# Automatic diacritization of Arabic transcripts for automatic speech recognition

**Conference Paper** · December 2005

CITATIONS
22

READS
96

**3 authors**, including:

Srinivas Bangalore
AT&T
**186** PUBLICATIONS **3,546** CITATIONS
SEE PROFILE

Shrikanth S Narayanan
University of Southern California
**879** PUBLICATIONS **14,417** CITATIONS
SEE PROFILE

# Automatic Diacritization of Arabic Transcripts for Automatic Speech Recognition

**Sankaranarayanan Ananthakrishnan, Shrikanth S. Narayanan**
Speech Analysis and Interpretation Laboratory
Department of Electrical Engineering (Systems)
University of Southern California
*http://sail.usc.edu*
ananthak@usc.edu, shri@sipi.usc.edu

**Srinivas Bangalore**
AT&T Labs – Research
180 Park Avenue
Florham Park, NJ 07932, U.S.A.
srini@research.att.com

## Abstract

Arabic orthography does not provide full vocalization of the text, and the reader is expected to infer short vowels from the context of the sentence. Inferring the full form of a word is useful when developing Arabic speech and language processing tools, since it is likely to reduce ambiguity in these tasks. In this paper, we present generative techniques for recovering vowels and other diacritics that are contextually appropriate, and present evaluation results for these techniques on widely available data sets.

## 1. Introduction

Arabic is a Semitic language spoken chiefly in the Middle East (including North Africa) by approximately 225 million people (ranked 4th by total number of speakers). In spite of its importance, computational tools, such as automatic speech recognition (ASR) and machine translation systems, for processing Arabic speech and text are woefully inadequate. Development of natural language processing tools for Arabic has been hindered by

- Non-availability of large corpora for training. Statistical approaches to language processing are currently well entrenched in the community, and they work very well, but one of their downsides is that they require large amounts of data to train. Most large corpora currently available cater to English, and resources for other languages, including Arabic, have only been recently made available.
- The existence of a number of dialects with a multitude of pronunciations. For example, Levantine is spoken in the region around Lebanon and Syria, Egyptian (or Nile) Arabic is spoken in most areas of north-eastern Africa, Gulf Arabic is spoken in Saudi Arabia and parts of Yemen, while variants of Maghreb Arabic are spoken in Saharan Africa, including Tunisia, Algeria, and Morocco.
- Issues with representation of Arabic script on current computer systems. This is more a nuisance than a real problem. Different "standards" have come up over time and mutual incompatibilities and lack of support for these in most popular software packages and web browsers have exacerbated it. It is not uncommon even now to find Arabic content in PDF or image formats, which is almost entirely useless for natural language processing.
- The fact that script Arabic does not provide full vocalization of the text – the reader is expected to infer short vowels and other missing cues from the context. This is one of the key issues we address in this paper.

In our work, we deal with Modern Standard Arabic (MSA) and are not concerned about dialectal variations at this point. The LDC is now making data available in large quantities, and most issues with representation have been solved by Unicode (UTF-8) and Buckwalter's transliteration scheme [1]. The latter provides a way to represent Arabic script using only 7-bit ASCII codes. Hence, we focus on the last problem in the list.

"... وروى الشرطي في لونغ بيتش يفيدد مارندر أنها المرة الثانية خلال يومين التي يحاول فيها ستيفن كنت استقلال الباص لاحصول على الميراث الموعود ..."

(a) Arabic script (read from right to left, top to bottom)

**"... wrwY Al$rTy fy lwng byt$ dyfyd mArndr >nhA Almrp AlvAnyp xlAl ywmyn Alty yHAwl fyhA styfn knt AstqlAl AlbAS llHSwl ElY AlmyrAv AlmwEwd ..."**

(b) Buckwalter transliteration (undiacritized)

**"... warawaY Al$uroTiy~u fiy luwnog biyt$ diyfiyd mArndr >an~ahA Almar~apu AlvAniyapu xilAla yawomayoni Al~atiy yuHAwilu fiyhA stiyfin kinot {isotiqolAla AlbASi lilHuSuwli EalaY AlmiyrAvi AlmawoEuwdi ..."**

(c) Buckwalter transliteration (fully diacritized)

Full vocalization of Arabic script is provided by the insertion of *diacritics* in the text. However, with the exception of important writings such as the Qu'ran, where vocalization errors are not permissible, these diacritics are almost never written, and in most cases are expected to be inferred from the context. Most of the Arabic corpora available for training acoustic and language models for ASR belong to this category. As will be described below, a single script form may correspond to one of several possible vocalizations, and this causes problems for both acoustic modeling (the location and identity of short vowels in the transcript is unknown) as well as language modeling (states for potentially different vocalizations will be merged, leading to poor estimates of the true distribution).

Diacritization is vital if the output of the ASR is to be used for downstream processing by a machine translation or dialogue unit, since different vocalizations of the same script form usually have different meanings. For instance, [2] gives the example of a root form, ***ktb*** (basic meaning 'write'), which may be interleaved with different patterns to obtain different surface forms such as ***kataba*** ('he wrote') or ***kutub*** ('books'). These forms cannot be disambiguated without knowledge of the context. Sample text from the Arabic treebank is shown above in script and transliterated (both undiacritized and diacritized) forms.

The problem of diacritization has been investigated by others [2,3,4] – in [2], the effort is directed at building an ASR for Egyptian colloquial Arabic, whereas [3] and [4] present general solutions directed at no specific application. The focus of our present work is to develop techniques for automatic diacritization of script Arabic so as to facilitate the development of speech recognition and NLU systems for MSA. We plan to move beyond simple *n*-gram based generative models that have been employed in previous studies toward richer structures that integrate more contextual and morphological information for predicting diacritics. We also plan to investigate the effects of improved diacritization on ASR performance (in terms of word-error-rate) and machine translation accuracy (in terms of the BLEU score).

## 2. Data Corpus

In order to train an automatic diacritizer for Arabic text, we use the Arabic Treebank (catalogues LDC2005T02, LDC2004T02 and LDC2005T20) released by the LDC. These corpora contain newswire text from the AFP, Ummah, and An-Nahar news agencies totaling about 554,000 words after clitics were merged.

| *Corpus* | *Size (words)* |
|---|---|
| AFP | 127,915 |
| Ummah | 127,818 |
| An-Nahar | 298,796 |

**Table 1** Composition of Arabic Treebank data

They contain (in Buckwalter's transliteration scheme) both the undiacritized Arabic script and the corresponding diacritized version, in addition to part-of-speech tags and a treebank-style parsed annotation. The diacritized text also has the affixes (including case-endings) separated from the stems of words. For now, we ignore the other annotations and focus on the parallel corpora with and without diacritics (we do consider part-of-speech tags in a few trial experiments). The data are split in two sets – a training set made up of about 541,000 words and a test set of about 13,300 words. Statistical models (described in the following sections) are built using the training corpus and diacritics are inserted into words in the test corpus so that the most likely sequence of diacritized words is obtained at the output.

## 3. Models for diacritization

We adopt both statistical and knowledge-based approaches to solving the diacritization problem. In this section, we describe various approaches that we use to predict the diacritized version of an undiacritized Arabic text.

### 3.1 Baseline

We use the same, simple, maximum-likelihood unigram baseline described in [2,3]. We replace each undiacritized word in the test corpus with the corresponding diacritized version that occurs most frequently in the training corpus according to the following equation:

$$w_i^{d'} = \arg\max_{w^d} p\left(w^d \mid w_i^u\right) \qquad (1)$$

In the above equation, $w_i^{d'}$ is the chosen diacritized form for the $i^{\text{th}}$ word in the input (undiacritized) stream, represented by $w_i^u$.

Due to the highly inflected nature of Arabic and the relatively small size of our training corpus, about 6% of the undiacritized words in the test set did not occur in the training set – these unseen words have no statistics associated with them. We deal with the unseen words by simply copying the undiacritized form into the diacritized version. This simple baseline gives about 75% accuracy at the word level.

### 3.2 Word-level language model

This is simply an extension of the baseline model to use more left context. More formally, the diacritizer picks $w_i^d$ as the diacritized form of input word $w_i^u$ and previous history $\left(w_{i-1}^u, w_{i-1}^d\right), \left(w_{i-2}^u, w_{i-2}^d\right)$ (for the trigram case) according to the following joint model maximization criterion.

$$w_d^* = \arg\max_{w_1^d, w_2^d, \ldots, w_n^d} \prod_{i=2}^{n} p\left(w_i^d, w_i^u \mid w_{i-1}^d, w_{i-1}^u, w_{i-2}^d, w_{i-2}^u\right) \qquad (2)$$

Given the fact that we are dealing with compound words that have not been separated morphologically, sparsity is a major problem with this model, and we did not expect much improvement over the baseline. This model gives a word level accuracy of just over 77.3% for trigram context, an improvement of about 2.3% over the baseline.

### 3.3 Character-level language model

One way to alleviate the effects of sparsity and unseen words in the test corpus is to operate at the character level. We assume that each character in the undiacritized text may "generate" diacritics. For example, an alignment at the character level for the tuple (`AlmwEwd`, `AlmawoEuwdi`) is given by (`A` → `A`, `l` → `l`, `m` → `ma`, `w` → `wo`, `E` → `Eu`, `w` → `w`, `d` → `di`). A "NULL diacritic" ensures that it is possible for a character to generate no diacritic. This generative process is currently strictly left-to-right, which is not true for all cases, and we plan to extend it in future work. We used a 4-gram character model that gave us a word-level accuracy of 74.8%.

We combined the word- and character-level models in a "post-processing" approach. For each unknown word (words not seen in the training set) in the test corpus, the output of the word-level diacritizer was the token ***<UNK>***. During the post-processing phase, each occurrence of ***<UNK>*** was replaced by the output of the character-level diacritizer for that word. There are probably more gains to be had by integrating the word- and character-models before the decoding phase – this is on our agenda for future work.

### 3.4 Re-ranking knowledge sources

The motivation for incorporating knowledge sources is as follows. One major issue with the statistical models described above is that of unseen words in the test data. Even though we use over 541,000 words to train the models, about 800 of 13,300 words in the test corpus (~ 6%) were not seen during training. Unseen words completely break the word-based models, and even though we try to predict these using the character-based approach described above, the prediction is open-set and leads to several errors. The character model accuracy is 76.5% on seen words, but only 48.2% on unseen words. If, given the unseen, undiacritized baseforms, we are able to constrain the search space of possible diacritizations and re-rank them with the character-based model, we might eliminate several errors. To this end, we use the Buckwalter Morphological Analyzer [5], which, given an undiacritized baseform, produces a list of all possible diacritizations with a complete morphological analysis. It accomplishes this in a rule-based fashion using a dictionary of prefixes, stems and suffixes in Arabic. The morphological analyzer operates on individual words in a completely context-independent fashion, and does not attach a score to any item on the list.

We use the morphological analyzer as follows: for each unseen word in the test corpus, we generate a list of possible diacritizations $BW(w_u)$. We then compute a score for each hypothesis in this list using the character-based generative model $(\lambda)$ described in the previous section. The best-scoring hypothesis is then inserted into the appropriate location in the output generated by the word-only model.

$$w_d^* = \arg\max_{w \in BW(w_u)} p_\lambda(w) \qquad (3)$$

This approach gives a very modest improvement (0.06% absolute improvement) over the other methods, but opens up the possibility of using the morphological analyzer to construct a lattice of possible diacritizations and search only within this space, as opposed to open set prediction (even at the word level).

Another interesting aspect to consider would be the morphological structure of Arabic words, which we currently do not exploit. Explicit handling of clitics (separating the word into prefix, stem and suffix) may offer additional improvements in diacritization accuracy as presented in [4].

### 3.5 Exploiting part-of-speech information

A significant proportion of errors in the diacritized output has to do with incorrect prediction of case-endings, which are indicated by the suffixes {*aiuFKN*} in the Buckwalter transliteration. We believe that the use of syntactic information, particularly part-of-speech, can go a long way in reducing the occurrence of such errors. We carried out an experiment in which we assumed that the correct part-of-speech tags for the undiacritized words in the test corpus are known, and attempted to predict the diacritics with this additional information. This gave us a significant boost in output accuracy (3.4% absolute improvement at the word level). Note that while this setup cannot be part of a fully automatic diacritizer, it does illustrate the usefulness of part-of-speech for diacritization.

However, automatic PoS tagging of undiacritized Arabic is itself a hard problem, because most words are associated with a compound PoS tag that accounts for the prefix, stem, and suffix. As an example, the word `<i$ArapK`, which can be morphologically decomposed into `<i$Ar+ap+K` (meaning "indication" or "sign", fem. sing., indef. gen. case), has the compound tag `NOUN+NSUFF_FEM_SG+CASE_INDEF_GEN`. Although the set of compound part-of-speech tags as annotated in the Treebank is smaller than the number of unique lexical items, it is unclear at this point whether this tagging problem is easier than the diacritization problem. A preliminary trigram based tagger gave us about 70% accuracy on the compound tags with the undiacritized words as input.

We are currently looking at ways to exploit dependencies (or lack thereof) among the various elements of the compound part-of-speech tags to simplify this problem. Graphical models provide an elegant way to represent these interdependencies and could be promising in this regard.

| *Model* | *Word Accuracy %* |
|---|---|
| Baseline (unigram) | 74.96% |
| Word trigram | 77.30% |
| Character 4-gram | 74.80% |
| Word trigram + character 4-gram | 80.21% |
| Word trigram + Buckwalter (KS) re-ranking | 80.27% |
| Word trigram + true POS* | 83.59% |
| Word trigram + character 4-gram + true POS* | 86.50% |

**Table 2** Results for Arabic diacritization (* note that the last two results with "true" POS are not a part of the completely automatic diacritizer; they are included to illustrate the effect that additional knowledge of POS can have on diacritization accuracy)

Excellent results have been reported for the part-of-speech tagging problem using Buckwalter's morphological analyzer [6], and we are looking into this possibility.

Diacritization performance results for each of the methods previously discussed are shown in Table 2.

## 4. Speech Recognition for Arabic

We used the diacritizer described and built in the previous sections to process Arabic script data that was used for both acoustic and language modeling for a large vocabulary ASR for the broadcast news task. Even though the process of diacritization is noisy, the hope is that models built from diacritized data will be superior to those built using just script Arabic.

### 4.1 Language modeling

We use a simple trigram language model to develop our baseline system. We make use of three different data sources for language modeling. The first is the Arabic Treebank data, which we used for the diacritization experiments. In terms of transcription accuracy, this data set is the cleanest, since all the text in this set is correctly diacritized. We also diacritize (using our automated approach with the word- and character-based models) transcripts from the LDC TDT4 Broadcast News data set (catalogues LDC2005S11, LDC2005T16) and a subset of the Arabic Gigaword corpus (LDC2003T12), totaling about 16.5 million words. Language models are built using both diacritized and undiacritized text in order to evaluate the effectiveness of diacritization on ASR performance. Table 3 shows the split size of the different subsets.

| *Corpus* | *Size (words)* |
|---|---|
| Arabic Treebank | 554,000 |
| TDT4 Transcripts | 423,258 |
| Gigaword (XIA subset) | 15.6m |

**Table 3** Corpora for building Arabic language model

### 4.2 Acoustic modeling

We built standard context-dependent triphone acoustic models based on the HMM framework. The reference text was diacritized and each word

in the pronunciation lexicon was mapped to a sequence of phonemes. In other words, a given (diacritized) Arabic text could be mapped deterministically into a linear sequence of phonemes. This is the standard approach to training acoustic models in languages such as English.

The training algorithm started from a set of bootstrap acoustic models trained using about four hours of fully-voweled Arabic data from the LDC WestPoint corpus (LDC2002S02). The data used for training our acoustic models came from the LDC TDT4 News Broadcast corpus, which consists of about 90 hours of radio and television news broadcasts along with transcriptions and topic segmentation. We use standard 39-dimensional MFCC features (including delta and acceleration) for our baseline system.

### 4.3 Experimental results

The Arabic ASR was evaluated on a 12-hour subset of the TDT4 corpus, which we set aside for testing purposes. We stripped the final recognition results of diacritics for evaluation against the reference. This is done since the true diacritized reference is not available for the TDT4 data set. Table 4 shows word error rate (WER) figures for the baseline system.

| *Training reference* | *ASR WER %* |
|---|---|
| Diacritized | 41.1% |

**Table 4** Baseline ASR performance for Arabic

## 5. Discussion and future directions

In this paper, we described the development of a high-performance automatic diacritization tool for pre-processing raw Arabic text for use in natural language applications. We used this tool to process Arabic corpora for acoustic and language modeling and demonstrated that the use of diacritized text leads to an increase in ASR accuracy. This paper presents work that is very much in progress. Development of accurate diacritization tools opens up a number of possibilities for acoustic and language modeling for both ASR and machine translation. We list below some of the possibilities that we are looking to explore in future work.

### 5.1 Diacritization

Although we argue that diacritization of Arabic script is useful for building natural language processing tools for that language, the noise (errors) introduced by the automatic diacritizer may adversely affect their performance. Given that speakers of Arabic are able to achieve almost complete disambiguation when reading undiacritized text, it is certain that our accuracy figures probably do not represent the ceiling of what may be achieved through better statistical models in combination with knowledge sources. In this regard, we plan to implement the following ideas in future work.

#### *5.1.1 Maximum-entropy approach*

We experimented with non-generative models such as maximum-entropy, which enable us to use features beyond just lexical items (such as part-of-speech, morphology, etc., although we have not yet tried these features), and allow us to take into account both left and right context. Preliminary experiments at the character level (with left and right context) provided results that were a little worse than the *n*-gram based character model (about 70% word level accuracy). This could be due to the fact that the generative model is not data-deficient at the character level and is thus able to capture patterns in the stream better than the maximum-entropy based model. It would be an interesting study to extend the maxent models to the word level, where sparsity is a limiting issue with the generative model, and the former may then have an advantage over the latter.

#### *5.1.2 Syntactic constraints*

Examination of the output of our automatic diacritizer revealed that a significant proportion of errors was caused by the choice of incorrect case-endings. These errors may be avoided to a large extent by imposing syntactic constraints such as case agreement. While in theory case agreement can be captured at the word-level, in practice this is

not possible due to the relatively small size of the training corpus. Our initial experiments with part-of-speech showed that if syntactic information is included in the diacritization process, many of these errors can be eliminated. We plan to extend these ideas in future work in conjunction with the maximum-entropy models discussed in the previous section.

### 5.2 Acoustic modeling

Once the diacritized transcriptions corresponding to an Arabic utterance are available, building acoustic models becomes a straightforward task. The effect of errors in diacritization can be mitigated by the following iterative process.

a) Train the acoustic models using the best-hypothesis transcription, but keep an *n*-best lattice of possible transcriptions for each utterance.
b) With the acoustic models thus obtained, construct a pronunciation network for each training utterance from the *n*-best transcriptions, and let the recognizer choose the best pronunciation. This is more likely to be the correct transcription than the one used to train the acoustic models in the first stage.
c) Re-train the acoustic models with the new best hypothesis.
d) Carry out steps (b) and (c) iteratively to the point where the best-hypothesis transcriptions do not change across iterations.

The idea is that we are attempting to use acoustic evidence to "correct" the diacritization provided by the text-only system.

To build acoustic models from *undiacritized* text, the word entries in the pronunciation lexicon can be stripped of diacritics while preserving the original pronunciation (derived from diacritized text). In this lexicon, it is possible for one (undiacritized) word to have multiple pronunciations. The reference text in this case is undiacritized. With this approach, a given word sequence is mapped into a lattice of possible pronunciations (rather than a linear sequence of phonemes), and the acoustic model training algorithm chooses the best path through this pronunciation lattice. We expect that the ambiguity at the lexical level is resolved at the phonetic level by the bootstrap acoustic models. We plan to compare the performance of ASR models built in this fashion with the results obtained using diacritized text for training.

### 5.3 Language modeling

Traditional *n*-gram language models can easily be built from script or diacritized Arabic. Using the diacritized version will ensure that states corresponding to different vocalizations will not be merged, leading to better estimates of probability distributions. However, the main advantage of working in the diacritized domain is that it enables us to scale beyond the *n*-gram level. Using diacritized transcriptions, it is easy to perform morphological analysis and part of speech tagging of the text using the Buckwalter analyzer. These can be used as additional feature streams in factored language models [7] or maximum-entropy language models. These features have been shown to be extremely beneficial for modeling highly inflected languages such as Arabic [7].

### 5.4 Speech synthesis

Although our emphasis in this paper has been on automatic speech recognition, full diacritization of script Arabic is also extremely useful for text-to-speech synthesis (TTS) applications for that language. In fact, diacritization is even more important for speech synthesis than it is for speech recognition, since it is imperative for TTS that the correct vowel(s) are pronounced – an incorrect vowel may completely alter the meaning of the utterance.

## 6. Conclusions

In this paper, we outlined the diacritization problem for Arabic transcripts for the development of natural language processing tools. We described and implemented different types of generative statistical models at the word and character level in order to re-introduce missing information based on context. These models were evaluated on standard data sets (the Arabic Treebank). Our experiments also confirmed that using part-of-speech improves diacritization performance significantly. Finally,

we used our diacritization system to annotate Arabic transcripts for acoustic and language modeling in order to build a baseline speech recognizer for Arabic broadcast news. We built and tested the recognizer on the TDT4 broadcast news data set, and presented word error rate figures for this evaluation set.

## 7. Acknowledgements

We would like to thank David Schulz of AT&T Research for his useful comments on a draft version of this paper, and also for the guidance he provided us with the Arabic language. We are also thankful to Enrico Bocchieri and S. Parthasarathy, also of AT&T Research, for their support with the acoustic and language modeling tools used in building the ASR.

## 8. References

[1] T. Buckwalter, *Buckwalter Arabic Transliteration*, 2002. http://www.qamus.org/transliteration.htm

[2] K. Kirchhoff et. al., *Novel Speech Recognition Models for Arabic*, Final Report, JHU Summer Research Workshop, Baltimore, MD, 2002.

[3] Y. Gal, *An HMM Approach to Vowel Restoration in Arabic and Hebrew*, Proc. ACL Workshop on Computational Approaches to Semitic Languages, pp. 27-33, Philadelphia, PA, July 2002.

[4] R. Nelken and S. M. Shieber, *Arabic Diacritizaton using Weighted Finite-State Transducers*, Proc. ACL Workshop on Computational Approaches to Semitic Languages, Ann Arbor, MI, June 2005.

[5] Tim Buckwalter, *Buckwalter Morphological Analyzer Version 2.0*, Linguistic Data Consortium, University of Pennsylvania, 2004.

[6] Nizar Habash and Owen Rambow, *Arabic Tokenization, Part-of-Speech Tagging and Morphological Disambiguation in One Fell Swoop*, Proc. ACL Workshop on Computational Approaches to Semitic Languages, Ann Arbor, MI, June 2005.

[7] D. Vergyri, K. Kirchhoff, K. Duh and A. Stolcke, *Morphology-based language modeling for Arabic speech recognition*, Proc. ICSLP, Jeju, South Korea, 2004.