Arabic Tokenization, Part-of-Speech Tagging and Morphological Disambiguation in One Fell Swoop

Nizar Habash and **Owen Rambow**
Center for Computational Learning Systems
Columbia University
New York, NY 10115, USA
{habash,rambow}@cs.columbia.edu

Abstract

We present an approach to using a morphological analyzer for tokenizing and morphologically tagging (including part-of-speech tagging) Arabic words in one process. We learn classifiers for individual morphological features, as well as ways of using these classifiers to choose among entries from the output of the analyzer. We obtain accuracy rates on all tasks in the high nineties.

1 Introduction

Arabic is a morphologically complex language.[1] The morphological analysis of a word consists of determining the values of a large number of (orthogonal) features, such as basic part-of-speech (i.e., noun, verb, and so on), voice, gender, number, information about the clitics, and so on.[2] For Arabic, this gives us about 333,000 theoretically possible completely specified morphological analyses, i.e., morphological tags, of which about 2,200 are actually used in the first 280,000 words of the Penn Arabic Treebank (ATB). In contrast, English morphological tagsets usually have about 50 tags, which cover all morphological variation.

As a consequence, **morphological disambiguation** of a word in context, i.e., choosing a complete morphological tag, cannot be done successfully using methods developed for English because of data sparseness. Hajič (2000) demonstrates convincingly that morphological disambiguation can be aided by a morphological analyzer, which, given a word without any context, gives us the set of all possible morphological tags. The only work on Arabic tagging that uses a corpus for training and evaluation (that we are aware of), (Diab et al., 2004), does not use a morphological analyzer. In this paper, we show that the use of a morphological analyzer outperforms other tagging methods for Arabic; to our knowledge, we present the best-performing wide-coverage tokenizer on naturally occurring input and the best-performing morphological tagger for Arabic.

2 General Approach

Arabic words are often ambiguous in their morphological analysis. This is due to Arabic's rich system of affixation and clitics and the omission of disambiguating short vowels and other orthographic diacritics in standard orthography ("undiacritized orthography"). On average, a word form in the ATB has about 2 morphological analyses. An example of a word with some of its possible analyses is shown in Figure 1. Analyses 1 and 4 are both nouns. They differ in that the first noun has no affixes, while the second noun has a conjunction prefix (+و +*w* 'and') and a pronominal possessive suffix (ي+ +*y* 'my').

In our approach, tokenizing and morphologically tagging (including part-of-speech tagging) are the same operation, which consists of three phases. First, we obtain from our morphological analyzer a list of all possible analyses for the words of a given sentence. We discuss the data and our lexicon in

[1]We would like to thank Mona Diab for helpful discussions. The work reported in this paper was supported by NSF Award 0329163. The authors are listed in alphabetical order.

[2]In this paper, we only discuss inflectional morphology. Thus, the fact that the stem is composed of a root, a pattern, and an infi x vocalism is not relevant except as it affects broken plurals and verb aspect.

Proceedings of the 43rd Annual Meeting of the ACL, pages 573–580,
Ann Arbor, June 2005. ©2005 Association for Computational Linguistics

#	lexeme	gloss	POS	Conj	Part	Pron	Det	Gen	Num	Per	Voice	Asp
1	wAliy	ruler	N	NO	NO	NO	NO	masc	sg	3	NA	NA
2	<ilaY	and to me	P	YES	NO	YES	NA	NA	NA	NA	NA	NA
3	waliy	and I follow	V	YES	NO	NO	NA	neut	sg	1	act	imp
4	|l	and my clan	N	YES	NO	YES	NO	masc	sg	3	NA	NA
5	|liy~	and automatic	AJ	YES	NO	NO	NO	masc	sg	3	NA	NA

Figure 1: Possible analyses for the word والي *wAly*

more detail in Section 4.

Second, we apply classifiers for ten morphological features to the words of the text. The full list of features is shown in Figure 2, which also identifies possible values and which word classes (POS) can express these features. We discuss the training and decoding of these classifiers in Section 5.

Third, we choose among the analyses returned by the morphological analyzer by using the output of the classifiers. This is a non-trivial task, as the classifiers may not fully disambiguate the options, or they may be contradictory, with none of them fully matching any one choice. We investigate different ways of making this choice in Section 6.

As a result of this process, we have the original text, with each word augmented with values for all the features in Figure 2. These values represent a complete morphological disambiguation. Furthermore, these features contain enough information about the presence of clitics and affixes to perform tokenization, for any reasonable tokenization scheme. Finally, we can determine the POS tag, for any morphologically motivated POS tagset. Thus, we have performed tokenization, traditional POS tagging, and full morphological disambiguation in one fell swoop.

3 Related Work

Our work is inspired by Hajič (2000), who convincingly shows that for five Eastern European languages with complex inflection plus English, using a morphological analyzer[3] improves performance of a tagger. He concludes that for highly inflectional languages "the use of an independent morphological dictionary is the preferred choice [over] more annotated data". Hajič (2000) uses a general exponential model to predict each morphological feature separately (such as the ones we have listed in Figure 2), but he trains different models for each ambiguity left unresolved by the morphological analyzer, rather than training general models. For all languages, the use of a morphological analyzer results in tagging error reductions of at least 50%.

We depart from Hajič's work in several respects. First, we work on Arabic. Second, we use this approach to also perform tokenization. Third, we use the SVM-based Yamcha (which uses Viterbi decoding) rather than an exponential model; however, we do not consider this difference crucial and do not contrast our learner with others in this paper. Fourth, and perhaps most importantly, we do not use the notion of ambiguity class in the feature classifiers; instead we investigate different ways of using the results of the individual feature classifiers in directly choosing among the options produced for the word by the morphological analyzer.

While there have been many publications on computational morphological analysis for Arabic (see (Al-Sughaiyer and Al-Kharashi, 2004) for an excellent overview), to our knowledge only Diab et al. (2004) perform a large-scale corpus-based evaluation of their approach. They use the same SVM-based learner we do, Yamcha, for three different tagging tasks: word tokenization (tagging on letters of a word), which we contrast with our work in Section 7; POS tagging, which we discuss in relation to our work in Section 8; and base phrase chunking, which we do not discuss in this paper. We take the comparison between our results on POS tagging and those of Diab et al. (2004) to indicate that the use of a morphological analyzer is beneficial for Arabic as

[3]Hajič uses a lookup table, which he calls a "dictionary". The distinction between table-lookup and actual processing at run-time is irrelevant for us.

Feature Name	Description	Possible Values	POS that Carry Feature	Default
POS	Basic part-of-speech	See Footnote 9	all	X
Conj	Is there a cliticized conjunction?	YES, NO	all	NO
Part	Is there a cliticized particle?	YES, NO	all	NO
Pron	Is there a pronominal clitic?	YES, NO	V, N, PN, AJ, P, Q	NO
Det	Is there a cliticized definite determiner +ال *Al*+?	YES, NO	N, PN, AJ	NO
Gen	Gender (intrinsic or by agreement)	masc(uline), fem(inine), neut(er)	V, N, PN, AJ, PRO, REL, D	masc
Num	Number	sg (singular), du(al), pl(ural)	V, N, PN, AJ, PRO, REL, D	sg
Per	Person	1, 2, 3	V, N, PN, PRO	3
Voice	Voice	act(ive), pass(ive)	V	act
Asp	Aspect	imp(erfective), perf(ective), imperative	V	perf

Figure 2: Complete list of morphological features expressed by Arabic morphemes that we tag; the last column shows on which parts-of-speech this feature can be expressed; the value 'NA' is used for each feature other than POS, Conj, and Part if the word is not of the appropriate POS

well.

Several other publications deal specifically with segmentation. Lee et al. (2003) use a corpus of manually segmented words, which appears to be a subset of the first release of the ATB (110,000 words), and thus comparable to our training corpus. They obtain a list of prefixes and suffixes from this corpus, which is apparently augmented by a manually derived list of other affixes. Unfortunately, the full segmentation criteria are not given. Then a trigram model is learned from the segmented training corpus, and this is used to choose among competing segmentations for words in running text. In addition, a huge unannotated corpus (155 million words) is used to iteratively learn additional stems. Lee et al. (2003) show that the unsupervised use of the large corpus for stem identification increases accuracy. Overall, their error rates are higher than ours (2.9% vs. 0.7%), presumably because they do not use a morphological analyzer.

There has been a fair amount of work on entirely unsupervised segmentation. Among this literature, Rogati et al. (2003) investigate unsupervised learning of stemming (a variant of tokenization in which only the stem is retained) using Arabic as the example language. Unsurprisingly, the results are much worse than in our resource-rich approach. Darwish (2003) discusses unsupervised identification of roots; as mentioned above, we leave root identification to future work.

4 Preparing the Data

The data we use comes from the Penn Arabic Treebank (Maamouri et al., 2004). Like the English Penn Treebank, the corpus is a collection of news texts. Unlike the English Penn Treebank, the ATB is an ongoing effort, which is being released incrementally. As can be expected in this situation, the annotation has changed in subtle ways between the incremental releases. Even within one release (especially the first) there can be inconsistencies in the annotation. As our approach builds on linguistic knowledge, we need to carefully study how linguistic facts are represented in the ATB. In this section, we briefly summarize how we obtained the data in the representation we use for our machine learning experiments.[4]

We use the first two releases of the ATB, ATB1 and ATB2, which are drawn from different news sources. We divided both ATB1 and ATB2 into de-

[4]The code used to obtain the representations is available from the authors upon request.

velopment, training, and test corpora with roughly 12,000 word tokens in each of the development and test corpora, and 120,000 words in each of the training corpora. We will refer to the training corpora as TR1 and TR2, and to the test corpora as, TE1 and TE2. We report results on both TE1 and TE2 because of the differences in the two parts of the ATB, both in terms of origin and in terms of data preparation.

We use the ALMORGEANA morphological analyzer (Habash, 2005), a lexeme-based morphological generator and analyzer for Arabic.[5] A sample output of the morphological analyzer is shown in Figure 1. ALMORGEANA uses the databases (i.e., lexicon) from the Buckwalter Arabic Morphological Analyzer, but (in analysis mode) produces an output in the lexeme-and-feature format (which we need for our approach) rather than the stem-and-affix format of the Buckwalter analyzer. We use the data from first version of the Buckwalter analyzer (Buckwalter, 2002). The first version is fully consistent with neither ATB1 nor ATB2.

Our training data consists of a set of all possible morphological analyses for each word, with the unique correct analysis marked. Since we want to learn to choose the correct output using the features generated by ALMORGEANA, the training data must also be in the ALMORGEANA output format. To obtain this data, we needed to match data in the ATB to the lexeme-and-feature representation output by ALMORGEANA. The matching included the use of some heuristics, since the representations and choices are not always consistent in the ATB. For example, نحو *nHw* 'towards' is tagged as AV, N, or V (in the same syntactic contexts). We verified whether we introduced new errors while creating our data representation by manually inspecting 400 words chosen at random from TR1 and TR2. In eight cases, our POS tag differed from that in the ATB file; all but one case were plausible changes among Noun, Adjective, Adverb and Proper Noun resulting from missing entries in the Buckwalter's lexicon. The remaining case was a failure in the conversion process relating to the handling of broken plurals at the lexeme level. We conclude that our data representation provides an adequate basis for performing machine learning experiments.

[5]The ALMORGEANA engine is available at http://clipdemos.umiacs.umd.edu/ALMORGEANA/.

An important issue in using morphological analyzers for morphological disambiguation is what happens to *unanalyzed* words, i.e., words that receive no analysis from the morphological analyzer. These are frequently proper nouns; a typical example is برلوسكوني *brlwskwny* 'Berlusconi', for which no entry exists in the Buckwalter lexicon. A backoff analysis mode in ALMORGEANA uses the morphological databases of prefixes, suffixes, and allowable combinations from the Buckwalter analyzer to hypothesize all possible stems along with feature sets. Our Berlusconi example yields 41 possible analyses, including the correct one (as a singular masculine PN). Thus, with the backoff analysis, unanalyzed words are distinguished for us only by the larger number of possible analyses (making it harder to choose the correct analysis). There are not many unanalyzed words in our corpus. In TR1, there are only 22 such words, presumably because the Buckwalter lexicon our morphological analyzer uses was developed onTR1. In TR2, we have 737 words without analysis (0.61% of the entire corpus, giving us a coverage of about 99.4% on domain-similar text for the Buckwalter lexicon).

In ATB1, and to a lesser degree in ATB2, some words have been given no morphological analysis. (These cases are not necessarily the same words that our morphological analyzer cannot analyze.) The POS tag assigned to these words is then NO_FUNC. In TR1 (138,756 words), we have 3,088 NO_FUNC POS labels (2.2%). In TR2 (168,296 words), the number of NO_FUNC labels has been reduced to 853 (0.5%). Since for these cases, there is no meaningful solution in the data, we have removed them from the evaluation (but not from training). In contrast, Diab et al. (2004) treat NO_FUNC like any other POS tag, but it is unclear whether this is meaningful. Thus, when comparing results from different approaches which make different choices about the data (for example, the NO_FUNC cases), one should bear in mind that small differences in performance are probably not meaningful.

5 Classifiers for Linguistic Features

We now describe how we train classifiers for the morphological features in Figure 2. We train one classifier per feature. We use Yamcha (Kudo and Matsumoto, 2003), an implementation of support vector machines which includes Viterbi decoding.[6] As training features, we use two sets. These sets are based on the ten morphological features in Figure 2, plus four other "hidden" morphological features, for which we do not train classifiers, but which are represented in the analyses returned by the morphological analyzer. The reason we do not train classifiers for the hidden features is that they are only returned by the morphological analyzer when they are marked overtly in orthography, but they are not disambiguated in case they are not overtly marked. The features are indefiniteness (presence of nunation), idafa (possessed), case, and mood. First, for each of the 14 morphological features and for each possible value (including 'NA' if applicable), we define a binary machine learning feature which states whether in *any* morphological analysis for that word, the feature has that value. This gives us 58 machine learning features per word. In addition, we define a second set of features which abstracts over the first set: for all features, we state whether any morphological analysis for that word has a value other than 'NA'. This yields a further 11 machine learning features (as 3 morphological features never have the value 'NA'). In addition, we use the untokenized word form and a binary feature stating whether there is an analysis or not. This gives us a total of 71 machine learning features per word. We specify a window of two words preceding and following the current word, using all 71 features for each word in this 5-word window. In addition, two dynamic features are used, namely the classification made for the preceding two words. For each of the ten classifiers, Yamcha then returns a confidence value for each possible value of the classifier, and in addition it marks the value that is chosen during subsequent Viterbi decoding (which need not be the value with the highest confidence value because of the inclusion of dynamic features).

We train on TR1 and report the results for the ten Yamcha classifiers on TE1 and TE2, using all simple tokens,[7] including punctuation, in Figure 3. The baseline BL is the most common value associated in the training corpus TR1 with every feature for a given word form (unigram). We see that the baseline for TE1 is quite high, which we assume is due to the fact that when there is ambiguity, often one interpretation is much more prevelant than the others. The error rates on the baseline approximately double on TE2, reflecting the difference between TE2 and TR1, and the small size of TR1. The performance of our classifiers is good on TE1 (third column), and only slightly worse on TE2 (fifth column). We attribute the increase in error reduction over the baseline for TE2 to successfully learned generalizations.

Method	BL	Class	BL	Class
Test	TE1	TE1	TE2	TE2
POS	96.6	97.7	91.1	95.5
Conj	99.9	99.9	99.7	99.9
Part	99.9	99.9	99.5	99.7
Pron	99.5	99.6	98.8	99.0
Det	98.8	99.2	96.8	98.3
Gen	98.6	99.2	95.8	98.2
Num	98.8	99.4	96.8	98.8
Per	97.6	98.7	94.8	98.1
Voice	98.8	99.3	97.5	99.0
Asp	98.8	99.4	97.4	99.1

Figure 3: Accuracy of classifiers (Class) for morphological features trained on TR1, and evaluated on TE1 and TE2; BL is the unigram baseline trained on TR1

We investigated the performance of the classifiers on unanalyzed words. The performance is generally below the baseline BL. We attribute this to the almost complete absence of unanalyzed words in training data TR1. In future work we could attempt to improve performance in these cases; however, given their small number, this does not seem a priority.

[6]We use Yamcha's default settings: standard SVM with 2nd degree polynomial kernel and 1 slack variable.

[7]We use the term *orthographic token* to designate tokens determined only by white space, while *simple tokens* are orthographic tokens from which punctuation has been segmented (becoming its own token), and from which all tatweels (the elongation character) have been removed.

6 Choosing an Analysis

Once we have the results from the classifiers for the ten morphological features, we combine them to choose an analysis from among those returned by the morphological analyzer. We investigate several options for how to do this combination. In the following, we use two numbers for each analysis. First, the **agreement** is the number of classifiers agreeing with the analysis. Second, the **weighted agreement** is the sum, over all classifiers, of the classification confidence measure of that value that agrees with the analysis. The agreement, but not the weighted agreement, uses Yamcha's Viterbi decoding.

• The majority combiner (Maj) chooses the analysis with the largest agreement.

• The confidence-based combiner (Con) chooses the analysis with the largest weighted agreement.

• The additive combiner (Add) chooses the analysis with the largest sum of agreement and weighted agreement.

• The multiplicative combiner (Mul) chooses the analysis with the largest product of agreement and weighted agreement.

• We use Ripper (Cohen, 1996) to learn a rule-based classifier (Rip) to determine whether an analysis from the morphological analyzer is a "good" or a "bad" analysis. We use the following features for training: for each morphological feature in Figure 2, we state whether or not the value chosen by its classifier agrees with the analysis, and with what confidence level. In addition, we use the word form. (The reason we use Ripper here is because it allows us to learn lower bounds for the confidence score features, which are real-valued.) In training, only the correct analysis is good. If exactly one analysis is classified as good, we choose that, otherwise we use Maj to choose.

• The baseline (BL) chooses the analysis most commonly assigned in TR1 to the word in question. For unseen words, the choice is made randomly.

In all cases, any remaining ties are resolved randomly.

We present the performance in Figure 4. We see that the best performing combination algorithm on TE1 is Maj, and on TE2 it is Rip. Recall that the Yamcha classifiers are trained on TR1; in addition, Rip is trained on the output of these Yamcha classifiers on TR2. The difference in performance between TE1 and TE2 shows the difference between the ATB1 and ATB2 (different source of news, and also small differences in annotation). However, the results for Rip show that retraining the Rip classifier on a new corpus can improve the results, without the need for retraining all ten Yamcha classifiers (which takes considerable time).

Corpus	TE1		TE2	
Method	All	Words	All	Words
BL	92.1	90.2	87.3	85.3
Maj	96.6	95.8	94.1	93.2
Con	89.9	87.6	88.9	87.2
Add	91.6	89.7	90.7	89.2
Mul	96.5	95.6	94.3	93.4
Rip	96.2	95.3	94.8	94.0

Figure 4: Results (percent accuracy) on choosing the correct analysis, measured per token (including and excluding punctuation and numbers); BL is the baseline

Figure 4 presents the accuracy of tagging using the whole complex morphological tagset. We can project this complex tagset to a simpler tagset, for example, POS. Then the minimum tagging accuracy for the simpler tagset must be greater than or equal to the accuracy of the complex morphological tagset. Even if a combining algorithm chooses the wrong analysis (and this is counted as a failure for the evaluation in this section), the chosen analysis may agree with some of the correct morphological features. We discuss our performance on the POS feature in Section 8.

7 Evaluating Tokenization

The term "tokenization" refers to the segmenting of a naturally occurring input sequence of orthographic symbols into elementary symbols ("tokens") used in subsequent processing steps (such as parsing) as basic units. In our approach, we determine all morphological properties of a word at once, so we can use this information to determine tokenization. There is not a single possible or obvious tokenization scheme: a tokenization scheme is an analytical tool devised by the researcher. We evaluate in this section how well our morphological disambiguation

Meth.	Word Acc.	Token Acc.	Token Prec.	Token Rec.	Token F-m.
BL	99.1	99.6	98.6	99.1	98.8
Maj	99.3	99.6	98.9	99.3	99.1

Figure 5: Results of tokenization on TE1: word accuracy measures for each input word whether it gets tokenized correctly, independently of the number of resulting tokens; the token-based measures refer to the four token fields into which the ATB splits each word

determines the ATB tokenization. The ATB starts with a simple tokenization, and then splits the word into four fields: conjunctions; particles (prepositions in the case of nouns); the word stem; and pronouns (object clitics in the case of verbs, possessive clitics in the case of nouns). The ATB does not tokenize the definite article +ال *Al+*.

We compare our output to the morphologically analyzed form of the ATB, and determine if our morphological choices lead to the correct identification of those clitics that need to be stripped off.[8] For our evaluation, we only choose the Maj chooser, as it performed best on TE1. We evaluate in two ways. In the first evaluation, we determine for each simple input word whether the tokenization is correct (no matter how many ATB tokens result). We report the percentage of words which are correctly tokenized in the second column in Figure 5. In the second evaluation, we report on the number of output tokens. Each word is divided into exactly four token fields, which can be either filled or empty (in the case of the three clitic token fields) or correct or incorrect (in the case of the stem token field). We report in Figure 5 accuracy over all token fields for all words in the test corpus, as well as recall, precision, and f-measure for the non-null token fields. The baseline BL is the tokenization associated with the morphological analysis most frequently chosen for the input word in training.

While the token-based evaluation is identical to that performed by Diab et al. (2004), the results are not directly comparable as they did not use actual input words, but rather recreated input words from the regenerated tokens in the ATB. Sometimes this can simplify the analysis: for example, a ة *p (ta marbuta)* must be word-final in Arabic orthography, and thus a word-medial ة *p* in a recreated input word reliably signals a token boundary. The rather high baseline shows that tokenization is not a hard problem.

8 Evaluating POS Tagging

The POS tagset Diab et al. (2004) use is a subset of the tagset for English that was introduced with the English Penn Treebank. The large set of Arabic tags has been mapped (by the Linguistic Data Consortium) to this smaller English set, and the meaning of the English tags has changed. We consider this tagset unmotivated, as it makes morphological distinctions because they are marked in English, not Arabic. The morphological distinctions that the English tagset captures represent the complete morphological variation that can be found in English. However, in Arabic, much morphological variation goes untagged. For example, verbal inflections for subject person, number, and gender are not marked; dual and plural are not distinguished on nouns; and gender is not marked on nouns at all. In Arabic nouns, arguably the gender feature is the more interesting distinction (rather than the number feature) as verbs in Arabic always agree with their nominal subjects in gender. Agreement in number occurs only when the nominal subject precedes the verb. We use the tagset here only to compare to previous work. Instead, we advocate using a reduced part-of-speech tag set,[9] along with the other orthogonal linguistic features in Figure 2.

We map our best solutions as chosen by the Maj model in Section 6 to the English tagset, and we furthermore assume (as do Diab et al. (2004)) the gold standard tokenization. We then evaluate against the gold standard POS tagging which we have mapped

[8]The ATB generates normalized forms of certain clitics and of the word stem, so that the resulting tokens are not simply the result of splitting the original words. We do not actually generate the surface token form from our deep representation, but this can be done in a deterministic, rule-based manner, given our rich morphological analysis, e.g., by using ALMORGEANA in generation mode after splitting off all separable tokens.

[9] We use V (Verb), N (Noun), PN (Proper Noun), AJ (Adjective), AV (Adverb), PRO (Nominal Pronoun), P (Preposition/Particle), D (Determiner), C (Conjunction), NEG (Negative particle), NUM (Number), AB (Abbreviation), IJ (Interjection), PX (Punctuation), and X (Unknown).

Corpus		TE1		TE2	
Method	Tags	All	Words	All	Words
BL	PTB	93.9	93.3	90.9	89.8
	Smp	94.9	94.3	92.6	91.4
Maj	PTB	97.6	97.5	95.7	95.2
	Smp	98.1	97.8	96.5	96.0

Figure 6: Part-of-speech tagging accuracy measured for all tokens (based on gold-standard tokenization) and only for word tokens, using the Penn Treebank (PTB) tagset as well as the smaller tagset (Smp) (see Footnote 9); BL is the baseline obtained by using the POS value from the baseline tag used in Section 6

similarly. We obtain a score for TE1 of 97.6% on all tokens. Diab et al. (2004) report a score of 95.5% for all tokens on a test corpus drawn from ATB1, thus their figure is comparable to our score of 97.6%. On our own reduced POS tagset, evaluating on TE1, we obtain an accuracy score of 98.1% on all tokens. The full dataset is shown in Figure 6.

9 Conclusion and Outlook

We have shown how to use a morphological analyzer for tokenization, part-of-speech tagging, and morphological disambiguation in Arabic. We have shown that the use of a morphological analyzer is beneficial in POS tagging, and we believe our results are the best published to date for tokenization of naturally occurring input (in undiacritized orthography) and POS tagging.

We intend to apply our approach to Arabic dialects, for which currently no annotated corpora exist, and for which very few written corpora of any kind exist (making the dialects bad candidates even for unsupervised learning). However, there is a fair amount of descriptive work on dialectal morphology, so that dialectal morphological analyzers may be easier to come by than dialect corpora. We intend to explore to what extent we can transfer models trained on Standard Arabic to dialectal morphological disambiguation.

References

Imad A. Al-Sughaiyer and Ibrahim A. Al-Kharashi. 2004. Arabic morphological analysis techniques: A comprehensive survey. *Journal of the American Society for Information Science and Technology*, 55(3):189–213.

Tim Buckwalter. 2002. Buckwalter Arabic Morphological Analyzer Version 1.0. Linguistic Data Consortium, University of Pennsylvania, 2002. LDC Catalog No.: LDC2002L49.

William Cohen. 1996. Learning trees and rules with set-valued features. In *Fourteenth Conference of the American Association of Artificial Intelligence*. AAAI.

Kareem Darwish. 2003. Building a shallow Arabic morphological analyser in one day. In *ACL02 Workshop on Computational Approaches to Semitic Languages*, Philadelpia, PA. Association for Computational Linguistics.

Mona Diab, Kadri Hacioglu, and Daniel Jurafsky. 2004. Automatic tagging of arabic text: From raw text to base phrase chunks. In *5th Meeting of the North American Chapter of the Association for Computational Linguistics/Human Language Technologies Conference (HLT-NAACL04)*, Boston, MA.

Nizar Habash. 2005. Arabic morphological representations for machine translation. In Abdelhadi Soudi, Antal van den Bosch, and Guenter Neumann, editors, *Arabic Computational Morphology: Knowledge-based and Empirical Methods*, Text, Speech, and Language Technology. Kluwer/Springer. in press.

Jan Hajič. 2000. Morphological tagging: Data vs. dictionaries. In *1st Meeting of the North American Chapter of the Association for Computational Linguistics (NAACL'00)*, Seattle, WA.

Taku Kudo and Yuji Matsumoto. 2003. Fast methods for kernel-based text analysis. In *41st Meeting of the Association for Computational Linguistics (ACL'03)*, Sapporo, Japan.

Young-Suk Lee, Kishore Papineni, Salim Roukos, Ossama Emam, and Hany Hassan. 2003. Language model based Arabic word segmentation. In *41st Meeting of the Association for Computational Linguistics (ACL'03)*, pages 399–406, Sapporo, Japan.

Mohamed Maamouri, Ann Bies, and Tim Buckwalter. 2004. The penn arabic treebank : Building a large-scale annotated arabic corpus. In *NEMLAR Conference on Arabic Language Resources and Tools*, Cairo, Egypt.

Monica Rogati, J. Scott McCarley, and Yiming Yang. 2003. Unsupervised learning of arabic stemming using a parallel corpus. In *41st Meeting of the Association for Computational Linguistics (ACL'03)*, pages 391–398, Sapporo, Japan.