POS Tagging of Dialectal Arabic: A Minimally Supervised Approach

Abstract

Natural language processing technology for the dialects of Arabic is still in its infancy, due to the problem of obtaining large amounts of text data for spoken Arabic. In this paper we describe the development of a part-of-speech (POS) tagger for Egyptian Colloquial Arabic. We adopt a minimally supervised approach that only requires raw text data from several varieties of Arabic and a morphological analyzer for Modern Standard Arabic. No dialect-specific tools are used. We present several statistical modeling and cross-dialectal data sharing techniques to enhance the performance of the baseline tagger that and compare the results to those obtained by a supervised tagger trained on hand-annotated data and, by a state-of-the-art Modern Standard Arabic tagger applied to Egyptian Arabic.

1 Introduction

Part-of-speech (POS) tagging is a core natural language processing task that can benefit a wide range of downstream processing applications. Tagging is often the first step towards parsing or chunking (Osborne, 2000; Koeling, 2000), and knowledge of POS tags can benefit statistical language models for speech recognition or machine translation (Heeman, 1998; Vergyri et al., 2004). Many approaches for POS tagging have been developed in the past, including rule-based tagging (Brill, 1995), HMM taggers (Brants, 2000; Cutting and others, 1992), maximum-entropy models (Rathnaparki, 1996), cyclic dependency networks (Toutanova et al., 2003), etc. All of these approaches require either a large amount of annotated training data (for supervised tagging) or lexicon listing all possible tags for each word (for unsupervised tagging). Taggers have been developed for a variety of languages, including Modern Standard Arabic (MSA) (Khoja, 2001; Diab et al., 2004). Since large amount of text material as well as standard lexicons can be obtained in these cases, POS tagging is a straightforward task.

The dialects of Arabic, by contrast, are spoken rather than written languages. 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. Moreover, there is no universally agreed upon writing standard for dialects (though several standardization efforts are underway); any large-scale data collection and transcription effort therefore requires extensive training of annotators to ensure consistency. Due to this data acquisition bottleneck, the development of NLP technology for dialectal Arabic is still in its infancy. In addition to the problems of sparse training data and lack of writing standards, tagging of dialectal Arabic is difficult for the following reasons:

- Resources such as lexicons, morphological analyzers, tokenizers, etc. are scarce or non-existent for dialectal Arabic.
- Dialectal Arabic is a spoken language. Tagging spoken language is typically harder than tag-

ging written language, due to the effect of disfluencies, incomplete sentences, varied word order, etc.

- The rich morphology of Arabic leads to a large number of possible word forms, which increases the number of out-of-vocabulary (OOV) words.
- The lack of short vowel information results in high lexical ambiguity.

In this paper we present an attempt at developing a POS tagger for dialectal Arabic in a minimally supervised way. Our goal is to utilize existing resources and data for several varieties of Arabic in combination with unsupervised learning algorithms, rather than developing dialect-specific tools. The resources available to us are the CallHome Egyptian Colloquial Arabic (ECA) corpus, the LDC Levantine Arabic (LCA) corpus, the LDC MSA Treebank corpus, and a generally available morphological analysis tool (the LDC-distributed Buckwalter stemmer) for MSA. The target dialect is ECA, since this is the only dialectal corpus for which POS annotations are available. Our general approach is to bootstrap the tagger in an unsupervised way using POS information from the morphological analyzer, and to subsequently improve it by integrating additional data from other dialects and by general machine learning techniques. We compare the result against the performance of a tagger trained in a supervised way and an unsupervised tagger with a hand-developed ECA lexicon.

2 Data

The ECA corpus consists of telephone conversations between family members or close friends, with one speaker being located in the U.S. and the other in Egypt. We use the combined train, eval96 and hub5 sections of the corpus for training, the dev set for development and the eval97 set for evaluation. The LCA data consists of telephone conversations on a pre-defined topic between Levantine speakers previously unknown to each other; all of the available data was used. The Treebank corpus is a collection of MSA newswire text from Agence France Press. We use parts 1 (v3.0), 2 (v2.0) and 3 (v1.0). The sizes of the various corpora are shown in Table 1.

The ECA corpus was originally transcribed in a "romanized" form; a script representation was then derived automatically from the romanized transcripts. For this reason, the script does not entirely conform to the MSA standard: the romanized forms may represent actual pronunciations and contain such MSA $\rightarrow$ ECA changes as /θ/ $\rightarrow$ /s/ or /t/ and /ð/ $\rightarrow$ /z/ or /d/. The resulting representation cannot be unambiguously mapped back to MSA script; therefore the variants /s/ or /t/, for instance, are represented by س or ت, rather than ث. This introduces additional noise into the data, but it also mimics the real-world situation of variable spelling standards that need to be handled by a robust NLP system. We use the script representation of this corpus for all our experiments. The ECA corpus is accompanied by a lexicon containing the morphological analysis of all words, i.e. an analysis in terms of stem and morphological characteristics such as person, number, gender, POS, etc. Since the analysis is based on the romanized form, a single tag can be assigned to the majority of words (75% of all tokens) in the corpus. We use this assignment as the reference annotation for our experiments to evaluate the output of our tagger. The remaining 25% tokens (ambiguous words) are ignored since the correct reference tag cannot be determined.

Both the LCA and the MSA Treebank data sets were transcribed in standard MSA script. The LCA corpus only consists of raw orthographic transcriptions; no further annotations exist for this data set. Each word in the Treebank corpus is associated with all of its possible POS tags; the correct tag has been marked manually. We use the undecomposed word forms rather than the forms resulting from splitting off conjunctions, prepositions, and other clitics. Although improved tokenization can be expected to result in better tagging performance, tokenization tools for dialectal Arabic are currently not available, and our goal was to create comparable conditions for tagging across all of our data sets. Preprocessing of the data thus only included removing punctuation from the MSA data and removing word fragments from the spoken language corpora. Other disfluencies (fillers and repetitions) were retained since they are likely to have predictive value. Finally, singleton words inconsistent spellings were removed from the

LCA data. The properties of the different data sets (number of words, n-grams, and ambiguous words) are displayed in Table 1.

	ECA			LCA	MSA
	train	dev	test		
sentences	25k	6k	2.7k	114k	20k
# tokens	156k	31k	12k	476k	552k
# types	15k	5k	1.5k	16k	65k
# bigrams	81k	20k	7k	180k	336k
# trigrams	125k	26k	10k	320k	458k
% ambig.	24.4	27.8	28.2	—	—

Table 1: Corpus statistics for ECA, LCA and MSA.

The only resource we utilize in addition to raw data is the LDC-distributed Buckwalter stemmer. The stemmer analyzes MSA script forms and outputs all possible morphological analyses (stems and POS tags, as well as diacritizations) for the word. The analysis is based on an internal stem lexicon combined with rules for affixation. Although the stemmer was developed primarily for MSA, it can accommodate a certain percentage of dialectal words. Table 2 shows the percentages of word types and tokens in the ECA and LCA corpora that received an analysis from the Buckwalter stemmer. Since both sets contain dialectal as well as standard MSA forms, it is not possible to determine precisely how many of the unanalyzable forms are dialectal forms vs. words that were rejected for other reasons, such as misspellings. The higher percentage of rejected word types in the ECA corpus is most likely due to the non-standard script forms described above.

	Type		Token	
N	ECA	LCA	ECA	LCA
0	37.6	23.3	18.2	28.2
1	34.0	52.5	33.6	40.4
2	19.4	17.7	26.4	19.9
3	7.2	5.2	16.2	10.5
4	1.4	1.0	5.0	2.3
5	0.1	0.1	0.4	0.6

Table 2: Percentage of word types/tokens with N possible tags, as determined by the Buckwalter stemmer. Words with 0 tags are unanalyzable.

The POS tags used in the LDC ECA annotation and in the Buckwalter stemmer are rather fine-grained; furthermore, they are not identical. We therefore mapped both sets of tags to a unified, simpler tagset consisting only of the major POS categories listed in Table 2. The mapping from the original Buckwalter tag to the simplified set was based on the conversion scheme suggested in (Bies, 2003). The same underlying conversion rules were applicable to most of the LDC tags; those cases that could not be determined automatically were converted by hand.

Symbol	Gloss	(%)
CC	coordinating conjunction	7.15
DT	determiner	2.23
FOR	foreign word	1.18
IN	preposition	7.46
JJ	adjective	6.02
NN	noun	19.95
NNP	proper noun	3.55
NNS	non-singular noun	3.04
NOTAG	non-word	0.05
PRP	pronoun	5.85
RB	adverb	4.13
RP	particle	9.71
UH	disfluency, interjection	9.55
VBD	perfect verb	6.53
VBN	passive verb	1.88
VBP	imperfect verb	10.62
WP	relative pronoun	1.08

Table 3: Collapsed tagset and percentage of occurrence of each tag in the ECA corpus.

Prior to the development of our tagger we computed the cross-corpus coverage of word n-grams in the ECA development set, in order to verify our assumption that utilizing data from other dialects might be helpful. Table 4 demonstrates that the n-gram coverage of the ECA development set increases slightly by adding LCA and MSA data.

	Types			Tokens		
	1gr	2gr	3gr	1gr	2gr	3gr
ECA	64	33	12	94	58	22
LCA	31	9	1.4	69	20	3.7
ECA + LCA	68	35	13	95	60	23
MSA	32	3.7	0.2	66	8.6	0.3
ECA + MSA	71	34	12	95	60	22

Table 4: Percentages of n-gram types and tokens in ECA dev set that are covered by the ECA training set, the LCA set, combined ECA training + LCA set, and MSA sets. Note that adding the LCA or MSA improves the coverage slightly.

3 Baseline Tagger

We use a statistical trigram tagger in the form of a hidden Markov model (HMM). Let $w_{0:M}$ be a sequence of words $(w_0, w_1, \ldots, w_M)$ and $t_{0:M}$ be the corresponding sequence of tags. The trigram HMM computes the conditional probability of the word and tag sequence $p(w_{0:M}, t_{0:M})$ as:

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

The **lexical model** $p(w_i|t_i)$ characterizes the distribution of words for a specific tag; the **contextual model** $p(t_i|t_{i-1}, t_{i-2})$ is trigram model over the tag sequence. For notational simplicity, in subsequent sections we will denote $p(t_i|t_{i-1}, t_{i-2})$ as $p(t_i|h_i)$, where h_i represents the tag history. The HMM is trained to maximize the likelihood of the training data. In supervised training, both tag and word sequences are observed, so the maximum likelihood estimate is obtained by relative frequency counting, and, possibly, smoothing. During unsupervised training, the tag sequences are hidden, and the Expectation-Maximization Algorithm is used to iteratively update probabilities based on expected counts. Unsupervised tagging requires a lexicon specifying the set of possible tags for each word. Given a test sentence $w'_{0:M}$, the Viterbi algorithm is used to find the tag sequence maximizing the probability of tags given words: $t^*_{0:M} = argmax_{t_{0:M}} p(t_{0:M}|w'_{0:M})$. Our taggers are implemented using the Graphical Models Toolkit (GMTK) (Bilmes and Zweig, 2002), which allows training a wide range of probabilistic models with both hidden and observed variables.

As a first step, we compare the performance of four different baseline systems on our ECA dev set. First, we trained a supervised tagger on the MSA treebank corpus (System I), in order to gauge how a standard system trained on written Arabic performs on dialectal speech. The second system (System II) is a supervised tagger trained on the manual ECA POS annotations. System III is an unsupervised tagger on the ECA training set. The lexicon for this system is derived from the reference annotations of the training set – thus, the correct tag is not known during training, but the lexicon is constrained by expert information. The difference in accuracy between Systems II and III indicates the loss due to the unsupervised training method. Finally, we trained a system using only the unannotated ECA data and a lexicon generated by applying the MSA analyzer to the training corpus and collecting all resulting tags for each word. In this case, the lexicon is much less constrained; moreover, many words do not receive an output from the stemmer at all. This is the training method with the least amount of supervision and therefore the method we are interested in most.

Table 5 shows the accuracies of the four systems on the ECA development set. Also included is a breakdown of accuracy by analyzable (AW), unanalyzable (UW), and out-of-vocabulary (OOV) words. Analyzable words are those for which the stemmer outputs possible analyses; unanalyzable words cannot be processed by the stemmer. The percentage of unanalyzable word tokens in the dev set is 18.8%. The MSA-trained tagger (System I) achieves an accuracy of 97% on a held-out set (117k words) of MSA data, but performs poorly on ECA due to a high OOV rate (43%). By contrast, the OOV rate for taggers trained on ECA data is 9.5%. The minimally-supervised tagger (System IV) achieves a baseline accuracy of 62.76%. In the following sections, we present several methods to improve this system, in order to approximate as closely as possible the performance of the supervised systems. [1]

System	Total	AW	UW	OOV
I	39.84	55.98	21.05	19.21
II	92.53	98.64	99.09	32.20
III	84.88	90.17	99.11	32.64
IV	62.76	67.07	20.74	21.84

Table 5: Tagging accuracy (%) for the 4 baseline systems. AW = analyzable words, UW unanalyzable words, OOV = out-of-vocabulary words.

4 System Improvements

4.1 Adding Affix Features

The low accuracy of unanalyzable and OOV words may significantly impact downstream applications.

[1] The accuracy of a naive tagger which labels all words with the most likely tag (NN) achieves an accuracy of 20%. A tagger which labels words with the most likely tag amongst its possible tags achieves an accuracy of 52%.

One common way to improve accuracy is to add word features. In particular, we are interested in features that can be derived automatically from the script form, such as affixes. Affix features are added in a Naive Bayes fashion to the basic HMM model defined in Eq.1. In addition to the lexical model $p(w_i|t_i)$ we now have prefix and suffix models $p(a_i|t_i)$ and $p(b_i|t_i)$, where a and b are the prefix and suffix variables, respectively. The affixes used are: هم-, ها-, كم-, له-, لك-, لي-, ني-, ة-, ال-, فال-, وال-, بال-, لّ-. Affixes are derived for each word by simple substring matching. More elaborate techniques are not used due to the philosophy of staying within a minimally supervised approach that does not require dialect-specific knowledge.

4.2 Constraining the Lexicon

The quality of the lexicon has a major impact on unsupervised HMM training. Banko et. al. (2004) demostrated that tagging accuracy improves when the number of possible tags per word in a "noisy lexicon" can be restricted based on corpus frequency. In the current setup, words that are not analyzable by the MSA stemmer are initally assigned all possible tags, with the exception of obvious restricted tags like the begin and end-of-sentence tags, NOTAG, etc. Our goal is to constrain the set of tags for these unanalyzable words. To this end, we cluster both analyzable and unanalyzable words, and to reduce the set of possible tags for unanalyzable words based on its cluster membership. Several different clustering algorithms could in principle be used; here we utilize Brown's clustering algorithm (Brown and others, 1992), which iteratively merges word clusters with high mutual information based on distributional criteria. The tagger lexicon is then derived as follows:

1. Generate K clusters of words from data.
2. For each cluster C, calculate $P(t|C) = \sum_{w \in A,C} P(t|w)P(w|C)$ where t and w are the word and tag, and A is the set of analyzable words.
3. The cluster's tagset is determined by choosing all tags t' with $P(t'|C)$ above a certain threshold γ.
4. All unanalyzable words within this cluster are given these possible tags.

The number of clusters K and the threshold γ are variables that affect the final tagset for unanalyzable words. Using $K = 200$ and $\gamma = 0.05$ for instance, the number of tags per unanalyzable word reduces to an average of four and ranges from one to eight tags. There is a tradeoff regarding the degree of tagset reduction: choosing fewer tags results in less confusability but may also involve the removal of the correct tag from a word's lexicon entry. However, we have observed that tagset reduction generally leads to improvement compared to the baseline system with an unconstrained lexicon.

The improvements gained from adding affix features to System III and constraining the lexicon are shown in Table 6. We notice that adding affix features yields improvements in OOV accuracy. The relationship between the constrained lexicon and unanalyzable word accuracy is less straighforward. In this case, the degradation of unanalyzable word accuracy is due to the fact that the constrained lexicon over-restricts the tagsets of some frequent unanalyzable words. However, the constrained lexicon improves the overall accuracy in general, and is therefore a viable method for system improvement. Finally, the combination of affix features and constrained lexicon results in a tagger with 69.83% accuracy, which is a 7% absolute improvement over System IV.

System	Total	AW	UW	OOV
System IV	62.76	67.07	20.74	21.84
+affixes	67.48	71.30	22.82	**29.82**
+constrained lex	66.25	70.29	21.28	26.32
+both	**69.83**	**74.10**	**24.65**	27.68

Table 6: Improvements in tagging accuracy from adding affix features and constraining lexicon.

5 Cross-Dialectal Data Sharing

Next we examine whether unannotated corpora in other dialects (LCA) can be used to further improve the ECA tagger. The problem of data sparseness for Arabic dialects would be less severe if we were able to exploit the commonalities between similar dialects. In natural language processing, Kim & Khudanpur (2004) have explored techniques for using parallel corpora in Chinese and English for

the purpose of language modeling. Parallel corpora have also been used to infer morphological analyzers, POS taggers, and noun phrase bracketers by projections via word alignments (Yarowsky et al., 2001). In (Hana et al., 2004), Czech data is used to develop a morphological analyzer for Russian.

In contrast to these works, we do not require parallel or comparable corpora or a bilingual dictionary, since these resources may be difficult to obtain. Our goal is to develop general algorithms for utilizing the commonalities across dialects for developing a tool for a specific dialect. Although dialects can differ very strongly, they are similar in that they exhibit morphological simplifications and a different word order compared to MSA (e.g. SVO rather than VSO order), and close dialects may share part of their vocabularies.

Each of the tagger components (i.e. contextual model $p(t_i|h_i)$, lexical model $p(w_i|t_i)$, and affix model $p(a_i|t_i)p(b_i|t_i)$) can be shared during training. In the following, we present two approaches for sharing data between dialects, each derived from following different assumptions about the underlying data generation process.

5.1 Contextual Model Interpolation

Contextual model interpolation is widely-used data-sharing technique. Its underlying assumption is that models trained on data from different sources can be "mixed" in order to provide the most appropriate probability distribution for the target data. In our case, the we have LCA as an out-of-domain data source, and ECA as the in-domain data source, with the former being about 4 times larger than the latter. If properly combined, the larger amount of out-of-domain data might improve the robustness of the in-domain tagger. We therefore use a linear interpolation the in-domain and out-of-domain contextual models. The joint probability $p(w_{0:M}, t_{0:M})$ becomes:

$$\prod_{i=0}^{M} p_E(w_i|t_i)(\lambda p_E(t_i|h_i) + (1-\lambda)p_L(t_i|h_i)) \quad (2)$$

Here λ defines the interpolation weights for the ECA contextual model $p_E(t_i|h_i)$ and the LCA contextual model $p_L(t_i|h_i)$. $p_E(w_n|t_n)$ is the ECA lexical model. The interpolation weight λ is estimated by maximizing the likelihood of a held-out data set given the combined model. As an extension to the basic interpolation approach, we allow the interpolation weights to be a function of the current tag: $\lambda(t_i)$, since class-dependent interpolation has shown improvements over class-independent interpolation in many other applications, such as language modeling (Bulyko et al., 2003).

5.2 Joint Training of Contextual Model

As an alternative to model interpolation, we consider training a single model *jointly* from the two different data sets. The underlying assumption of this technique is that tag sequences in LCA and ECA are generated by the *same* process, whereas the observations (the words) are generated from the tag by two different processes in the two different dialects. The HMM model for joint training is expressed as:

$$\prod_{i=0}^{M}(\alpha_i p_E(w_i|t_i) + (1-\alpha_i)p_L(w_i|t_i))p_{E+L}(t_i|h_i)$$
$$\text{where} \quad \alpha_i = \begin{cases} 1 \text{ if word } w_i \text{ is ECA} \\ 0 \text{ otherwise} \end{cases} \quad (3)$$

A single conditional probability table is used for the contextual model, whereas the lexical model switches between two different parameter tables, one for LCA observations and another for ECA observations. During training, the contextual model is trained jointly from both ECA and LCA data; however, the data is divided into ECA and LCA portions when updating the lexical models. Both the contextual and lexical models are trained within the same training pass. A graphical model representation of this system is shown in Figure 1. This model can easily be implemented using the functionality of switching parents (Bilmes, 2000) provided by GMTK.

During decoding, the tagger can in principle switch its lexical model to ECA or LCA, depending on the input; this system thus is essentially a multi-dialect tagger. In the experiments reported below, however, we exclusively test on ECA, and the LCA lexical model is not used. Due to the larger amount of data available for contextual model, joint training can be expected to improve the performance on the target dialect. The affix models can be trained jointly in a similar fashion.

5.3 Data sharing results

The results for the two data sharing approaches are shown in Table 7. The systems Interpolate-λ and Interpolate-$\lambda(t_i)$ are taggers built by interpolation and class-dependent interpolation, respectively. For joint training, we present results for two systems: JointTrain(1:4) is trained on the existing collection ECA and LCA data, which has a 1:4 ratio in terms of corpus size; JointTrain(2:1) weights the ECA data twice, in order to bias the training process more towards ECA's distribution. In addition, we provide results for two additional taggers: the first (CombineData) is simply trained "naively" on the pooled data from both ECA and LCA, without any weighting, interpolation, or changes to the probabilistic model. The second (CombineLex) uses a contextual model trained on ECA and a lexical model estimated from both ECA and LCA data. The latter was trained in order to assess the potential for improvement due to the reduction in OOV rate on the dev set when adding the LCA data (cf. Table 4). All the above systems utilize the constrained lexicon, as it consistently gives improvements.

Table 7 shows that, as expected, the brute-force combination of training data is not helpful and degrades performance. Combining data for estimating the lexicon only results in higher accuracy but still does not attain the baseline performance. The same is true of the taggers using model interpolation. The best performance is obtained by the system using the joint contextual model with separate lexical models, with 2:1 weighting of ECA vs. LCA data. Finally, we added word affix information to the best shared-data system, which resulted in an accuracy of 70.88%. This result is directly comparable to the best system in Section 4 (last row of Table 6)[2].

The analysis of tagging errors revealed that the most frequent confusions are between VBD and NNS, VBP and VBD, and JJ and NN. Commonly mistagged words also includes cases like يعني, which is labeled as a particle in the ECA reference data but is most often tagged as a verb, which is also a possible tag for this word.

[2]We also experimented with joint training of ECA+MSA. This gave good OOV accuracy, but overall it did not improve over the best system in Section 4.

System	Total	AW	UW	OOV
CombineData	60.79	64.21	20.27	26.10
CombineLex	65.13	69.47	18.81	22.34
Interpolate-λ	62.82	67.42	16.98	17.44
Interpolate-$\lambda(t_i)$	63.49	67.96	17.19	19.33
JointTrain(1:4)	62.53	66.18	27.78	26.52
JointTrain(2:1)	**66.95**	**71.02**	**31.72**	**26.81**
JointTrain(2:1)+affix				
w/ ECA+LCA	**70.88**	**75.20**	28.17	**34.06**
w/ ECA+MSA	67.85	71.50	17.76	31.76

Table 7: Tagging accuracy for various data sharing methods.

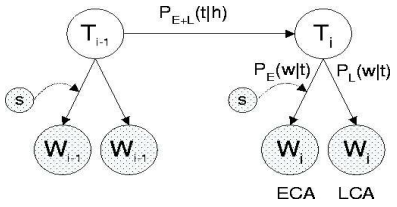

Figure 1: Graphical Model of Joint Training: switching between different lexical models while sharing the underlying contextual model. The variable s represents the α term in Eq. 3 and chooses the lexical model depending on the origin of the word.

6 Discussion and Future Work

Table 8 highlights the performance of the various taggers on the ECA evaluation set. The accuracy of the unsupervised HMM tagger (System IV) improves from 58.47% to 66.61% via the affix features and constrained lexicon, and to a 68.48% by including joint training.

Several of the methods proposed here deserve further work: first, additional ways of constraining the lexicon can be explored, which may also include imposing probability distributions on the possible tags for unanalyzable words. Other clustering algorithms (e.g. root-based clustering of Arabic words (De Roeck and Al-Fares, 2000)), may be used instead of, or in addition to, distribution-based clustering.

Cross-dialectal data sharing for tagging also deserves more research. For instance, the performance of the contextual model interpolation might be increased if one trains interpolation weights depen-

dent on the classes based on previous two tags. Joint training of contextual model and data sharing for lexical models can be combined; other dialectal data may also be added into the same joint training framework. Finally, we would like to extend these methods to create other dialectal Arabic tools, such as stemmer and a more fine-grained part-of-speech tagger. Stems, POS, and fine-grained POS can be combined into a factorial hidden Markov model framework, so that relationships between the stems and POS as well as data sharing between dialects can be simultaneously exploited to build a better system.

In conclusion, we have presented the first steps towards developing dialectal Arabic tagger with minimal supervision. We have shown that adding affix features and constraining the lexicon for unanalyzable words are simple resource-light methods to improve tagging accuracy. We also explore the possibility of improving an ECA tagger using LCA data and present two data sharing methods. The combination of these techniques yield a 10% improvement over the baseline.

System	Total	AW	UW	OOV
System IV	58.47	64.71	22.34	17.50
+affix+lexicon	66.61	72.87	20.17	**25.49**
Interpolate II	60.07	66.56	20.55	17.61
JointTr.+affix	**68.48**	**76.20**	**48.44**	17.76
CombineLex	61.35	68.12	16.02	16.87

Table 8: Tagging accuracy on ECA evaluation set

Acknowledgements

This material is based on work funded the NSF and the CIA under NSF Grant No. IIS-0326276. 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*.

A. Bies. 2003. Guide to collapsing Arabic tagset. http://www.ircs.upenn.edu/arabic/Jan03release/arabic-POStags-collapse-to-PennPOStags.txt.

J. Bilmes and G. Zweig. 2002. The Graphical Models Toolkit: an open-source software system for speech and time series processing. In *Proc. of ICASSP*.

J. Bilmes. 2000. Dynamic Bayesian multi-networks. In *The 16th Conf. on Uncertainty in Artificial Intelligence*.

T. Brants. 2000. TnT - a statistical part-of-speech tagger. In *Proc. of 6th Applied Natural Language Processing Conf.*

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

P.F. Brown et al. 1992. Class-based n-gram models of natural language. *Computational Linguistics*, 18(4):467–479.

I. Bulyko, M. Ostendorf, and A. Stolcke. 2003. Getting more mileage from web text sources for conversational speech language modeling using class-dependent mixtures. In *Proc. of HLT*.

D. Cutting et al. 1992. A practical part-of-speech tagger. In *Proc. of the 3rd Conference on Applied Natural Language Processing*.

A. De Roeck and W. Al-Fares. 2000. A morphologically sensitive clustering algorithm for identifying arabic roots. In *Proceedings of the 38th Annual Meeting of the ACL*.

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*.

J. Hana, A. Feldman, and C. Brew. 2004. A resource-light approach to russian morphology: Tagging russian using czech resources. In *Proc. of EMNLP 2004*, July.

P. Heeman. 1998. POS vs. classes in language modeling. In *Proc. of 6th Workshop on Very Large Corpora*, pages 179–187.

S. Khoja. 2001. APT: Arabic part-of-speech tagger. In *Proc. of the NAACL Student Workshop*.

W. Kim and S. Khudanpur. 2004. Lexical triggers and latent semantic analysis for cross-lingual language model adaptation. *ACM Trans. on Asian Language Info. Processing*, 3:94–112.

R. Koeling. 2000. Chunking with maximum entropy models. In *Proceedings of CoNLL*.

M. Osborne. 2000. Shallow parsing as part-of-speech tagging. In *Proc. of CoNLL*.

A. Rathnaparki. 1996. A maximum-entropy part-of-speech tagger. In *Proc. of EMNLP*.

K. Toutanova, D. Klein, C. Manning, and Y. Singer. 2003. Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proceedings of HLT-NAACL*.

D. Vergyri, K. Kirchhoff, K. Duh, and A. Stolcke. 2004. Morphology-based language modeling for Arabic speech recognition. In *Proc. of ICSLP*.

D. Yarowsky, G. Ngai, and R. Wicentowski. 2001. Inducing multilingual pos taggers and np bracketers via robust projection across aligned corpora. In *Proc. of HLT*.