Farasa: A New Fast and Accurate Arabic Word Segmenter

Kareem Darwish and Hamdy Mubarak
Qatar Computing Research Institute
Hamad Bin Khalifa University (HBKU), Doha, Qatar
{kdarwish,hmubarak}@qf.org.qa

Abstract

In this paper, we present Farasa (meaning insight in Arabic), which is a fast and accurate Arabic segmenter. Segmentation involves breaking Arabic words into their constituent clitics. Our approach is based on SVM^{rank} using linear kernels. The features that we utilized account for: likelihood of stems, prefixes, suffixes, and their combination; presence in lexicons containing valid stems and named entities; and underlying stem templates. Farasa outperforms or equalizes state-of-the-art Arabic segmenters, namely QATARA and MADAMIRA. Meanwhile, Farasa is nearly one order of magnitude faster than QATARA and two orders of magnitude faster than MADAMIRA. The segmenter should be able to process one billion words in less than 5 hours. Farasa is written entirely in native Java, with no external dependencies, and is open-source.

Keywords: Arabic morphology, word segmentation, stemming

1. Introduction

Segmenting Arabic words is one of the most important processing steps for Arabic and is used as a building block for many Arabic natural language processing (NLP) technologies such as search, part-of-speech tagging, parsing, and machine translation. Word segmentation involves breaking words into their constituent clitics. For example, the word "wktAbnA"[1] (وكتابنا meaning: "and our book") is composed of three clitics "w+ktAb+nA", namely the conjunction article "w" as prefix, the stem "ktAb", and possessive pronoun "nA" as suffix. Due to the importance of the problem, many segmenters have appeared in the past 20 years. They ranged from rule-based morphological analyzers (Beesley et al., 1989; Beesley, 1996; Buckwalter, 2002; Khoja, 2001), to light stemmers (Aljlayl and Frieder, 2002; Darwish and Oard, 2007), and statistical word segmenters (Darwish, 2002; Habash et al., 2009; Diab, 2009; Darwish et al., 2014). Statistical word segmenters are considered state-of-the-art with reported segmentation accuracy above 98%.

In this paper we introduce a new segmenter, called Farasa (meaning: insight in Arabic), that is at par or better than the state-of-the-art and is very fast. It is an SVM^{rank}-based segmenter that uses a variety of features and lexicons to rank different possible segmentations of a word. The features that we utilized account for: likelihood of stems, prefixes, suffixes, and their combination; presence in lexicons containing valid stems and named entities; and underlying stem templates. For Farasa, we make the fundamental simplifying assumption that word-context is not necessary to find the correct segmentation of a word. Though a word may have multiple valid segmentations, our results show that this assumption minimally impacts our results. We also report evaluation results on a new test set that we developed to measure segmentation accuracy. We also compare to other state-of-the-art segmenters, namely MADAMIRA (Pasha et al., 2014) and QCRI Advanced Tools For ARAbic (QATARA) (Darwish et al., 2014). We also wanted to compare to the Stanford Arabic segmenter (Monroe et al., 2014), but its segmentation scheme was different from the one we used in Farasa. Many of the current results reported in the literature are done on subsets of the Penn Arabic Treebank (ATB) (Maamouri et al., 2005). Since the aforementioned tools are trained on the ATB, testing on a subset of the ATB is problematic due to its limited lexical diversity and the similarity between the training and test sets. This generally leads to results that are often artificially high. Our new dataset is composed of recent WikiNews articles from the years 2013 and 2014.

The contributions of this paper are:

- the development a new Arabic word segmenter that is more accurate and much faster than the current state-of-the-art segmenters.
- the introduction of a new test set to evaluate word segmentation. We plan to make this tool and data freely available.

A demo of the segmenter is available online at:
`http://qatsdemo.cloudapp.net/farasa/`

2. Related Work

Due to the morphological complexity of the Arabic language, morphological processing such as word segmentation helps recover the units of meaning or their proxies, such as stems (or perhaps roots). Most early Arabic morphological analyzers generally used finite state transducers (Beesley et al., 1989; Beesley, 1996; Kiraz, 1998). Their use is problematic for two reasons. First, they were designed to produce as many analyses as possible without indicating which analysis is most likely. This property of the analyzers complicate subsequent NLP as many applications require the most likely solution. Second, the use of finite state transducers inherently limits coverage, which is the number of words that the analyzer can analyze, to the cases programmed into the transducers. Other similar approaches attempt to find all possible prefix and suffix combinations in a word and then try to match the remaining

[1]Buckwalter encoding is used exclusively in the paper.

stem to a list of possible stems (Khoja and Garside, 2001; Maamouri et al., 2010). This approach has the same shortcomings as the finite state transducer approach. Another approach to morphology is so-called light stemming. In this approach, leading and trailing letters in a word are removed if they match entries in lists of common prefixes and suffixes respectively. The advantage of this approach is that it requires no morphological processing and is hence very efficient. However, incorrect prefixes and suffixes are routinely removed. This approach was used to develop Arabic stemmers by Aljlayl et al. (Aljlayl and Frieder, 2002), Darwish and Oard (Darwish and Oard, 2007), and Larkey et al. (Larkey et al., 2002).

More recent analyzers can statistically perform deep word stemming. For example, Darwish attempted to solve this problem by developing a statistical morphological analyzer for Arabic called Sebawai that attempts to rank possible analyses to pick the most likely one (Darwish, 2002). Lee et al. (Lee et al., 2003) developed IBM-LM, which adopted a trigram language model (LM) trained on a portion of the manually segmented LDC Arabic Treebank (Maamouri et al., 2005), in an attempt to improve the coverage and linguistic correctness over existing statistical analyzers such as Sebawai (Darwish, 2002). IBM-LM's analyzer combined a trigram LM (to analyze a word within its context in the sentence) with a prefix-suffix filter (to eliminate illegal prefix-suffix combinations, hence improving correctness) and unsupervised stem acquisition (to improve coverage). Lee et al. report a 2.9% error rate in analysis compared to 7.3% error reported by Darwish for Sebawai (Lee et al., 2003; Darwish, 2002). Diab (Diab, 2009) created a stemmer called AMIRA that used an SVM classifier to ascertain the optimal segmentation for a word in context. The classifier was trained on the Arabic Penn Treebank data. Essentially, Diab treated the problem as a sequence labeling problem and reported a stemming error rate of about 1%. MADA (Habash et al., 2009) and MADAMIRA (Pasha et al., 2014), the successor to both MADA and AMIRA, are morphological taggers for MSA that are used widely for processing Arabic in the context of machine translation (Sajjad et al., 2013). Our work in this paper uses statistical methods to perform word segmentation.

Though there has been some work on morphological analysis of Arabic dialects (Habash et al., 2013), particularly Egyptian and Levantine. Most of the dialectal morphological phenomena primarily affect verbs with little effect on nouns. Work on dialects is beyond the scope of this paper.

3. Segmentation

3.1. Features:

In this section we introduce the features and lexicons that we used for segmentation. For any given word (out of context), all possible character-level segmentations are found and ones leading to a sequence of $prefix_1$+...+$prefix_n$+stem+$suffix_1$+...+$suffix_m$, where: $prefix_{1..n}$ are valid prefixes; $suffix_{1..m}$ are valid suffixes; and prefix and suffix sequences are legal, are retained. Our valid prefixes are:
f, w, l, b, k, Al, s. (ف، و، ل، ب، ك، ال، س).
Our valid suffixes are: A, p, t, k, n, w, y, At, An, wn, wA, yn, kmA, km, kn, h, hA, hmA, hm, hn, nA, tmA, tm, and tn (ا، ة، ت، ك، ن، و، ي، ات، ان، ون، وا، ين، كما، كم، كن، ه، ها، هما، هم، هن، نا، تما، تم، تن).
Using these prefixes and suffixes, we generated a list of valid prefix and suffix sequences. For example, sequences where a coordinating conjunction (w or f) precedes a preposition (b, l, k), which in turn precedes a determiner (Al), is legal, for example in the word fbAlktab فبالكتاب (gloss: "and in the book") which is segmented to (f+b+Al+ktAb ف+ ب+ ال+ كتاب). Conversely, a determiner is not allowed to precede any other prefix. We used the following features:

- **Leading Prefixes:** conditional probability that a leading character sequence is a prefix.
- **Trailing Suffixes:** conditional probability that a trailing character sequence is a suffix.
- **LM Prob (Stem):** unigram probability of stem based on a language model that we trained from a corpus containing over 12 years worth of articles of Aljazeera.net (from 2000 to 2011). The corpus is composed of 114,758 articles containing 94 million words. This feature helps the segmenter determine if a stem is a valid word.
- **LM Prob:** unigram probability of stem with first suffix. This would help capture cases where a particular stem most likely appears with a specific suffix such as "jdA" (جدا meaning: "much") where the proper segmentation is "jd+A" and the word "jd" appears rarely.
- **Prefix—Suffix:** probability of prefix given suffix.
- **Suffix—Prefix:** probability of suffix given prefix.
- **Stem Template:** whether a valid stem template can be obtained from the stem. Stem templates are patterns that transform an Arabic root into a stem. For example, apply the template CCAC on the root "ktb" "كتب" produces the stem "ktAb" "كتاب" (meaning: book). To find stem templates, we adopted QATARA's stem-template module (Darwish et al., 2014), and we modified it to improve its coverage and accuracy. Character sequences leading to valid stem-templates are more likely to be stems.
- **Stem Lexicon:** whether the stem appears in a lexicon of automatically generated stems. This can help identify valid stems. This list is generated by placing roots into stem templates to generate a stem, which is retained if it appears in the aforementioned Aljazeera corpus.
- **Gazetteer Lexicon:** whether the stem that has no trailing suffixes appears in a gazetteer of person and location names. The gazetteer was extracted from Arabic Wikipedia in the manner described by (Darwish et al., 2012). We retained just word unigrams.

- **Function Words:** whether the stem is a function word such as “ElY” “على” (on) and “mn” “من” (from).

- **AraComLex:** whether the stem appears in the AraComLex[2] Arabic lexicon, which contains 31,753 stems of which 24,976 are nouns and 6,777 are verbs (Attia et al., 2011).

- **Buckwalter Lexicon:** whether the stem appears in the Buckwalter lexicon as extracted from the AraMorph package (Buckwalter, 2002).

- **Length Difference:** difference in length from the average stem length.

3.2. Learning:

We constructed feature vectors for each possible segmentation and marked correct segmentation for each word. We then used SVM^{rank} (Joachims, 2006) to learn the feature weights that would hopefully rank correct segmentations higher than incorrect ones. We used a linear kernel with a trade-off factor between training errors and margin (C) equal to 100, which is based on offline experiments done on a dev set. During testing, all possible segmentations with valid prefix-suffix combinations are generated, and the different segmentations are scored using the classifier. We had two varieties of Farasa. In the first, Farasa$_{Base}$, the classifier is used to segment all words directly. It also uses a small lookup list of concatenated stop-words where the letter “n” “ن” is dropped such as “EmA” “عما” (“En+mA” “عن+ ما”), and “mmA” “مما” (“mn+mA” “من+ ما”). In the second, Farasa$_{Lookup}$, previously seen segmentations during training are cached, and classification is applied on words that were unseen during training. The cache includes words that have only one segmentation during training, or words appearing 5 or more times with one segmentation appearing more than 70% of times.

3.3. Training and Testing:

For training, we used parts 1 (version 4.1), 2 (version 3.1), and 3 (version 2) of the the Penn Arabic Treebank (ATB). As mentioned earlier, many of the current results reported in the literature are done on subsets of the ATB. Testing done on a subset of the ATB is problematic due to its limited lexical diversity, leading to artificially high results. We created a new test set composed of 70 WikiNews articles (from 2013 and 2014) that cover a variety of themes, namely: politics, economics, health, science and technology, sports, arts, and culture. The articles are evenly distributed among the different themes (10 per theme). The articles contain 18,271 words.

Figure 1 shows a sample from the WikiNews corpus which has for each word its segmentation, POS tag, and diacritization. Currently, We use this corpus for evaluating segmentation accuracy for different systems, and we plan to expand its usage to cover more linguistic aspects. Generally speaking, segmentation of foreign named entities is one of the main reasons of errors for all segmentation systems as shown in the figure.

[2]`http://sourceforge.net/projects/aracomlex/`

System	Tokenization Error Rate
MADAMIRA	1.24%
QATARA	1.77%
Farasa$_{base}$	1.24%
Farasa$_{lookup}$	1.06%

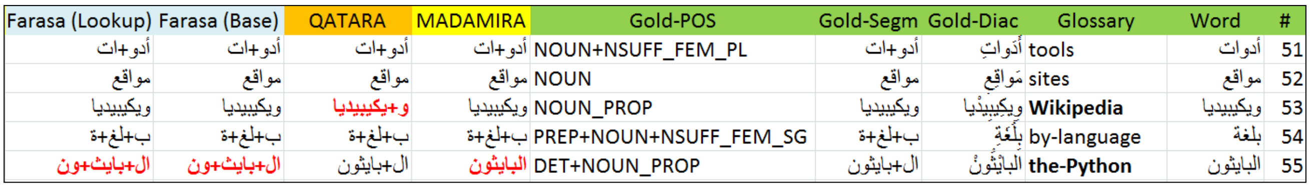
Table 1: Comparing Farasa word segmentation results to MADAMIRA and QATARA.

Table 1 compares the results of MADAMIRA, QATARA, and both varieties of Farasa. As the results show, Farasa yields lower segmentation errors than existing systems. For MADAMIRA, we used a segmentation scheme that matches ours exactly and considers the same prefixes and suffixes. Table 2 shows some output examples from Farasa$_{Lookup}$, MADAMIRA, and QATARA. From analyzing the errors in FARASA, we found that most of the errors were due to either: foreign named entities such as lynks (meaning: Linux) and bAlysky (meaning: Palisky); or to long words with more than four segmentations such as wlmfAj>thmA (w+l+mfAj>+t+hmA meaning and to surprise both of them). We think that adding larger gazetteers of foreign names may help reduce the first kind of errors. For the second type of errors, the classifier generates the correct segmentation, but it receives often a slightly lower score than the incorrect segmentation. Perhaps adding more features can help correct such errors.

To test the speed, we used Farasa without any caching to analyze 7.4 million words on a MacBook Pro with 2.5 GHz Intel Core i5 processor and 16 gigabytes of RAM. The model loading time was 7 seconds and segmentation time was exactly 129 seconds (a total of 2 minutes and 16 second). This is compared to more than 2.5 hours for MADAMIRA and more than 18 minutes for QATARA. Extrapolating from that, the segmenter can process 1 billion words in less than 5 hours. The tokenizer is written entirely in native Java code with no external dependencies.

4. Conclusion

In this paper we introduced Farasa, a new Arabic segmentation package, which uses SVM^{rank} to rank the possible legal segmentations of an Arabic word. We also introduced a new test set for the task based on news articles that cover different themes. We show that our system is more accurate than current state-of-the-art systems, while being much faster. The tool is implemented entirely in Java without any external dependencies. We plan to make both Farasa and test set publicly available. For future work, we plan to improve the underlying gazetteers and to improve our features to reduce segmentation errors even further.



Farasa (Lookup)	Farasa (Base)	QATARA	MADAMIRA	Gold-POS	Gold-Segm	Gold-Diac	Glossary	Word	#
أدو+ات	أدو+ات	أدو+ات	أدو+ات	NOUN+NSUFF_FEM_PL	أدو+ات	أَدواتِ	tools	أدوات	51
مواقع	مواقع	مواقع	مواقع	NOUN	مواقع	مَواقِع	sites	مواقع	52
ويكيبيديا	ويكيبيديا	و+يكيبيديا	ويكيبيديا	NOUN_PROP	ويكيبيديا	ويكِيبِيدْيا	**Wikipedia**	ويكيبيديا	53
ب+لغ+ة	ب+لغ+ة	ب+لغ+ة	ب+لغ+ة	PREP+NOUN+NSUFF_FEM_SG	ب+لغ+ة	بِلُغَةِ	by-language	بلغة	54
ال+بايث+ون	ال+بايث+ون	ال+بايثون	البايثون	DET+NOUN_PROP	ال+بايثون	البايْثُونْ	**the-Python**	البايثون	55

Figure 1: Sample from WikiNews corpus with segmentation errors highlighted

Word	FarasaLookup	MADA	QATARA
bmsbAr (بمسبار)	**b+msbAr (ب + مسبار)**	bmsbAr (بمسبار)	**b+msbAr (ب + مسبار)**
fysbwk (فيسبوك)	**fysbwk (فيسبوك)**	f+ysb+w+k (ف + يسب + و + ك)	**fysbwk (فيسبوك)**
kwrwnA (كورونا)	**kwrwnA (كورونا)**	kwr+w+nA (كور + و + نا)	**kwrwnA (كورونا)**
mlSqAt (ملصقات)	**mlSq+At (ملصق + ات)**	mlSqAt (ملصقات)	**mlSq+At (ملصق + ات)**
lynks (لينكس)	l+ynks (ل + ينكس)	l+ynks (ل + ينكس)	**lynks (لينكس)**
wlmfAj>thmA (ولمفاجأتهما)	wlmfAj>thmA (ولمفاجأتهما)	w+l+mfAj+>th+mA (و+ل+مفاج+أته+ما)	w+lmfAj>+th+mA (و+لمفاجأ+ته+ما)

Table 2: Examples that show some of the failings of the different systems. Correct segmentation is in bold.

5. Bibliographical References

Aljlayl, M. and Frieder, O. (2002). On arabic search: improving the retrieval effectiveness via a light stemming approach. In *CIKM-2002*, pages 340–347.

Attia, M., Pecina, P., Toral, A., Tounsi, L., and van Genabith, J. (2011). An open-source finite state morphological transducer for modern standard arabic. In *Proceedings of the 9th International Workshop on Finite State Methods and Natural Language Processing*, pages 125–133.

Beesley, K., Buckwalter, T., and Newton, S. (1989). Two-level finite-state analysis of arabic morphology. In *Proceedings of the Seminar on Bilingual Computing in Arabic and English*, pages 6–7.

Beesley, K. R. (1996). Arabic finite-state morphological analysis and generation. In *Proceedings of the 16th conference on Computational linguistics-Volume 1*, pages 89–94. Association for Computational Linguistics.

Buckwalter, T. (2002). Buckwalter {Arabic} morphological analyzer version 1.0.

Darwish, K. and Oard, D. W. (2007). Adapting morphology for arabic information retrieval*. In *Arabic Computational Morphology*, pages 245–262. Springer.

Darwish, K., Magdy, W., and Mourad, A. (2012). Language processing for arabic microblog retrieval. In *ACM CIKM-2012*, pages 2427–2430.

Darwish, K., Abdelali, A., and Mubarak, H. (2014). Using stem-templates to improve arabic pos and gender/number tagging. In *LREC-2014*.

Darwish, K. (2002). Building a shallow arabic morphological analyzer in one day. In *Computational Approaches to Semitic Languages, ACL-2002*, pages 1–8.

Diab, M. (2009). Second generation amira tools for arabic processing: Fast and robust tokenization, pos tagging, and base phrase chunking. In *2nd International Conference on Arabic Language Resources and Tools*.

Habash, N., Rambow, O., and Roth, R. (2009). Mada+tokan: A toolkit for arabic tokenization, diacritization, morphological disambiguation, pos tagging, stemming and lemmatization. In *Proceedings of the 2nd International Conference on Arabic Language Resources and Tools (MEDAR), Cairo, Egypt*, pages 102–109.

Habash, N., Roth, R., Rambow, O., Eskander, R., and Tomeh, N. (2013). Morphological analysis and disambiguation for dialectal arabic. In *HLT-NAACL*, pages 426–432.

Joachims, T. (2006). Training linear svms in linear time. In *ACM SIGKDD-2006*, pages 217–226. ACM.

Khoja, S. and Garside, R. (2001). *Automatic tagging of an Arabic corpus using APT*. Ph.D. thesis, University of Utah, Salt Lake City, Utah.

Khoja, S. (2001). Apt: Arabic part-of-speech tagger. In *NAACL*, pages 20–25.

Kiraz, G. A. (1998). Arabic computational morphology in the west. In *Proceedings of the 6th International Conference and Exhibition on Multi-lingual Computing*, pages 3–5. Citeseer.

Larkey, L. S., Ballesteros, L., and Connell, M. E. (2002). Improving stemming for arabic information retrieval: light stemming and co-occurrence analysis. In *Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 275–282. ACM.

Lee, Y.-S., Papineni, K., Roukos, S., Emam, O., and Hassan, H. (2003). Language model based arabic word segmentation. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics-Volume 1*, pages 399–406. Association for Computational Linguistics.

Maamouri, M., Bies, A., Buckwalter, T., Jin, H., and Mekki, W. (2005). Arabic treebank: Part 3 (full corpus) v 2.0 (mpg+ syntactic analysis). *Linguistic Data Consortium, Philadelphia*.

Maamouri, M., Graff, D., Bouziri, B., Krouna, S., Bies, A., and Kulick, S. (2010). Standard arabic morphological analyzer (sama) version 3.1. *Linguistic Data Consortium, Catalog No.: LDC2010L01*.

Monroe, W., Green, S., and Manning, C. D. (2014). Word segmentation of informal arabic with domain adaptation. *ACL, Short Papers*.

Pasha, A., Al-Badrashiny, M., Diab, M., El Kholy, A., Eskander, R., Habash, N., Pooleery, M., Rambow, O., and Roth, R. M. (2014). Madamira: A fast, comprehensive tool for morphological analysis and disambiguation of arabic. In *LREC-2014*, Reykjavik, Iceland.

Sajjad, H., Guzmn, F., Nakov, P., Abdelali, A., Murray, K., Obaidli, F. A., and Vogel, S. (2013). QCRI at IWSLT 2013: Experiments in Arabic-English and English-Arabic spoken language translation. In *Proceedings of the 10th International Workshop on Spoken Language Technology (IWSLT-13)*, December.