Lexicon Acquisition for Dialectal Arabic Using Transductive Learning

Kevin Duh
Dept. of Electrical Engineering
University of Washington
Seattle, WA, USA
duh@ee.washington.edu

Katrin Kirchhoff
Dept. of Electrical Engineering
University of Washington
Seattle, WA, USA
katrin@ee.washington.edu

Abstract

We investigate the problem of learning a part-of-speech (POS) lexicon for a resource-poor language, dialectal Arabic. Developing a high-quality lexicon is often the first step towards building a POS tagger, which is in turn the front-end to many NLP systems. We frame the lexicon acquisition problem as a transductive learning problem, and perform comparisons on three transductive algorithms: Transductive SVMs, Spectral Graph Transducers, and a novel Transductive Clustering method. We demonstrate that lexicon learning is an important task in resource-poor domains and leads to significant improvements in tagging accuracy for dialectal Arabic.

1 Introduction

Due to the rising importance of globalization and multilingualism, there is a need to build natural language processing (NLP) systems for an increasingly wider range of languages, including those languages that have traditionally not been the focus of NLP research. The development of NLP technologies for a new language is a challenging task since one needs to deal not only with language-specific phenomena but also with a potential lack of available resources (e.g. lexicons, text, annotations). In this study we investigate the problem of learning a part-of-speech (POS) lexicon for a resource-poor language, dialectal Arabic.

Developing a high-quality POS lexicon is the first step towards training a POS tagger, which in turn is typically the front end for other NLP applications such as parsing and language modeling. In the case of resource-poor languages (and dialectal Arabic in particular), this step is much more critical than is typically assumed: a lexicon with too few constraints on the possible POS tags for a given word can have disastrous effects on tagging accuracy. Whereas such constraints can be obtained from large hand-labeled corpora or high-quality annotation tools in the case of resource-rich languages, no such resources are available for dialectal Arabic. Instead, constraints on possible POS tags must be inferred from a small amount of tagged words, or imperfect analysis tools. This can be seen as the problem of learning complex, structured outputs (multi-class labels, with a different number of classes for different words and dependencies among the individual labels) from partially labeled data.

Our focus is on investigating several machine learning techniques for this problem. In particular, we argue that lexicon learning in resource-poor languages can be best viewed as transductive learning. The main contribution of this work are: (1) a comprehensive evaluation of three transductive algorithms (Transductive SVM, Spectral Graph Transducer, and a new technique called Transductive Clustering) as well as an inductive SVM on this task; and (2) a demonstration that lexicon learning is a worthwhile investment and leads to significant improvements in the tagging accuracy for dialectal Arabic.

The outline of the paper is as follows: Section 2 describes the problem in more detail and discusses the situation in dialectal Arabic. The transductive framework and algorithms for lexicon learning are elaborated in Section 3. Sections 4 and 5 describe the data and system. Experimental results are presented in Section 6. We discuss some related work in Section 7 before concluding in Section 8.

2 The Importance of Lexicons in Resource-poor POS Tagging

2.1 Unsupervised Tagging

The lack of annotated training data in resource-poor languages necessitates the use of unsupervised taggers. One commonly-used unsupervised tagger is the Hidden Markov model (HMM), which models the joint distribution of a word sequence $w_{0:M}$ and tag sequence $t_{0:M}$ as:

$$P(t_{0:M}, w_{0:M}) = \prod_{i=0}^{M} p(w_i|t_i)p(t_i|t_{i-1}, t_{i-2}) \tag{1}$$

This is a trigram HMM. Unsupervised learning is performed by running the Expectation-Maximization (EM) algorithm on raw text. In this procedure, the tag sequences are unknown, and the probability tables $p(w_i|t_i)$ and $p(t_i|t_{i-1}, t_{i-2})$ are iteratively updated to maximize the likelihood of the observed word sequences.

Although previous research in unsupervised tagging have achieved high accuracies rivaling supervised methods (Kupiec, 1992; Brill, 1995), much of the success is due to the use of artificially *constrained* lexicons. Specifically, the lexicon is a wordlist where each word is annotated with the set of all its possible tags. (We will call the set of possible tags of a given word the *POS-set* of that word; an example: POS-set of the English word `bank` may be {`NN`, `VB`}.) Banko and Moore (2004) showed that unsupervised tagger accuracies on English degrade from 96% to 77% if the lexicon is not constrained such that only high frequency tags exist in the POS-set for each word.

Why is the lexicon so critical in unsupervised tagging? The answer is that it provides additional knowledge about word-tag distributions that may otherwise be difficult to glean from raw text alone. In the case of unsupervised HMM taggers, the lexicon provides constraints on the probability tables $p(w_i|t_i)$ and $p(t_i|t_{i-1}, t_{i-2})$. Specifically, the lexical probability table is initialized such that $p(w_i|t_i) = 0$ if and only if tag t_i is not included in the POS-set of word w_i. The transition probability table is initialized such that $p(t_i|t_{i-1}, t_{i-2}) = 0$ if and only if the tag sequence (t_i, t_{i-1}, t_{i-2}) never occurs in the tag lattice induced by the lexicon on the raw text. The effect of these zero-probability initialization is that they will always stay zero throughout the EM procedure (modulo the effects of smoothing). This therefore acts as hard constraints and biases the EM algorithm to avoid certain solutions when maximizing likelihood. If the lexicon is accurate, then the EM algorithm can learn very good predictive distributions from raw text only; conversely, if the lexicon is poor, EM will be faced with more confusability during training and may not produce a good tagger. In general, the addition of rare tags, even if they are correct, creates a harder learning problem for EM.

Thus, a critical aspect of resource-poor POS tagging is the acquisition of a high-quality lexicon. This task is challenging because the lexicon learning algorithm must not be resource-intensive. In practice, one may be able to find analysis tools or incomplete annotations such that only a partial lexicon is available. The focus is therefore on effective machine learning algorithms for inferring a full high-quality lexicon from a partial, possibly noisy initial lexicon. We shall now discuss this situation in the context of dialectal Arabic.

2.2 Dialectal Arabic

The Arabic language consist of a collection of spoken dialects and a standard written language (Modern Standard Arabic, or MSA). The dialects of Arabic are of considerable importance since they are used extensively in almost all everyday conversations. NLP technology for dialectal Arabic is still in its infancy, however, due to the lack of data and resources. Apart from small amounts of written dialectal material in e.g. plays, novels, chat rooms, etc., data can only be obtained by recording and manually transcribing actual conversations. Annotated corpora are scarce because annotation requires another stage of manual effort beyond transcription work. In addition, basic resources such as lexicons, morphological analyzers, tokenizers, etc. have been developed for MSA, but are virtually non-existent for dialectal Arabic.

In this study, we address lexicon learning for Levantine Colloquial Arabic. We assume that only two resources are available during training: (1) raw text transcriptions of Levantine speech and (2) a morphological analyzer developed for MSA.

The lexicon learning task begins with a partial lexicon generated by applying the MSA analyzer to the Levantine wordlist. Since MSA differs from Levantine considerably in terms of syntax, morphology, and lexical choice, not all Levantine words receive an analysis. In our data, 23% of the words are un-analyzable. Thus, the goal of lexicon learning is to infer the POS-sets of the un-analyzable words, given the partially-annotated lexicon and raw text.

Details on the Levantine data and overall system are provided in Sections 4 and 5. We discuss the learning algorithms in the next section.

3 Learning Frameworks and Algorithms

Let us formally define the lexicon learning problem. We have a wordlist of size $m + u$. A portion of these words (m) are annotated with POS-set labels, which may be acquired by manual annotation or an automatic analysis tool. The set of labeled words $\{X_m\}$ is the training set, also referred to as the partial lexicon. The task is to predict the POS-sets of the remaining u unlabeled words $\{X_u\}$, the test set. The goal of lexicon learning is to label $\{X_u\}$ with low error. The final result is a full lexicon that contains POS-sets for all $m + u$ words.

3.1 Transductive Learning with Structured Outputs

We argue that the above problem formulation lends itself to a transductive learning framework. Standard *inductive learning* uses a training set of fully labeled samples in order to learn a classification function. After completion of the training phase, the learned model is then used to classify samples from a new, previously unseen test set. *Semi-supervised* inductive learning exploits unlabeled data in addition to labeled data to better learn a classification function. *Transductive learning*, first described by Vapnik (Vapnik, 1998) also describes a setting where both labeled and unlabeled data are used *jointly* to decide on a label assignment to the unlabeled data points. However, the goal here is not to learn a general classification function that can be applied to new test sets multiple times but to achieve a high-quality one-time labeling of a particular data set. Transductive learning and inductive semi-supervised learning are sometimes confused in the literature. Both approaches use unlabeled data in learning – the key difference is that a transductive classifier only optimizes the performance on the given unlabeled data while an inductive semi-supervised classifier is trained to perform well on any new unlabeled data.

Lexicon learning fits in the transductive learning framework as follows: The test set $\{X_u\}$, i.e. the unlabeled words, is static and known during learning time; we are not interested in inferring POS-sets for any words outside the word list.

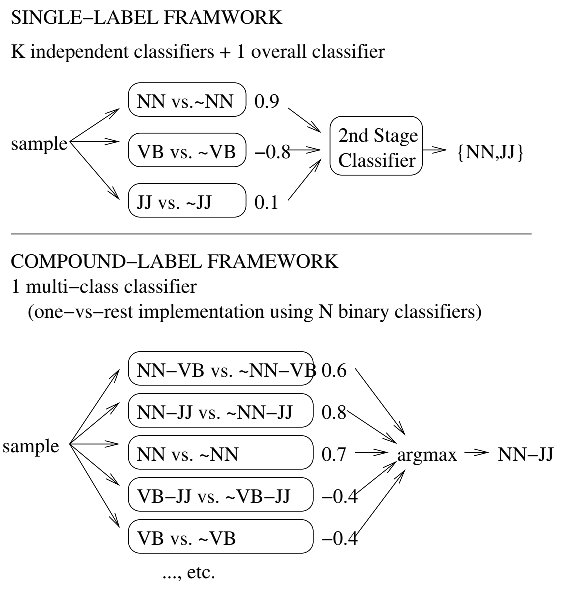

Figure 1: Learning with Structured Outputs using single or compound labels

An additional characterization of the lexicon learning problem is that it is a problem of learning with complex, structured outputs. The label for each word is its POS-set, which may contain one to K POS tags (where K is the size of the tagset, K=20 in our case). This differs from traditional classification tasks where the output is a single scalar variable.

Structured output problems like lexicon learning can be characterized by the granularity of the basic unit of labels. We define two cases: single-label and compound-label. In the single-label framework (see Figure 1), each individual POS tag is the target of classification and we have K binary classifiers each hypothesizing whether a word has a POS tag k ($k = 1, \ldots, K$). A second-stage classifier takes the results of the K individual classifiers and outputs a POS-set. This classifier can simply take all POS tags hypothesized positive by the individual binary classifiers to form the POS-set, or use a more sophisticated scheme for determining the number of POS tags (Elisseeff and Weston, 2002).

The alternative compound-label framework treats each POS-set as an atomic label for classification. A POS-set such as {"NN", "VB"} is "compounded" into one label "NN-VB", which results in a different label than, say, "NN" or "NN-JJ". Suppose there exist N distinct POS-sets in the training data; then we have N atomic units for labeling. Thus a (N-ary) multi-class classifier is employed to directly predict the POS-set of a word. If only binary classifiers are available (i.e. in the case of Support Vector Machines), one can use one-vs-rest, pairwise, or error correcting code schemes to implement the multi-class classification.

The single-label framework is potentially ill-suited for capturing the dependencies between POS tags. Dependencies between POS tags arise since some tags, such as "NN" and "NNP" can often be tagged to the same word and therefore co-occur in the POS-set label. The compound-label framework implicitly captures tag co-occurrence, but potentially suffers from training data fragmentation as well as the inability to hypothesize POS-sets that do not already exist in the training data. In our initial experiments, the compound-label framework gave better classification results; thus we implemented all of our algorithms in the multi-class framework (using the one-vs-rest scheme and choosing the argmax as the final decision).

3.2 Transductive Clustering

How does a transductive algorithm effectively utilize unlabeled samples in the learning process? One popular approach is the application of the so-called *cluster assumption*, which intuitively states that samples close to each other (i.e. samples that form a cluster) should have similar labels.

Transductive clustering (TC) is a simple algorithm that directly implements the cluster assumption. The algorithm clusters labeled and unlabeled samples jointly, then uses the labels of labeled samples to infer the labels of unlabeled words in the same cluster. This idea is relatively straightforward, yet what is needed is a principled way of deciding the correct number of clusters and the precise way of label transduction (e.g. based on majority vote vs. probability thresholds). Typically, such parameters are decided heuristically (e.g. (Duh and Kirchhoff, 2005a)) or by tuning on a labeled development set; for resource-poor languages, however, no such set may be available.

As suggested by (El-Yaniv and Gerzon, 2005), the TC algorithm can utilize a theoretical error bound as a principled way of determining the parameters. Let $\hat{R}_h(X_m)$ be the empirical risk of a given hypothesis (i.e. classifier) on the training set; let $R_h(X_u)$ be the test risk. (Derbeko et al., 2004) derive an error bound which states that, with probability $1-\delta$, the risk on the test samples is bounded by:

$$\mathrm{R}_h(X_u) \leq \hat{R}_h(X_m) + \sqrt{\left(\frac{m+u}{u}\right)\left(\frac{u+1}{u}\right)\left(\frac{\ln\frac{1}{p(h)}+\ln\frac{1}{\delta}}{2m}\right)} \qquad (2)$$

i.e. the test risk is bounded by the empirical risk on the labeled data, $\hat{R}_h(X_m)$, plus a term that varies with the prior $p(h)$ of the hypothesis or classifier. This is a PAC-Bayesian bound (McAllester, 1999). The prior $p(h)$ indicates ones prior belief on the hypothesis h over the set of all possible hypotheses. If the prior is low or the empirical risk is high, then the bound is large, implying that test risk may be large. A good hypothesis (i.e. classifier) will ideally have a small value for the bound, thus predicting a small expected test risk.

The PAC-Bayesian bound is important because it provides a theoretical guarantee on the quality of a hypothesis. Moreover, the bound in Eq. 2 is particularly useful because it is easily computable on any hypothesis h, assuming that one is given the value of $p(h)$. Given two hypothesized labelings of the test set, h_1 and h_2, the one with the lower PAC-Bayesian bound will achieve a lower expected test risk. Therefore, one can use the bound as a principled way of choosing the parameters in the Transductive Clustering algorithm: First, a large number of different clusterings is created; then the one that achieves the lowest PAC-Bayesian bound is chosen. The pseudo-code is given in Figure 2.

(El-Yaniv and Gerzon, 2005) has applied the Transductive Clustering algorithm successfully to binary classification problems and demonstrated improvements over the current state-of-the-art Spectral Graph Transducers (Section 3.4). We use the algorithm as described in (Duh and Kirchhoff, 2005b), which adapts the algorithm to structured output problems. In particular, the modification involves a different estimate of the priors $p(h)$, which was assumed to be uniform in (El-Yaniv and Gerzon, 2005). Since there are many possible h, adopting a uniform prior will lead to small values of $p(h)$ and thus a loose bound for all h. Probability mass should only be spent on POS-sets that are possible, and as such, we calculate $p(h)$ based on frequencies of compound-labels in the training data (i.e. an empirical prior).

3.3 Transductive SVM

Transductive SVM (TSVM) (Joachims, 1999) is an algorithm that implicitly implements the cluster

```
1   For $\tau = 2 : C$            ($C$ is set arbitrarily to a large number)
2     Apply a clustering algorithm to generate $\tau$ clusters on $X_{m+u}$.
3     Generate label hypothesis $h_\tau$ (by labeling each cluster with the most frequent label among its labeled samples)
4     Calculate the bound for $h_\tau$ as defined in Eq. 2.
5   Choose the hypothesis $h_\tau$ with the lowest bound; output the corresponding classification of $X_u$.
```

Figure 2: Pseudo-code for transductive clustering.

assumption. In standard inductive SVM (ISVM), the learning algorithm seeks to maximize the margin subject to misclassification constraints on the training samples. In TSVM, this optimization is generalized to include additional constraints on the unlabeled samples. The resulting optimization algorithm seeks to maximize the margin on both labeled and unlabeled samples and creates a hyperplane that avoids high-density regions (e.g. clusters).

3.4 Spectral Graph Transducer

Spectral Graph Transducer (SGT) (Joachims, 2003) achieves transduction via an extension of the normalized mincut clustering criterion. First, a data graph is constructed where the vertices are labeled or unlabeled samples and the edge weights represent similarities between samples. The mincut criteria seeks to partition the graph such that the sum of cut edges is minimized. SGT extends this idea to transductive learning by incorporating constraints that require samples of the same label to be in the same cluster. The resulting partitions decide the label of unlabeled samples.

4 Data

4.1 Corpus

The dialect addressed in this work is Levantine Colloquial Arabic (LCA), primarily spoken in Jordan, Lebanon, Palestine, and Syria. Our development/test data comes from the Levantine Arabic CTS Treebank provided by LDC. The training data comes from the Levantine CTS Audio Transcripts. Both are from the Fisher collection of conversational telephone speech between Levantine speakers previously unknown to each other. The LCA data was transcribed in standard MSA script and transliterated into ASCII characters using the Buckwalter transliteration scheme[1]. No diacritics are used in either the training or development/test data. Speech effects such as disfluencies and noises were removed prior to our experiments.

The training set consists of 476k tokens and 16.6k types. It is not annotated with POS tags – this is the raw text we use to train the unsupervised HMM tagger. The test set consists of 15k tokens and 2.4k types, and is manually annotated with POS tags. The development set is also POS-annotated, and contains 16k tokens and 2.4k types. We used the reduced tagset known as the Bies tagset (Maamouri et al., 2004), which focuses on major part-of-speech and excludes detailed morphological information.

Using the compound-label framework, we observe 220 and 67 distinct compound-labels (i.e. POS-sets) in the training and test sets, respectively. As mentioned in Section 3.1, a classifier in the compound-label framework can never hypothesize POS-sets that do not exist in the training data: 43% of the test vocabulary (and 8.5% by token frequency) fall under this category.

4.2 Morphological Analyzer

We employ the LDC-distributed Buckwalter analyzer for morphological analyses of Arabic words. For a given word, the analyzer outputs all possible morphological analyses, including stems, POS tags, and diacritizations. The information regarding possible POS tags for a given word is crucial for constraining the unsupervised learning process in HMM taggers.

The Buckwalter analyzer is based on an internal stem lexicon combined with rules for affixation. It was originally developed for the MSA, so only a certain percentage of Levantine words can be correctly analyzed. Table 1 shows the percentages of words in the LCA training text that received N possible POS tags from the Buckwalter analyzer. Roughly 23% of types and 28% of tokens received no tags (N=0) and are considered un-analyzable.

5 System

Our overall system looks as follows (see Figure 3): In Step 1, the MSA (Buckwalter) analyzer is applied to the word list derived from the raw training text. The result is a partial POS lexicon,

[1]http://www.ldc.upenn.edu/myl/morph/buckwalter.html

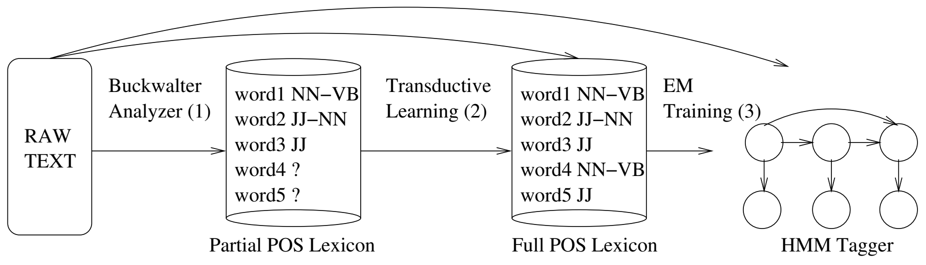

Figure 3: Overall System: (1) Apply Buckwalter Analyzer to dialectal Arabic raw text, obtaining a partial POS lexicon. (2) Use Transductive Learning to infer missing POS-sets. (3) Unsupervised training of HMM Tagger using both raw text and inferred lexicon.

N	Type	Token
0	23.3	28.2
1	52.5	40.4
2	17.7	19.9
3	5.2	10.5
4	1.0	2.3
5	0.1	0.6

Table 1: Percentage of word types/tokens with N possible tags, as determined by the Buckwalter analyzer. Words with 0 tags are un-analyzable.

which lists the set of possible POS tags for those words for which the analyzer provided some output. All possibilities suggested by the analyzer are included.

The focus of Step 2 is to infer the POS-sets of the remaining, unannotated words using one of the automatic learning procedures described in Section 3. Finally, Step 3 involves training an HMM tagger using the learned lexicon. This is the standard unsupervised learning component of the system. We use a trigram HMM, although modifications such as the addition of affixes and variables modeling speech effects may improve tagging accuracy. Our concern here is the evaluation of the lexicon learning component in Step 2.

An important problem in this system setup is the possibility of error propagation. In Step 1, the MSA analyzer may give incorrect POS-sets to analyzable words. It may not posit the correct tag (low recall), or it may give too many tags (low precision). Both have a negative effect on lexicon learning and EM training. For lexicon learning, Step 1 errors represent corrupt training data; For EM training, Step 1 error may cause the HMM tagger to never hypothesize the correct tag (low recall) or have too much confusibility during training (low precision). We attempted to measure the extent of this error by calculating the tag precision/recall on words that occur in the test set: Among the 12k words analyzed by the analyzer, 1483 words occur in the test data. We used the annotations in the test data and collected all the "oracle" POS-sets for each of these 1483 words.[2] The average precision of the analyzer-generated POS-sets against the oracle is 56.46%. The average recall is 81.25%. Note that precision is low–this implies that the partial lexicon is not very constrained. The recall of 81.25% means that 18.75% of the words may never receive the correct tag in tagging. In the experiments, we will investigate to what extent this kind of error affects lexicon learning and EM training.

6 Experiments

6.1 Lexicon learning experiments

We seek to answer the following three questions in our experiments:

- How useful is the lexicon learning step in an end-to-end POS tagging system? Do the machine learning algorithms produce lexicons that result in higher tagging accuracies, when compared to a baseline lexicon that simply hypothesizes all POS tags for un-analyzable words? The answer is a definitive **yes**.

- What machine learning algorithms perform the best on this task? Do transductive learning outperform inductive learning? The empirical answer is that TSVM performs best, SGT performs worst, and TC and ISVM are in the middle.

[2]Since the test set is small, these "oracle" POS-sets may be missing some tags. Thus the true precision may be higher (and recall may be lower) than measured.

Orthographic features:
w_i matches `/^pre/`, `pre` = {set of data-derived prefixes}
w_i matches `/suf$/`, `suf` = {set of data-derived suffixes}
Contextual features:
w_{i-1} = `voc`, `voc` = {set of words in lexicon}
t_{i-1} = `tag`, `tag` = {set of POS tags}
t_{i+1} = `tag`, `tag` = {set of POS tags}
w_{i-1} is an un-analyzable word
w_{i+1} is an un-analyzable word

Table 2: Binary features used for predicting POS-sets of un-analyzable words.

- What is the relative impact of errors from the MSA analyzer on lexicon learning and EM training? The answer is that Step 1 errors affect EM training more, and lexicon learning is comparably robust to these errors.

In our problem, we have 12k labeled samples and 3970 unlabeled samples. We define the feature of each sample as listed in Table 2. The contextual features are generated by co-occurrence statistics gleaned from the training data. For instance, for a word `foo`, we collect all bigrams consisting of `foo` from the raw text; all features [w_{t-1} = `voc`] that correspond to the bigrams (`voc`, `foo`) are set to 1. The idea is that words with similar orthographic and/or contextual features should receive similar POS-sets.

All results, unless otherwise noted, are tagging accuracies on the test set given by training a HMM tagger on a specific lexicon. Table 3 gives tagging accuracies of the four machine learning methods (TSVM, TC, ISVM, SGT) as well as two baseline approaches for generating a lexicon: (all tags) gives all 20 possible tags to the un-analyzable words, whereas (open class) gives only the subset of open-class POS tags.[3] The results are given in descending order of overall tagging accuracy.[4] With the exception of TSVM (63.54%) vs. TC (62.89%), all differences are statistically significant. As seen in the table, applying a machine learning step for lexicon learning is a worthwhile effort since it always leads to better tagging accuracies than the baseline methods.

[3]Not all un-analyzable words are open-class. Close-class words may be un-analyzable due to dialectal spelling variations.

[4]Note that the unknown word accuracies do not follow the same trend and are generally quite low. This might be due to the fact that POS tags of unknown words are usually best predicted by the HMM's transition probabilities, which may not be as robust due to the noisy lexicon.

Method	Accuracy	UnkAcc
TSVM	63.54	26.19
TC	62.89	26.71
ISVM	61.53	27.68
SGT	59.68	25.82
open class	57.39	27.08
all tags	55.64	25.00

Table 3: Tagging Accuracies for lexicons derived by machine learning (TSVM, TC, ISVM, SGT) and baseline methods. Accuracy=Overall accuracy; UnkAcc=Accuracy of unknown words.

The poor performance of SGT is somewhat surprising since it is contrary to results presented in other papers. We attributed this to the difficulty in constructing the data graph. For instance, we constructed k-nearest-neighbor graphs based on the cosine distance between feature vectors, but it is difficult to decide the best distance metric or number of neighbors. Finally, we note that besides the performance of SGT, transductive learning methods (TSVM, TC) outperform the inductive ISVM.

We also compute precision/recall statistics of the final lexicon on the test set words (similar to Section 5) and measure the average size of the POS-sets (||POSset||). As seen in Table 4, POS-set sizes of machine-learned lexicon is a factor of 2 or 3 smaller than that of the baseline lexicons. On the other hand, recall is better for the baseline lexicons. These observations, combined with the fact that machine-learned lexicons gave better tagging accuracy, suggests that we have a *constrained* lexicon effect here: i.e. for EM training, it is better to constrain the lexicon with small POS-sets than to achieve high recall.

Method	Precision	Recall	‖POSset‖
TSVM	58.15	88.85	1.89
TC	59.19	87.88	1.80
ISVM	58.09	88.44	1.87
SGT	53.98	82.60	1.87
open class	54.03	96.77	3.39
all tags	53.31	98.53	5.17

Table 4: Statistics of the Lexicons in Table 3.

Next, we examined the effects of error propagation from the MSA analyzer in Step 1. We attempted to correct these errors by using POS-sets of words derived from the development data. In particular, of the 1562 partial lexicon words that also occur in the development set, we found 1044 words without entirely matching POS-sets. These POS-sets are replaced with the oracle POS-sets derived from the development data, and the result is treated as the (corrected) partial lexicon of Step 1. In this procedure, the average POS-set size of the partial lexicon decreased from 2.13 to 1.10, recall increased from 82.44% to 100%, and precision increased from 57.15% to 64.31%. We apply lexicon learning to this corrected partial lexicon and evaluate tagging results, shown in Table 5. The fact that all numbers in Table 5 represent significant improvements over Table 3 implies that error propagation is not a trivial problem, and automatic error correction methods may be desired.

Method	Accuracy	UnkAcc
TSVM	66.54	27.38
ISVM	65.08	26.86
TC	64.05	28.20
SGT	63.78	27.23
all tags	62.96	27.91
open class	61.26	27.83

Table 5: Tag accuracies by correcting mistakes in the partial lexicon prior to lexicon learning. Interestingly, we note ISVM outperforms TC here, which differs from Table 3.

Finally, we determine whether error propagation impacts lexicon learning (Step 2) or EM training (Step 3) more. Table 6 shows the results of TSVM for four scenarios: correcting analyzer errors in the the lexicon: (A) prior to lexicon learning, (B) prior to EM training, (C) both, or (D) none. As seen in Table 6, correcting the lexicon at Step 3 (EM training) gives the most improvements, indicating that analyzer errors affects EM training more than lexicon learning. This implies that lexicon learning is relatively robust to training data corruption, and that one can mainly focus on improved estimation techniques for EM training (Wang and Schuurmans, 2005) if the goal is to alleviate the impact of analyzer errors. The same evaluation on the other machine learning methods (TC, ISVM, SGT) show similar results.

6.2 Comparison experiments: Expert lexicon and supervised learning

Our approach to building a resource-poor POS tagger involves (a) lexicon learning, and (b) unsupervised training. In this section we examine cases where (a) an expert lexicon is available, so that lexicon learning is not required, and (b) sentences are annotated with POS information, so that supervised training is possible. The goal of these experiments is to determine when alternative approaches involving additional human annotations become worthwhile in this task.

Scenario	Step2	Step3	TSVM
(B)	N	Y	66.70
(C)	Y	Y	66.54
(A)	Y	N	64.93
(D)	N	N	63.54

Table 6: Effect of correcting the lexicon in different steps. Y=yes, lexicon corrected; N=no, POS-set remains the same as analyzer's output.

(a) Expert lexicon: First, we build an expert lexicon by collecting all tags per word in the development set (i.e. "oracle" POS-sets). Then, the tagger is trained using EM by treating the development set as raw text (i.e. ignoring the POS annotations). This achieves an accuracy of 74.45% on the test set. Note that this accuracy is significantly higher than the ones in Table 3, which represent unsupervised training on more raw text (the training set), but with non-expert lexicons derived from the MSA analyzer and a machine learner. This result further demonstrates the importance of obtaining an accurate lexicon in unsupervised training. If one were to build this expert lexicon by hand, one would need an annotator to label the POS-sets of 2450 distinct lexicon items.

(b) Supervised training: We build a supervised tagger by training on the POS annotations of the development set, which achieves 82.93% accuracy. This improved accuracy comes at the cost of annotating 2.2k sentences (16k tokens) with complete POS information.

Finally, we present the same results with reduced data, taking first 50, 100, 200, etc. sentences in the development set for lexicon or POS annotation. The learning curve is shown in Table 7. One may be tempted to draw conclusions regarding supervised vs. unsupervised approaches by directly comparing this table with the results in Section 6.1; we avoid doing so since taggers in Sections 6.1 and 6.2 are trained on different data sets (training vs. development set) and the accuracy differences are compounded by issues such

Supervised		Unsupervised, Expert	
#Sentence	Acc	#Vocab	Acc
50	47.82	123	47.13
100	55.32	188	54.65
200	61.17	299	57.37
400	69.17	497	64.36
800	76.92	953	70.36
1600	81.73	1754	72.99
2200	82.93	2450	74.45

Table 7: (1) Supervised training accuracies with varying numbers of sentences. (2) Accuracies of unsupervised training using a expert lexicon of different vocabulary sizes.

as ngram coverage, data-set selection, and the way annotations are done.

7 Related Work

There is an increasing amount of work in NLP tools for Arabic. In supervised POS tagging, (Diab et al., 2004) achieves high accuracy on MSA with the direct application of SVM classifiers. (Habash and Rambow, 2005) argue that the rich morphology of Arabic necessitates the use of a morphological analyzer in combination with POS tagging. This can be considered similar in spirit to the learning of lexicons for unsupervised tagging.

The work done at a recent JHU Workshop (Rambow and others, 2005) is very relevant in that it investigates a method for improving LCA tagging that is orthogonal to our approach. They do not use the raw LCA text as we have done. Instead, they train a MSA supervised tagger and adapt it to LCA by a combination of methods, such using a MSA-LCA translation lexicon and redistributing the probabibility mass of MSA words to LCA.

8 Conclusion

In this study, we investigated several machine learning algorithms on the task of lexicon learning and demonstrated its impact on dialectal Arabic tagging. We achieve a POS tagging accuracy of 63.54% using a transductively-learned lexicon (TSVM), outperforming the baseline (57.39%). This result brings us one step closer to the accuracies of unsupervised training with expert lexicon (74.45%) and supervised training (82.93%), both of which require significant annotation effort. Future work includes a more detailed analysis of transductive learning in this domain and possible solutions to alleviating error propagation.

Acknowledgments

We would like to thank Rebecca Hwa for discussions regarding the JHU project. This work is funded in part by NSF Grant IIS-0326276 and an NSF Graduate Fellowship for the 1st author. Any opinions, findings, and conclusions expressed in this material are those of the authors and do not necessarily reflect the views of these agencies.

References

M. Banko and R. Moore. 2004. Part-of-speech tagging in context. In *Proc. of COLING 2004*.

E. Brill. 1995. Unsupervised learning of disambiguation rules for part of speech tagging. In *Proc. of the Third Workshop on Very Large Corpora*.

P. Derbeko, R. El-Yaniv, and R. Meir. 2004. Explicit learning curves for transduction and application to clustering and compression algorithms. *Journal of Artificial Intelligence Research*, 22:117-142.

M. Diab, K. Hacioglu, and D. Jurafsky. 2004. Automatic tagging of Arabic text: from raw text to base phrase chunks. In *Proceedings of HLT/NAACL*.

K. Duh and K. Kirchhoff. 2005a. POS tagging of dialectal arabic: a minimally-supervised approach. In *ACL 2005, Semitic Languages Workshop*.

K. Duh and K. Kirchhoff. 2005b. Structured multi-label transductive learning. In *NIPS Workshop on Advances in Structured Learning for Text/Speech Processing*.

R. El-Yaniv and L. Gerzon. 2005. Effective transductive learning via objective model selection. *Pattern Recognition Letters*, 26(13):2104-2115.

A. Elisseeff and J. Weston. 2002. Kernel methods for multi-labeled classification. In *NIPS*.

N. Habash and O. Rambow. 2005. Arabic tokenization, morphological analysis, and part-of-speech tagging in one fell swoop. In *ACL*.

T. Joachims. 1999. Transductive inference for text classification using support vector machines. In *ICML*.

T. Joachims. 2003. Transductive learning via spectral graph partitioning. In *ICML*.

J. Kupiec. 1992. Robust part-of-speech tagging using a hidden Markov model. *Computer Speech and Language*, 6.

M. Maamouri, A. Bies, and T. Buckwalter. 2004. The Penn Arabic Treebank: Building a large-scale annotated Arabic corpus. In *NEMLAR Conf. on Arabic Language Resources and Tools*.

D. McAllester. 1999. Some PAC-Bayesian theorems. *Machine Learning*, 37(3):255-36.

O. Rambow et al. 2005. Parsing Arabic dialects. Technical report, Final Report, 2005 JHU Summer Workshop.

V. Vapnik. 1998. *Statistical Learning Theory*. Wiley Interscience.

Q. Wang and D. Schuurmans. 2005. Improved estimation for unsupervised part-of-speech tagging. In *IEEE NLP-KE*.