniques, rule based and sometimes using a hybrid aproach.

The earliest attempts at POS tagging were based on 2 stage architecture (Harris, 1962; Klein and Simmons, 1963; Greene and Rubin, 1971). The ENGTOWL tagger (Voutilainene, 1995) is based on the same 2 stage architecture but with more sophisticated lexicon and disambiguation rules. Probabilities were first used by Stolz et al. (1965) and various statistical taggers were built in the 1980's (Marshall, 1983). Trigrams'n'Tags (TnT) is an efficient statistical POS tagger based on the HMM model which works by training on tagged data. Kupiec (1992) and Cutting et al. (1992) showed that it is also possible to train a HMM tagger on unlabelled data using EM algorithm.

Some of the statistical models used for POS tagger are the Hidden Markov Models (HMMs) (Cutting, 1992), Maximum Entropy Models (MEMs) (Ratnaparkhi, 1999), CRFs (Sha and Pereira, 2002) and TBL (Brill, 1992).

Works on POS tagging on Indian languages include HMM based tagging for Bengali (Dandapat and Sarkar, 2006), use of Conditional Random Fields (Agrawal, 2007), Conditional Random Fields and post processing with Transformation Based Learning (PVS and Gali, 2007), Decision Forests(Pammi and Prahallad, 2007), Hybrid which uses HMM and rule based approach (Pattabhi et al., 2007).

HMM and CRF based tagger are the two major statistical POS tagging methods. It was proved that CRF performs better than HMM because it takes linguistic features along with words. However training a CRF tagger is time consuming whereas HMM takes much less time. So, introducing linguistic features in the HMM based approach may increase the accuracy of tagging.

This paper explains the use of HMM with morphological information for Hindi and Telugu. The performance of both HMM and CRF based taggers improved after effective handling of compound words in Hindi.

The training and testing data for the current experiments was provided by IL-ILMT[2]. The

[1] Gender, Number and Person

[2] IL-ILMT is project sponsored by MC&IT, Govt. of India Reference No: 11(10)/2006-HCC(TDIL)

training size is 185782 and 190006 tokens for Hindi and Telugu respectively. The testing size is 23483 tokens for Hindi and 21351 tokens for Telugu. The tagset used is as described in Guidlines for POS and Chunk Annotation for Indian Languages (Bharati et al., 2006)

The paper is arranged as follows; in Section 2 we describe the tools used, in section 3 we describe the approach followed. Section 4 describes our experiments and results. Analysis of our results is discussed in Section 5. We conclude the paper with some future direction in Section 6.

2 Tools Used

The following tools are used for our experiments on statistical POS tagging:

1. Brants TnT (Brants, 2000), a HMM based tagger
2. CRF++, a CRF based tagger

In all the experiments we used TnT, except in experiment 4.1.2 where we compared the compound word handling using both.

2.1 TnT

Brant's TnT uses second order Markov models for part-of-speech tagging. The states of the model represent tags, outputs represent the words. Transition probabilities depend on the states, thus pairs of tags. Output probabilities only depend on the most recent category. For a given sequence of words w_i . . . w_T of length T and tagset $t_1 \ldots t_T$,

$$\underset{t_1 \ldots t_T}{\operatorname{argmax}} \left[\prod_{i=1}^{T} P(t_i|t_{i-1}, t_{i-2}) P(w_i|t_i) \right] P(t_{T+1}|t_T) \tag{1}$$

is calculated. The additional tags t_{-1} , t_0, and t_{T+1} are used for beginning of sequence and end of sequence markers. Using these additional tags, even if they stem from rudimentary processing of punctuation marks, slightly improves tagging results. This is different from formulas presented in other publications, which just stop with a "loose end" at the last word. If sentence boundaries are not marked in the input, TnT adds these tags if it encounters one of [.!?;] as a token. Transition and output probabilities are estimated from a tagged corpus. Maximum likelihood probabilities P are used which are derived from the relative frequencies:

$$\text{Unigrams:} \quad \hat{P}(t_3) = \frac{f(t_3)}{N} \tag{2}$$

$$\text{Bigrams:} \quad \hat{P}(t_3|t_2) = \frac{f(t_2, t_3)}{f(t_2)} \tag{3}$$

$$\text{Trigrams:} \quad \hat{P}(t_3|t_1, t_2) = \frac{f(t_1, t_2, t_3)}{f(t_1, t_2)} \tag{4}$$

$$\text{Lexical:} \quad \hat{P}(w_3|t_3) = \frac{f(w_3, t_3)}{f(t_3)} \tag{5}$$

for all t_l, t_2, t_3 in the tagset and w3 in the lexicon. N is the total number of tokens in the training corpus. Maximum likelihood probability is zero if the corresponding numerators and denominators are zero. Contextual frequencies are smoothed with linear interpolation and lexical frequencies are completed by handling words that are not in the lexicon with the help of suffix analysis.

2.2 CRF

Conditional random field is a probabilistic framework for labeling and segmenting data. It is a form of undirected graphical model that defines a single log-linear distribution over label sequences given a particular observation sequence. CRFs define conditional probability distributions $P(\mathbf{Y}|\mathbf{X})$ of label sequences given input sequences. Lafferty et al. (2001) define the probability of a par-ticular label sequence Y given observation se-quence X to be a normalized product of potential functions each of the form

$$\exp(\sum \lambda_j \mathbf{t}_j (Y_{i-1}, Y_i, X, i) + \sum \mu_k \mathbf{s}_k (Y_i, X, i)) \tag{6}$$

where $t_j(Y_{i-1}, Y_i, X, i)$ is a transition feature function of the entire observation sequence and the labels at positions *i* and *i-1* in the label sequence; $s_k (Y_i, X, i)$ is a state feature function of the label at position I and the observation sequence; and λ_j and μ_k are parameters to be estimated from training data.

$$F_j(Y,X) = \sum f_j (Y_{i-1}, Y_i, X, i) \tag{7}$$

where each f_j (Y_{i-1}, Y_i, X, i) is either a state function $s(Y_{i-1}, Y_i, X, i)$ or a transition function $t(Y_{i-1}, Y_i, X, i)$. This allows the probability of a label sequence **Y** given an observation sequence **X** to be written as

$$P(\boldsymbol{Y}|\boldsymbol{X}, \boldsymbol{\lambda}) = (1/Z(\mathbf{X})) \exp(\sum \boldsymbol{\lambda}_j F_j(\boldsymbol{Y},\boldsymbol{X})) \qquad \textbf{(8)}$$

Z(**X**) is a normalization factor.

3 Approach

The following sections describe our methods to improve POS tagging for Hindi and Telugu.

3.1 Introducing Features in TnT

As described earlier above CRF considers the linguistic features to improve the accuracy. But it is very slow. As TnT is fast we introduced morphological features to it. Introducing features will disambiguate the words to some extent as well as it increases the accuracy of tagging when a new word is encountered.

Since TnT takes its input in only two columns, we can't give the features in extra columns like CRF. So, we added the features to the token both at the beginning and at the end using '_'. So, we effectively modified the token as token_feature or feature_token. The features used are root, GNP, category and case obtained from the morph. We introduced combinations of the above four features each separated by '_'.

Token	Features	Token with Features
laDake	Root	laDake_laDaka
laDake	Category	laDake_n
laDake	Root and Category	laDake_laDaka_n

Table 1. Table showing adding features to a token.

3.2 Handling Compound Words

Until now compound words in Hindi are marked with POS tag for the last word and 'XC' tag for all the previous words (Example 2). Tagging compound words in this way is not effective as the other words in the compound have no relation with the last word. We cannot solve this problem by adding features as in section 3.1. So, we proceeded as follows:
We replace the 'X' in the 'XC' tag with the POS tag of the last word in the compound word so that the machine can learn effectively that it's a compound word. We did it in two ways. One way is to replace 'X' with the respective tag and leave the 'C' in the 'XC' tag as it is. The other way is to remove the 'C' also from the tag, which means we are substituting the POS tag of the last word in the place of the 'XC' tag. Example 2 below shows both these methods as Experiment 1 and Experiment 2.

Example 2:

	Mahatma	Gandhi	Road
Original Annotation:	XC	XC	NN
Experiment1:	NNC	NNC	NN
Experiment2:	NN	NN	NN

4 Experiments

We started with adding linguistic features like root, GNP to TnT and experimented with different sizes of the corpus for Hindi and Telugu. We also handled compound words only in case of Hindi as described in 4.1.2.

We added combinations of features like root, GNP information, category, case to the words in TnT both for Hindi and Telugu. We experimented with different training sizes.

We used the morph analyzers for Hindi and Telugu to get the linguistic features. We tried adding one feature at a time to the word.

We also grouped different features in all combinations using '_' and added them to the word. We also experimented with adding prefixes and suffixes of different lengths to the words as they may help in handling unknown words effectively.
The following sections describe our experiments for Hindi and Telugu.

4.1 Hindi

4.1.1 Introducing Features in TnT

We used the morph analyzer for hindi developed by Akshara Bharathi et al. It gives all the possible results for each word which contains root, gender, number, person, category, etc. Since the morph analyzer can potentially give multiple analysis for a single word, only the first set was considered and used as features in our experiment. If the analyzer does not give the morph for a word, we leave the word as it is.

We experimented by gradually increasing the training size by 10000 each time beginning with 30000 till 185000.

Feature	Difference in accuracy for the maximum size(%)
Root	0.07
Gnp information	0.81
Category	0.82
Gnp_category	0.85
Root_GNP_case	0.71

Table 2. Table showing accuracies for maximum Training size for Hindi

As seen in Table 2, addition of the feature root did not shown any significant increase in the accuracy.

The maximum precision at the final training size (185782) was 91.35% for word_GNP_category as seen in the above table. Table2 shows the precisions before and after adding the GNP_category feature to the words for Hindi.

Training Size	Initial	Word_GNP_cat	Difference
30118	81.48	84.12	2.64
39633	82.12	84.62	2.5
49145	82.51	84.92	2.41
58666	82.82	85.22	2.4
68170	82.99	85.27	2.28
77677	86.16	87.63	1.47
87313	86.45	87.89	1.44
96784	86.74	88.16	1.42
106313	87.38	88.61	1.23
115842	88.12	89.34	1.22
125272	90.27	91.21	0.94
134384	90.36	91.28	0.92
143647	90.45	91.34	0.89
152811	90.47	91.31	0.84
161889	90.53	91.32	0.79
171105	90.5	91.37	0.87
185782	90.5	91.35	0.85

Table 3. Table showing the POS tagging precision(%) for different sizes of the training corpus for Hindi

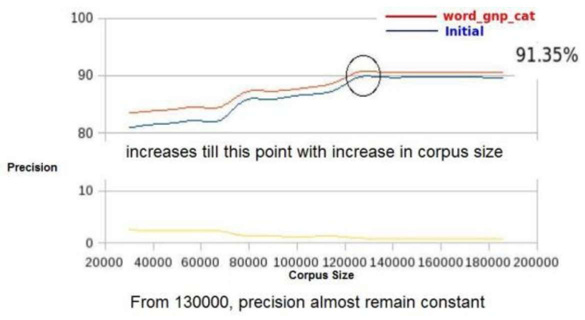

Fig 1. Graph showing the accuracies plotted for various training sizes for initial and word_GNP_category for Hindi.

Table 3 shows that the initial difference was up to 2.6%, but finally it came down to 0.8%. After 130000 words, the difference remained almost the same (0.8%).

Introducing suffix as feature in TnT, the maximum accuracy is 90.54% for suffix length of 2 which is a 0.04% increase from the baseline. Prefix has no affect on the precision.

4.1.2 Handling Compound Words

After adding features to the words, we did two experiments for replacing ‘XC’ tag as described in Section 3.2. We replaced ‘XC’ tag in both training and testing corpora. This increased the accuracy by 1% in TnT and 0.5% in CRF. We did the experiments for both bigrams and trigrams in TnT.

In this experiment, CRF takes a window size of 4 and the other features given were the first 4 prefixes and the last 7 suffixes of the word.

The below Tables 4, 5 show the results of the experiment for Hindi using both TnT and CRF.

Results:

TnT	Initial	Experiment1	Experiment2
Bigram	89.67%	91.61%	91.70%
Trigram	90.50%	92.26%	92.19%

Table 4. Results for Hindi using TnT

CRF	Initial	Experiment1	Experiment2
	92.63%	93.09%	93.13%

Table 5. Results for Hindi using CRF

4.2 Telugu

4.2.1 Introducing Features in TnT

We used the morph analyzer for Telugu developed by Akshara Bharathi et al. This morph analyzer does not give gender and person information. It gives only Root, category and number information. So only these features are added to the token.

Similar to the Hindi case we took only the first morph among all the given morphs for each word and also repeated the experiment for varying training sizes. The maximum training size for Telugu is 190006.

Feature	Difference in accuracy for the maximum Training size(%)
Root	0.55
Category	0.72
Number	0.21

Table 6. Accuracies for maximum training size for Telugu

The maximum precision at the final training size was 91.23% for word_category as seen in Table 6. Table 7 shows the precisions before and after adding the GNP_category feature to the words for Telugu.

Training	Initial	Word_GNP_cat	Difference
30003	87.4	89.51	2.11
40002	88.01	89.76	1.75
50010	88.39	90	1.61
60004	88.94	90.2	1.26
70006	89.01	90.25	1.24
80005	89.31	90.35	1.04
90009	89.52	90.53	1.01
100010	89.64	90.69	1.05
110002	89.76	90.63	0.87
120003	90.03	90.91	0.88
130010	90.18	90.88	0.7
140006	90.22	90.92	0.7
150004	90.28	90.95	0.67
160003	90.35	91.1	0.75
170001	90.51	91.13	0.62
180005	90.53	91.2	0.67
190006	90.51	91.23	0.72

Table 7. POS tagging precision (%) for different sizes of the training corpus for Telugu

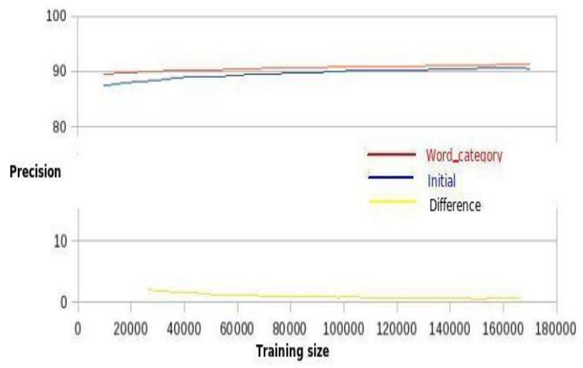

Fig 2. Graph showing the accuracies plotted for various training sizes for initial and word_category for Telugu.

Initial difference was up to 2.11%, finally it came down to 0.72%. Prefix and suffix features did not have any significant affect on the precision.

5 Analysis

Root and prefix-suffix features have no significant affect on the results. This may be because they do not disambiguate between the words, though they may prove effective when unknown words are encountered in the testing corpus. So, there is only 0.04% increase with prefixes. And when root is added as feature, there is 0.07% increase, which is not significant.

Table 8 below shows our accuracies compared with the best POS tagging results of IJCAI'07 shallow parsing contest (ISPC) (Bharati and Mannem, 2007). Only experiments in section 4.1.1 and 4.2.1 are performed as the data is already in the format used in experiment 4.1.2.

	Hindi	Telugu
Previous Best	78.66%	77.37%
Our Result	79.96%	76.30%

Table 8. Comparison of our results with the previous best results of POS tagging for Hindi and Telugu

The Previous Best results shown in the above table are using CRF and TBL whereas our Results are using TnT and linguistics features by the morph on the same data provided by ISPC. The training size is 21425 tokens for Telugu and 21470 tokens for Hindi and the testing size is 5193 tokens and 4924 tokens for Telugu and Hindi respectively. The efficiency for Hindi is 1.3% more than the previous best but in the case of Telugu our result is 1.03% less than the previous best. This is due to the inefficiency and low coverage of the morph in case of Telugu.

Note that the previous best results are obtained by using CRF without the new features added by us like category, case, root and GNP. It is not that TnT outperformed CRF with the same feature set. CRF might give better results with these features, but we have not experimented. What we say is HMM based taggers can be improved by providing richer information.

Unknown words are handled in TnT by assigning weights to the tags of the words having same suffix as that of the unknown words. Since we are adding the features at the end, we are effectively giving the features as the suffixes. It helps in predicting the tag because features are characteristic of different kinds of words. As an example, note that for all the nouns if we add 'n' as the category information, and an unknown noun comes, all the suffixes of the nouns match with its suffix, increasing the probability of it to be tagged as a noun.

GNP and category information is crucial in disambiguating the words as they differ with the context for the same word, therefore their inclusion shows a good increase in the performance.

With small training size the features made a big increase in the accuracy. As we increase the training size, the increase in accuracy over the baseline gradually decreased.

Root has no significance in case of Hindi but its affect is significant in Telugu because Telugu is morphologically richer than Hindi.

6 Conclusion & Future work

The experiments described in this paper show that accuracy using TnT can be improved by adding features to it and by handling compound words. The overall accuracy for Hindi POS tagging is 92.36% using TnT and 93.13% using CRF.

The individual improvement of the accuracies in the two experiments in the sections 4.1.1 and 4.1.2 for Hindi was 0.85% and 1% respectively and the combined improvement is 1.85%. So, adding features did not affect the way compound words are handled.

It would be interesting to observe the commonly occurring errors in POS tagging and how prefixes and suffixes are handling these errors. Separate results for unknown words will help in understanding the handling of prefixes and suffixes.

The results show that the CRF accuracy is still more than that of TnT. But CRF uses only prefix and suffix information as features. So, we were not able to effectively use prefix and suffix in TnT as adding these did not show any significant increase in the accuracy. This is also true in the case of other features. We need to use those features more effectively. Instead of adding features using '_', training module of HMM based tagging can be changed so that it can take different feature values and can learn effectively.

Acknowledgements

We would like to express our gratitude to Dr. Dipti Misra Sharma and Prof. Rajeev Sangal for their guidance and support. We would also like to thank Samar Husain and Gali Karthik for their valuable suggestions.

References

Himanshu Agrawal, 2007. POS Tagging and Chunking for Indian Languages. *Proceeding of Twentieth International Joint Conference on Artificial Intelligence(2007).*

Akshar Bharati and Prashanth R. Mannem, 2007. Introduction to Shallow Parsing Contest on South Asian Languages. *Proceeding of Twentieth International Joint Conference on Artificial Intelligence(2007).*

Akshar Bharati, Rajeev Sangal, Dipti Misra Sharma, Lakshmi Bai, 2006. Annotating Corpora Guidelines For POS And Chunk Annotation For Indian Languages, *Language Technologies Research Centre, IIIT, Hyderabad(2006).*

Thorsten Brants. 2000. TnT – a Statistical Part-of-Speech Tagger. *Proceeding of the sixth conference on Applied Natural Language Processing (2000) pg 224-231.*

Eric Brill. 1995. Transformation-based error driven learning and natural language processing: A case study in part-of-speech tagging**.** *Computational Linguistics.*

Aniket Dalal, Kumar Naggaraj, Uma Sawant, Sandeep Shelke, 2006. Hindi Part-of-Speech Tagging and Chunking : A Maximum Entropy Approach. *Proceeding of NLPAI Machine Learning Contest-2006.*

Sandipan Dandapat, Sudeshna Sarkar, 2006. Part of Speech Tagging for Bengali with Hidden Markov Model. *Proceeding of NLPAI Machine Learning Contest-2006.*

John Lafferty, Andrew McCallum and Fernando Pereira, 2007. Conditional Random Fields:

Probabilistic Models for Segmenting and Labeling Sequence Data.

Sathish Chandra Pammi and Kishore Prahallad, 2007. POS Tagging and Chunking Using Decision Forests. *Proceeding of Twentieth International Joint Conference on Artificial Intelligence(2007).*

Avinesh PVS and Karthik Gali, 2007. Part-Of-Speech Tagging and Chunking using Conditional Random Fields and Transformation Based Learning . *Proceeding of Twentieth International Joint Conference on Artificial Intelligence(2007).*

Adwait Ratnaparkhi. 1998. Maximum Entropy Model For Natural Language Ambiguity Resolution, *Dissertation in Computer and Information Science, Uni-versity Of Pennslyvania, 1998.*

Pattabhi R K Rao T, Vijay Sundar Ram R, Vijayakrishna R and Sobha L, 2007. A Text Chunker and Hybrid POS Tagger for Indian Languages. *Proceeding of Twentieth International Joint Conference on Artificial Intelligence(2007).*

Fei Sha and Fernando Pereira. 2003. Shallow Parsing with Conditional Random Fields. *Proceedings of HLT-NAACL.*

Charles Sutton, 2006. *An Introduction to Conditional Random Fields for Relational Learning.*

CRF++: Yet Another Toolkit

http://crfpp.sourceforge.net/