A Journal of Software Engineering and Applications, **2012, 5, 124-127**
doi:10.4236/jsea.2012.512b024 Published Online December 2012 (http://www.scirp.org/journal/jsea)

A HMM-Based System To Diacritize Arabic Text

M S Khorsheed

National Center for Robotics & Intelligent Systems, King Abdulaziz City for Science & Technology (KACST), POB 6086, Riyadh, Saudi Arabia.
Email: mkhorshd@kacst.edu.sa

Received 2012.

ABSTRACT

The Arabic language comes under the category of Semitic languages with an entirely different sentence structure in terms of Natural Language Processing. In such languages, two different words may have identical spelling whereas their pronunciations and meanings are totally different. To remove this ambiguity, special marks are put above or below the spelling characters to determine the correct pronunciation. These marks are called diacritics and the language that uses them is called a diacritized language. This paper presents a system for Arabic language diacritization using Hidden Markov Models (HMMs). The system employs the renowned HMM Tool Kit (HTK). Each single diacritic is represented as a separate model. The concatenation of output models is coupled with the input character sequence to form the fully diacritized text. The performance of the proposed system is assessed using a data corpus that includes more than 24000 sentences.

Keywords: Arabic; Hidden Markov Models; Text-to-speech; Diacritization

1. Introduction

The pronunciation of a word in some languages, like English, is almost always fully determined by its constituting characters. In these languages, the sequence of consonants and vowels determines the correct corresponding voice while pronouncing a word. Such languages are called non-diacritized languages. On the other hand, there are languages, like Arabic, where the pronunciation of their words cannot be fully determined by their spelling characters only.

The Arabic alphabet consists of 28 basic letters, which consists of strokes and dots. Ten of them have one dot, three have two dots, two have three dots. Dots can be above, below or in the middle of the letter as shown in **Figure 1**. The shape of the letter is context sensitive, depending on its location within a word. A letter can have up to four different shapes: isolated (Is), beginning (In) connection from the left, middle (M) connection from the left and right, and end (E) connection from the right, **Figure 2**.

Arabic characters may have diacritics which are written as strokes and can change the pronunciation and the meaning of the word. Diacritics may appear as strokes above the character as Fatha, Dhamma, Sukun, Shadda or Maddah, or below the character as Kasra. Tanween is a form of discretizing Arabic script but with double Fatha, double Dhamma or double Kasra. Tanween occurs only at the end of the word. **Figure 3** shows various diacritics which may appear with Arabic characters.

Diacritics perform an essential function in pronouncing a certain word. To illustrate this with an example: consider the word appears in **Figure 4**, which can be pronounced: (1) college, or (2) kidney. Thus, for an undiacritized Arabic word there may be a vast number of pronunciations whereas for the diacritized Arabic word it is usually a one-to-one table matching process. In spite of this importance, text may be undiacritized and readers of Arabic are accustomed to inferring the meaning from the context, but when it comes to computer processing, the computer still needs to be provided with algorithms to mimic the human ability to identify the proper vocalization

Figure 1. Arabic character samples showing dots above, below and in middle of a character.

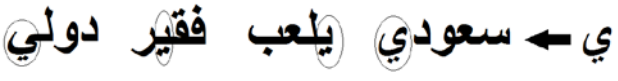

Figure 2. Sample character in four different shapes

of the text. Such a tool is an essential infrastructure for applications as Text-to-Speech, Transliteration and Automatic Translation. The problem of automatic generation of the Arabic diacritic marks is known in the literature under various translations though in this report it is referred to as diacritization.

RDI of Egypt [1] has developed a system called ArabDiac 2.0 based on a proprietary morphological analyzer, as well as syntactical rules and statistical information. The system is claimed to achieve 95% accuracy on the word level. However, the detailed of their technical approach has not been published. Among other systems are Al-Alamia of Kuwait [2] and CIMOS of France [3]. The formal approach to the problem of restoration of the diacrical marks of Arabic text involves a complex integration of the Arabic morphological, syntactic, and semantic rules [4]. A morphological rule matches the undiacritized word to known patterns or templates and recognizes prefixes and suffixes. Syntax applies specific syntactic rules to determine the final diacritical marks by applying Finite State Automata. Semantics help to resolve ambiguous cases and to filter out hypothesis.

2. System Overview

Figure 5 shows the block diagram of the proposed system. The global model is a network of interconnected small models. Each one of those models represents a diacritic. Table 1 illustrates the fourteen possible diacritics besides the no diacritic sign.

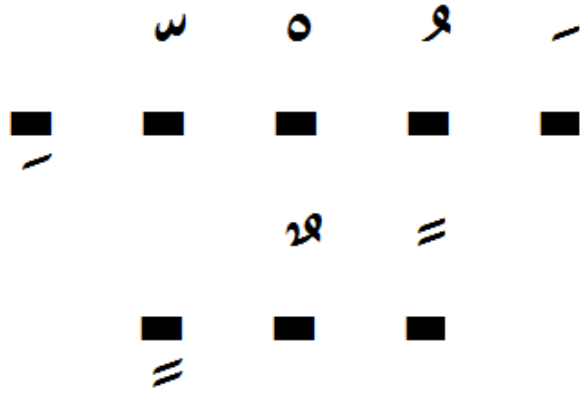

Figure 3. Various diacritics for Arabic characters.

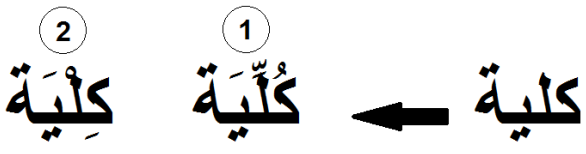

Figure 4. A sample word with two pronunciation.

Table 1. A list of possible diacritics and the corresponding model names.

Diacritic	Model name
Fatha	Mfa
Shadda+Fatha	Msf
Tanween Fatha	Mtf
Shadda+ Tanween Fatha	Stf
Damma	Mda
Shadda+Damma	Msd
Tanween Damma	Mtd
Shadda+ Tanween Damma	Std
Kasra	Mka
Shadda+Kasra	Msk
Tanween Kasra	Mtk
Shadda+ Tanween Kasra	Stk
Madda	Mmd
Sukun	Mso
Nodiacritic	Non

The system may be divided into two stages. The first stage is performed prior to HTK, and includes feature extraction. The objective is to transfer the Arabic text into a sequence of discrete symbols. Each character is coded according to its location within the word; start, middle or end. The feature space includes 110 features. This consists of 84 features for the 28 basic characters, 24 features for the additional characters mentioned earlier, one feature for the space character and another feature to represents the start/end of a sentence.

Stage two is performed within HTK. It couples the extracted features with the corresponding ground truth to estimate the diacritic model parameters. The final output of this stage is a lexicon{free system to diacritize Arabic text. During recognition, an input pattern of discrete symbols representing the text line is injected to the global model which outputs a stream of diacritics that are combined with the input character sequence to form the diacritized text.

3. HMM Tool Kit

The hidden Markov model toolkit (HTK) [5] is a portable toolkit for building and manipulating hidden Markov models. It is primarily designed for building HMM{based speech recognition systems. HTK was originally developed at the Speech Vision and Robotics Group of the Cambridge University Engineering Department (CUED).

Much of the functionality of HTK is built into the li-

brary modules available in C source code. These modules are designed to run with the traditional command line style interface, so it is simple to write scripts to control HTK tools execution. The HTK tools are categorized into four phases: data preparation, training, testing and result analysis tools.

The data preparation tools are designed to obtain the speech data from data bases, CD ROM or record the speech manually. These tools parameterize the speech data and generate the associated speech labels. For the current work, the task of those tools is performed prior to the HTK, as previously explained, then the result is converted to HTK data format.

HTK allows HMMs to be built with any desired topology using simple text files. The training tools adjusts HMM parameters using the prepared training data, representing text lines, coupled with the data transcription. These tools apply the Baum{Welch re-estimation procedure [6] to maximize the likelihood probabilities of the training data given the model.

HTK provides a recognition tool to decode the sequence of observations and output the associated state sequence. The recognition tool requires a network to describe the transition probabilities from one model to another. The dictionary and language model can be input to the tool to help the recognizer to output the correct state sequence.

The result analysis tool evaluates the performance of the recognition system by matching the recognizer output data with the original reference transcription. This comparison is performed using dynamic programming to align the two transcriptions, the output and the ground truth, and then count the number of: substitution (S) and deletion (D).

The optimal string match calculates a score for matching the output sample with respect to the reference line. The procedure works such that identical labels match with score 0, a substitution carries a score of 10 and a label deletion carries a score of 7. The optimal string match is the label alignment which has the lowest possible score. Once the optimal alignment has been found, the correction rate (CR) is then:

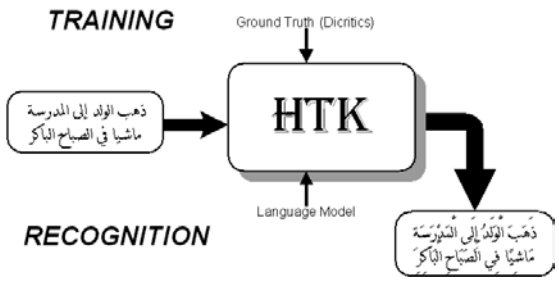

Figure 5. The HTK based system.

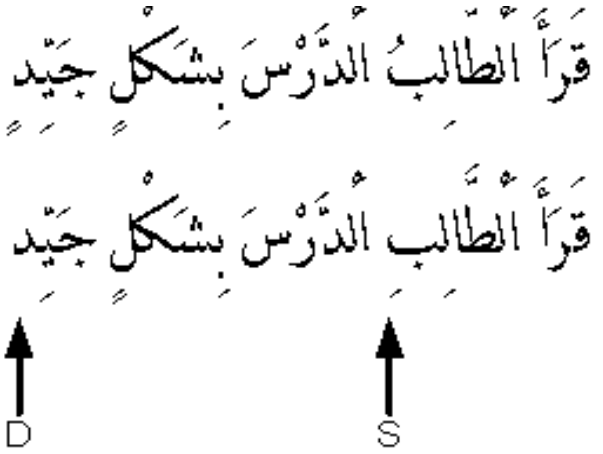

Figure 6. Optimal string matching: the upper line shows the reference text where the lower line is an output sample.

Table 2. CR using two different training sets .

	1000 sentences	2000 sentences
Test set Training set	72.76%	72.80%
Test set Training set	72.50%	72.67%

$$CR = \frac{H}{N} \times 100\% \quad (1)$$

where N is the total number of labels in the recognizer output sequence and H is the number of correct (hit) labels

$$H = N - D - S \quad (2)$$

Accordingly,

$$CR = \frac{N - D - S}{N} \times 100\% \quad (3)$$

Figure 6 illustrates an Arabic reference line and a possible system output. The sentence includes 25 labels. The system output includes one substitution and one deletion. The correction rate equals 92%.

4. Recognition Results

To assess the performance of the proposed system , we built a data corpus which includes more than 24000 Arabic sentences with more than 200000 words. The corpus contains Arabic text that covers different subjects to avoid biasing the likelihood probability of the language model. The global model consists of 15 models stated in Table 1.

The first set of experiments was performed with a training data set of 10000 sentences. The test data set was either similar to or different from the training data set and it ranges from 3000 to 9000 sentences. In both

cases CR ranges between 72:29% and 72:79%. The same experiments were performed again but this time with 20000 sentences as a training data set and a test data set that ranges between 4000 and 16000 sentences. CR scored almost the same result with an average value 72:83%. Table 2 briefly illustrates the average correction rates of those experiments. This concludes that 10000 sentences are quite enough to train the global model.

5. Acknowledgement

This research project is completely funded by King Abdulaziz City for Science & Technology.

REFERENCES

[1] RDI, "ArabDiac" *http://www.rdi-eg.com/rdi/Research*.

[2] SAKHR, *http://www.sakhr.com/*.

[3] CIMOS, *http://www.cimos.com/*.

[4] A. Farghaly and J. Snellart, "Intuitive coding of the Arabic lexicon," Louisiana-USA, 23 September 2003.

[5] S. Young, G. Evermann, D. Kershaw, G. Moore, J. Odell, D. Ollason, V. Valtchev, and P. Woodland, "The HTK Book," Cambridge University Engineering Dept., 2001.

[6] L. Rabiner and B. Juang, "Fundamentals Of Speech Recognition," Prentice Hall, 1993.

