Volume 3, Issue 8, August 2013 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Mix Hidden Markov Model Based Part-of-Speech Tagging for Urdu in Limited Resource Scenario

Smt.M.Humera Khanam* *Department of CSE& S.V. University India*
Prof.K.V.Madhumurthy *Department of CSE&S.V University India*
Md.A.Khudhus *J.E, BSNL India*

Abstract—In this paper we depict a Mixture Hidden Markov Model (HMM) based stochastic algorithm for Part of speech (POS) tagging. HMM is the most successfully used simple language model (n-gram) for POS tagging that uses diminutive amount of knowledge about the language, apart from simple contextual information[1]. In view of the fact that only a small annotated training data is available for Urdu POS tagging, a simple HMM based approach does not give up very good domino effect. Along with HMM we use Morphological analyser (MA) [2] and Stemmer [3] to improve the performance of tagger. Further, we use the semi-supervised learning technique by augmenting the small annotated training data set with a larger unannotated data set to improve the outcome of the tagger in limited resource scenario. We implemented the tagger on both bigram (first order) and trigram HMM (second order) model. Thus we shall call these proposed models as first order supervised Hidden Markov Model (HMM-S1), first order semi-supervised Hidden Markov Model (HMM-SS1), second order supervised Hidden Markov Model (HMM-S2), second order semi supervised Hidden Markov Model (HMM-SS2), HMM-S1+Suf, HMM-S1+MA, HMM-SS1+Suf, HMM-SS1+MA, HMM-S2+Suf, HMM-S2+MA, HMM-SS2+Suf, HMM-SS2+MA, finally we combined both MA and Stemmer to obtain higher accuracy, hence the new models are called as HMM-S1+MA+Suf, HMM-SS1+MA+Suf, HMM-S2+MA+Suf, HMM-SS2+MA+Suf. Totally we have 16 extended HMM that are implemented and tested by different data sets and results are analysed.

Keywords— POS tagger, Urdu, Mix Hidden Markov Model, Morphological Analyser, Stemmer.

I. INTRODUCTION

Part-of-Speech (POS) tagging is the process of assigning a part of speech or lexical class marker to each word in corpus. Tags are also usually applied to punctuation markers, thus tagging for natural language is the same process as tokenization for computer languages although tags for natural languages are much more ambiguous [1] and plays fundamental role in various Natural Language Processing (NLP) applications such as speech recognition, information extraction, machine translation and word sense disambiguation etc. POS tagging particularly plays very important role in word-free languages because such languages have relatively complex morphological structure of sentences than other languages.

II. OVERVIEW OF HIDDEN MARKOV MODEL

A Hidden Markov Model (HMM) is one of the statistical based language model that can be used to solve classification problems. The model can be representing with an interconnected set of *states* which are connected by a set of *transition probabilities*. Transition probability is a probability of the state moves from one state to another state. A process starts at a fussy state and moves to a new state as governed by the transition probabilities in discrete time intervals. As the process enters into a state one of a set of *output symbol* also known as *observation* is emitted by the process. The symbol emitted, is dependent on the probability distribution of the particular state. The output of the HMM is a sequence of observations. In an HMM, the exact state sequence corresponding to a particular output symbol is hidden and the probabilities of the states lies between 0 to 1[1].

A. Basic definitions and notation

According to Rabiner [4], five tuples are required to represent Hidden Morkov Model. They are finite set of States (S), finite set of Output symbols (O), State transition probability (A), Output symbol emission probability (B) and Initial state probability (π_i). Figure 1 shows the General representation of an HMM.

1. $S=\{s_1,s_2,s_3,\ldots.s_N\}$ where N is the number of distinct states in a model. In case of Part-of-speech tagging $N \in \{T\}$ where T is the tagset that will be used by the system. Each tag in the tagset corresponds to one state in the HMM.
2. $O=\{o_1,o_2,o_3,\ldots o_M\}$ where M is the number of different output symbols in the HMM. In case of Part-of-Speech tagging, the number of output symbols M is the number of words in the lexicon of the system.
3. $A = \{a_{ij}\}$ where a_{ij} is the probability of the state that moves from state *i* to *j* in one transition. In case of part-of-

speech tagging the states correspond to tags, so a_{ij} is the probability that the model will move from tag t_i to t_j where (t_i, $t_j \in \{T\}$). In other words, $a_{ij=} P(t_j \mid t_i)$ is the probability that t follows t_i .This probability is usually estimated from the annotated corpus during training.

4. B = $\{b_j(k)\}$ where $b_j(k)$ represents the probability that the k-th output symbol will be emitted when the model is in state j. For POS tagging, this is the probability that the word w_k will be emitted when the process is in state t_j i.e $p(w_k \mid t_j)$.This probability can also be estimated from the training corpus.

5. $\pi = \{\pi_i\}$ where π_i is the initial probability of the model that will starts at state i. In POS tagging, this is the probability that the sentence will begin with a particular tag t_i..

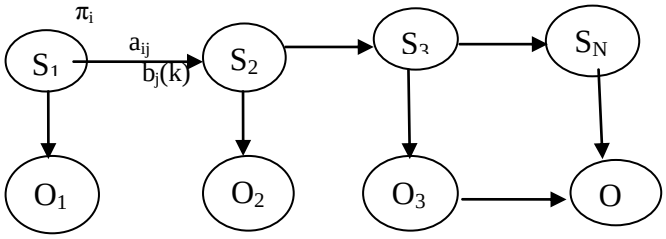


Fig 1: General representation of an HMM

III. OUR APPROACH

In our approach we use HMM, MA and Stemmer for automatic POS tagging of Urdu text. MA has been used to reduce the space complexity and time complexity of the tagger and stemmer has been used to tag unknown words based on Sufix information. Figure 2 shows the architecture of our approach, it contains mainly three components, namely Language model, Disambiguator and possible class restriction module. These components are explained in detail in the following sections.

A .Language model

Language model is the representation of the knowledge about the task of POS disambiguation. The knowledge may come from several resources and can be encoded in various representations. In particular to HMM, the language model is represented by the model parameters $\mu = (\pi_i, A, B)$. We aim to estimate the model parameters $\mu = (\pi_i, A, B)$ of the HMM using corpora. The model parameters of the HMM are estimated based on the annotated data during supervised learning. Unannotated data are used to re-estimate the model parameters during semi-supervised learning. The model parameters are re-estimated using Baum-Welch algorithm. The taggers will be implemented based on both bigram and trigram HMM models.

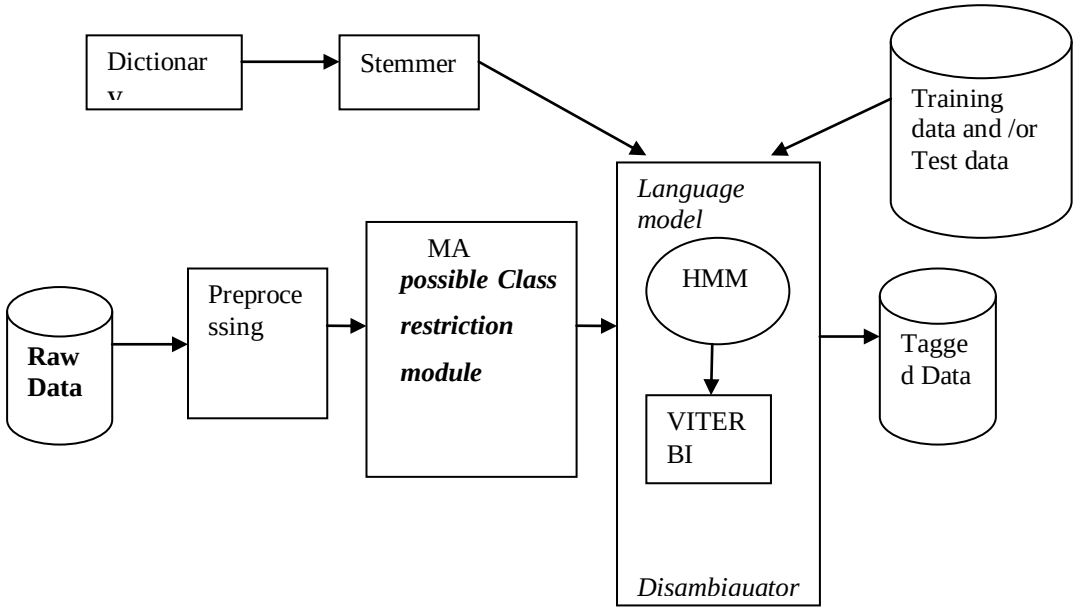


Figure 2: The Mixed HMM based POS tagging architecture

1).The Algorithm for HMM Tagging

The HMM tagger isn't just choosing the best tag for a simple word but the best sequence of tags for an entire sentence. We use the viterbi approximation [1] and choose the most probable tag sequence for each sentence. This approach thus assumes that we are trying to compute for each sentence the most probable sequence of tags $T=t_1,t_2,\ldots,t_n$ given the sequence of words in the sentence (w):

$$\hat{T} = \underset{T \in t}{\text{argmax}}\ P(T|W)$$

By Bayes Law, P(T|W) can be expressed as:

$$P(T|W) = \frac{P(T)\,P(W|T)}{P(W)}$$

Thus we are attempting to choose the sequence of tags that maximizes $\frac{P(T)\,P(W|T)}{P(W)}$

$$\hat{T} = \text{argmax}\ \frac{P(T)\,P(W|T)}{P(W)}$$

Since we are looking for the most likely tag sequence for a sentence given a particular word sequence, the probability of the word sequence P(W) will be the same for each tag sequence and we can ignore it.

$$\hat{T} = \underset{T \epsilon t}{\text{argmax}}\ P(T)\ P(W|T)$$

From the chain rule of probability:

$$P(T)\ P(W|T) = \prod_{i=1}^{n} P(w_i|w_1t_1 \ldots w_{i-1}t_{i-1}t_i)P(t_i|w_1t_1 \ldots w_{i-1}t_{i-1})$$

First we make the simplifying assumption that the probability of a word is dependent only its tag:

$$P(w_i|w_1t_1 \ldots w_{i-1}t_{i-1}t_i) = P(w_i|t_i)$$

Next, we make the assumption that the tag history can be approximated by the most recent two tags:

$$P(t_i|w_1t_1 \ldots w_{i-1}t_{i-1}) = P(t_i|t_{i-2}t_{i-1})$$

Thus we are choosing the tag sequence that maximizes:

$$P(t_1)P(t_2|t_1)\prod_{i=3}^{n} P(t_i|t_{i-2}t_{i-1})\left[\prod_{i=1}^{n} P(w_i|t_i)\right]$$

We can use maximum likelihood estimation from relative frequencies to estimate probabilities:

$$P(t_i|t_{i-2}t_{i-1}) = \frac{c(t_{i-2}t_{i-1}t_i)}{c(t_{i-2}t_{i-1})}$$

$$P(w_i|t_i) = \frac{c(w_i,t_i)}{c(t_i)}$$

This model can also be smoothed to avoid zero probabilities.
Finding the most probable tag sequence can be done with the viterbi algorithm describe in next section.

B. Disambiguator

the main function of disambiguator is to decide the best possible tag for each word in a sentence according to the language model. Viterbi algorithm has been used for disambiguation.

1).The Viterbi Algorithm

Function VITERBI(observations of len T, state-graph) **returns** best-path


```
num-state ← NUM-OF-STATES(state-graph)
Create a path probability matrix viterbi[num-states+2,T+2]
 viterbi[0,0]←1.0
for each time step t from 0 to T do
    for each state s from 0 to num-states do
        for each transition s^1 from s specified by state-graph
            new-score←viterbi[s,t]*a[s,s^1]*b_s^1(0_t)
            if ((viterbi[s^1,t+1]=0) || (new-score > viterbi[s^1,t+1]))
                then
                        viterbi[s1,t+1]←new-score
                           back-pointer[s1,t+1]←  s
```

Back trace from highest probability state in the final column of viterbi[] and return path.

***C. possible class restriction* module** *possible class restriction* module estimates the set of possible tags {*T*}, for every word in a sentence. We use Morphological Analyzer [2] as *possible class restriction* module. This module consists of a list of lexical units associated with the list of possible tags. In this approach the assumption are made that every word can be associated with all the tags in the tagset (i.e. *a set of 35 tags in the tagset* {*T*}). Further, assume that the POS tag of a word *w* can take the values from the set $T_{MA}(w)$, where $T_{MA}(w)$ is computed by the Morphological Analyzer.

The Language model, disambiguator and possible class restriction module are related and combine them into a single tagger description. The input to the disambiguation algorithm takes the list of lexical units with the associated list of possible tags. The disambiguation module provides the output tag for each lexical unit using the encoded information from the language model. The following subsections gives a detailed design of the above three components in this work.

IV. EXPERIMENTAL SETUP AND RESULTS

In this section, we outline our experimental setup and discuss the effect of MA and stemmer on the system performance. First we implemented baseline model to know the difficulty of the POS tagging task. In this model the tag probabilities depend only on the present word:

$$P(t_{1\ldots\ldots}t_n \mid w_{1\ldots\ldots}w_n) = \prod_{i=1,n} P(ti \mid wi)$$

he effect of this is that the each word in the test data will be assigned the tag which occurred most frequently for that word in the training data. Note that the baseline model has an accuracy of 78.88%.

Next we implemented our proposed models (HMM-S1, HMM-S1+Suf, HMMS1+MA, HMM-S1+Suf+MA, HMM-SS1, HMM-SS1+Suf, HMM-SS1+MA, HMM-SS1+Suf+MA, HMM-S2, HMM-S2+MA, HMM-S2+Suf, HMM-S2+MA+Suf, HMM-SS2, HMM-SS2+Suf, HMM-SS2+MA, HMM-SS2+MA+Suf) under bigram and trigram HMM based stochastic tagging schemes. We use the same training data to estimate the parameters for all the models. The model parameters for supervised HMM based models are estimated from the annotated text corpus. For semi-supervised learning, the HMM learned through supervised training is considered as the initial model. Further, a larger unannotated training data has been used to re-estimate the model parameters of the semi-supervised HMM. The experiments were conducted with three different sizes (3K, 5K and 7K words) of the training data to understand the relative performance of the models. The experiments (HMM-S2) have carried out with the freely available ACOPOST (for "A Collection of Part-Of-Speech Taggers) [5], which is based on a supervised trigram HMM with suffix tree information for unknown words. These experiments give us some insight about the performance of the tagging task in comparison with the order of the Markov model in a poor resource scenario.

A. Training data

Data for our experiments was taken from Department of Urdu. This data set consisted of 15786 words of different story books, and manually tagged the data with 35 different tags [7] at Sri vekateswara University, Tirupati. We use this tagset because it gives better results among the other tagsets. This data was spread as 3000,5000 and 7000 words across three files. We perform three fold cross validation on this data set. Model parameters has been estimated using this data set during supervised learning.

Another data set of 10,000 words unlabeled data was taken from CIIL Mysore [8] and this was used to re-estimate the model parameter during semi-supervised learning.

B. Test data

All the models have been tested on a set of randomly drawn 7000 words from the training corpus. It has been noted that 15% words in the open testing text are unknown with respect to the training set, which is also a little higher compared to the European languages [9]

C. Pre-processing

Our system process data in two phases. In first phase, resources necessary for tagging the text are generated. The generated resources include cleaned data and tokenized data.

In second phase, another resource is generated in preprocessing phase is the list of suffixes for all words during stemming. For every word in the corpus, dictionary stores information about the list of possible tags.

V. IMPLEMENTATION

Our POS tagger was developed in Java and uses the ACOPOST[5] Tagger and TnT [6] Tagger for experiments. This taggers make use of HMM algorithm to estimate the model parameters during supervised learning and Baum Welch algorithm [10] during semi supervised learning. During tagging phase, Viterbi algorithm[11] is employed to find the most promising tag sequence. Typical execution times on an Intel Pentium 4 machine with Linux are approximately 15 seconds for training and 3 seconds for tagging.

A. Results and System Performance

we use three measures to evaluate the performance of our system, namely, Overall word tagging accuracy, Known word tagging accuracy and unknown word tagging accuracy. Overall word tagging accuracy has been defined as the ratio of the correctly tagged words to the total number of words.

$$Word\ Tagging\ Accuracy\,(\%) = \frac{Correctly\ tagged\ words\ by\ the\ system}{Total\ no.of\ words\ in\ the\ corpus}\ x\ 100$$

Known word tagging accuracy has been defined as the ratio of the correctly tagged know word to the total number of known words.

$$Know\ word\ Accuracy(\%) = \frac{Correctly\ tagged\ know\ words\ by\ the\ system}{Total\ no.of\ words\ in\ the\ evalution\ set}\ x\ 100$$

Similarly unknown word tagging accuracy has been defined as the ratio of the correctly tagged unknown words to the total number of unknown words.

$$Unknow\ word\ Tagging\ Accuracy(\%) = \frac{Correctly\ tagged\ unknow\ words\ by\ the\ system}{Total\ no.of\ unknow\ words\ in\ the\ evalution\ set}\ x\ 100$$

We performed 3-fold cross validation on the data. Table 1 and Figure4 shows the overall improvement in accuracy of each of the models along with the increase in the size of annotated data when supervised learning algorithm (HMM-S1, HMM-S1+suf, HMM-S1+MA and HMM-S1+suf+MA) have been used to estimate the model parameters.

TABLE 1 :TAGGING ACCURACY OF DIFFERENT SUPERVISED MODELS WITH DIFFERENT CORPUS

Training Corpus (words)	Methods and accuracies in %			
	HMM-S1	HMM-S1+Suf	HMM-S1+MA	HMM-S1+Suf+MA
3000	57.53	75.13	82.39	84.73
5000	70.61	79.76	84.06	87.35
7000	77.29	83.85	86.64	88.75

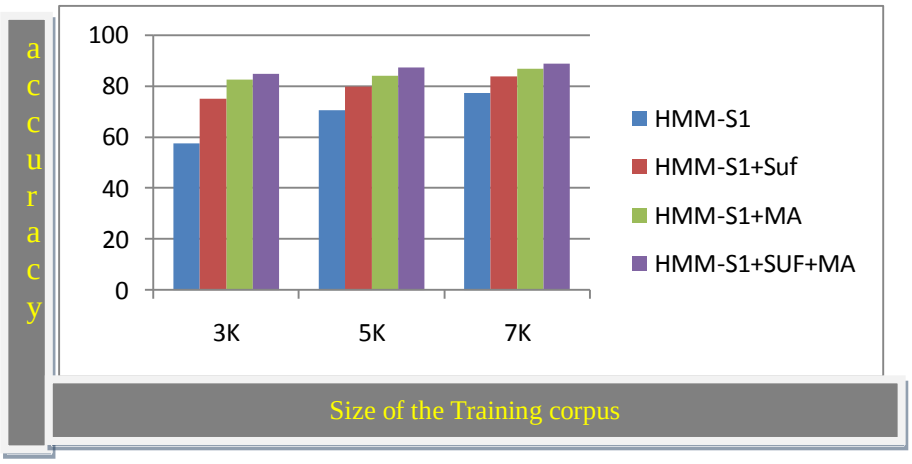


Figure 4: The accuracy growth of different supervised HMM models.

Tagging accuracy of known word has been measured separately for all the models to understand their performance in a poor resource scenario. Table 2 and figure 5 shows known word accuracy under different HMM models during supervised learning.

TABLE 2 :KNOWN WORD TAGGING ACCURACY OF DIFFERENT SUPERVISED MODELS WITH DIFFERENT CORPUS

Training Corpus (words)	Know word accuracy in different models			
	HMM-S1	HMM-S1+Suf	HMM-S1+MA	HMM-S1+Suf+MA
3000	74.19	84.35	85.81	87.81
5000	79.42	84.58	84.77	88.29
7000	83.07	86.90	86.82	88.95

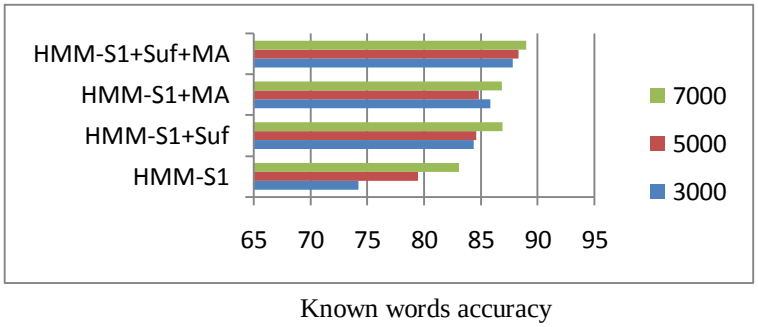
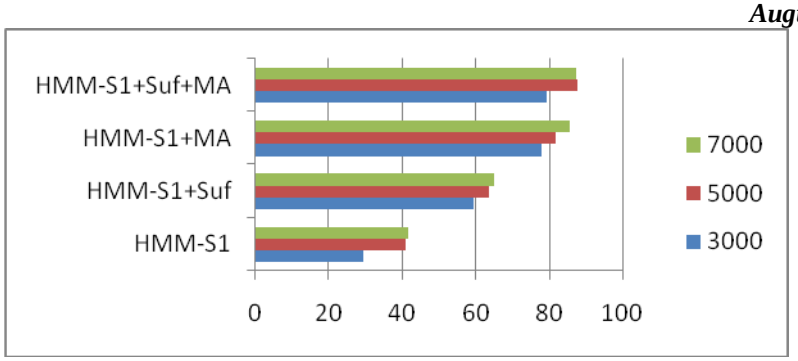


Figure5: Known words accuracy growth of different supervised HMM models.

We also measured the accuracy of Unknown word separately for all the models to understand their performance in a poor resource scenario. Table .3 and figure 6 shows Unknown word accuracy under different supervised HMM models. It is obvious that the number of unknown words is always high when less amount of annotated data is available. So, the models which handles the unknown words better are considered to be a well fitted model for the POS disambiguation task in a poor resource scenario.

TABLE 3 :UNKNOWN WORD TAGGING ACCURACY OF DIFFERENT SUPERVISED HMM MODELS WITH DIFFERENT CORPUS

Training Corpus (words)	Unknow word accuracy in different models			
	HMM-S1	HMM-S1+Suf	HMM-S1+MA	HMM-S1+Suf+MA
3000	29.42	59.54	77.80	79.53
5000	40.88	63.53	81.67	84.21
7000	41.78	65.13	85.56	87.55



Unknown words accuracy

Figure 6:Unknown words accuracy growth of different supervised HMM models

It is interesting to note that known word accuracy under the supervised HMM models improve (about 4% for both the cases) while suffix or MA (HMM-S1+suf, HMM-S1+MA) is being used in the models over the simple HMM model (HMM-S1). However a combination of both suffix and MA (HMM-S1+suf+MA) gives about 6% improvement compared to the HMM-S1 model. However, the improvement in unknown word accuracy is much higher due to the uses of suffix and/or MA. The uses of suffix (HMM-S1+suf) and MA (HMM-S1+MA) gives an improvement of 25% and 44% respectively compared to the HMM-S1 model. The improvement is much higher (35%) while both suffix and MA are used in the HMM model (HMM-S1+suf+MA).

Similar trend has been observed for the semi-supervised HMM models. Table 4 and Figure 7 shows the improvement in over all accuracy of each of the semi-supervised models (HMMSS1+suf, HMM-SS1+suf, HMM-SS1+MA, HMM-SS1+suf+MA) along with the increase in the size of annotated training data.

TABLE 4 :TAGGING ACCURACY OF DIFFERENT SEMI-SUPERVISED MODELS WITH DIFFERENT CORPUS

Training Corpus (words)	Methods and accuracies in %			
	HMM-SS1	HMM-SS1+Suf	HMM-SS1+MA	HMM-SS1+Suf+MA
3000	63.40	75.08	83.04	84.41
5000	70.67	79.31	84.47	87.16
7000	77.16	83.76	86.41	87.95

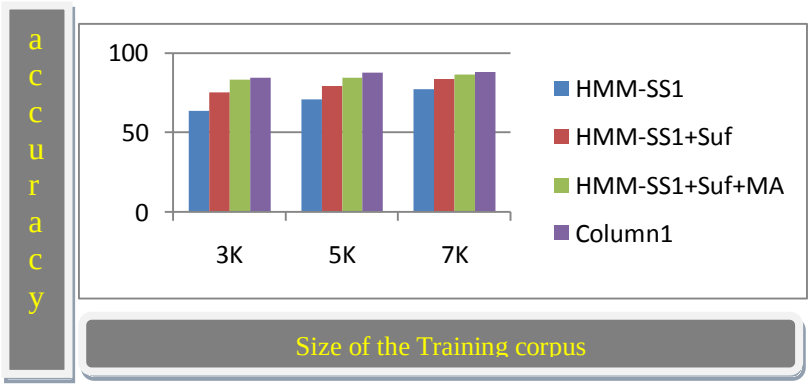


Figure 7: Overall accuracy growth of different semi-supervised HMM tagging models.

Tagging accuracy of known word has been measured separately for all the models to understand their performance during semi-supervised learning model in limited resource scenario. Table 5 and figure 8 shows known word accuracy under different semi-supervised learning HMM models.

TABLE 5 :Known Word Tagging Accuracy Of Different Semi-Supervised Hmm Models With Different Corpus

Training Corpus (words)	known word accuracy in different models			
	HMM-SS1	HMM-SS1+Suf	HMM-SS1+MA	HMM-SS1+Suf+MA
3000	81.84	84.20	86.23	87.35
5000	83.33	84.83	86.32	88.10
7000	85.09	86.73	87.96	89.56

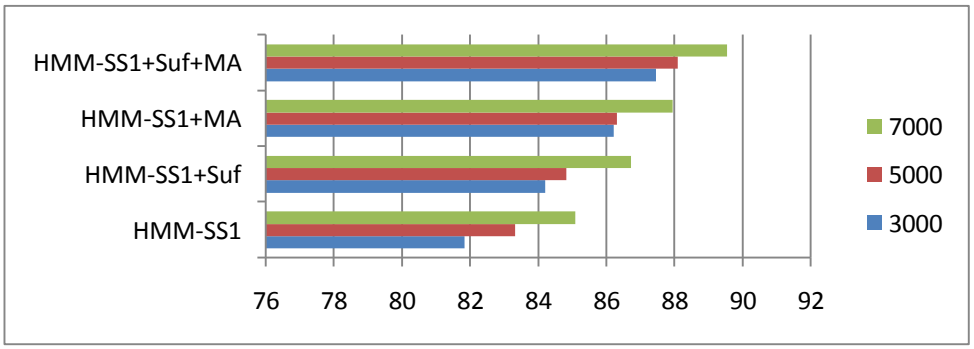


Figure8: Known word accuracy of different semi-supervised HMM tagging models.

We also measured the accuracy of Unknown words in all models during semi-supervised learning to understand their performance. Table6 and figure 9 shows Unknown word accuracy under different supervised HMM models.

TABLE 6 :Unknown Word Tagging Accuracy Of Different Semi-Supervised Hmm Models With Different Corpus

Training Corpus (words)	known word accuracy in different models			
	HMM-SS1	HMM-SS1+Suf	HMM-SS1+MA	HMM-SS1+Suf+MA
3000	32.13	61.90	78.26	81.79
5000	33.41	65.83	81.34	86.98
7000	36.27	68.23	84.04	89.73

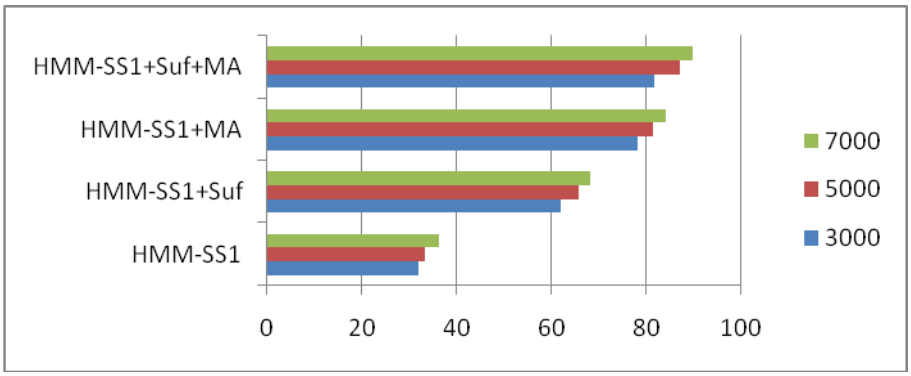


Figure 9: Unknown word accuracy of different semi-supervised HMM tagging models.

Furthermore, the experiments were carried out with second order HMM (HMM-S2) with existing taggers: TnT [6].which is based on a supervised trigram HMM with suffix tree information for unknown words. These experiments give us some insight about the performance of the tagging task in comparison with the order of the Markov model in a poor resource scenario.The same training text has been used to estimate the parameters for all the models(HMM-S2, HMM-S2+suf, HMM-S2+MA, HMM-S2+Suf+MA, HMM-SS2, HMM-SS2+Suf, HMM-SS2+MA, HMM-SS2+Suf+MA).

VI. Conclusion

This paper has described an approach for automatic stochastic tagging of natural language text. The models described here are very efficient for automatic tagging even when the amount of available annotated text is small. The models have a much higher accuracy than the naive baseline model. However, the performance of the current system is not as good as that of the best POS-taggers available for English and other European languages. The best performance is achieved for the supervised bigram HMM learning model along with morphological restriction on the possible grammatical categories of a word and suffix information for handling unknown words. In fact, in all the models discussed above the use of MA enhances the performance of the POS tagger significantly. We conclude that the use of morphological features is especially helpful to develop a reasonable POS tagger when tagged resources are limited.Although HMM performs reasonably well for part-of-speech disambiguation task, it uses only local features (*current word, previous one or two tags*) for POS tagging. Uses of only local features may not work well for a morphologically rich and relatively free order word language Urdu.

References

[1]. Jurafsky, D and Martin H. James, (2000), Speech and Language Processing, Prentice Hall.

[2]. M. Humayoun, H. Hammarström, and A. Ranta. Urdu Morphology, Orthography and Lexicon Extraction. CAASL-2: The Second Workshop on Computational Approaches to Arabic Script-based Languages, July 21-22, 2007, LSA 2007 Linguistic Institute, Stanford University. 2007. (Design decision of version 1.2 are based on this paper) (pdf) (presentation)

[3]. Urdu stemmer Assas-band - Center for Language Engineering www.cle.org.pk/software/langproc/UrduStemmer.htm -

[4]. L.R. Rabiner (1989): "A tutorial on Hidden Markov Models and selected applications in speech recognition." Proceedings of the IEEE 77: 257–286.

[5]. ACOPOST 1.8.6 for Linux - A open source collection of ... linux.softpedia.com/get/Text-Editing.../ACOPOST-96269.shtml - Cached Free – Linux 30 Jan 2013 ... ACOPOST is an open source software that provides a collection of free POS taggers. It also provides a uniform environment for testing, and ...

[6]. TnT -- Statistical Part-of-Speech Tagging - Universität des Saarlandes www.coli.uni-saarland.de/~thorsten/tnt/ - Cached - Similar 26 Oct 1998 ... TnT, the short form of Trigrams'n'Tags, is a very efficient statistical part-of-speech tagger that is trainable on different languages and virtually

[7]. Urdu Parts of Speech (POS) Tagset - Center for Language ... www.cle.org.pk/.../UrduPOStagger/Urdu%20POS%20Tagset%200.3.pdf - CachedThe current Urdu Tagset consists of syntactic categories, and improves upon the earlier versions available (Muaz et al. (2009)1, Sajjad (2007)2, Sajjad and ...

[8] Central Institute of Indian Languages www.ciil.org/ - Cached - Similar CIIL Related Sites · Library · RTI · Tenders · To Reach Mysore ... Government of India Manasagangothri, Hunsur Road, Mysore 570006. Tel: +91-821-2515820 .

[9] Dandapat, Sudeshna Sarkar, Anupam BasuDepartment of Computer Science and Engineering Indian Institute of Technology Kharagpur India 721302 "Automatic Part-of-Speech Tagging for Bengali: An Approach for Morphologically Rich Languages in a Poor Resource Scenario"

[10]. Principles of Autonomy and Decision MakingLecture 21: Intro to Hidden Markov Modelsthe Baum-Welch algorithm.

