Smoothing a Lexicon-based POS Tagger for Arabic and Hebrew

Saib Mansour
Computer Science, Technion
Haifa, 32000, Israel
saib@cs.technion.ac.il

Khalil Sima'an
ILLC
Universiteit van Amsterdam
Amsterdam, The Netherlands
simaan@science.uva.nl

Yoad Winter
Computer Science, Technion
Haifa, 32000, Israel
and Netherlands Institute for Advanced Study
Wassenaar, The Netherlands
winter@cs.technion.ac.il

Abstract

We propose an enhanced Part-of-Speech (POS) tagger of Semitic languages that treats Modern Standard Arabic (henceforth *Arabic*) and Modern Hebrew (henceforth *Hebrew*) using the same probabilistic model and architectural setting. We start out by porting an existing Hidden Markov Model POS tagger for Hebrew to Arabic by exchanging a morphological analyzer for Hebrew with Buckwalter's (2002) morphological analyzer for Arabic. This gives state-of-the-art accuracy (96.12%), comparable to Habash and Rambow's (2005) analyzer-based POS tagger on the same Arabic datasets. However, further improvement of such analyzer-based tagging methods is hindered by the incomplete coverage of standard morphological analyzer (Bar Haim et al., 2005). To overcome this coverage problem we supplement the output of Buckwalter's analyzer with synthetically constructed analyses that are proposed by a model which uses character information (Diab et al., 2004) in a way that is similar to Nakagawa's (2004) system for Chinese and Japanese. A version of this extended model that (unlike Nakagawa) incorporates synthetically constructed analyses also for known words achieves 96.28% accuracy on the standard Arabic test set.

1 Introduction

Part-of-Speech tagging for Semitic languages has been an active topic of research in recent years. (Diab et al., 2004; Habash and Rambow, 2005; Bar-Haim et al., 2005) are some examples for this line of work on Modern Standard Arabic and Modern Hebrew. POS tagging systems aim at classifying input sequences of lexemes by assigning each such sequence a corresponding sequence of most probable POS tags. It is often assumed that for each input lexeme there is a set of *a priori* possible POS tag categories, or a probability function over them, and the tagger has to choose from this limited set of candidate categories. We henceforth use the term *lexicon* to refer to the set of lexemes in a language and the mapping that assigns each of them candidate POS tags, possibly with additional probabilities.

Two ways to obtain a lexicon can be distinguished in recent works on POS tagging in Semitic languages. *Data-driven* approaches like (Diab et al. 2004) employ the lexicon only implicitly when extracting features on possible POS tags from annotated corpora that are used for training the POS tagger. *Lexicon-based* approaches (Habash and Rambow, 2005; Bar-Haim et al., 2005) use a lexicon that is extracted from a manually constructed morphological analyzer (Buckwalter 2002 and Segal 2001 respectively).

In this paper we show that although lexicon-based taggers for Arabic and Hebrew may initially outperform data-driven taggers, they do not exhaust the advantages of data-driven approaches.

Proceedings of the 5th Workshop on Important Unresolved Matters, pages 97–103,
Prague, Czech Republic, June 2007.

Consequently, we propose a hybrid model of data-driven methods and lexicon-based methods, and show its advantages over both models, in a way that is reminiscent of Nakagawa's (2004) results for Chinese and Japanese.

As a first step, we develop a Part-of-Speech tagger that treats Arabic and Hebrew using the same probabilistic model and architectural setting. We start out from MorphTagger, a lexicon-based tagger for Hebrew developed by Bar-Haim et al. (2005), which uses standard Hidden Markov Model techniques. We port the existing MorphTagger implementation to Arabic by exchanging Segal's (2001) morphological analyzer with Buckwalter's (2002) morphological analyzer, and then training the tagger on the Arabic Treebank (Maamouri et al., 2001). Remarkably, this gives state-of-the-art accuracy (96.12%) on the same Arabic datasets as Habash and Rambow (2005). To the best of our knowledge, this is the first time the same POS tagging architecture is used both for Arabic and Hebrew texts with comparable accuracy.

Despite the initial advantages of this setting, our empirical study shows that in both languages, further improvement in accuracy is hindered by the incompleteness of the morphological analyzer. By "incompleteness" we refer not only to the well-studied problem of unknown words (out-of-vocabulary). Our results show that for both Arabic and Hebrew, a more serious problem involves words for which the analyzer provides a set of analyses that does not contain the correct one. We find out that this is the case for 3% of the words in the development set. This obviously sets an upper bound on tagger accuracy using methods that are purely based on a manually constructed lexicon. We refer to this problem as the "incomplete lexicon" problem.

We focus on devising a solution to the incomplete lexicon problem by smoothing. We supplement the output of Buckwalter's analyzer with synthetically constructed analyses that are proposed by a model which uses character information (Diab et al., 2004) in a way that is similar to Nakagawa's (2004) system for Japanese. Unlike Nakagawa's method, however, our smoothing method incorporates synthetically constructed analyses also for known words, though only when all available taggings of the sentence have low probabilities according to our model. A version of this extended model achieves a modest improvement (96.28%) in accuracy over the baseline on the standard Arabic test set.

This paper is structured as follows. In section 2 we start with a brief discussion of previous work. Section 3 describes our adaptation of Bar Haim et al.'s POS tagging system to Arabic. In section 4 we show that an architecture like Bar Haim et al.'s, which relies on a morphological analyzer, is likely to suffer from coverage problems under any configuration where it is used as a stand-alone. In section 5 we present our new architecture and the method of combining the models. Section 6 concludes.

2 Relation to Previous Works

Quite a few works have dealt with extending a given POS tagger, mainly by smoothing it using extra-information about untreated words. For example, (Church, 1988) uses the simple heuristic of predicting proper nouns from capitalization. This method is not applicable to Arabic and Hebrew, which lack typographical marking of proper nouns. More advanced methods like those described by Weischedel et al. (1993) incorporate the treatment of unknown words within the probability model. Weischedel et al. use derivational and inflectional endings to infer POS tags of unknown words. Nakagawa (2004) addresses the problem of unknown words for Japanese and Chinese, and uses a hybrid method of word-level and character-level information. In his model, Nakagawa uses character information (only) when handling unknown words, claiming that in word-level methods information about known words helps to achieve higher accuracy compared to character-level models. On the other hand, when it comes to unknown words, Nakagawa uses a character-level method, which is hypothesized to be more robust in such cases than word-level methods.

Virtually all works that dealt with coverage problems of POS taggers have concentrated on the problem of "unknown" words – words that have no analysis in the initial tagging system. However, in the context of analyzer-based tagging systems, we also have to deal with the problem of "known" words that miss the correct analysis in the morphological analyzer. In the Arabic and Hebrew datasets we have examined, this problem is more severe than the unknown words problem. Unlike previous works, we propose to smooth the word-segment driven model also for "known" words. To avoid overgeneration, this is done only when all taggings of the sentence have low probability.

3 Adapting a Hebrew POS-tagger to Arabic

Bar Haim et al.'s (2005) POS tagging system, MorphTagger, was developed initially for Hebrew. Our work is mainly developed for Arabic and tested over Arabic data. Due to the similarity in the morphological processes in Hebrew and Arabic and the generality of Bar Haim et al.'s architecture, the adaptation process was fairly simple. However, as far as we know this is the first implementation of a unified model for Arabic and Hebrew that achieves state-of-the-art accuracy. MorphTagger requires two components: a morphological analyzer to produce a set of analyses for every lexeme, and a POS tagged corpus for acquiring an HMM disambiguator. The HMM disambiguator assigns a probability to every pair $\langle w_1^n, t_1^n \rangle$, where $w_1^n = w_1 ... w_n$ is a sentence and $t_1^n = t_1 ... t_n$ a corresponding sequence of POS tags hypothesized by the analyzer. This probability is approximated in a standard HMM fashion:

$$P(w_1^n, t_1^n) = P(t_1^n) P(w_1^n \mid t_1^n) = \prod_{i=1}^{n} P(t_i \mid t_{i-1}, t_{i-2}) P(w_i \mid t_i)$$

For an input sentence w_1^n, the pair $\langle w_1^n, t_1^n \rangle$ with the highest probability is selected. The language ($P(t_i \mid t_{i-1}, t_{i-2})$) and lexical ($P(w_i \mid t_i)$) models' parameters are estimated from the tagged corpus by Maximum-Likelihood Estimator (MLE) followed by Katz backoff smoothing for the language model and Add-λ smoothing for the lexical model, where a small λ=1 count is given to analyzes provided by the analyzer but not found in the training corpus. Furthermore, MorphTagger employs an array of other smoothing techniques explained in Bar Haim et al. (2005).

Our implementation of MorphTagger for Arabic was developed using Buckwalter's (2002) Morphological Analyzer v1.0 (BMA1.0), and the Arabic Treebank part 1 v2.0 (ATB1), Part 2 v2.0 (ATB2) and Part 3 v1.0 (ATB3). The ATB was chosen not only because of its size and comprehensiveness, but also because Buckwalter's analyzer was developed in accordance with the ATB, which makes the task of combining information from both sources easier. In all our experiments we use a tag-set of 24 tags which was mapped from the original tag-set (191 tags in ATB1) using the mapping script of the ATB distribution.

To check the ambiguity level and the difficulty of the task at hand, we ran BMA1.0 over a testing set extracted from ATB1. The average number of analyses per word is 1.83, and the average number of segmentations per word is 1.2, however, the task of disambiguating Arabic is still not easy, as 46% of the data is ambiguous. Those results are comparable to the results of Bar Haim et al. for Hebrew, according to which the average number of analyses per word is 2.17 with 1.25 segmentations on average per word, and 54% of the words are ambiguous.

The performance of MorphTagger over Arabic was measured using the same test settings of Diab et al. (2004). Habash and Rambow (2005) use a different test setting drawn from ATB1. Although we could not reproduce the exact setting of Habash and Rambow, comparison to their reported accuracy is still quite telling due to the similarity of the data. The comparison between the accuracy of the three systems is summarized in Table 1. The results in this table were obtained using the correct ("gold") segmentation and applying the standard F-measure for POS tagging accuracy. The result of Diab et al. was reproduced on their setting, and the result of Habash and Rambow is as reported in their paper.

System	Tagging accuracy
MorphTagger	96.12
Diab et al.	95.81
Habash and Rambow	97.5

Table 1 - Comparison between systems over ATB1

The result achieved by MorphTagger slightly exceeds Diab et al.'s result (on the same test setting) and is slightly inferior to Habash and Rambow's reported result. Overall, it is an encouraging result that the MorphTagger system that was developed for Hebrew could be easily ported to Arabic and yield state-of-the-art results.

In Table 2, we present the accuracies achieved for MorphTagger on a cross validated, 10-fold test, including the standard deviation results in parentheses. The results are reported both for gold-segmentation (GS) and without GS.

Test setting	Accuracy per word (%)		$F_{\beta=1}$ per Word-segment (%)	
	Segmentation	Tagging	Segmentation	Tagging
GS	100	94.89 (0.62)	100	95.436 (0.53)
without GS	99.015 (0.24)	94.374 (0.64)	98.854 (0.28)	94.727 (0.56)

Table 2 - MorphTagger performance cross validated

Note that by tagging accuracy per word we mean the percentage of words correctly segmented and tagged. The tagging F-measure is calculated in the standard way, counting the correctly tagged word-segments and dividing it by the number of "gold" word-segments for recall, and further by the number of outputted word-segments for precision.

Analyzing the POS tagging errors of MorphTagger, we found that about 2.8% of the words in ATB1 were not correctly analyzed by the morphological analyzer. Such "incomplete lexicon" problems inevitably lead to tagging errors in MorphTagger's architecture. This problem is more serious still on data taken from ATB2 and ATB3, where respectively 4.5% and 5.3% of the data led to "incomplete lexicon" problems. We conclude that a morphological analyzer can be used to improve upon Diab et al.'s results, as done in Habash and Rambow and in our straightforward application of MorphTagger to Arabic. However, this method still suffers from considerable coverage problems, which are discussed in the following section.

4 Coverage of Morphological Analysis for Arabic

In order to analyze the coverage problem, we tested the coverage of BMA1.0 over parts of the ATB which were composed from articles taken on different periods of times. The results are summarized in Table 3. The schema of the table includes, for each part of the ATB: (i) the number of tokens that include at least one Arabic character (henceforth "*Arabic words*"[1]); (ii) Out-of-Vocabulary (OOV) words, unanalyzed by BMA1.0; (iii) the percentage of proper nouns (NNP) out of the OOV words; (iv) the number of "no correct" words – words for which BMA1.0 found at least one solution but the correct analysis according to the ATB was not among them; and (v,vi,vii) the number of proper nouns (NNP), nouns (NN) and adjectives (JJ) from "no correct". A problem that is unique to the ATB is that some words in the corpus were not manually annotated and were given the NO_FUNC tag. Those words are counted as Arabic words, but are ignored in the rest of the statistics of Table 3.

The noticeable difference in OOV words between ATB1 and ATB2/ATB3 is expected, because the lexicon of BMA1.0 was developed using information extracted from ATB1. ATB2 and ATB3, which were developed after BMA1.0 was released (using a more advanced version of Buckwalter's analyzer), show a different picture. In those two parts the OOV problem is not too hard: a heuristic that would assign NNP to each OOV word would be sufficient in most of the cases. However, the "No Correct" problem is more difficult: NNPs account for 5% in ATB2 and 18% in ATB3 of these words, which are mostly dominated by missing adjectives and missing nouns (54% jointly in ATB2 and 37% jointly in ATB3).

Taken together, the OOV problem and the "No Correct" problem mean that more than 5% of the words in ATB2 and ATB3 cannot be tagged correctly using BMA1.0 unless further data are added to those provided by the morphological analyzer. A similar coverage result was reached for Hebrew by Bar Haim et al., using a morphological analyzer for Hebrew (Segal, 2001). Bar Haim et al. report that for about 4% of the Hebrew words in their corpus, the correct analysis was missing. From these data we conclude that on top of systems like the ones proposed by Bar Haim et al. and Habash and Rambow, we need to enhance the morphological analyzer using additional analyses.

[1] This definition of Arabic words is taken from Buckwalter's analyzer.

ATB part	Arabic words	OOV	NNP of OOV	No Correct	NNP of No Correct	NN of No Correct	JJ of No Correct
1	123798	126 (0.11%)	21 (16.67%)	3369 (2.82%)	0	517 (15.35%)	980 (29.09%)
2	125729	958 (0.77%)	497 (51.88%)	5663 (4.53%)	282 (4.98%)	1254 (22.14%)	1818 (32.1%)
3	293026	6405 (2.2%)	5241 (81.83%)	15484 (5.32%)	2864 (18.5%)	2238 (14.45%)	3494 (22.57%)

Table 3 - Coverage of Buckwalter's Analyzer

5 Smoothing Using a Data-driven Character-based Model

So far we have shown that POS tagging models that use a morphological analyzer achieve high accuracy but suffer from coverage problems that can not be solved by a simple heuristic. On the other hand, models that use character-based information are likely to make relatively good predictions for words that are out of the vocabulary of the morphological analyzer. We hypothesize that this may be especially true for Semitic languages, due to their rich and systematic pattern (template) paradigms. Such patterns add constant characters to root characters, and features of substrings of words may therefore help in predicting POS tags from those patterns.

Our baseline models for the experiments are MorphTagger with a NNP heuristic (MorphTagger+NNP) and ArabicSVM (Diab et al.'s system). As we have already reported in section 3, MorphTagger+NNP achieved 96.12% tagging accuracy and ArabicSVM achieved 95.87% over the same testing data used by Diab et al. One simple hybrid model would be adding the analyses produced by the SVM to the morphological analyzer analyses and disambiguate these analyses using MorphTagger's HMM. This system has improved accuracy – it achieved accuracy of 96.18%, higher than both of the base models.

The problem with such model is over-generation of the SVM: when checked over ATB1 and ATB2, 40% of the new analyses introduced by the SVM are correct analyses, and 60% are wrong. To avoid this problem, we suggest conditioning the addition of SVM analyses on the sentence's tagging probability calculated by the HMM model. This is justified due to the fact that there is correlation between the probability of the tagging of a sentence given by a language model and the accuracy of the tagging. The relation is shown in Figure 1.

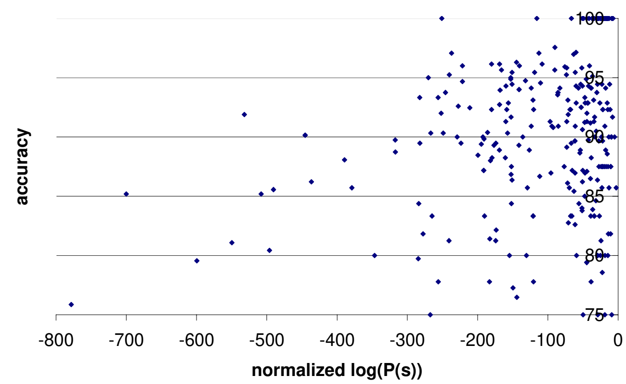

Figure 1 Probability VS Accuracy

Figure 1 shows the relation between the accuracy of the tagging and the normalized logarithmic probability of the tagging. We normalize the probability of the tagging by the sentence length as longer sentences usually have lower probabilities.

Following the previous conclusions, we propose a hybrid model which adds the analyses of the SVM only in cases where the tagging probability by the basic MorphTagger system is lower than an empirically calculated threshold. If the HMM is confident about the tagging it produces, the probability of the tagging will be high enough to pass the threshold, and then the tagging will be outputted without adding the SVM analyses which might add noise to the morphological analyzer output. A general algorithm is shown in Figure 2.

Given a sentence *s*, perform the following steps:

1. Produce analyses for each word in *s* using the morphological analyzer combined with the corpus analyses.
2. Calculate lexical and contextual probabilities using available annotated corpora (using Maximum Likelihood Estimation).
3. Run Viterbi's Algorithm for HMM disambiguation, and calculate a rank of the tagging which is composed from the probability given by the model and the length of the sentence.
4. If [rank>threshold] output tagging.

4'. [Otherwise] run the character based model over the sentence and add the new analyses generated.

5'. Combine the analyses generated by the morphological analyzer and the character-based model, update the lexical probabilities and rerun the model.

Figure 2 - Enhanced Tagging Algorithm

Note that in the algorithm, a new (word, tag) pair introduced by the morphological analyzer or by the character model does not appear in the tagged corpus, therefore a small count $\lambda=1$ is given in such cases. This method can be improved further, especially for the analyses produced by the data-driven character-based method.

The accuracy we obtained using this system was 96.28% which shows slight improvement over the previous simple hybrid system. Examining the errors in the simple hybrid method and the conditioned method, we see that the improvement is not smooth: the conditioned model includes errors which did not exist in the simple model. These errors occur when correct analyses of the character-based model were discarded. In general, however, the conditioned method chooses more correct analyses. It should be noted that adding the character-based model analyses boosted the coverage from 97% to 98%, but the accuracy did not improve to the same level. The main cause for this is the weak relation between the probability of a sentence and the accuracy. As it is difficult to model this relation, we believe that more time should be invested to improve the HMM probabilities especially for the character model analyses, which can boost the chances of choosing good analyses.

6 Conclusions and Future Work

This paper demonstrates that it is possible to successfully port a POS tagger originally built for Hebrew to Arabic using a morphological analyzer and a tagged corpus. The POS tagger (called MorphTagger) achieves state-of-the-art results both on Hebrew and Arabic. Despite this positive result we find that further improvement of accuracy is hindered by the coverage of the morphological analyzer. Contrary to earlier work on POS tagging, the problem turns out not so much in unknown (OOV) lexemes as much as in known lexemes for which the correct tag is missing. We showed empirical evidence that this problem arises for the available treebanks and morphological analyzers for both Arabic and Hebrew. We propose an approach that smoothes a given lexical model (obtained from a morphological analyzer and an annotated corpus) by adding synthetically constructed analyses, obtained from a POS tagger that combines character-level information. Unlike earlier work, we apply this smoothing only when the probabilistic model assigns probabilities lower than a threshold to all possible POS taggings of the input sentence. This way we obtain moderate improvement in Arabic POS tagging.

The problem of missing lexeme-POS pairs in POS taggers for Semitic languages is more severe than in languages like English. We conjecture that this is because of the more complex morphology of Semitic languages.

In future work it might be worthwhile to consider morphological processes that are more complex than the standard affixation (suffixing/prefixing) processes in order to generalize better over cases in the training data. Such a generalization may provide better coverage of lexeme-POS pairs and would increase the upper bound on accuracy.

References

Roy Bar Haim, Khalil Sima'an and Yoad Winter. 2005. *Choosing an Optimal Architecture for Segmentation and POS-Tagging of Modern Hebrew*. ACL Workshop on Computational Approaches to Semitic Languages. A revised and extended version to appear in Journal of *Natural Language Engineering*.

Eric Brill. 1995. *Transformation-based error-driven learning and natural language processing: a case study in part of speech tagging*. In Computational Linguistics 21, pages 543-565.

Tim Buckwalter. 2002. *Arabic Morphological Analyzer Version 1.0*. Linguistic Data Consortium, University of Pennsylvania.

Kenneth W. Church. 1988. *A stochastic parts program and noun phrase parser for unrestricted text*. Proceedings of the second conference on Applied natural language processing, Pages 136-143.

Mona Diab, Kadri Hacioglu, and Daniel Jurafsky. 2004. *Automatic Tagging of Arabic Text: From Raw Text to Base Phras e Chunks*. In HLT-NAACL: Short Papers, pages 149-152.

Nizar Habash and Owen Rambow. 2005. *Arabic Tokenization, Part-of-Speech Tagging and Morphological Disambiguation in One Fell Swoop*. In Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics, pages 573-580, Ann Arbor.

Young-Suk Lee, Kishore Papineni, Salim Roukos, Ossama Emam, and Hany Hassan. 2003. *Language model based Arabic word segmentation*. In ACL, pages 399-406.

Mohamed Maamouri and Ann Bies. 2004. *Developing an Arabic treebank: Methods, guidelines, procedures, and tools*. In Proceedings of the Workshop on Computational Approaches to Arabic Script-based Languages (COLING), Geneva.

Christopher D. Manning and Hinrich Schütze. 1999. *Foundations of Statistical Natural Language Processing*. The MIT press, Cambridge, Massachusetts.

Tetsuji Nakagawa. 2004. *Chinese and Japanese word segmentation using word-level and character-level information*. In Proceedings of the 20th International Conference on Computational Linguistics, pages 466-472, Geneva.

Erel Segal. 2001. *Hebrew morphological analyzer for Hebrew undotted texts*. Master's thesis, Computer Science Department, Technion, Haifa, Israel.

Ralph Weischedel, Marie Meteer, Richard Schwartz, Lance Ramshaw and Jeff Palmucci. 1993. *Coping with Ambiguity and Unknown Words through Probabilistic Models*. Computational Linguistics (Special issue on using large corpora: II) volume 19, pages 361-382.