Arabic Part-Of-Speech Tagging Using Transformation-Based Learning

Shabib AlGahtani **William Black** **John McNaught**

The University of Manchester
School Computer Science, Manchester M1 9PL
United Kingdom
[algahtani, wblack, mcnaught]@cs.man.ac.uk

Abstract

Corpus-based methods have been widely used to tackle NLP tasks after the advent of annotated corpora with a notable success. Inevitably, shifting from classical rule-based to corpus-based method has a major drawback. That is, most of corpus-based ones produce statistical models that are hard to interpret and modify along with their higher complexity in terms of required processing power and memory allocation. Fortunately, that drawback is eliminated with Transformation-based learning technique which is one corpus-based method that embraces the power of both worlds; overcoming obscurity and complexity without relinquishing state-of-the-art accuracy. This paper examines the application of TBL to the task of tagging Modern Standard Arabic text. For unknown words guessing, an n-gram technique has been adopted to select best tag from a list of candidates outputted from a morphological analyzer exploiting previous context. The developed tagger achieved an accuracy of 98.6% when evaluated on the train set and 96.9% on the test set. Furthermore, the same unknown words module has been slightly modified and successfully applied to the task of word-tokenization with an accuracy of 99.6%.

Introduction

The goal of **N**atural **L**anguage **P**rocessing **(NLP)** is to resolve human language ambiguity in different analysis levels. Recently, a new trend has been adopted to tackle NLP tasks with the development of annotated corpora; giving preference for corpus-based methods over the classical rule-based methods. In some experiments, corpus-based methods outperform classical approach. The main difference between them occurs in providing the system with the required linguistic knowledge. Rule-based systems rely on human knowledge while corpus-based uses corpora to build a data model used afterwards to resolve the ambiguity in certain task without the need for strong linguistic skills making it an attractive approach to researchers who lack strong skills (Brill and Moony 1992). However, the main bottleneck of that approach is the availability of large corpora which **L**anguage **D**ata **C**onsortium **(LDC)** has solved by launching a number of megaprojects to develop annotated corpora targeting a wide range of languages including Arabic.

In the NLP framework, **P**art-**O**f-**S**peech **(POS)** tagging is the process of assigning syntactic role to each word in context and hence considered to be a crucial step that highly affects other subsequent NLP tasks. With respect to **M**odern **S**tandard **A**rabic **(MSA)**, official written language, the influence of POS tagging is even larger due to its characteristics that impose a number of processing challenges. One example exists in Arabic Named Entity Recognition where POS tagging is vital with the absence of capitalization in proper nouns. In Semitic languages including Arabic, the phenomena of clitics attachment is another challenge added to POS tagging complexity. The process of finding the boundaries between stem and cltics attached to it is called word tokenization or segmentation. The ambiguity of a word has two parts; finding the right segmentation and finding the right tag of each segment. This study will be dealing with both processes in MSA.

Previous Work

POS tagging started early in the field by TAGGIT, a rule-based system that relies on the character pattern and previous tag to disambiguate the current word. Later, CLAWS was developed by the University of Lancaster in early 1980s as probabilistic version of TAGGIT (Garside, 1987). Since then, a wide range of corpus-based methods have been applied to NLP tasks including POS tagging leveraging large corpora. Ratnaparkhi used Maximum Entropy Models to build POS classifier which successfully combined a wider context of tagging history and morphological features yielding to an accuracy of 96.6% on the Penn Tree Bank (1996). Standard taggers were developed using Hidden Markov Models which was imported from speech recognition and applied to tagging such as Kupeic's work (1992) which achieved 96.3% accuracy (Abny, 1996). Error driven approach, called **T**ransformation-**B**ased **L**earning **(TBL),** was introduced by Eric Brill in 1994 and achieved an accuracy of 97.2% in same corpus outperforming HMM tagger.

Recent advances in POS tagging were done by concept of bidirectional learning which resulted in state of the art accuracy, above 97%, in English as published in (Tsuruoka et al, 2005) and (Libin, S. et.al, 2007). Bidirectional learning uses previous and successive context explicitly to find the tag of the current word.

Most corpus-based methods produce models that are not easily analyzed and improved compared to a set of clear and concise transformation rules produced by TBL. With them, TBL shares the idea of automatic extraction of language regularities from corpora in the training phase but the tagging phase is fully rule-based techniques (Brill & Mooney 1997).

In Arabic POS tagging, Shereen Khoja used a hybrid technique of statistical and ruel-based with a morph-syntactic tagset (2001). Later, Support Vector Machines were used to separately implement a character based word-

tokenizer and a POS tagger with a collapsed tagset achieving 99.7% on F measure and 95.5% on word-tokenization and tagging respectively (Diab et al, 2004). With the help of the Arabic rich morphological features, Habash and Rambow were able to tackle both problems in this task in one step achieving an accuracy of 97.5% (2005). An HMM Hebrew tagger was ported to Arabic yielding to an accuracy of 96.1% (Mansour, 2007).

In this study, TBL was favoured for its explicit consideration of both side of the word as bidirectional learning to some extent, assuming to be more competent with free-order language than techniques previously used in the literature added to its ability to combine advantages of both corpus-based and rule-based systems with less affected by data sparseness.

Arabic Segmentation and POS tagging

MSA processing is highly affected by the missing diacritical adding more complexity to both syntactic and semantic analysis. That is due to the fact that diacritics reduce the number of possible classes of the word. This feature is not present in English but could be imagined by dropping vowels from words; e.g. dropping vowels from *is* would result in three possible outcomes; *us*, *is* and *as*. Still, vowels could be restored by the context to decide the right word.

Arabic word is composed of stem plus affixation to indicate tense, gender and number. In addition to affixes, clitics are attached to beginning, end or both. Clitics are segments that represent an independent syntactic role; mainly conjunctions, preposition and pronouns. Prepositions and conjunctions are attached to beginning of the word while pronouns at the end (Diab et al, 2004). Clitics are composed of general Arabic characters that could be part of the stem. To view this problem of clitics attachment in English, consider passing English text through a noisy channel with a possibility of dropping the space delimiter between words resulting in word concatenation. If the following sentence was received:

Those cars useless fuel.

The only possibility of concatenation occurs in the word *useless* as it might have two possibilities; one is being correct and the other is the result of concentrating the word *use* and *less*. If we use the POS tagging information of the previous word *cars*, it would be more sensible to choose *use less* since verbs are more likely to follow nouns than adjectives.

Bar-Haim in (2005) referred to each unit of the word that represent an independent tag as segment. In Arabic, the word (ولدك), transliterated as (*wldk*), has three valid segmentations; *wld+k*, *w+ld+k* and *w+l+dk*. Each of them corresponds to a number of POS tagging annotation. The segmentation *wld+k*, might have a number of POS tagging annotation; one possibility is CC+NN+PRP$. Combining both the segmentation with the tagging information constitutes a full analysis; w/CC+ld/VBD+k/PRP. These two tasks are bound in a way that the correct tagging analysis always encodes the right segmentation.

Transformation-Based Learning

Transformation-based learning is an error-driven approach to induce the retagging rules from a training corpus. The learning algorithm starts by building a lexicon containing each word with all possible tags and the frequency of each tag. Then, it maintains two versions of the corpus; gold standard corpus that contains word/tag and training corpus that contains only words. Next, it assigns the most frequent tag to each word in the training corpus which referred as initial state tagging. After that, it compares the initially annotated corpus it with gold-standard corpus and the largest error class. Focusing on that error class, it applies a set of predefined templates to correct it and chooses the rule with highest gain; gain means number of corrections in the training corpus. That rule is stored in a list after being applied to the training corpus. Then, it calculates the largest error class from the updated training corpus again and so on until no correction could be made. In the tagging phase, the raw text will use the lexicon to tag each word with its most frequent tag then the list of learned rules are applied.

Mainly, each rule template has a tag transformation and a triggering event. The tag transformation will be fired only if the triggering condition is met. The predefined templates are divided into two categories; non-lexicalized and lexicalized rules. Non-lexicalized rule depends only on surrounding tagging information to change the tag of the current tag while the lexicalized one could make reference to words. The 24 rule templates used with English are listed below in Figure 1. The first rule is a non-lexicalized one that is interpreted as: change tag A to tag B if tag C occurs at position -1; previous tag. The 14th rule is a lexicalized one interpreted as change tag A to tag B if word C occurs at position -1.

```
A -> B tag C @ [ -1 ] .
A -> B tag C @ [ 1 ] .
A -> B tag C @ [ -2 ] .
A -> B tag C @ [ 2 ] .
A -> B tag C @ [ -1 -2 ] .
A -> B tag C @ [ 1 2 ] .
A -> B tag C @ [ -1 -2 -3 ] .
A -> B tag C @ [ 1 2 3 ] .
A -> B tag C @ [ -1 ] & tag D @ [ 1 ] .
A -> B tag C @ [ -1 ] & tag D @ [ -2 ] .
A -> B tag C @ [ 1 ] & tag D @ [ 2 ] .
A -> B word C @ [ 0 ] & tag D @ [ -2 ] .
A -> B word C @ [ 0 ] & word D @ [ -2 ] .
A -> B word C @ [ -1 ] .
A -> B word C @ [ 1 ] .
A -> B word C @ [ -2 ] .
A -> B word C @ [ 2 ] .
A -> B word C @ [ -1 -2 ] .
A -> B word C @ [ 1 2 ] .
A -> B word C @ [ 0 ] & word D @ [ -1 ] .
A -> B word C @ [ 0 ] & word D @ [ 1 ] .
A -> B word C @ [ 0 ] & tag D @ [ -1 ] .
A -> B word C @ [ 0 ] & tag D @ [ 1 ] .
A -> B word C @ [ 0 ] .
A -> B word C @ [ 0 ] & tag D @ [ 2 ] .
A -> B word C @ [ 0 ] & word D @ [ 2 ] .
```

Figure 1: Templates

word	Transliterated	Translated	Analysis
قرأ	qr>	Read	qr>/VBD
ولدك	wldk	your boy and diverted you and to demolish	wld/NN+k/PRP$ w/CC+ld/VBD+k/PRP w/CC+l/IN+dk/NN
الكتاب	AlktAb	The book	AlktAb /NN

Table 1: Arabic sentence example

Methodology

With the clitics attachment feature in Arabic, the POS tag of a word could be compound leading to tagset extension which, in turn, adding more complexity to this task. Also, that adds the problem of data sparseness. So, the decision was to consider segment-level tagger instead of word level. First stage of this approach is maintaining a segment-level annotated corpus which will be used to produce retagging rules. That involves segmenting the available word-level corpus as a pre-processing step. The rules induction algorithm described above is run on the pre-processed corpus. The induced rules are applicable to segment-level text which is not the case with natural text. In the tagging phase, the algorithm exploits the close relation between tagging and segmentation process in Arabic performing tagging and segmenting words in the same time. This process relies on a morphological analyzer to produce all possible analysis and uses bigrams to choose the right analysis. As clitics are limited, words that seem to have clitics, in other words ambiguous, are processed while words, that don't, will be tagged with their most frequent tag directly taken from lexicon. By focusing on specific words, the accuracy of the initial state is leveraged and hence add more confident to the ngram module. In the ngram module, the task is to choose only the right segmentation which does not require considering long previous context and add the burden of data sparseness; experiments showed that bigrams performed well. As an example, consider the following example shown in Table 1:

قرأ ولدك الكتاب.

Here, the only word that is ambiguous in terms of its segmentation is *wldk* since it starts and ends with clitic-like segments; w could be a conjunction and k could be a pronoun. In initial state of tagging that sentence, words will be tagged with their most frequent tag assuming that they exists in the lexicon while *wldk* will be tagged as unknown. A morphological analyzer is used to only process *wldk* and produce the three analyses listed in the Table 1. Now, bigram of previous tag and the segmentation of *wldk* will be used to select the correct segmentation based on the previous tag. The analysis associated with the highest of the following probabilities will be selected:

P(wld+k|VBD), P(w+ld+k|VBD) and P(w+l+dk|VBD)

If all these bigrams never occurs in the corpus resulting in zero probability, select the analysis with highest probability of the tagging analysis of each one given the previous tag:

P(NN PRP$|VBD),
P(CC VBD PRP|VBD),
P(CC IN NN|VBD)

In the prevoius example, it would more sensible to select the first segmentation of the word *wld+k* as it is more likely for NN to follow VBD.

Implementation & Experiments

The corpus used in this experiment is the **A**rabic **T**ree **B**ank **(ATB)** which was produced in four parts by LDC and contains news text from four official newspapers of different regions in the Middle East. The total number of words is 770k. The annotations include morphological analysis and syntactic trees of sentences. To our task, only the morphological analysis is need.

The morphological analysis annotation was firstly mapped to the Arabic collapsed tagset distributed with ATB. Then, each word was that have compound tag were split so that annotation is segment-level; the resulted corpus has segment/tag unit. The total number of segments in the corpus after the segmentation is 920k. With respect to tagging, 35% of the data was ambiguous; segments with more than one tag.

The morphological analyzer used produce word analysis was **B**uckwalter **M**orphological **A**nalyzer **(BMA)**. The same mapping scheme was used to map the output of BMA to be consistent with collapsed tagset used in mapping the ATB instead of morphological tagset. The training phase is performed on segmented text. As an output, a lexicon is built and a set of retagging rules are induced.

The tagging phase has two different algorithms used depending on the format of the input text. Figure 2 shows the algorithm used to segment and tag an un-segmented text, the general case. In the initial state annotator, not only words that don't exist in the lexicon but also words that seem to have clitics attachment are passed to BMA. The main concern in this stage is finding only the correct segmentation and not the correct full analysis. If the tagging of the selected analysis is not correct, it will be corrected afterwards by the retagging rules induced in the training phase.

If the tagger is to be run on a segmented text, only segments that don't exist in the lexicon will be tagged as unknown; sometimes referred as out-of-vocabulary. The ngram module's task is to select the highest tag probability of current segment produced by BMA. The tag of unknown segment is conditioned on the previous tag. After the initial state with unknown words guessing, retagging rules are applied straightforward to the initially annotated text.

1. Assign most frequent tag to all words and UNK to unknown words in the input list of words. Any word that starts or ends with clitics-like sequence of characters will also be tagged as UNK.
2. Pass the list of words with their initial tagging to BMA.
3. BMA will only process words tagged as UNK and find their solutions.
4. Each solution of UNK words outputted by BMA is mapped to the collapsed tagset; each solution has a number possible segmentation and tagging.
5. Each word in the input list will have one of the following:

- Single tag if it exist in the lexicon
- One or more analysis if found by BMA
- Tagged as NNP if not found by BMA.

6. Use the frequency of bigram constructed from the previous tag and the segmentation of the word in focus to select the right solution produced by BMA. As a back-off scheme, use the bigram constructed from previous tag and the tag of current word's segments.
7. Split the word according to the selection done in previous step so the input list contains segments only and then apply retagging rules learned from pre-tokenized corpus.

Figure 2: Joint tagging and segmenting algorithm

In order to evaluate the performance of TBL tagger, two experiments have been conducted on two different corpora. The first experiment was only on ATB 1.0 for the sake of comparing tagger performance with the previous work as most taggers were evaluated against that part due to its availability when developed. The second experiment was on the four parts of the ATB to see how the tagger would perform on diverse genres assuming discrepancy. In both experiments, the corpus was split into 90% train set and 10% test set. The rules are induced first from the pre-segmented corpus. Then the first evaluation is conducted on the training pre-segmented corpus which doesn't sounds rational but it is an attempt to examine the quality of the induced rules. Then, the next run will be on the pre-segmented test set. Finally, the tokenization module is evaluated on the non-segmented (word-level) train set.

Results and discussion

Using rule templates used for English, a set of non-lexicalized and lexicalized and rules were produced from the pre-segmented corpus in both experiments. The total number of rules was 255 in exp1 while that number increased to 1500 in exp2. The first portion of the rules was changing PRP to PRP$ which capture the Arabic feature of pronouns serving as both; personal pronoun and possessive pronoun depending on whether the previous tag is noun or verb. A sample of the induced rules is listed in Figure 3. The first five rules are non-lexicalized and the rest are lexicalized.

```
PRP$ -> PRP tag IN @ [ -1 ] .
PRP  -> PRP$ tag NN @ [ -1 ] .
PRP$ -> PRP tag VBP @ [ -1 ] .
PRP  -> PRP$ tag NNS @ [ -1 ] .
PRP$ -> PRP tag VBD @ [ -1 ] & tag WP @ [ -2 ] .
JJ   -> NN word AlEAm @ [ 0 ] & tag CD @ [ 1 ] .
IN   -> NN word bEd @ [ 0 ] & tag IN @ [ -1 ] .
VBD -> NN word b @ [ -1 ] .
```

Figure 3: Sample rules

Table2 shows results obtained in each stage of the two experiments. The quality of the induced rules was superior on the train set achieving an accuracy of 98.5% and 97.9%. When evaluated on test set, the tagger achieved an accuracy of 96.9% in exp1 and 96.15 in exp2. The main cause for the accuracy drop was the accuracy of the initial state annotator caused by the tagging inconstancy in different parts of ATB, e.g. months were tagged as NNP in part 1.0 while they were tagged as NN in the other parts of the ATB. Empty slots in the table indicate that test was not completed.

The ngram module used for unknown segments guessing achieved an accuracy of 85% exp1 and 80% in exp2 due to the fact that BMA was developed from ATB 1.0. However, accuracy was not highly affected because larger training data enriches the lexicon and hence reduce the possibility of unknown words except proper nouns that would be tagged as NNP in not found by BMA.

Table 3, shows a performance comparison of the tagger on segmented text with three developed taggers described in (Diab, 2004; Habash, 2005; Mansour 2007). The same test data used to evaluate those taggers was not available but a similar selection scheme was used instead. Randomly, 10% of ATB 1.0 was selected for testing the TBL tagger comparing its performance with their reported accuracy. It is obvious that TBL tagger outperforms all in addition to its concise and easily interpreted rules.

In exp1, the segmentation module achieved an accuracy of 99.6% exp1 and 99.2 % in exp2. The coverage of BMA has a larger effect in the segmentation module as the usage of lexicon is reduced since all word that seems to have clitics attached are passed to BMA. That superior accuracy was achieved due to the low number of words that has multiple segmentations in the corpus. Further experiments are to be conducted in order to precisely evaluate the segmentation module as it was tested on a small data. Furthermore, the F measure would give a more meaningful measure of the performance of that module and give the ability to compare it with other techniques; to be included in further study.

	Exp1	Exp2
ATB part	1.0	1,2,3,4
Corpus size	165k	920k
Train set size	150k	828k
Lexicon size	15k	43k
Number of rules	255	1500
Initial state accuracy on train set	95.65%	--
Accuracy after rule application	98.6	97.9
Test set size	15k	92k
Unknown words in test set	5.3%	2.8%
Acc. Initial state when UNK as NN	91.1%	--
Acc. on test set UNK as NN + rules	93.67	--
Acc. Initial state. when UNK as NNP	92.37	--
Acc. on test set UNK as NN + rules	94.91	--
Acc. Initial state. when using nGram	94.52	93.4%
Words not found by BMA	18%	25%
Acc. Unknown word guessing	85%	80%
Acc. with all modules	**96.90%**	**96.14%**
Accuracy improvement	2.38	2.7%
Transformations Acc. (correct/all)	94%	92%
Largest error class and %	NN as JJ	NN as JJ
Acc. Segmentation module	99.6%	99.2%

Table 2: Experiments results

System	Accuracy %
Diab	95.49
Habash	97.5
Mansour	96.12
TBL	96.9

Table 3: Systems comparison on ATB 1.0

Future Work

This study showed that TBL outperform any other technique used for Arabic tagging on top of its simplicity and lower complexity and with same templates used with other languages. With respect to segmentation, it is quit telling that short previous tagging context using most frequent tag information will still perform well in that task.

Encouraged by the performance of the tagger, the plan is to train the tagger on un-segmented text. That way, the compound tags will have both tagging and segmentation information eliminating the need for a morphological analyzer. The new experiment involves modifying the learning algorithm to consider validating that the new compound tag is one of the possible cases of the word based on the presence of clitics-like segments exploiting the phenomena that tagging analysis always maps to only one segmentation. Also, the conditions of the contextual templates have to be applied based on single tag context rather than compound. Furthermore, Brill has introduced other TBL templates to deal with unknown words using only morphological features which assumed to be appropriate for Arabic words if the templates were modified. The proposed templates will use clitics attached to the word as a feature along with context of tags to guess unknown words rather than morphological features.

Acknowledgment

The author (Shabib) would like to acknowledge Saudi government, MEDAR and LDC for their outmost support.

References

Abney, S. (1997). Part-of-Speech Tagging and Partial Parsing. Corpus Based Methods in Language and Speech Processing (pp. 118-136) Text, Speech and Language Technology Series, Kluwer Academic Publishers

Bar-Haim, R. , Khalil Sima'an and Yoad Winter (2005). Choosing an Optimal Architecture for Segmentation and POS-Tagging of Modern Hebrew. ACL 2005 Workshop on Computational Approaches to Semitic Languages.

Brill, E. (1995). Transformation-Based Error-Driven Learning and Natural Language Processing: A Case Study in Part-of-Speech Tagging. Computational Linguistics 21(4):543–565.

Brill, E. and R. J. Mooney (1997). An Overview of Empirical Natural Language Processing. 18: 13-24.

Diab, M. , Kadri Hacioglu and Daniel Jurafsky (2004). Automatic Tagging of Arabic Text: From raw text to Base Phrase Chunks. Proceedings of HLT-NAACL 2004.

Garside, R. (1987). The CLAWS Word-tagging System. In: R. Garside, G. Leech and G. Sampson (eds), The Computational Analysis of English: A Corpus-based Approach. London: Longman

Habash, N. and Owen Rambow (2005). Arabic Tokenization, Morphological Analysis, and Part-of-Speech Tagging in One Fell Swoop. In Proceedings of the Conference of American Association for Computational Linguistics (ACL05).

Kupiec, J. (1992). Robust part-of-speech tagging using a hidden Markov model. Computer Speech and Language, 6(3):225-242.

Khoja, S. (2001). APT: Arabic Part-of-speech Tagger. Proc. of the Student Workshop at NAACL 2001.

Libin, S. , Giorgio Satta, and Aravind K. Joshi. (2007). Guided learning for bidirectional sequence classification. Proc. Of ACL-2007.

Mansour, S. , Khalil Sima'an and Yoad Winter (2007). Smoothing a Lexicon-based POS tagger for Arabic and Hebrew. In proceedings of ACL 2007 Workshop on Computational Approaches to Semitic Languages: Common Issues and Resources. Prague, Czech Republic, 2007.

Ratnaparkhi, A. (1996). A maximum entropy model for part-of-speech tagging, Association for Computational Linguistics: 133–142.

Tsuruoka, Y. and Jun'ichi Tsujii (2005). Bidirectional Inference with the Easiest-First Strategy for Tagging Sequence Data. In the Proceedings of HLT/EMNLP 2005. Vancouver , BC , Canada , pp. 467-474, October 2005.