Unsupervised Part-of-Speech Acquisition for Resource-Scarce Languages

Sajib Dasgupta and **Vincent Ng**
Human Language Technology Research Institute
University of Texas at Dallas
Richardson, TX 75083-0688
{sajib,vince}@hlt.utdallas.edu

Abstract

This paper proposes a new bootstrapping approach to unsupervised part-of-speech induction. In comparison to previous bootstrapping algorithms developed for this problem, our approach aims to improve the quality of the seed clusters by employing seed words that are both distributionally and morphologically reliable. In particular, we present a novel method for combining morphological and distributional information for seed selection. Experimental results demonstrate that our approach works well for English and Bengali, thus providing suggestive evidence that it is applicable to both morphologically impoverished languages and highly inflectional languages.

1 Introduction

The availability of a high-quality lexicon is crucial to the development of fundamental text-processing components such as part-of-speech (POS) taggers and syntactic parsers. While hand-crafted lexicons are readily available for resource-rich languages such as English, the same is not true for *resource-scarce* languages. Unfortunately, manually constructing a lexicon requires a lot of linguistic expertise, and is practically infeasible for highly inflectional and agglutinative languages, which contain a very large number of lexical items. Given the scarcity of annotated data for acquiring the lexicon in a supervised manner, researchers have instead investigated *unsupervised* POS induction techniques for automating the lexicon construction process. In essence, the goal of unsupervised POS induction is to learn the set of possible POS tags for each lexical item from an unannotated corpus.

The most common approach to unsupervised POS induction to date has been motivated by Harris's (1954) distributional hypothesis: words with similar co-occurrence patterns should have similar syntactic behavior. More specifically, unsupervised POS induction algorithms typically operate by (1) representing each *target* word (i.e., a word to be tagged with its POS) as a *context* vector that encodes its left and right context, (2) clustering distributionally similar words, and (3) manually labeling each cluster with a POS tag by inspecting the members of the cluster.

This *distributional* approach works under the assumption that the context vector of each word encodes sufficient information for enabling accurate word clustering. However, many words are *distributionally unreliable*: due to data sparseness, they occur infrequently and hence their context vectors do not capture reliable statistical information. To overcome this problem, Clark (2000) proposes a bootstrapping approach, in which he (1) clusters the most distributionally reliable words, and then (2) incrementally augments each cluster with words that are distributionally similar to those already in the cluster.

The goal of this paper is to propose a new bootstrapping approach to unsupervised POS induction that can operate in a resource-scarce setting. Most notably, our approach aims to improve the quality of the seed clusters by employing seed words that are both *distributionally* and *morphologically* reliable. In particular, we present a novel method for combining morphological and distributional information for seed selection. Furthermore, given our

Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, pp. 218–227, Prague, June 2007.

emphasis on resource-scarce languages, our approach does not rely on any language resources. In particular, the morphological information that it exploits is provided by an unsupervised morphological analyzer.

It is perhaps not immediately clear why morphological information would play a crucial role in the induction process, especially since the distributional approach has achieved considerable success for English POS induction (see Lamb (1961), Schütze (1995) and Clark (2000)). To understand the role and significance of morphology, it is important to first understand why the distributional approach works well for English. Recall from the above that the distributional approach assumes that the information encoded in the context vector of each word, which typically consists of the 250 most frequent words of a given language, is sufficient for accurately clustering the words. This approach works well for English because the most frequent English words are composed primarily of closed-class words such as "to" and "is", which provide strong clues to the POS of the target word. However, this assumption is not necessarily valid for fairly free word order and highly inflectional languages such as Bengali. The reason is that (1) co-occurrence statistics collected from free word order languages are not as reliable as those from fixed word order languages; and (2) many of the closed-class words that appear in the context vector for English words are realized as inflections in Bengali. The absence of these highly informative words implies that the context vectors may no longer capture sufficient information for accurately clustering Bengali words, and hence the use of morphological information becomes particularly important for unsupervised POS induction for these inflectional languages.

We will focus primarily on labeling *open-class* words with their POS tags. Our decision is motivated by the fact that closed-class words generally comprise a small percentage of the lexical items of a language. In Bengali, the percentage of closed-class words is even smaller than that in English: as mentioned before, many closed-class words in English are realized as suffixes in Bengali.

Although our attempt to incorporate morphological information into the distributional POS induction framework was originally motivated by inflectional languages, experimental results show that our approach works well for both English and Bengali, suggesting its applicability to both morphologically impoverished languages and highly inflectional languages. Owing to the lack of publicly available resources for Bengali, we manually created a 5000-word Bengali lexicon for evaluation purposes. Hence, one contribution of our work lies in the creation of an annotated dataset for Bengali. By making this dataset publicly available[1], we hope to facilitate the comparison of different unsupervised POS induction algorithms and to stimulate interest in Bengali language processing.

The rest of the paper is organized as follows. Section 2 discusses related work on unsupervised POS induction. Section 3 describes our tagsets for English and Bengali. The next three sections describe the three steps of our bootstrapping approach: cluster the words using morphological information (Section 4), remove potentially mislabeled words from each cluster (Section 5), and bootstrap each cluster using a weakly supervised learner (Section 6). Finally, we present evaluation results in Section 7 and conclusions in Section 8.

2 Related Work

Several unsupervised POS induction algorithms have also attempted to incorporate morphological information into the distributional framework, but our work differs from these in two respects.

Computing morphological information. Previous POS induction algorithms have attempted to derive morphological information from dictionaries (Hajič, 2000) and knowledge-based morphological analyzers (Duh and Kirchhoff, 2006). However, these resources are generally not available for resource-scarce languages. Consequently, researchers have attempted to derive morphological information heuristically (e.g., Cucerzan and Yarowsky (2000), Clark (2003), Freitag (2004)). For instance, Cucerzan and Yarowsky (2000) posit a character sequence *x* as a suffix if there exists a sufficient number of distinct words *w* in the vocabulary such that the concatentations *wx* are also in the vocabulary. It is conceivable that such heuristically computed morphological information can be inaccurate, thus rendering the usefulness of a more accurate morphological analyzer. To address this problem, we exploit morphological information provided by an unsupervised word segmentation algorithm.

[1] See http://www.utdallas.edu/~sajib/posDatasets.html.

Tag	Description	Treebank tags
JJ	Adjective	JJ
JJR	Adjective, comparative	JJR
JJS	Adjective, superlative	JJS
NN	Singular noun	NN, NNP
NNS	Plural noun	NNS, NNPS
RB	Adverb	RB
VB	Verb, non-3rd ps. sing. present	VB, VBP
VBD	Verb, past tense or past participle	VBD, VBN
VBG	Verb, gerund/present participle	VBG
VBZ	Verb, 3rd ps. sing. present	VBZ

Table 1: The English tagset

Tag	Description	Examples
JJ	Adjective	vhalo, garam, kharap
NN	Singular noun	kanna, ridoy, shoshon
NN2	2nd order inflectional noun	dhopake, kalamtike
NN6	6th order inflectional noun	gharer, manusher
NN7	7th order inflectional noun	dhakai, barite, graame
NNP	Proper noun	arjun, ahmmad
NNS	Plural noun	manushgulo, pakhider
NNSH	Noun ending with "sh"	barish, jatrish
VB	Finite verb	kheyechi, krlam, krI
VBN	Non-finite verb	kre, giye, jete, kadte

Table 2: The Bengali tagset

Using morphological information. Perhaps due to the overly simplistic methods employed to compute morphological information, morphology has only been used as what Biemann (2006) called *add-on's* in existing POS induction algorithms, which remain primarily distributional in nature. In contrast, our approach more tightly integrates morphology into the distributional framework. As we will see, we train SVM classifiers using both morphological and distributional features to select seed words for our bootstrapping algorithm, effectively letting SVM combine these two sources of information and perform automatic feature weighting. Another appealing feature of our approach is that when labeling each unlabeled word with its POS tag, an SVM classifier also returns a numeric value that indicates how confident the word is labeled. This opens up the possibility of having a human improve our automatically constructed lexicon by manually checking those entries that are tagged with low confidence by an SVM classifier.

Recently, there have been attempts to perform (mostly) unsupervised POS tagging without relying on a POS lexicon. Haghighi and Klein's (2006) *prototype-driven* approach requires just a few prototype examples for each POS tag, exploiting these labeled words to constrain the labels of their distributionally similar words when training a generative log-linear model for POS tagging. Smith and Eisner (2005) train a log-linear model for POS tagging in an unsupervised manner using *contrastive estimation*, which seeks to move probability mass to a positive example *e* from its *neighbors* (i.e., negative examples created by perturbing *e*).

3 The English and Bengali Tagsets

Given our focus on automatically labeling open class words, our English and Bengali tagsets are designed to essentially cover all of the open-class words. Our English tagset, which is composed of ten tags, is shown in Table 1. As we can see, a tag in our tagset can be mapped to more than one Penn Treebank tags. For instance, we use the tag "NN" for both singular and plural common nouns. Our decision of which Penn Treebank tags to group together is based on that of Schütze (1995).

Our Bengali tagset, which also consists of ten tags, is adapted from the one proposed by Saha et al. (2004) (see Table 2). It is worth noting that unlike English, we assign different tags to Bengali proper nouns and common nouns. The reason is that for English, it is not particularly crucial to distinguish the two types of nouns during POS induction, since they can be distinguished fairly easily using heuristics such as initial capitalization. For Bengali, such simple heuristics do not exist, as the Bengali alphabet does not have any upper and lower case letters. Hence, it is important to distinguish Bengali proper nouns and common nouns during POS induction.

4 Clustering the Morphologically Similar Words

As mentioned before, our approach aims to more tightly integrate morphological information into the distributional POS induction framework. In fact, our POS induction algorithm begins by clustering the *morphologically similar* words (i.e., words that combine with the same set of suffixes). The motivation for clustering morphologically similar words can be attributed to our hypothesis that words having similar POS should combine with a similar set of suffixes. For instance, verbs in English combine with suffixes like "ing", "ed" and "s", whereas adjectives combine with suffixes like "er" and "est". Note, however, that the suffix "s" can attach to both verbs and nouns in English, and so it is not likely to be a useful feature for identifying the POS of a word. The question, then, is how to determine which suffixes are useful for the POS identification task in an *unsupervised* setting where we do not have any prior knowledge of language-specific grammatical constraints. This section proposes a method for identifying the "useful" suffixes and employing them to cluster the morphologically similar words. As we will see, our clustering algorithm not only produces soft clusters, but it also automatically determines the number of clusters for a particular language.

Before we describe how to identify the useful suffixes, we need to (1) induce all of the suffixes and (2) morphologically segment the words in our vocabulary.[2] However, neither of these tasks is simple for a truly resource-scarce language for which we do not have a dictionary or a knowledge-based morphological analyzer. As mentioned in the introduction, our proposed solution to both tasks is to use an unsupervised morphological analyzer that can be built just from an unannotated corpus. In particular, we have implemented an unsupervised morphological analyzer that outperforms Goldsmith's (2001) Linguistica and Creutz and Lagus's (2005) Morfessor for our English and Bengali datasets and compares favorably to the best-performing morphological parsers in MorphoChallenge 2005[3] (see Dasgupta and Ng (2007)).

Given the segmentation of each word and the most frequent 30 suffixes[4] provided by our morphological analyzer, our clustering algorithm operates by (1) clustering the similar suffixes and then (2) assigning words to each cluster based on the suffixes a word combines with. To cluster similar suffixes, we need to define the similarity between two suffixes. Informally, we say that two suffixes x and y are similar if a word that combines with x also combines with y and vice versa. In practice, we will rarely posit two suffixes as similar under this definition unless we assume access to a complete vocabulary – an assumption that is especially unrealistic for resource-scarce languages. As a result, we relax this definition and consider two suffixes x and y similar if $P(x \mid y) > t$ and $P(y \mid x) > t$, where $P(x \mid y)$ is the probability of a word combining with suffix x given that it combines with suffix y, and t is a threshold that we set to 0.4 in all of our experiments. Note that both probabilities can be estimated from an unannotated corpus.[5] Given this definition of similarity, we can cluster the similar suffixes using the following steps:

Creating the initial clusters. First, we create a *suffix graph*, in which we have (1) one node for each of the 30 suffixes, and (2) a directed edge from suffix x to suffix y if $P(y \mid x) > 0.4$. We then identify the strongly connected components of this graph using depth-first search. These strongly connected components define our initial partitioning of the 30 suffixes. We denote the suffixes assigned to a cluster the *primary keys* of the cluster.

Improving the initial clusters. Recall that we ultimately want to cluster the words by assigning each word w to the cluster in which w combines with all of its primary keys. Given this goal, it is conceivable that singleton clusters are not desirable. For instance, a cluster that has "s" as its only primary key is not useful, because although a lot of words combine with "s", they do not necessarily have the same POS. As a result, we improve each initial cluster by adding more suffixes to the cluster, in hopes of improving the resulting clustering of the words by placing additional constraints on each cluster. More specifically, we add a suffix y to a cluster c if, for each primary key x of c, $P(y \mid x) > 0.4$. If this condition is satisfied, then y becomes a *secondary key* of c. For each initial cluster c', we perform this check using each of the suffixes x' not in c' to see if x' can be added to c'. If, after this expansion step, we still have a cluster c^* defined by a single primary key x that also serves as a secondary key in other clusters, then x is *probably ambiguous* (i.e., x can probably attach to words belonging to different POSs); and consequently, we remove c^*. We denote the resulting set of clusters by C.

Populating the clusters with words. Next, for each word w in our vocabulary, we check whether w can be assigned to any of the clusters in C. Specifically, we assign w to a cluster c if w can combine with each of its primary keys and at least half of its secondary keys.

Labeling and merging the clusters. After populating each cluster with words, we manually label

[2] A vocabulary is simply a set of (distinct) words extracted from an unannotated corpus. We extracted our English and Bengali vocabulary from WSJ and Prothom Alo, respectively.

[3] http://www.cis.hut.fi/morphochallenge2005/

[4] We found that 30 suffixes are sufficient to cluster the words.

[5] For instance, we compute $P(x \mid y)$ as the ratio of the number of distinct words that combines with both x and y to the number of distinct words that combine with y only.

each of them with a POS tag from the tagset. We found that all of the clusters are labeled as NN, VB, or JJ. The reason is that the clustered words are mostly root words. We then merge all the clusters labeled with the same POS tag, yielding only three "big" clusters. Note that these "big" clusters are *soft* clusters, since a word can belong to more than one of them. For instance, "cool" can combine with "s" or "ing" to form a VB, and it can also combine with "er" or "est" to form a JJ.

Generating sub-clusters. Recall that each "big" cluster contains a set of suffixes and also a set of words that combines with those suffixes. Now, for each "big" cluster c, we create one sub-cluster c_x for each suffix x that appears in c. Then, for each word w in c, we use our unsupervised morphological analyzer to generate $w+x$ and add the surface form to the corresponding sub-cluster.

Labeling the sub-clusters. Finally, we manually label each sub-cluster with a POS tag from our tagset. For example, all the words ending in "ing" will be labeled as VBG. As before, we merge two clusters if they are labeled with the same POS tag. The resulting clusters are our morphologically formed clusters.

5 Purifying the Seed Set

The clusters formed thus far cannot be expected to be perfectly accurate, since (1) our unsupervised morphological analyzer is not perfect, and (2) morphology alone is not always sufficient for determining the POS of a word. In fact, we found that many adjectives are mislabeled as nouns for both languages. For instance, "historic" is labeled as a noun, since it combines with suffixes like "al" and "ally" that "accident" combines with. In addition, many words are labeled with the POS that does not correspond to their most common word sense. For instance, while words like "chair", "crowd" and "cycle" are more commonly used as nouns than verbs, they are labeled as verbs by our clustering algorithm. The reason is that suffixes that typically attach to verbs (e.g., "s", "ed", "ing") also attach to these words. Such labelings, though not incorrect, are undesirable, considering the fact that these words are to be used as seeds to bootstrap our morphologically formed clusters in a distributional manner. For instance, since "chair" and "crowd" are distributionally similar to nouns, their presence in the verb clusters can potentially contaminate the clusters with nouns during the bootstrapping process. Hence, for the purpose of effective bootstrapping, we also consider these words "mislabeled".

To identify the words that are potentially mislabeled, we rely on the following assumption: words that are morphologically similar should also be distributionally similar and vice versa. Based on this assumption, we propose a *purification* method that posits a word w as potentially mislabeled (and therefore should be removed or relabeled) if the POS of w as predicted using distributional information differs from that as determined by morphology.

The question, then, is how to predict the POS tag of a word using distributional information? Our idea is to use "supervised" learning, where we train and test on the seed set. Conceptually, we (1) train a *multi-class* classifier on the morphologically labeled words, each of which is represented by its context vector, and (2) apply the classifier to relabel the *same* set of words. If the new label of a word w differs from its original label, then morphology and context disagree upon the POS of w; and as mentioned above, our method then determines that the word is potentially misclassified. Note, however, that (1) the training instances are not perfectly labeled and (2) it does not make sense to train a classifier on data that is seriously mislabeled. Hence, we make the assumption that a large percentage (> 70%) of the training instances is correctly labeled[6], and that our method would work with a training set labeled at this level of accuracy. In addition, since we are training a classifier based on distributional features, we train and test on only *distributionally reliable* words, which we define to be words that appear at least five times in our corpus. Distributionally unreliable words will all be removed from the morphologically formed clusters, since we cannot predict their POS using distributional information.

In our implementation of this method, rather than train a multi-class classifier, we train a set of binary classifiers using SVM*light* (Joachims, 1999) together with the distributional features for determining the POS tag of a given word.[7] More specifically, we train one classifier for each pair of

[6] An inspection of the morphologically formed clusters reveals that this assumption is satisfied for both languages.

[7] In this and all subsequent uses of SVM*light*, we set all the training parameters to their default values.

POS tags. For instance, since we have ten POS tags for English, we will train 45 binary classifiers.[8] To determine the POS tag of a given English word *w*, we will use these 45 pairwise classifiers to independently assign a label to *w*. For instance, the NN-JJ classifier will assign either NN or JJ to *w*. We then count how many times *w* is tagged with each of the ten POS tags. If there is a POS tag *t* whose count is nine, it means that all the nine classifiers associated with *t* have classified *w* as *t*, and so our method will label *w* as *t*. Otherwise, we remove *w* from our seed set, since we cannot confidently label it using our classifier ensemble.

To create the training set for the NN-JJ classifier, for instance, we can possibly use all of the words labeled with NN and JJ as positive and negative instances, respectively. However, to ensure that we do not have a skewed class distribution, we use the same number of instances from each class to train the classifier. More formally, let I_{NN} be the set of instances labeled with NN, and I_{JJ} be the set of instances labeled with JJ. Without loss of generality, assume that $|I_{NN}| < |I_{JJ}|$, where $|X|$ denotes the size of the set X. To avoid class skewness, we have to sample from I_{JJ}, since it is the larger set. Our sampling method is motivated by bagging (Breiman, 1996). More specifically, we create 10 training sets from I_{JJ}, each of which has size $|I_{NN}|$ and is formed by sampling with replacement from I_{JJ}. We then combine each of these 10 training sets separately with I_{NN}, and train 10 SVM classifiers from the 10 resulting training sets. Given a test instance *i*, we first apply the 10 classifiers independently to *i* and obtain the signed confidence values[9] of the predictions provided by the classifiers. We then take the average of the 10 confidence values, assigning *i* the positive class if the average is at least 0, and negative otherwise.

As mentioned above, we use distributional features to represent an instance created from a word *w*. The distributional features are created based on Schütze's (1995) method. Specifically, the left context and the right context of *w* are each encoded using the most frequent 500 words from the vocabulary. A feature in the left (right) context has the value 1 if the corresponding word appears to the left (right) of *w* in our corpus, and 0 otherwise. However, we found that using distributional features alone would erroneously classify words like "car" and "cars" as having the same POS because the two words are distributionally similar. In general, it is difficult to distinguish words in NN from those in NNS by distributional means. The same problem occurs for words in VB and VBD. To address this problem, we augment the feature set with suffixal features. Specifically, we create one binary feature for each of the 30 most frequent suffixes that we employed in the previous section. The feature corresponding to suffix *x* has the value 1 if *x* is the suffix of *w*. Moreover, we create an additional suffixal feature whose value is 1 if none of the 30 most frequent suffixes is the suffix of *w*.

6 Augmenting the Seed Set

After purification, we have a set of clusters filled with distributionally and morphologically reliable seed words that receive the same POS tag when predicted independently by morphological features and distributional features. Our goal in this section is to augment this seed set. Since we have a small seed set (5K words for English and 8K words for Bengali) and a large number of unlabeled words, we believe that it is most natural to apply a weakly supervised learning algorithm to bootstrap the clusters. Specifically, we employ a version of self-training together with SVM as the underlying learning algorithm.[10] Below we first present the high-level idea of our self-training algorithm and then discuss the implementation details.

Conceptually, our self-training algorithm works as follows. We first train a multi-class SVM classifier on the seed set for determining the POS tag of a word using the morphological and distributional features described in the previous section, and then apply it to label the unlabeled (i.e., unclustered) words. Words that are labeled with a confidence value that exceeds the current threshold (which is initially set to 1 and -1 for positively and negatively labeled instances, respectively) will be

[8] We could have trained just one 10-class classifier, but the fairly large number of classes leads us to speculate that this multi-class classifier will not achieve a high accuracy.

[9] Here, a large positive number indicates that the classifier confidently labels the instance as NN, and a large negative number represents confident prediction for JJ.

[10] As a related note, Clark's (2001) bootstrapping algorithm uses KL-divergence to measure the distributional similarity between an unlabeled word and a labeled word, adding to a cluster the words that are most similar to its current member. For us, SVM is a more appealing option because it automatically combines the morphological and distributional features.

added to the seed set. In the next iteration, we retrain the classifier on the augmented labeled data, apply it to the unlabeled data, and add to the labeled data those instances whose predicted confidence is above the current threshold. If none of the instances has a predicted confidence above the current threshold, we reduce the threshold by 0.1. (For instance, if the original thresholds are 1 and -1, they will be changed to 0.9 and -0.9.) We then repeat the above procedure until the thresholds reach 0.5 and -0.5. [11] Finally, we apply the resulting bootstrapped classifier to label all of the unlabeled words that have a corpus frequency of at least five, using a threshold of 0.

In our implementation of the self-training algorithm, rather than train a multi-class classifier in each bootstrapping iteration, we train pairwise classifiers (recall that for English, 45 classifiers are formed from 10 POS tags) using the morphological and distributional features described in the previous section. Again, since we employ distributional features, we apply the 45 pairwise classifiers only to the distributionally reliable words (i.e., words with corpus frequency at least 5). To classify an unlabeled word *w*, we apply the 45 pairwise classifiers to independently assign a label to *w*.[12] We then count how many times *w* is tagged with each of the ten POS tags. If there is a POS tag whose count is nine and all of these nine votes are associated with confidence that exceeds the current threshold, then we add *w* to the labeled data together with its assigned tag.

7 Evaluation

7.1 Experimental Setup

Corpora. Recall that our bootstrapping algorithm assumes as input an unannotated corpus from which we (1) extract our *vocabulary* (i.e., the set of words to be labeled) and (2) collect the statistics needed in morphological and distributional clustering. We use as our English corpus the Wall Street Journal (WSJ) portion of the Penn Treebank (Marcus et al., 1993). Our Bengali corpus is composed of five years of articles taken from the Bengali newspaper *Prothom Alo*.

Vocabulary creation. To extract our English vocabulary, we pre-processed each document in the WSJ corpus by first tokenizing them and then removing the most frequent 500 words (as they are mostly closed class words), capitalized words, punctuations, numbers, and unwanted character sequences (e.g., "***"). The resulting English vocabulary consists of approximately 35K words. We applied similar pre-processing steps to the Prothom Alo articles to generate our Bengali vocabulary, which consists of 80K words.

Test set preparation. Our English test set is composed of the 25K words in the vocabulary that appear at least five times in the WSJ corpus. The gold-standard POS tags for each word *w* are derived automatically from the parse trees in which *w* appears. To create the Bengali test set, we randomly chose 5K words from the vocabulary that appear at least five times in Prothom Alo. Each word in the test set was then labeled with its POS tags by two of our linguists.

Evaluation metric. Following Schütze (1995), we report performance in terms of recall, precision, and F1. Recall is the percentage of POS tags correctly proposed, precision is the percentage of POS tags proposed that are correct, and F1 is simply the harmonic mean of recall and precision. To exemplify, suppose the correct tagset for "crowd" is {NN, VB}; if our system outputs {VB, JJ, RB}, then recall is 50%, precision is 33%, and F1 is 40%. Importantly, all of our results will be reported on *word types*. This prevents the frequently occurring words from having a higher influence on the results than their infrequent counterparts.

7.2 Results and Discussion

The baseline system. We use as our baseline system one of the best existing unsupervised POS induction algorithms (Clark, 2003). More specifically, we downloaded from Clark's website[13] the code that implements a set of POS induction algorithms he proposed. Among these implementations, we chose *cluster_neyessenmorph*, which combines morphological and distributional infor-

[11] We decided to stop the bootstrapping procedure at thresholds of 0.5 and -0.5, because the more bootstrapping iterations we use, the lower are the quality of the bootstrapped data as well as the accuracy of the bootstrapped classifier.

[12] As in purification, each pairwise classifier is implemented as a set of 10 classifiers, each of which is trained on an equal number of instances from both classes. Testing also proceeds as before: the label of an instance is derived from the average of the confidence values returned by the 10 classifiers, and the confidence value associated with the label is just the average of the 10 confidence values.

[13] http://www.cs.rhul.ac.uk/home/alexc/

mation and achieves the best performance in his paper. When running his program, we use WSJ and Prothom Alo as the input corpora. In addition, we set the number of clusters produced to be 128, since this setting yields the best result in his paper. Results of the baseline system for the English and Bengali test sets are shown under the “After Bootstrapping” column in row 1 of Tables 3 and 4. As we can see, the baseline achieves F1-scores of 59% and 45% for English and Bengali, respectively. The other results in row 1 will be discussed below.

Our induction system. Recall that our unsupervised POS induction algorithm operates in three steps. To better understand the performance contribution of each of these steps, we show in row 2 of Tables 3 and 4 the results of our system after we (1) morphologically cluster the words, (2) purify the seed set, and (3) augment the seed set. Importantly, the numbers shown for each step are computed over the set of words in the test set that are labeled at the end of that step. For instance, the morphological clustering algorithm labeled 11K English words and 25K Bengali words, and so recall, precision and F1-score are computed over the subset of these labeled words that appear in the test set. Similarly, after bootstrapping, all the words that appear at least five times in our corpus are labeled; since our labeled data is now a superset of our test data, the numbers in the last column are the results of our algorithm for the entire test set.

As we can see, after morphological clustering, our system achieves F1-scores of 79% and 78% for English and Bengali, respectively. When measured on exactly the same set of words, the baseline only achieves F-scores of 59% and 56%. In fact, comparing rows 1 and 2, we outperform the baseline in each of the three steps of our algorithm. In particular, our system yields F1-scores of 73% and 77% for the entire English and Bengali test sets, thus outperforming the baseline by 14% and 18% for English and Bengali, respectively.

Two additional points deserve mentioning. First, for both languages, the highest F1-score is achieved after the purification step. A closer analysis of the labeled words reveals the reason. For English, many of the nouns incorrectly labeled as verbs by the morphological clustering algorithm were subsequently removed during the purification step when distributional similarity was used on top of morphological similarity. For Bengali, many proper nouns were assigned by the morphological clustering algorithm to the clusters dominated by common nouns (because the two types of Bengali nouns are morphologically similar), and many of these mislabeled proper nouns were subsequently removed during purification. Second, as expected, precision drops after the seed augmentation step, since the quality of the labeled data deteriorates as bootstrapping progresses. Nevertheless, with a lot more words labeled in the bootstrapping step, we still achieve F1-scores of 73% for English and 76% for Bengali.

The remaining rows of the Tables 3 and 4 show the performance of our algorithm for each tag in our two POS tagsets. Different observations can be made for the two languages. For English, the poor results for VBZ and NNS can be attributed to the fact that it is not easy to distinguish between these two tags: “s” is a typical suffix for words that are NNS and words that are the third person singular of a verb. In addition, results for verbs are better than those for nouns, since verbs are easier to identify using only morphological knowledge.

For Bengali, results for adjectives are not good, since (1) adjectives and nouns have very similar distributional property in Bengali and (2) there are not enough suffixes to induce the adjectives morphologically. Moreover, we achieve high precision but low recall for proper nouns. This implies that most of the words that our algorithm labels as proper nouns are indeed correct, but there are also many proper nouns that are mislabeled. A closer examination of the clusters reveals that many of these proper nouns are mislabeled as common nouns, presumably because these two types of Bengali nouns are morphologically and distributionally similar and therefore it is difficult to separate them. We will leave the identification of Bengali proper nouns as a topic for future research.

7.3 Additional Experiments

Labeling rare words with morphological information. Although our discussion thus far has focused on words whose corpus frequency is at least five, it would be informative to examine how well our algorithm performs on *rare, distributionally unreliable* words (i.e., words with corpus frequency less than five). Recall that our morphological clustering algorithm also clusters rare words. In fact, these rare words comprise 15% of the English words and 18% of the Bengali words in our morphological formed clusters. Perhaps more impor-

	After Morphological Clustering			After Purification			After Bootstrapping		
	P	R	F1	P	R	F1	P	R	F1
Baseline	84.1	45.3	58.9	84.9	51.4	64.1	75.6	48.0	59.0
Ours	85.9	74.0	**79.4**	89.3	74.4	**81.7**	80.4	66.8	**73.1**
JJ	88.7	49.1	63.2	91.4	51.9	66.1	57.7	62.9	60.2
JJR	91.1	86.2	88.6	92.1	92.0	92.0	62.1	83.1	71.0
JJS	100	98.3	99.1	100	100	100	81.3	86.9	83.9
NN	91.6	43.7	59.2	94.8	42.8	58.8	95.2	47.1	62.8
NNS	90.6	39.2	53.5	93.5	41.3	57.2	96.6	44.7	60.9
RB	100	76.1	86.4	100	82.2	90.6	98.8	63.5	77.3
VB	74.0	97.7	84.1	79.8	96.0	87.1	65.7	92.8	76.9
VBD	96.6	98.9	97.7	97.6	100	98.8	96.7	91.9	93.3
VBG	89.9	100	94.7	91.1	100	95.7	90.8	93.5	92.1
VBZ	60.9	99.9	74.7	65.1	96.8	77.7	52.8	92.6	67.3

Table 3: POS induction results for English based on word type

	After Morphological Clustering			After Purification			After Bootstrapping		
	P	R	F1	P	R	F1	P	R	F1
Baseline	82.1	42.3	55.5	83.1	45.3	58.3	78.1	43.3	49.3
Ours	74.1	81.3	**77.5**	83.4	78.0	**80.7**	74.1	79.2	**76.6**
JJ	50.0	51.8	50.9	56.1	55.0	55.5	57.5	51.4	54.3
NN	63.0	96.8	76.4	67.0	96.0	78.9	62.2	92.2	74.3
NN2	96.3	100	98.1	99.0	100	99.5	99.0	99.0	99.0
NN6	95.5	89.2	92.2	97.2	90.0	93.9	97.1	91.0	93.9
NN7	88.4	94.1	89.7	92.1	99.2	93.1	90.1	78.7	84.1
NNP	87.2	37.3	52.3	92.8	43.8	59.4	92.7	51.5	66.1
NNS	62.7	93.1	75.0	66.8	93.5	77.9	65.2	94.1	77.1
NNSH	91.0	100	95.6	91.0	100	95.7	91.0	100	95.7
VB	68.9	93.0	79.2	77.0	94.6	84.9	73.9	91.8	81.9
VBN	84.3	49.1	62.1	82.4	50.1	62.9	56.1	46.7	50.1

Table 4: POS induction results for Bengali based on word type

tantly, when measuring performance on just these morphologically clustered rare words, our algorithm achieves F1-scores of 81% and 79% for English and Bengali, respectively. These results provide empirical support for the claim that morphological information can be usefully employed to label rare words (Clark, 2003).

Soft clustering. Many words have more than one POS tag. For instance, "received" can be labeled as VBD and JJ. Although our morphological clustering algorithm can predict some of these ambiguities, those are at the "big" cluster level. At the sub-cluster level, the algorithm imposes a hard clustering on the words. In other words, no word appears in more than one sub-cluster.

Ideally, a POS induction algorithm should produce soft clusters due to lexical ambiguity. In fact, Jardino and Adda (1994), Schütze (1997) and Clark (2000) have attempted to address the ambiguity problem to a certain extent. We have also experimented with a very simple method for handling ambiguity in our bootstrapping algorithm: when augmenting the seed set, instead of labeling a word with a tag that receives 9 votes from the 45 pairwise classifiers, we label a word with any tag that receives at least 8 votes, effectively allowing the assignment of more than one label to a word. However, our experimental results (not shown due to space limitations) indicate that the incorporation of this method does not yield better overall performance, since many of the additional labels are erroneous and hence their presence deteriorates the quality of the bootstrapped data.

8 Conclusions

We have proposed a new bootstrapping algorithm for unsupervised POS induction. In contrast to existing algorithms developed for this problem, our algorithm is designed to (1) operate under a resource-scarce setting in which no language-specific tools or resources are available and (2) more tightly integrate morphological information with the distributional POS induction framework. In particular, our algorithm (1) improves the quality of the seed clusters by employing seed words that are distributionally and morphologically reliable and (2) uses support vector learning to combine morphological and distributional information. Our results show that it outperforms Clark's algorithm for English and Bengali, suggesting that it is applicable to both morphologically impoverished and highly inflectional languages.

Acknowledgements

We thank the five anonymous EMNLP-CoNLL referees for their valuable comments. We also thank Zeeshan Abedin and Mahbubur Rahman Haque for creating the Bengali lexicon.

References

Chris Biemann. 2006. Unsupervised part-of-speech tagging employing efficient graph clustering. In *Proceedings of the COLING/ACL 2006 Student Research Workshop*.

Leo Breiman. 1996. Bagging predictors. *Machine Learning* 24(2):123-140.

Alexander Clark. 2000. Inducing syntactic categories by context distributional clustering. In *Proceedings of CoNLL*, pages 91-94.

Alexander Clark. 2003. Combining distributional and morphological information for part of speech induction. In *Proceedings of the EACL*.

Mathias Creutz and Krista Lagus. 2005. Unsupervised morpheme segmentation and morphology induction from text corpora using Morfessor 1.0. In *Computer and Information Science, Report A81*, Helsinki University of Technology.

Silviu Cucerzan and David Yarowsky. 2000. Language independent, minimally supervised induction of lexical probabilities. In *Proceedings of the ACL*, pages 270-277.

Sajib Dasgupta and Vincent Ng. 2007. High-performance, language-independent morphological segmentation. In *Proceedings of NAACL-HLT*, pages 155-163.

Kevin Duh and Katrin Kirchhoff. 2006. Lexicon acquisition for dialectal Arabic using transductive learning. In *Proceedings of EMNLP*, pages 399-407.

Dayne Freitag. 2004. Toward unsupervised whole-corpus tagging. In *Proceedings of COLING*, pages 357-363.

John Goldsmith. 2001. Unsupervised learning of the morphology of a natural language. In *Computational Linguistics* 27(2):153-198.

Aria Haghighi and Dan Klein. 2006. Prototype-driven learning for sequence models. In *Proceedings of HLT-NAACL*, pages 320-327.

Jan Hajič. 2000. Morphological tagging: Data vs. dictionaries. In *Proceedings of the NAACL*, pages 94-101.

Zellig Harris. 1954. Distributional structure. In *Word*, 10(2/3):146-162.

Michele Jardino and Gilles Adda. 1994. Automatic determination of a stochastic bi-gram class language model. In *Proceedings of Grammatical Inference and Applications, Second International Colloquium, ICGI-94*, pages 57-65.

Thorsten Joachims. 1999. Making large-scale SVM learning practical. In *Advances in Kernel Methods – Support Vector Learning*, pages 44-56. MIT Press.

Sydney Lamb. 1961. On the mechanization of syntactic analysis. In *Proceedings of the 1961 Conference on Machine Translation of Languages and Applied Language Analysis*, Volume 2, pages 674-685. HMSO, London.

Mitchell P. Marcus, Mary Ann Marcinkiewicz, and Beatrice Santorini. 1993. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313-330.

Andrei Mikheev. 1997. Automatic rule induction for unknown word-guessing. *Computational Linguistics*, 23(3):405-423.

Goutam Kumar Saha, Amiya Baran Saha, and Sudipto Debnath. 2004. Computer assisted Bangla words POS tagging. In *Proceedings of the International Symposium on Machine Translation NLP and TSS (iTRANS, 2004)*.

Hinrich Schütze. 1995. Distributional part-of-speech tagging. In *Proceedings of the EACL*, pages 141-148.

Hinrich Schütze. 1997. *Ambiguity Resolution in Language Learning*. CSLI Publications.

Noah Smith and Jason Eisner. 2005. Contrastive estimation: Training log-linear models on unlabeled data. In *Proceedings of the ACL*, pages 354-362.