Farasa: A Fast and Furious Segmenter for Arabic

Ahmed Abdelali Kareem Darwish Nadir Durrani Hamdy Mubarak
Qatar Computing Research Institute
Hamad Bin Khalifa University
Doha, Qatar
{aabdelali,kdarwish,ndurrani,hmubarak}@qf.org.qa

Abstract

In this paper, we present Farasa, a fast and accurate Arabic segmenter. Our approach is based on SVM-rank using linear kernels. We measure the performance of the segmenter in terms of accuracy and efficiency, in two NLP tasks, namely Machine Translation (MT) and Information Retrieval (IR). Farasa outperforms or is at par with the state-of-the-art Arabic segmenters (Stanford and MADAMIRA), while being more than one order of magnitude faster.

1 Introduction

Word segmentation/tokenization is one of the most important pre-processing steps for many NLP task, particularly for a morphologically rich language such as Arabic. Arabic word segmentation involves breaking words into its constituent prefix(es), stem, and suffix(es). For example, the word "wktAbnA"[1] "وكتابنا" (gloss: "and our book") is composed of the prefix "w" "و" (and), stem "ktAb" "كتاب" (book), and a possessive pronoun "nA" "نا" (our). The task of the tokenizer is to segment the word into "w+ ktAb +nA" "و+ كتاب +نا". Segmentation has been shown to have significant impact on NLP applications such as MT and IR.

Many Arabic segmenters have been proposed in the past 20 years. These include rule based analyzers (Beesley et al., 1989; Beesley, 1996; Buckwalter, 2002; Khoja, 2001), 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%.

We introduce a new segmenter, Farasa ("insight" in Arabic), an SVM-based segmenter that uses a variety of features and lexicons to rank possible segmentations of a word. The features include: likelihoods of stems, prefixes, suffixes, their combinations; presence in lexicons containing valid stems or named entities; and underlying stem templates.

We carried out extensive tests comparing Farasa with two state-of-the-art segmenters: MADAMIRA (Pasha et al., 2014), and the Stanford Arabic segmenter (Monroe et al., 2014), on two standard NLP tasks namely MT and IR. The comparisons were done in terms of accuracy and efficiency. We trained Arabic↔English Statistical Machine Translation (SMT) systems using each of the three segmenters. Farasa performs clearly better than Stanford's segmenter and is at par with MADAMIRA, in terms of BLEU (Papineni et al., 2002). On the IR task, Farasa outperforms both with statistically significant improvements. Moreover, we observed Farasa to be at least an order of magnitude faster than both. Farasa also performs slightly better than the two in an intrinsic evaluation. Farasa has been made freely available.[2]

2 Farasa

Features: In this section we introduce the features and lexicons that we used for seg-

[1] Buckwalter encoding is used exclusively in this paper

[2] Tool available at: http://alt.qcri.org/tools/farasa/

Proceedings of NAACL-HLT 2016 (Demonstrations), pages 11–16,
San Diego, California, June 12-17, 2016.

mentation. 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.

- LM Prob: unigram probability of stem with first suffix.

- 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 used the module described in Darwish et al. (2014).

- 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) and 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 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.

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 feature weights. 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 test, 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.

Training and Testing: For training, we used parts 1 (version 4.1), 2 (version 3.1), and 3 (version 2) of

	MADAMIRA	Farasa$_{base}$	Farasa$_{lookup}$
Accuracy	98.76%	98.76%	98.94%

Table 1: Segmentation Accuracy

the the Penn Arabic Treebank (ATB). Many of the current results reported in the literature are done on subsets of the Penn Arabic Treebank (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. Table 1 compares segmentation accuracy for both versions of Farasa with MADAMIRA, where both were configured to segment all possible affixes. We did not compare to Stanford, because it only segments based on the ATB segmentation scheme. Farasa$_{lookup}$ performs slightly better than MADAMIRA. From analyzing the errors in Farasa, we found that most of the errors were due to either: foreign named entities such as "lynks" "لينكس" (meaning: Linux) and "bAlysky" "باليسكي" (meaning: Palisky); or to long words with more than four segmentations such as "wlmfAj}thmA" "ولمفاجئتهما" ("w+l+mfAj}+t+hmA" "و+ ل+ مفاجئ+ ت+ هما") (meaning "and to surprise both of them"). Perhaps, adding larger gazetteers of foreign names would 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.

3 Machine Translation

Setup: We trained Statistical Machine Translation (SMT) systems for Arabic↔English, to compare Farasa with Stanford and MADAMIRA[3]. The comparison was done in terms of BLEU (Papineni et al., 2002) and processing times. We used concatenation of IWSLT TED talks (Cettolo et al., 2014) (containing 183K Sentences) and NEWS corpus (containing 202K Sentences) to train phrase-based systems.

Seg	iwslt$_{12}$	iwslt$_{13}$	Avg	Time
MADAMIRA	30.4	30.8	30.6	4074
Stanford	30.0	30.5	30.3	395
Farasa	30.2	30.8	30.5	80

Table 2: Arabic-to-English Machine Translation, BLEU scores and Time (in seconds)

Systems: We used Moses (Koehn et al., 2007), a state-of-the-art toolkit with the the settings described in (Durrani et al., 2014a): these include a maximum sentence length of 80, Fast-Aligner for word-alignments (Dyer et al., 2013), an interpolated Kneser-Ney smoothed 5-gram language model with KenLM (Heafield, 2011), used at runtime, MBR decoding (Kumar and Byrne, 2004), Cube Pruning (Huang and Chiang, 2007) using a stack size of 1,000 during tuning and 5,000 during testing. We tuned with the k-best batch MIRA (Cherry and Foster, 2012). Among other features, we used lexicalized reordering model (Galley and Manning, 2008), a 5-gram Operation Sequence Model (Durrani et al., 2011), Class-based Models (Durrani et al., 2014b)[4] and other default parameters. We used an unsupervised transliteration model (Durrani et al., 2014c) to transliterate the OOV words. We used the standard tune and test set provided by the IWSLT shared task to evaluate the systems.

In each experiment, we simply changed the segmentation pipeline to try different segmentation. We used ATB scheme for MADAMIRA which has shown to outperform its alternatives (S2 and D3) previously (Sajjad et al., 2013).

Results: Table 2 compares the Arabic-to-English SMT systems using the three segmentation tools. Farasa performs better than Stanford's Arabic segmenter giving an improvement of +0.25, but slightly worse than MADAMIRA (-0.10). The differences are not statistically significant. For efficiency, Farasa is faster than Stanford and MADAMIRA by a factor of 5 and 50 respectively.[5] The run-time of MADAMIRA makes it cumbersome to run on bigger corpora like the multiUN (UN) (Eisele and

[3]Release-01292014-1.0 was used in the experiments

[4]We used mkcls to cluster the data into 50 clusters.

[5]Time is the average of 10 runs on a machine with 8 Intel i7-3770 cores, 16 GB RAM, and 7 Seagate disks in software RAID 5 running Linux 3.13.0-48

Seg	iwslt$_{12}$	iwslt$_{13}$	Avg	Time
MADAMIRA	19.6	19.1	19.4	1781
Stanford	17.4	17.2	17.3	692
Farasa	19.2	19.3	19.3	66

Table 3: English-to-Arabic Machine Translation, BLEU scores and Time (in seconds)

Chen, 2010) which contains roughly 4M sentences. This factor becomes even daunting when training a segmented target-side language model for English-to-Arabic system. Table 3 shows results from English-to-Arabic system. In this case, Stanford performs significantly worse than others. MADAMIRA performs slightly better than Farasa. However, as before, Farasa is more than multiple orders of magnitude faster.

4 Information Retrieval

Setup: We also used extrinsic IR evaluation to determine the quality of stemming compared to MADAMIRA and the Stanford segmenter. We performed experiments on the TREC 2001/2002 cross language track collection, which contains 383,872 Arabic newswire articles, containing 59.6 million words), and 75 topics with their relevance judgments (Oard and Gey, 2002). This is presently the best available large Arabic information retrieval test collection. We used Mean Average Precision (MAP) and precision at 10 (P@10) as the measures of goodness for this retrieval task. Going down from the top a retrieved ranked list, Average Precision (AP) is the average of precision values computed at every relevant document found. P@10 is the same as MAP, but the ranked list is restricted to 10 results. We used SOLR (ver. 5.6)[6] to perform all experimentation. SOLR uses a *tf-idf* ranking model. We used a paired 2-tailed t-test with p-value less than 0.05 to ascertain statistical significance. For experimental setups, we performed letter normalization, where we conflated: variants of "alef", "ta marbouta" and "ha", "alef maqsoura" and "ya", and the different forms of "hamza". Unlike MT, Arabic IR performs better with more elaborate segmentation which improves matching of core units of meaning, namely stems. For MADAMIRA, we used the D34MT scheme, where all affixes are segmented. Stanford tokenizer only provides the ATB tokenization scheme. Farasa was used with the default scheme, where all affixes are segmented.

[6]http://lucene.apache.org/solr/

Stemming	MAP	P@10	Time
Words	0.20	0.34	-
MADAMIRA	0.26 ^{w,s}	0.39 ^{w}	21:27:21
Stanford	0.22 ^{w}	0.37	03:43:25
Farasa	0.28 ^{w,s,m}	0.43 ^{w,s,m}	00:15:26

Table 4: Retrieval Results in MAP and P@10 and Processing Time (in hh:mm:ss). For statistical significance, w = better than words, s = better than Stanford, and m = better than MADAMIRA

Results: Table 4 summarizes the retrieval results for using words without stemming and using MADAMIRA, Stanford, and Farasa for stemming. The table also indicates statistical significance and reports on the processing time that each of the segmenters took to process the entire document collection. As can be seen from the results, Farasa outperformed using words, MADAMIRA, and Stanford significantly. Farasa was an order of magnitude faster than Stanford and two orders of magnitude faster than MADAMIRA.

5 Analysis

The major advantage of using Farasa is speed, without loss in accuracy. This mainly results from optimization described earlier in the Section 2 which includes caching and limiting the context used for building the features vector. Stanford segmenter uses a third-order (i.e., 4-gram) Markov CRF model (Green and DeNero, 2012) to predict the correct segmentation. On the other hand, MADAMIRA bases its segmentation on the output of a morphological analyzer which provides a list of possible analyses (independent of context) for each word. Both text and analyses are passed to a feature modeling component, which applies SVM and language models to derive predictions for the word segmentation (Pasha et al., 2014). This hierarchy could explain the slowness of MADAMIRA versus other tokenizers.

6 Conclusion

In this paper we introduced Farasa, a new Arabic segmenter, which uses SVM for ranking. We compared our segmenter with state-of-the-art segmenters MADAMIRA and Stanford, on standard

MT and IR tasks and demonstrated Farasa to be significantly better (in terms of accuracy) than both on the IR tasks and at par with MADAMIRA on the MT tasks. We found Farasa by orders of magnitude faster than both. Farasa has been made available for use[7] and will be added to Moses for Arabic tokenization.

References

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

Mohammed Attia, Pavel Pecina, Antonio Toral, Lamia Tounsi, and Josef van Genabith. 2011. An open-source finite state morphological transducer for modern standard arabic. In *Workshop on Finite State Methods and Natural Language Processing*, pages 125–133.

Kenneth Beesley, Tim Buckwalter, and Stuart Newton. 1989. Two-level finite-state analysis of arabic morphology. In *Proceedings of the Seminar on Bilingual Computing in Arabic and English*, pages 6–7.

Kenneth R Beesley. 1996. Arabic finite-state morphological analysis and generation. In *ACL*, pages 89–94.

Tim Buckwalter. 2002. Buckwalter {Arabic} morphological analyzer version 1.0.

Mauro Cettolo, Jan Niehues, Sebastian Stüker, Luisa Bentivogli, and Marcello Federico. 2014. Report on the 11th IWSLT Evaluation Campaign. *IWSLT-14*.

Colin Cherry and George Foster. 2012. Batch Tuning Strategies for Statistical Machine Translation. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, HLT-NAACL'12, pages 427–436, Montréal, Canada.

Kareem Darwish and Douglas W Oard. 2007. Adapting morphology for arabic information retrieval*. In *Arabic Computational Morphology*, pages 245–262.

Kareem Darwish, Walid Magdy, and Ahmed Mourad. 2012. Language processing for arabic microblog retrieval. In *ACM CIKM-2012*, pages 2427–2430.

Kareem Darwish, Ahmed Abdelali, and Hamdy Mubarak. 2014. Using stem-templates to improve arabic pos and gender/number tagging. In *LREC-2014*.

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

[7] http://alt.qcri.org/tools/farasa/

Mona Diab. 2009. Second generation amira tools for arabic processing: Fast and robust tokenization, pos tagging, and base phrase chunking. In *Intl. Conference on Arabic Language Resources and Tools*.

Nadir Durrani, Helmut Schmid, and Alexander Fraser. 2011. A Joint Sequence Translation Model with Integrated Reordering. In *ACL'11*, pages 1045–1054, Portland, Oregon, USA, June.

Nadir Durrani, Barry Haddow, Philipp Koehn, and Kenneth Heafield. 2014a. Edinburgh's Phrase-based Machine Translation Systems for WMT-14. In *WMT'14*, pages 97–104, Baltimore, Maryland, USA.

Nadir Durrani, Philipp Koehn, Helmut Schmid, and Alexander Fraser. 2014b. Investigating the Usefulness of Generalized Word Representations in SMT. In *COLING'14*, pages 421–432, Dublin, Ireland.

Nadir Durrani, Hassan Sajjad, Hieu Hoang, and Philipp Koehn. 2014c. Integrating an Unsupervised Transliteration Model into Statistical Machine Translation. In *EACL'14*, pages 148–153, Gothenburg, Sweden.

Chris Dyer, Victor Chahuneau, and Noah A. Smith. 2013. A Simple, Fast, and Effective Reparameterization of IBM Model 2. In *Proceedings of NAACL'13*.

Andreas Eisele and Yu Chen. 2010. MultiUN: A Multilingual Corpus from United Nation Documents. In *LREC-2010*, Valleta, Malta, May.

Michel Galley and Christopher D. Manning. 2008. A Simple and Effective Hierarchical Phrase Reordering Model. In *EMNLP-2008*, pages 848–856, Honolulu, Hawaii, October.

Spence Green and John DeNero. 2012. A class-based agreement model for generating accurately inflected translations. In *ACL-2012*.

Nizar Habash, Owen Rambow, and Ryan Roth. 2009. Mada+ tokan: A toolkit for arabic tokenization, diacritization, morphological disambiguation, pos tagging, stemming and lemmatization. In *MEDAR*, pages 102–109.

Kenneth Heafield. 2011. KenLM: faster and smaller language model queries. In *Sixth Workshop on Statistical Machine Translation, EMNLP-2011*, pages 187–197, Edinburgh, Scotland, United Kingdom, July.

Liang Huang and David Chiang. 2007. Forest Rescoring: Faster Decoding with Integrated Language Models. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, ACL'07, pages 144–151, Prague, Czech Republic.

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

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

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi,

Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondrej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open source toolkit for statistical machine translation. In *ACL-2007*, Prague, Czech Republic.

Shankar Kumar and William J. Byrne. 2004. Minimum Bayes-Risk Decoding for Statistical Machine Translation. In *Proceedings of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, HLT-NAACL'04, pages 169–176, Boston, Massachusetts, USA.

Will Monroe, Spence Green, and Christopher D Manning. 2014. Word segmentation of informal arabic with domain adaptation. *ACL, Short Papers*.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *ACL-2002*, Philadelphia, PA, USA.

Arfath Pasha, Mohamed Al-Badrashiny, Mona Diab, Ahmed El Kholy, Ramy Eskander, Nizar Habash, Manoj Pooleery, Owen Rambow, and Ryan M Roth. 2014. Madamira: A fast, comprehensive tool for morphological analysis and disambiguation of arabic. In *LREC-2014*, Reykjavik, Iceland.

Hassan Sajjad, Francisco GuzmÃąn, Preslav Nakov, Ahmed Abdelali, Kenton Murray, Fahad Al Obaidli, and Stephan Vogel. 2013. QCRI at IWSLT 2013: Experiments in Arabic-English and English-Arabic spoken language translation. In *IWSLT-13*, December.