Part of Speech Tagging Using Statistical Approach for Nepali Text

Archit Yajnik

Abstract—Part of Speech Tagging has always been a challenging task in the era of Natural Language Processing. This article presents POS tagging for Nepali text using Hidden Markov Model and Viterbi algorithm. From the Nepali text, annotated corpus training and testing data set are randomly separated. Both methods are employed on the data sets. Viterbi algorithm is found to be computationally faster and accurate as compared to HMM. The accuracy of 95.43% is achieved using Viterbi algorithm. Error analysis where the mismatches took place is elaborately discussed.

Keywords—Hidden Markov model, Viterbi algorithm, POS tagging, natural language processing.

I. INTRODUCTION

NEPALI language is one of the most spoken languages in Sikkim, Nepal, Bhutan, etc. POS tagging plays a pivotal role in the development of Natural Language Processing applications like Parser and Morphological analyzer. Ample amount of articles are available in the literature on POS tagging task for Indian languages like Hindi, Marathi, Odia, Panjabi etc. but seldom efforts [1], [2] are carried out as far as POS tagging for Nepali text is concerned. Significant literature survey on POS tagging in Nepali language is made available in “AnnCorra: Annotating Corpora for Indian Languages” by Bharati et al. [4]. A hybrid approach demonstrated in [1] is applying Grammar rules on Statistical tagging task and achieved 93.15% of accuracy whereas [2] took the Support Vector Machine approach to perform this task and achieved around 90% of accuracy. Current approach is lucid as it does not incorporate any grammar rules; it is purely based on statistical approach.

Grammar rules may be applied to develop Natural Language Processing applications like Morphological Analysis, Local Word Grouping (LWG), Parsing, etc. based on the output of the tagger.

The article is divided in 6 sections. After introducing the POS tagging for Indian languages in Section I, the statistical techniques viz. Hidden Markov Model (HMM) and Viterbi algorithm based on HMM are briefed in Sections II and III. The methodology and experimental details is covered in Section IV followed by conclusion in Section V. At the end acknowledgment followed by references are presented.

II. HIDDEN MARKOV MODEL

POS tagging plays a vital role in the development of the applications of Natural Language Processing in which inputs are the words and outputs are the corresponding tags i.e. selecting the tag sequence of length n that is the most probable given the input word sequence. HMM is a probabilistic (nondeterministic) finite state automaton, with probabilistic outputs. Let x_i and y_i be the ith word and tag respectively forms an order pair (x_i, y_i) where i = 1, 2, ..., n.

For a given input sequence of words, the task of HMM tagger is to choose the tag sequence that maximizes

\[
p(x/y) = \max_y \frac{p(y|x)p(x/y)}{p(x)}
\]

where \( p(y/x) \) is a prior probability distribution over the tag \( y \).

\( p(x/y) \) is a conditional probability of yielding the input \( x \) given the tag \( y \). By using Bayes rule to derive the conditional probability \( p(y/x) \) for any order pair \( (x, y) \),

\[
p(y/x) = \frac{p(y)p(x/y)}{p(x)}
\]

where \( p(x) = \sum_{n} p(x, n) = \sum_{n} p(n)p(x/n) \) (from (1)).

Our goal is to derive the relation \( y = f(x) \) for the given word \( x \) as following model

\[
f(x) = \max_y p(y|x)
\]

From (2),

\[
f(x) = \arg \max_y \frac{p(y)p(x/y)}{p(x)}
\]

\( p(x) \) is independent of \( y \), hence, does not affect arg max. Therefore the equation looks like

\[
f(x) = \arg \max_y p(y)p(\tilde{x} | y)
\]

Archit Yajnik is with the Sikkim Manipal Institute of Technology, Sikkim Manipal University, India (phone: 0091-96353 19656; e-mail: archit.yajnikr@gmail.com).
Using first Markov assumption, in a sequence \( \{x_1, x_2, ..., x_n\} \)

\[
(x_n/x_{n-1}, x_{n-2}, ..., x_1) \approx p(x_n/x_{n-1})
\]

We can express the joint probability using Markov assumption,

\[
p(x_1, x_2, ..., x_n) = \prod_{i=1}^{n} p(x_i/x_{i-1})
\]  (4)

For the \( n \)th element of the sequence of word

\[
f(x_n) = \max_{y_n} \prod_{i=1}^{n} p(x_i/y_i) \prod_{i=2}^{n} p(y_i/y_{i-1})
\]  (5)

Assume that the probability of a word is only dependent on its own POS tag, then

\[
p(x_n/y_n) \approx \prod_{i=1}^{n} p(x_i/y_i)
\]  (6)

which constitutes the emission probability matrix.

If the probability of POS tag is only dependent on the previous POS tag, then

\[
p(y_n) \approx \prod_{i=2}^{n} p(y_i/y_{i-1})
\]  (7)

which constitutes the transition probability matrix. Using (6) and (7) in (5)

\[
f(x_n) = \max_{y_n} \prod_{i=1}^{n} p(x_i/y_i) \prod_{i=2}^{n} p(y_i/y_{i-1})
\]  (8)

This model is known as bigram model.

### III. VITERBI ALGORITHM

Viterbi algorithm makes some assumptions:

- There are two states of events with respect to time viz. Emitted events and hidden events.
- There has to be one – one correspondence of these to sequence of events.

Viterbi is a common decoding algorithm to find the \( y^* \), the most likely path of tags given the emissions[5].

\[
y^* = \arg \max_y p(y/x) = \arg \max_y p(y, x)
\]

It is based on Bottom-up dynamic programming approach. For an input sentence\( \{x_1, x_2, ..., x_n\} \). This is the problem of finding

\[
\arg \max_{y_1, ..., y_{n+1}} p(x_1, x_2, ..., x_n, y_1, ..., y_n, y_{n+1})
\]

where \( y_1, ..., y_n \) belongs to the set of Tags and \( y_{n+1} = \text{STOP} \).

\[
p(x_1, x_2, ..., x_n, y_1, ..., y_n, y_{n+1}) = \prod_{i=1}^{n} p(x_i/y_i) \prod_{i=1}^{n+1} p(y_i/y_{i-2}, y_{i-1})
\]  (9)

Application of Brute-force search technique in (9) yields Viterbi algorithm.

### IV. EXPERIMENT AND DISCUSSION

The database is generated from NELRALEC Tagset [6] with 41 tags. A report on Nepali Computational Grammar is made available by Prajwal Rupakheti et al. [7] which contains frequently used tag set. A POS tagged sentence of Nepali text with English conversion is illustrated in Figs. 1 (a) and (b) respectively. In Fig. 1 (a), DUM represents unmarked demonstrative pronoun, PKO indicates KO postposition whereas RBO and PPS demonstrate other adverb and possessive pronoun respectively. Moreover, NN and POP represent common noun and other post position respectively. At the end of the sentence VBX and YF indicate auxiliary verb and sentence final respectively. The tagset of the database used in the experiment is having the format shown in Fig. 1 (a).

Out of 574 mismatches (Table I) with Viterbi algorithm is depicted in Figs. 2 and 3. Out of 574 mismatches 309 are wrongly identified as NN (Common Noun) as shown in Fig. 1. Upon observing 309 incorrectly identified NN’s, 35 are misidentified as NNP (Proper Noun). Hence, out of the total 48 NNP Tags from Fig. 3, 35 are confused with NN. The transition probability of occurrence of NN after NN a 0.1519 which is the highest among all, whereas the occurrence of NNP after NN is 0.0056 and NN after NNP is 0.0065. Maximum mismatch (309) takes

<table>
<thead>
<tr>
<th>Sr. No</th>
<th>Technique</th>
<th>No of Mismatches</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>HMM</td>
<td>3455</td>
<td>76.97</td>
</tr>
<tr>
<td>2</td>
<td>Viterbi</td>
<td>574</td>
<td>95.43</td>
</tr>
</tbody>
</table>
place for NN only as appeared the tallest pick in Fig. 2 which is due to the highest probability from NN to NN.

V. CONCLUSION AND FUTURE WORK

Viterbi algorithm is found to be very accurate as compared to HMM for POS Tagging of Nepali text as depicted in Table I. The frequency of misclassification of the Tag NN (common noun) is 309 as shown in Fig. 2 which is the highest compared all other tags. In 35 out of 48 cases (Fig. 3) where the original tag is supposed to be NNP is wrongly tagged as NN. The second highest frequency of misclassified Tag CD (Cardinal Number) is 128. In 73 out of 128 cases (Fig. 2), the Cardinal Numbers are confused due to the availability of mixed and improper fractions in the database. Such errors can be minimized by applying proper grammar rule and that is the future task to work on. Another approach to improve accuracy is to apply Conditional Random Fields (CRF) with morphological analyzer presented by Patel and Gali [8].

![Fig. 2 Tag Wise Error Analysis for Viterbi](image-url)
ACKNOWLEDGMENT

Dr. Archit Yajnik is sincerely thankful to Dr. Samarjeet Borah, Associate Professor, Department of Computer Applications, Sikkim Manipal Institute of Technology, India for providing useful information about programing in Java. He extends his gratitude to Dr. Samar Sinha, Department of Nepali, Sikkim University, Gangtok, India for his valuable linguistic suggestions in Nepali Grammar and Annotated Corpora for POS Tagging in Nepali text. Without the project funded by Department of Science and Technology, New Delhi, this work would not have been initiated, so he expresses his sincere thanks to DST for providing an opportunity to work on the platform of NLP.

REFERENCES