TagMiner: A Semisupervised Associative POS Tagger Effective for Resource Poor Languages

Pratibha Rani, Vikram Pudi, and Dipti Misra Sharma

International Institute of Information Technology, Hyderabad, India
pratibha_rani@research.iiit.ac.in,{vikram, dipti}@iiit.ac.in

Abstract. We present here, TagMiner, a data mining approach for part-of-speech (POS) tagging, an important Natural language processing (NLP) classification task. It is a semi-supervised associative classification method for POS tagging. Existing methods for building POS taggers require extensive domain and linguistic knowledge and resources. Our method uses combination of a small POS tagged corpus and a raw untagged text data as training data to build the classifier model using association rules. Our tagger works well with very little training data also. The use of semi-supervised learning provides the advantage of not requiring a large high quality tagged corpus. These properties make it especially suitable for resource poor languages. Our experiments on various resource-rich, resource-moderate and resource-poor languages show good performance without using any language specific linguistic information. We note that inclusion of such features in our method may further improve the performance. Results also show that for smaller training data sizes our tagger performs better than state-of-the-art CRF tagger using same features as our tagger.

Keywords: Part-of-Speech Tagging, Associative Classification, Association Rules, Semi-supervised Classification, NLP

1 Introduction

Part-of-speech (POS) tagging is an important NLP classification task that takes a word or a sentence as input, assigns a POS tag or other lexical class marker to a word or to each word in the sentence, and produces the tagged text as output. For this task several rule based [7, 8], stochastic supervised [6, 15, 30], and unsupervised [2, 5, 16] approaches are available for a number of languages. All of these approaches (including the state-of-the-art taggers) require training data and linguistic resources like dictionaries in *large* quantities. These taggers do not perform well for languages which do not have much resources and training data, referred to as *resource poor languages*.

The creation of linguistic resources is a time consuming expensive process which requires expert linguistic knowledge. So, there is a need to develop semi-supervised and generic POS tagging methods which take advantage of raw untagged corpus and require less or no lexical resources. A few such available

In: P. Cellier, T. Charnois, A. Hotho, S. Matwin, M.-F. Moens, Y. Toussaint (Eds.): Proceedings of DMNLP, Workshop at ECML/PKDD, Nancy, France, 2014.

techniques are mentioned in Sect. 2. In order to perform well, these techniques require a *large* raw untagged corpus. Unfortunately, for many resource poor languages, even obtaining this is hard.

This motivates us to explore data mining methods to build generic POS tagger. Data mining, being composed of data driven techniques, is a promising direction to explore or to develop language/domain independent POS tagging methods. However, direct application of data mining concepts for this task is not feasible and requires handling various challenges like 1) mapping POS tagging task to association rule mining problem, 2) developing semi-supervised methods to extract association rules from training set of *tagged* and *raw untagged* data combined and 3) handling challenges of POS tagging task (discussed in Sect. 4.2), like class imbalance, data sparsity and phrase boundary problems.

Associative classification [28] is a well known data mining based classification approach which uses association rules [1] to build the classifier model. In this work, we apply associative classification for POS tagging and present Tag-Miner, a generic semi-supervised method for POS tagging. Our method uses a combination of a *small* POS tagged corpus and a raw untagged text data as training data to build a classifier model using a new concept of *context based association rule mining.* These association rules work as context based tagging rules. Our Experiments demonstrate that it gives good performance even without using any linguistic resources—except for a small POS tagged corpus—for resource-rich English, resource-moderate Hindi and resource-poor Telugu, Tamil and Bengali languages.

Our method is generic in two aspects: (1) it does not use any language specific linguistic information such as morphological features and there is ample scope to improve further by including such features, (2) it does not require a large, high quality, tagged corpus and uses the POS tags of the tagged corpus only to calculate scores of "context based lists" which are used to form association rules. This can be easily adapted for various languages. Also, as an additional benefit model made by our tagger is human understandable since it is based on association rules.

Our algorithm has following advantages, especially suitable for resource poor languages, arising due to the use of raw untagged data: (1) it tags unknown words without using smoothing techniques, (2) the coverage of words present in the classifier model is increased which in turn increases tagging accuracy and (3) it creates additional linguistic resources from raw untagged data in the form of word clusters.

Remainder of this paper is as follows: Section 2 surveys related work. Section 3 formally presents the problem. Section 4, 5 and 6 present details of our proposed approach. Section 7 gives details of the datasets, various experiments and discusses the performance. Section 8 concludes our work.

2 Related Work

Associative classifiers use association rules to build a classifier model. They have been successfully applied for various classification tasks, for example, [34] presents an associative classifier for mammography image classification and [26] uses it for predictive analysis in health care data mining. Some of the associative classifiers worth mentioning are CBA [21] which integrates association rules and classification by finding class association rules, CMAR [20] uses concept of multiple class-association rules, CPAR [33] is based on predictive association rules and ACME [29] exploits maximum entropy principle. A good review of various associative classifiers and the detailed analysis of this method can be found in [28]. In some other association rule based approaches [18] uses association rules in a hybrid system of Naive Bayes and genetic classifier for text classification and [23] presents a supervised *language specific* hybrid algorithm of statistical method and association rule mining to increase the POS tagging accuracy of Chinese text. To the best of our knowledge no semi-supervised method exists for association rule mining from training set of tagged and raw untagged data combined.

For POS tagging, one of the first semi-supervised methods was proposed by [10] which uses raw untagged corpus by incorporating features obtained from a small fraction of untagged data along with features obtained from a large tagged data. A good overview of the existing semi-supervised POS tagging methods and discussion on their limitations is provided by [27], which uses graph as a smoothness regularizer to train CRFs [19] in a semi-supervised manner from a large untagged data and a small tagged data. In another approach [25] presents a condensed nearest neighbor method for semi-supervised POS tagging and report 97.5% accuracy on WSJ dataset of English. Most of the existing semi-supervised POS tagging methods use a combination of complex learning methods and existing supervised tagging methods to learn from large untagged data and moderate sized tagged data. All these methods have been developed for resource rich English and other European languages.

To the best of our knowledge no semi-supervised tagging method has been employed for resource moderate Hindi and resource poor Telugu and Tamil languages. Also to the best of our knowledge no fully data mining based generic POS tagger exists for any language. Baseline POS taggers for various languages are discussed below. We note that all the reported accuracy values were obtained for very small sized test sets. All the mentioned POS taggers use linguistic (especially morphological) knowledge in some or the other form, while our approach uses only the POS tags of the tagged set in an indirect form and learns from the raw untagged data.

For Hindi language, [22] proposes a CRF model with Transformation Based Learning (TBL) with morphological features and reports 78.67% accuracy on SPSAL corpus. [14] reports 92.36% accuracy on ISPC corpus using special linguistic features in a HMM model. [24] proposes an HMM model with morphological features and reports 93.05% accuracy. For Telugu language, [22] applies Transformation Based Learning (TBL) on top of a CRF model and reports 77.37% accuracy on SPSAL corpus. [14] uses various special linguistic features in a HMM model and reports 91.23% accuracy on ISPC corpus.

For Bengali language, [11] presents various supervised and semi-supervised Maximum Entropy and HMM models using morphological features and report 87.9% accuracy for semi-supervised HMM model on CIIL corpus. [13] reports 92.35% accuracy using a voted approach among various models. For Tamil language, [31] presents a linear programming based SVM model and reports 95.63% accuracy.

3 Problem Definition

Automated POS tagging is a classification task which takes a word or a sentence as input, assigns a POS tag or other lexical class marker to a word or to each word in the sentence, and produces the tagged text as output. In semi-supervised paradigm the POS tagger is built from a corpus of untagged sentences and a set of tagged sentences. The POS tagging classification problem is formally defined as follows:

Given a set of tags $\mathit{\Gamma} = \{T_1, T_2, \ldots, T_n\}$, an **annotated set** of tagged sentences $AS = \{St_1, St_2, \ldots St_N\}$, where $St_i = \langle W_1/T_i, W_2/T_j \ldots W_n/T_k\rangle$ (where W_i is a word and T_i is a tag from $\mathit{\Gamma}$) and a **raw untagged** training corpus of sentences $D = \{S_1, S_2 \ldots S_M\}$, where $S_i = \langle W_1 W_2 \ldots W_m\rangle$, the goal is to build a classifier model $\mathit{\Phi}$ which outputs the best tag sequence $\langle T_1 T_2 \ldots T_l\rangle$ for an input sequence of words $\langle W_1 W_2 \ldots W_l\rangle$.

4 TagMiner

4.1 Mapping POS tagging task to Association Rule Mining problem

According to the *one sense per collocation* [32] hypothesis, the sense of a word in a document is effectively determined by its *context*. The notion of context has been used in various methods of POS tagging [2, 30]. A context can occur in multiple places in the text. We refer to the list of occurrences of a context as a *context based list*. We use this idea for building TagMiner. In our method, we mine context based association rules from training data containing both tagged and untagged text. Our method works as follows:

– We collect all possible words occurring in the same context from the raw untagged data into a list called *context based list* (formally defined later). In this way we are able to find groups of words of *similar categories* from the raw untagged data.
– Using the annotated set and the tag finding algorithm (in Fig. 1), we find association rules of the form: $Context \Rightarrow Tag$ for the context based lists. Each rule maps a context based list to a suitable POS tag. These association rules work as the context based classification rules.
– Lastly, we group these context based association rules according to their POS tags to form clusters. This set of clusters is used as the classifier model to tag words using the method described in Sect. 6 and Fig. 2.

By experimenting with two varieties of bi-gram (one with preceding word and the other with succeeding word as context) and trigram as possible contexts we found that trigram works best for our method. For a word instance W_i, we fix its context as a trigram containing W_i in the middle and we use this context to find the *context based list*. Any other notion of context can be used as long as it fits into the formalism given below.

Context Based List: If Ψ is a function mapping from a word instance W_i in the data to its context $\Psi(W_i)$, then $\Psi^{-1}(\Psi(W_i))$ is a list of words instances sharing the same context. We refer to this list as *context based list* of $\Psi(W_i)$. It denotes words of similar category or type as W_i in a specific context and can store multiple instances of a word. For a given trigram $(W_{i-1}\ W_i\ W_{i+1})$ of words, $\Psi(W_i) = (W_{i-1}, W_{i+1})$. The preceding word W_{i-1} and succeeding word W_{i+1} are called *context words* and $\Psi(W_i)$ is called the *context word pair* of W_i.

Context Based Association Rule: For each context based list L, our approach finds association rule of the form $L \Rightarrow T$. This rule maps the context based list L to a POS tag T with *support* and *confidence* parameters defined below. Since each list L is obtained from a unique context word pair, so each association rule uniquely associates a context to a POS tag and works as the context based tagging rule.

In the following definitions and formulas we develop the intuition and the method to compute the interestingness measures of the significant association rules. The complexity in defining support is due to the presence of raw untagged training data required for semi-supervised learning. The support is the frequency (count) of occurrences of the context in the dataset. Context based lists are made from raw untagged data D and we are interested in the words of this list for which we know the tag in annotated set AS. Hence, we define Support of a context as follows:

AllTagContextSupport: Number of unique words of a context based list L whose tags are available (in annotated set AS) is denoted as $AllTagContextSupport(L)$. This measure gives the number of tagged words of L.

ContextSupport: For a list of words L in which duplicates may be present, $ContextSupport(L)$ is defined as the set of unique words present in L.

Coverage: For a *context based list* L,

$$Coverage(L) = \frac{AllTagContextSupport(L)}{|ContextSupport(L)|} \tag{1}$$

This measure represents the confidence that enough number of tagged samples are present in L.

ContextTagSupport: Number of unique words of a *context based list* L present in annotated set AS with a particular tag T is denoted as $ContextTagSupport(L, T)$.

Confidence: For a *context based list* L and tag T,

$$Confidence(L, T) = \frac{ContextTagSupport(L, T)}{|ContextSupport(L)|} \tag{2}$$

This measure represents the confidence that considerable number of words in list L have a particular tag T and leads to rules of the form $Context \Rightarrow Tag$.

WordTagSupport: Frequency of tag T for a word W in the annotated set AS is denoted as $WordTagSupport(T, W)$.
WordTagScore: For a word W and tag T, $WordTagScore$ is defined as:

$$WordTagScore(W, T) = \frac{WordTagSupport(T, W)}{\max\limits_{T_i \in \Gamma} WordTagSupport(T_i, W)} \tag{3}$$

This represents how good the tag fits the word on a scale of 0 to 1.
ListTagScore: For a tag T in *context based list* L, $ListTagScore$ is defined as:

$$ListTagScore(L, T) = \frac{\sum\limits_{W_i \in ContextSupport(L)} WordTagScore(W_i, T)}{|\{W_i \in ContextSupport(L) : W_i/T \in AS\}|} \tag{4}$$

Where, **AS** is the annotated set. This formula represents the average frequency of tag T in *context based list* L. Intuitively, it represents how good the tag fits the list. Unfortunately, this is not always indicative of the correct tag for the list. For example, if a tag is overall very frequent, it can bias this score. Therefore, we compare this with the following score, inspired by the notion of *Conviction* [9].
BackgroundTagScore: For a tag T in annotated set AS, $BackgroundTagScore$ is defined as:

$$BackgroundTagScore(T) = \frac{\sum\limits_{W_i \in ContextSupport(AS)} WordTagScore(W_i, T)}{|\{W_i \in ContextSupport(AS) : W_i/T \in AS\}|} \tag{5}$$

This represents the average frequency of tag T in annotated set AS.

4.2 POS Tagging Challenges

POS tagging, especially for resource poor languages, involves three major challenges listed below. In our approach we handle each of them explicitly.

1. **Data sparsity problem**: Some POS tag classes are present in the annotated set with very few representations. This is not enough to derive statistical information about them. In our approach, the use of raw untagged data reduces this problem (shown in Sect. 7.4).
2. **Class imbalance problem**: POS tag classes are highly imbalanced in their occurrence frequency. While selecting a tag this may lead to biasing towards the most frequent tags. Existing solutions of class imbalance problem typically favor rare classes [12]. However, while tagging the *context based lists*, we need to find POS tags for them in such a way that we neither favor frequent tags nor rare tags. We tackle this problem using a novel *Minmax* approach to find the best preferred POS tag instead of the most frequent one (described in Sect. 5.2).
3. **Phrase boundary problem**: Some lists are formed at phrase boundaries where the context comes from two different phrases. We need to filter out those *context based lists* which do not contain words of similar categories. In this case, the context of a word instance need not represent strong context and so the context based list may contain unrelated words. We use suitable parameters to handle this problem (explained in Sect. 5.3).

```
1. for each tag $T_i \in \Gamma$ present in annotated set $AS$ do:
2.     Find $BackgroundTagScore(T_i)$    // Use Equation (5)
3. for context based list $L$ do:
4.     Find $Coverage(L)$    // Use Equation (1)
5.     if $Coverage(L) \geq MinCoverage$:
6.         $ContextTagSupport(L, T_{max}) = \max_{T_i \in \Gamma} ContextTagSupport(L, T_i)$
7.         $Maxconf = Confidence(L, T_{max})$    // Use Equation (2)
8.         if $Maxconf > MinConfidence$:
9.             $MaxTset = \{T_i \mid ContextTagSupport(L, T_i) == ContextTagSupport(L, T_{max})\}$
10.            $BestPrefTag = FindBestPrefTag(L, MaxTset)$
11.            Return $BestPrefTag$
12.        else: Return NOTVALIST
13.    else: Return NOTVALIST

14. $FindBestPrefTag(L, MaxTset)$:
15.     Initialize $PrefTagset = \{\}$
16.     for each word $W$ of $ContextSupport(L)$ present in $AS$ do:
17.         $Tagset(W) = \{T_i \mid W \text{ has tag } T_i \text{ in } AS\}$
18.         $UnqTagset = Tagset(W) \cap MaxTset$
19.         Find $MaxWTag \mid WordTagSupport(MaxWTag, W) == \max_{T_j \in UnqTagset} WordTagSupport(T_j, W)$
20.         $PrefTagset = PrefTagset \cup MaxWTag$
21.     Find $MinTag \in PrefTagset \mid \exists W_{min} \in ContextSupport(L)$ with $WordTagSupport(MinTag, W_{min}) == \min_{W_i \in ContextSupport(L)} WordTagSupport(MinTag, W_i)$
22.     Find $ListTagScore(L, MinTag)$    // Use Equation (4)
23.     if $ListTagScore(L, MinTag) \geq BackgroundTagScore(MinTag)$: Return $MinTag$
24.     else: Return NOTVALIST
```

Figure 1: Algorithm to find POS tag for a *context based list.*

5 Building Classifier Model from Context Based Lists

5.1 Finding Association Rule for a Context Based List

The first step in our classifier model building method is to compute *context based lists* from an untagged training corpus D. It may be noted that a context based list can store multiple instances of a word. We use a sliding window of size three to collect the context based lists from D, in a single iteration, taking care of sentence boundaries.

In the next step we use the algorithm shown in Fig. 1 to find association rules for all the context based lists. In this algorithm, $BackgroundTagScore$ of all the POS tags present in the annotated set AS (lines 1-2) are computed first. Then for a context based list satisfying the threshold values of $Coverage$ and $Confidence$ (lines 3-9), function $FindBestPrefTag$ (described in Sect. 5.2) finds the best preferred tag (lines 10-11, 14-24) from the set of tags with maximum $ContextTagSupport$ (lines 7-9).

For a context based list L present as antecedent in association rule $L \Rightarrow T$, tag T returned by this algorithm becomes the consequent. This algorithm outputs best preferred tags for all the context based lists and hence finds association rules for all of them.

5.2 Handling Class Imbalance Problem

We handle the *class imbalance problem* by using a novel *Minmax* approach in the function $FindBestPrefTag$ (lines 14-24 in Fig. 1) and parameters $ListTagScore$ and *BackgroundTagScore*. In *Minmax* approach the preferred tag T_i for *context based list* L, is the one which has maximum $ContextTagSupport(L, T_i)$ but minimum $WordTagSupport(T_i, W)$ among those words of list L which have tag T_i as the best tag in AS. This takes care that the selected tag is supported by majority of the words in the list and is not biased by the most frequent tag of the annotated set.

To find the best preferred tag in function $FindBestPrefTag$, from the set of all the tags with maximum *ContextTagSupport* value (line 9), at first we found those tags which were best tags (having maximum $WordTagSupport$ value) for the words of list L in AS (lines 15-20). Next, from this set of preferred tags we find the tag with minimum $WordTagSupport$ value (line 21). Then criteria $ListTagScore(L, T_i) \geq BackgroundTagScore(T_i)$ (lines 22-23) ensures that the selected tag has above average support in the annotated set and the context based list, both. If none of the tags satisfy this criteria, then we tag the list as "NOTVALIST" (line 24).

5.3 Handling Phrase Boundary Problem

To filter out context based lists with the *phrase boundary problem* (see Sect. 4.2) we use two suitable threshold values for parameters *Confidence* and *Coverage*. *Coverage* takes care of the fact that a context based list has considerable number of words to map it to a tag and *Confidence* ensures that the tag found for the list is the one which is supported by majority of the words in the list.

If context based list L has *Coverage* and *Confidence* values less than the corresponding threshold values $MinCoverage$ and $MinConfidence$, we tag L as "NOTVALIST" (lines 3-8, 12, 13 in Fig. 1). If L satisfies both of the threshold values then only we find the set of all the tags which have maximum value of $ContextTagSupport(L, T_i)$ and use this set (lines 9-10) to find the best preferred tag for the list (lines 14-24).

5.4 POS tag wise grouping of Association Rules to form Clusters

In the last step, we group context based lists according to their POS tags to get clusters of context based lists as classifier model. We exclude context based lists with tag "NOTVALIST" from the grouping process. Then we process these clusters to store word frequencies, corresponding context word pairs and their frequencies in each cluster. We represent the set of clusters as *Clustset*.

Since we are highly confident about the tags of the words present in the annotated set AS so, to improve cluster quality we apply a pruning strategy on the words of the clusters present in AS and remove those words from each cluster which do not have a matching cluster tag in AS. Finally, we get a set of clusters in which each cluster has a set of words with their frequencies and a set of associated context word pairs with their frequencies. Each cluster has a unique POS tag. These clusters are overlapping in nature and words can belong to multiple clusters.

6 POS tagging Method

To tag the words of a test sentence we make use of the test word's context word pair, preceding word and the word frequency in a cluster to decide the tag of the word (see Fig. 2). When a test word is found in only one cluster then we output the cluster tag as the tag of the test word. But when a test word is found in many clusters, then to select the suitable clusters following priority order is followed:

1. **Criteria 1:** Highest priority is given to the presence of matching context word pair of the test word in the clusters.
2. **Criteria 2:** Second highest priority is given to the presence of matching preceding word of the test word as first word of the context word pairs in clusters.
3. **Criteria 3:** Last priority is given to the frequency of the test word in the clusters.

For test words not present in any cluster we use criterion 1 and 2 to select appropriate clusters. Based on the priority order, only one of the criterion is used to select the suitable clusters. If we are not able to find any suitable cluster then we return "NOTAG" as the tag of the test word.

Even when we find suitable clusters, to increase precision, our method finds POS tags only for those cases where it is confident. It avoids to wrongly classify non confident cases and returns "NOTAG" for them. This is especially useful when the cost of misclassifying (false positive) is high. This also gives opportunity to integrate other language/domain specific POS taggers as they can be used for the non-confident cases.

After selecting the suitable clusters we need to make sure that we have enough confidence in the highest probability tag obtained from the clusters. To ensure this we use the parameter *TagProbDif*, which gives the fractional difference between the highest and the second highest cluster tag probabilities and is defined as follows:

$$TagProbDif = \frac{TagProb(C_{max}) - TagProb(C_{secmax})}{TagProb(C_{max})} \tag{6}$$

Where, C_{max} is the cluster with highest $TagProb(C_i)$ value and C_{secmax} is the cluster with second highest $TagProb(C_i)$ value. $TagProb(C_i)$ of a cluster is defined as follows:

$$TagProb(C_i) = \frac{\text{Frequency of X in } C_i}{\sum\limits_{\forall C_j \in Clustset} \text{Frequency of X in } C_j} \tag{7}$$

Where, X is set as follows: If the test word is present in cluster C_i then X = test word. For test word not present in any cluster, if the clusters are selected based on the presence of the context word pair of the test word then X = context

```
for each word $Wmid$ in sentence $S$ with context word pair $CW_p$ and $CW_s$ do:

 1. Initialize $PredClustset = \{\}$
 2. if $\exists$ cluster $C_i \in Clustset \mid Wmid \in C_i$:
    (a) Find $PClustset = \{C_i \mid Wmid \in C_i\}$
    (b) if $\exists$ cluster $C_j \in PClustset \mid CW_p$ and $CW_s$ pair is present as context word pair in
        cluster $C_j$:
            Find all such clusters from $PClustset$ and append to $PredClustset$   #Criteria 1
    (c) else:
          if $\exists$ cluster $C_j \in PClustset \mid CW_p$ is present as preceding word in a context word
             pair in cluster $C_j$:
               Find all such clusters from $PClustset$ and append to $PredClustset$   #Crit. 2
          else: Append $PredClustset = PredClustset \cup PClustset$   #Criteria 3
 3. else:
    (a) if $\exists$ cluster $C_i \in Clustset \mid CW_p$ and $CW_s$ pair is present as context word pair in
        cluster $C_i$:
            Find all such clusters from $Clustset$ and append to $PredClustset$   #Criteria 1
    (b) else:
          if $\exists$ cluster $C_i \in Clustset \mid CW_p$ is present as preceding word in a context word
             pair in cluster $C_i$:
               Find all such clusters from $Clustset$ and append to $PredClustset$   #Crit. 2
          else: Return NOTAG
 4. $\forall C_i \in PredClustset$ Find $TagProb(C_i)$   // Use Equation 7
 5. Find $C_{max} =$ cluster with highest $TagProb(C_i)$ value in $PredClustset$
 6. Find $C_{secmax} =$ cluster with second highest $TagProb(C_j)$ value in $PredClustset$
 7. Find $TagProbDif$   // Use Equation 6
 8. if $TagProbDif \geq Minprobdif$: Return $PredTag =$ POS tag label of cluster $C_{max}$
 9. else: Return NOTAG
```

Figure 2: Method to tag words of a sentence using set of clusters *Clustset*.

word pair. If the clusters are selected based on the presence of the preceding word of the test word as first word of the context word pairs in clusters then X = preceding word of the test word. In this way we are able to tag some *unseen*/*unknown* words also which are not present in the training data. This, in a way, acts as an alternative of smoothing technique for them.

After selecting the clusters (based on priority order) we compute their $TagProb$ values using (7) and then compute $TagProbDif$ using (6). For $TagProbDif$ value above a suitable threshold value $Minprobdif$ we output the tag of cluster with highest $TagProb$ value as the tag of the test word, otherwise we return "NOTAG"(see Fig. 2).

7 Experiments, Results and Observations

7.1 Dataset Details

We have done our experiments on resource-rich English[1](uses Biber tag set [17]), resource-moderate Hindi [3, 4] and resource-poor Telugu[2] [3], Tamil[3] and Ben-

[1] New York Times dataset of American National Corpus available at `http://americannationalcorpus.org/FirstRelease/contents.html`

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

[3] Available at `http://sanskrit.jnu.ac.in/ilci/index.jsp`

gali[4] languages. Table 1 gives details of all the language datasets. All the five language datasets have flat tag sets present in annotated training and test sets without any hierarchy and considerable number of lexical ambiguities are also present. We note that except English all the other four languages are morphologically rich and have free word-order property. The POS tag data distribution in the resource-moderate and resource-poor language datasets are highly imbalanced and sparse.

Table 1. Details of all language datasets with accuracy values obtained by **TagMiner**.

	Hindi	**Telugu**	**Tamil**	**Bengali**	**English**
No. of Words in Raw Untagged Training set	393303	104281	169705	85796	1293388
No. of Words in Annotated Training set	282548	83442	20207	21561	629532
No. of POS Tags in Annotated Training set	35	28	28	27	109
No. of Words in Test set	70811	20854	22352	20618	471977
No. of POS Tags in Test set	32	24	27	29	105
No. of Test Words tagged as NOTAG by TagMiner	1916	1634	2647	3448	9385
Average Accuracy (%) (Equation 8)	**87.8**	**87.6**	**83.46**	**76.17**	**88.5**
Resource Type	Moderate	Poor	Poor	Poor	Rich

7.2 Performance Analysis and Observations

We observed that following set of threshold values $MinConfidence = 60\%$, $MinCoverage = 60\%$ and $MinprobDif = 30\%$ for the three parameters gives best *AverageAcuracy* (defined below) values for all the five languages. Tables 1 and 2 show the results for this set of parameter values.

$$AverageAccuracy = \frac{\text{Number of correctly tagged test words}}{|\text{Test set}| - \text{No. of test words tagged as NOTAG}} \tag{8}$$

Where, $|\text{Test set}| =$ No. of words in the test set.

For both known and unknown test words, for all the five languages, maximum number of correct tagging was done by giving highest priority to presence of context word pair in the cluster. Here, *known* words means test set words which are present in untagged training set and *unknown* word means unseen test set words which are not present in the untagged training set. Note that words of annotated set are not included in the classifier model, only their tags

[4] Available at `http://sanskrit.jnu.ac.in/ilci/index.jsp`

are used indirectly while building the model. In the results shown in Table 1, around 46% unknown English words, 60% unknown Hindi words, 67% unknown Telugu words, 52% unknown Bengali words and 57% unknown Tamil words were correctly tagged using their context word pair. This shows the strength of our tagger to tag unknown words without using any smoothing technique used by other POS taggers.

In Table 2, we compare our results with a supervised CRF[5] tagger [19]. This tagger uses words, their POS tag and context word pair information from annotated data, while our tagger uses words and their context word pair information from untagged data and POS tag information from annotated data. We observe that for annotated data size $\leq 25K$ words, our tagger gives better *AverageAccuracy* than CRF tagger. Our tagger also gives better POS tag precisions and better tagging accuracies than CRF tagger for unknown words and performance improves by increasing the untagged data size up to a certain size. This shows that our tagger can be a better choice for resource poor languages. Also, as an additional benefit model made by our tagger is more human understandable than model made by CRF tagger.

7.3 Effect of Annotated (POS tagged) Data Size

We varied the size of annotated set of Tamil (see Table 3) while keeping the raw untagged set constant and observed that the coverage of words by the clusters in the classifier model increases with the increase in the size of annotated data, the tagging accuracy increases while the number of words missed by the model (tagged as “NOTAG”) decreases. For all languages we observed that increasing the annotated training data size improves cluster quality which increases the *AverageAcuracy* values but only up to a certain size. We also observed that there is only a slight decrease in *AverageAcuracy* value with decrease in annotated set size, so performance does not decrease drastically when the annotated set is made smaller. Our tagger gives above 70% *AverageAcuracy* for annotated data size as low as 5K and raw untagged data size 10K on all the languages. This justifies the use of small annotated set to build a semi-supervised POS tagging model for resource poor languages.

7.4 Effect of Raw Untagged Data Size

In Tables 1, 2 and 4 , we observe that increasing the raw untagged training data size initially increases word coverage of clusters which in turn increases the *AverageAcuracy* values but stabilizes after a certain size. For all languages we observed that the coverage of words by the clusters in the classifier model increases with the increase in the size of untagged data (while keeping the size of annotated set constant). This accounts for the increase in tagging accuracy and decrease in the number of words missed by the model (tagged as NOTAG). Other interesting observation is that *AverageAccuracy* does not vary much as

[5] Available at `http://crfpp.googlecode.com/svn/trunk/doc/index.html`, CRF model outputs tag for all test words. So, for CRF tagger AverageAccuracy = (No. of correctly tagged test words)/(No. of test words).

Table 2. Average Accuracy values for all languages obtained by CRF tagger and our tagger **TagMiner** for various annotated training set sizes (≤ 25000 words).

Lang.	**Test set size**	**Annotated Training set size**	**CRF**	**TagMiner**		
			Average Accuracy (%)	**Average Accuracy (%)**	**No. of NOTAG Words**	**Untagged Training set size**
Hindi	20227	5730	74.6	**79.1**	3195	10025
		10030	78.4	**79.05**	2740	10025
		15771	81.3	**82.1**	2116	25020
		25591	84.7	**85.0**	1903	50019
Telugu	20854	4994	59.4	**81.4**	5617	9994
		9994	67.1	**82.6**	3897	23435
		14995	71.2	**84.4**	3240	43434
		23435	75.3	**84.3**	2419	104281
Tamil	22352	5006	48.9	**75.0**	6957	40988
		9941	59.9	**79.7**	4357	80004
		15007	65.9	**82.1**	3778	80004
		20207	69.4	**83.1**	3495	80004
Bengali	20618	5010	47.3	**73.5**	7081	49997
		10003	56.2	**74.6**	5332	85796
		15009	59.3	**75.2**	4269	85796
		21561	63.0	**77.8**	4170	85796
English	24952	10671	70.4	**79.7**	3774	50444
		15298	72.9	**82.3**	3424	50444
		24825	76.4	**82.5**	2574	93679

Table 3. Effect of annotated data size on classifier model on 22352 Tamil test set words for $MinCoverage = 60\%$, $MinConfidence = 60\%$, $Minprobdif = 30\%$ and 169705 raw untagged words.

No. of Words in Annotated set	**No. of Clusters in model (No. of unique words)**	**No. of NOTAG test Words**	**Average Accuracy (%)**
5006	22 (2021)	5317	72.2
9941	25 (3553)	3575	79.0
15007	26 (4842)	2940	82.09
20207	26 (5774)	2647	83.46

the untagged data size varies, so our algorithm is able to perform well even with a small sized untagged data.

7.5 Effect of Various Parameters

We made the following observations about the effect of parameter values: (1) Increasing threshold values of $MinConfidence$ for parameter *Confidence*, it in-

Table 4. Effect of raw untagged data size on classifier model on 70811 Hindi test set words and 20854 Telugu test set words for $MinCoverage = 60\%$, $MinConfidence = 60\%$ and $Minprobdif = 30\%$ with 282548 annotated Hindi words and 104281 annotated Telugu words.

Language	**No. of Words in Raw set**	**No. of Clusters in model (No. of unique words)**	**No. of NOTAG test Words**	**Average Accuracy (%)**
Hindi	**50019**	25 (4366)	4714	**87.3**
	98331	28 (6081)	3664	87.7
	128329	28 (6865)	3220	87.9
	158337	29 (7546)	2890	87.9
	188326	29 (8112)	2793	88.0
	196659	29 (8260)	2748	88.0
	282554	30 (9517)	2484	88.0
	294979	30 (9663)	2450	**88.1**
	393303	30 (10817)	1916	87.8
Telugu	**23435**	23 (3600)	3079	**86.1**
	43434	23 (5091)	2276	87.4
	63436	23 (6221)	1828	87.4
	83442	23 (7198)	1749	**88.2**

creases the quality of clusters but at the same time it also increases the number of *context based lists* tagged as "NOTVALIST" which decreases the word coverage of clusters. (2) Decreasing threshold values of $MinCoverage$ for parameters *Coverage* although decreases the quality of clusters but at the same time it increases the word coverage of clusters by decreasing the number of *context based lists* tagged as "NOTVALIST". (3) By varying the threshold value of *Minprobdif* from 5% to 30% for parameter *TagProbDif* we found that increasing the threshold value increases the precision values of POS tags but slightly decreases their recall because the number of words tagged as "NOTAG" increases. Practical advantage of this parameter is that it ensures that tagging of ambiguous and non-confident cases is avoided. (4) The number of POS tag clusters obtained in the classifier model is almost independent of the selected threshold values of the parameters. For the datasets given in Table 1 and for the range of threshold values $MinConfidence = 60\%$ to 90% and $MinCoverage = 0\%$ to 75%, number of POS tag clusters found for English was 100 to 101, for Hindi was 29 to 31, for Tamil was 22 to 26, for Bengali was 25 and for Telugu was 23. We noted that the POS tags missing from the set of clusters were the rare POS tags having very low frequencies.

8 Conclusions and Future Work

In this work we developed TagMiner, a semi-supervised associative classification method for POS tagging. We used the concept of context based list and context based association rule mining. We developed a method to find interestingness measures required to find the association rules in a semi-supervised manner from a training set of tagged and raw untagged data combined. We showed that TagMiner gives good performance for resource rich as well as resource poor languages without using extensive linguistic knowledge. It works well even with less tagged training data and less untagged training data. It can also tag unknown words. To some extent, it handles class imbalance and data sparsity problems using the untagged data and a special method to find interestingness measures. It handles phrase boundary problem using a set of parameters. These advantages make it very suitable for resource poor languages and can be used as an initial POS tagger while developing linguistic resources for them.

Future work includes (1) using other contexts instead of trigram, (2) finding methods to include linguistic features in the current approach, (3) mining tagging patterns from the clusters to find tag of a test word and (4) using this approach for other lexical item classification tasks.

References

1. Agrawal, R., Imieliński, T., Swami, A.: Mining Association Rules Between Sets of Items in Large Databases. In: Proc. of SIGMOD. pp. 207–216 (1993)
2. Banko, M., Moore, R.C.: Part-of-Speech Tagging in Context. In: Proc. of COLING (2004)
3. Bharati, A., Misra Sharma, D., Bai, L., Sangal, R.: AnnCorra : Annotating Corpora Guidelines For POS And Chunk Annotation For Indian Languages. Tech. Rep. TR-LTRC-31, Language Technologies Research Centre, IIIT, Hyderabad (2006)
4. Bhatt, R., Narasimhan, B., Palmer, M., Rambow, O., Sharma, D.M., Xia, F.: A multi-representational and multi-layered treebank for Hindi/Urdu. In: Proc. of the Third Linguistic Annotation Workshop. pp. 186–189 (2009)
5. Biemann, C.: Unsupervised Part-of-Speech Tagging Employing Efficient Graph Clustering. In: Proc. of ACL (2006)
6. Brants, T.: TnT: a statistical part-of-speech tagger. In: Proc. of ANLP. pp. 224–231 (2000)
7. Brill, E.: A Simple Rule-Based Part of Speech Tagger. In: Proc. of ANLP. pp. 152–155 (1992)
8. Brill, E.: Transformation-Based Error-Driven Learning and Natural Language Processing: A Case Study in Part-of-Speech Tagging. Comput. Linguist. 21(4), 543–565 (1995)
9. Brin, S., Motwani, R., Ullman, J.D., Tsur, S.: Dynamic Itemset Counting and Implication Rules for Market Basket Data. In: Proc. of SIGMOD. pp. 255–264 (1997)
10. Cutting, D., Kupiec, J., Pedersen, J., Sibun, P.: A practical part-of-speech tagger. In: Proc. of the third conference on ANLP. pp. 133–140 (1992)
11. Dandapat, S., Sarkar, S., Basu, A.: Automatic Part-of-speech Tagging for Bengali: An Approach for Morphologically Rich Languages in a Poor Resource Scenario. In: Proc. of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions. pp. 221–224 (2007)
12. Dubey, H., Pudi, V.: Class Based Weighted K-Nearest Neighbor over Imbalance Dataset. In: Proc. of PAKDD (2). pp. 305–316 (2013)

13. Ekbal, A., Hasanuzzaman, M., Bandyopadhyay, S.: Voted Approach for Part of Speech Tagging in Bengali. In: Proc. of PACLIC. pp. 120–129 (2009)
14. Gadde, P., Yeleti, M.V.: Improving statistical POS tagging using Linguistic feature for Hindi and Telugu. In: Proc. of ICON (2008)
15. Gimenez, J., Marquez, L.: Svmtool: A general pos tagger generator based on support vector machines. In: Proc. of LREC. pp. 43–46 (2004)
16. Goldwater, S., Griffiths, T.: A fully Bayesian approach to unsupervised part-of-speech tagging. In: Proc. of the 45th Annual Meeting of the ACL. pp. 744–751 (June 2007)
17. Ide, N., Suderman, K.: The American National Corpus First Release. In: Proc. of LREC. pp. 1681–1684 (2004)
18. Kamruzzaman, S.M., Haider, F., Hasan, A.R.: Text Classification using Association Rule with a Hybrid Concept of Naive Bayes Classifier and Genetic Algorithm. CoRR abs/1009.4976 (2010)
19. Lafferty, J.D., McCallum, A., Pereira, F.C.N.: Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. In: Proc. of ICML. pp. 282–289 (2001)
20. Li, W., Han, J., Pei, J.: CMAR: Accurate and Efficient Classification Based on Multiple Class-Association Rules. In: Proc. of ICDM. pp. 369–376 (2001)
21. Liu, B., Hsu, W., Ma, Y.: Integrating Classification and Association Rule Mining. In: Proc. of KDD. pp. 80–86 (1998)
22. P.V.S., A., G., K.: Part Of Speech Tagging Using Conditional Random Fields and Transformation Based Learning. In: Proc. of IJCAI Workshop SPSAL. pp. 21–24 (2007)
23. Shaohong, Y., Guidan, F.: Research of POS Tagging Rules Mining Algorithm. Applied Mechanics and Materials 347-350, 2836–2840 (August 2013)
24. Shrivastava, M., Bhattacharyya, P.: Hindi POS Tagger Using Naive Stemming: Harnessing Morphological Information Without Extensive Linguistic Knowledge. In: Proc. of ICON (2008)
25. Søgaard, A.: Semisupervised condensed nearest neighbor for part-of-speech tagging. In: Proc. of ACL HLT: short papers - Volume 2. pp. 48–52 (2011)
26. Soni, S., Vyas, O.: Using Associative Classifiers For Predictive Analysis In Health Care Data Mining. Int. Journal Of Computer Application 4(5), 33–37 (July 2010)
27. Subramanya, A., Petrov, S., Pereira, F.: Efficient graph-based semi-supervised learning of structured tagging models. In: Proc. of EMNLP. pp. 167–176 (2010)
28. Thabtah, F.: A Review of Associative Classification Mining. The Knowledge Engineering Review 22(1), 37–65 (Mar 2007)
29. Thonangi, R., Pudi, V.: ACME: An Associative Classifier Based on Maximum Entropy Principle. In: Proc. of ALT. pp. 122–134 (2005)
30. Toutanova, K., Klein, D., Manning, C.D., Singer, Y.: Feature-rich part-of-speech tagging with a cyclic dependency network. In: Proc. of NAACL HLT'03 - Volume 1. pp. 173–180 (2003)
31. V., D., Kumar, A., G., S., P., S.K., S., R.: Tamil POS Tagging using Linear Programming. Int. Journal of Recent Trends in Engineering 1(2), 166–169 (May 2009)
32. Yarowsky, D.: One sense per collocation. In: Proc. of the workshop on Human Language Technology. pp. 266–271 (1993)
33. Yin, X., Han, J.: CPAR: Classification based on Predictive Association Rules. In: Proc. of SDM. pp. 331–335 (2003)
34. Zaïane, O.R., Antonie, M.L., Coman, A.: Mammography Classification By an Association Rule-based Classifier. In: Proc. of MDM/KDD. pp. 62–69 (2002)