MACHINE GENERATION OF ARABIC DIACRITICAL MARKS

Moustafa Elshafei
King Fahd Univiversity of Petroleum and Minerals, KSA

Husni Al-Muhtaseb
King Fahd Univiversity of Petroleum and Minerals, KSA

Mansour Alghamdi
King Abdulaziz City of Science and Technology, KSA

ABSTRACT

The absence of the vowelization marks from the modern Arabic text represents a major obstacle in machine translation and other text understanding applications. In this paper we present a formulation of the problem of automatic generation of the Arabic diacritical marks from unvoweled text using a Hidden Markov Model (HMM) approach. The model considers the word sequence of unvoweled Arabic text as an observation sequence, and the possible diacritized expressions of the words as hidden states. The optimal sequence of diacritized words (or states) is then obtained efficiently using a dynamic programming algorithm. We present the basic algorithm and its evaluation, and discuss its limitations as well as various ramifications for improving its performance.

Key Words: Arabic natural language, Text diacritization, Arabic HMI, Arabic text-to-Speech, machine translation.

1. INTRODUCTION

One of the problems facing computer processing of Arabic text is the absence of the diacritical marks in the modern printed text. Native Arabic readers can identify the proper vocalization of the text, 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 of the text. Such tool is an essential infrastructure for many applications as Text-to-Speech [1,2], and Automatic Translation [3,4].

Arabic writing system consists of 36 letter forms which represent the Arabic consonants. These are: ا, آ, أ, إ, ئ, ؤ, ء, ى, ة, ب, ت, ث, ج, ح, خ, د, ذ, ر, ز, س, ش, ص, ض, ط, ظ, ع, غ, ف, ق, ك, ل, م, ن, هـ, و and ي. Each Arabic letter represents a single consonant with some exceptions: أ, إ, ئ, ؤ and ء represent the glottal stop, but are written in different forms depending on the consonant position in the word and its adjacent phonemes. Almost all modern Arabic texts are written using the consonant symbols only, i. , the letters without the vowel symbols or the diacritical marks. Arabic diacritical marks are the vowelization marks {Sukoon ـْ , Fatha ـَ , Kasra ـِ , Dhamma ـُ }, the gemination mark {Shaddah ـّ }, and the suffixes { tan ـً , ten ـٍ , ton ـٌ }. The gemination diacritic is followed by a vowel diacritic (except Sukoon), or by a suffix diacritic. A word such as "علم" when diacritized can be: "عَلَم" flag, "عِلْم" science, "عُلِمَ" it was known, "عَلِمَ" he knew, "عَلَّمَ" he taught or "عُلِّمَ" he was taught. Arabic readers infer the appropriate diacritics based on the linguistic knowledge and the context.

The problem of automatic generation of the Arabic diacritic marks is known in the literature under various translations, e.g., automatic vocalization, vowelization, diacritization, accent restoration, and vowel restoration. The formal approach to the problem of restoration of the diacritical marks of Arabic text involves a complex integration of the Arabic morphological, syntactic, and semantic rules [5,6,7]. For example, Vergyri and Kirchhoff, [6] reported a word error rate of 27.3% and a character error rate of 11.54%, using acoustic + morphological + contextual methods. A morphological rule matches the undiacritized word to known patterns or templates and recognizes prefixes and suffixes [8]. Syntax applies specific syntactic rules to determine the final diacritical marks by applying Finite State Automata [9]. Semantics help to resolve ambiguous cases and to filter out hypothesis [10,11].

The approach of this paper falls under the general class of statistical methods in pattern recognition, and has been applied successfully in speech recognition field. Our argument here is that Arabic natives rely mainly on the human pattern matching power to select the right vowelization of words based mainly on its context. The word sequence of unvoweled Arabic text is considered an observation sequence from a Hidden Markov Model, where the hidden states are the possible diacritized expressions of the words. The optimal sequence of diacritized words (or states) are then selected to maximize the probability of the state sequence given the observation sequence. The HMM approach was also proposed by Gal in [12] for vowel restoration of the diacritical marks in Arabic and Hebrew. The text corpus was Qur'an, for which he reported a word accuracy of 86 % .

The Qur'an diacritization style and symbols differ in many aspect from the modern Arabic. The qur'an script contains diacritical marks which are particularly intended for recitation purpose. Our study is based on a corpus supplied by King Abdulaziz City of Science and Technology (KACST), SA, and was manually diacritized by professionals. The corpus is currently being expanded to include at least 50,000 Arabic sentences. The objective of this study is to provide the mathematical formulation of the HMM approach, and to evaluate it on a modern Arabic text. In this study we did not consider the generation of end case. Once the diacriticized text is generated, the generation of the end case can be performed by a separate post processing stage [9]. The HMM method achieves WER less than 0.5% when tested on sentences from the corpus, and WER of about 5.5% when tested on sentences from outside the corpus.

In Section 2, we present the formulation of the problem, while in Section 3 we outline the basic algorithm. In Section 4 we describe the training set and its processing, and in Section 5 we present detailed evaluation of the results and various modification to eliminate certain classes of restoration errors.

2. PROBLEM FORMULATION

We assume we have a large training set of Arabic text with full diacritical marks, $\mathbf{T_V}$, and its corresponding unvowelized text, $\mathbf{T_U}$. We then generate a word vocabulary list, $\mathbf{L_V} = \{v_i\}_1^{N_v}$, of the unique and fully vowelized words in $\mathbf{T_V}$. We also generate a table, $\mathbf{f_V}$, of the frequency of occurrence of each word in $\mathbf{L_V}$, such that $\mathbf{f_V}(k)$ is the number of occurrence of v_k in the training text $\mathbf{T_V}$. Similarly, we construct $\mathbf{L_U}$ of all unvowelized vocabulary words in $\mathbf{T_U}$. Let $\Gamma(.) : \mathbf{L_V} \rightarrow \mathbf{L_U}$ be the mapping from $\mathbf{L_V}$ to $\mathbf{L_U}$; For each word $u_k \in \mathbf{L_U}$ we define a subset $V_k \subset \mathbf{L_V}$ corresponding to all the vowelized words that are mapped to u_k , i.e. $V_k = \{v \in \mathbf{L}_V ; \Gamma(v) = u_k\}$.

Now, given a word sequence (without diacritical marks)

$$W = w_1 w_2 ....w_M ; w_t \in \mathbf{L_U} ; t = 1,2,..,M \tag{1}$$

We wish to determine the most probable diacritized word sequence:

$$D = d_1 d_2 ....d_M \tag{2}$$

Where $d_t = v_j = L_V(j)$ for some $j \in [1, N_v];$ for t=1,2,..M We also assume $w_t = u_k = L_u(k)$ for some $k \in [1, N_u];$ and for $t = 1,2,...M$, that is to say that all the words in (1) exist in $\mathbf{L_U}$.

The word sequence *D* may be chosen to maximize the posteriori probability $P(D \mid W)$, i.e. the best diacritized word sequence, $\hat{D}$, satisfies

$$\hat{D} = \arg\{\overset{\max}{D} P(D/W)\} \tag{3}$$

It is normally assumed in language modeling that the sequence of words obey the Markovian assumption, that is at any given *t*, w_t depends only on the previously words. In Bigram language modeling, each word is assumed to depend only on its previous word in a first order Markov chain, i.e. The conditional probability $P(D \mid W)$ can be written as

$$P(d_1 d_2 .....d_m \mid w_1 w_2 ......w_m) = P(d_1 \mid w_1) \prod_{t=2}^{m} P(d_t \mid d_{t-1}; w_{t-1} w_t) \tag{4}$$

The search for the best sequence of vowelized words which maximizes (3) considers the unvowelized text is generated from an HMM, where the observation sequence is the undiacritized word sequence *W*, while the possible vowelized words, $v_{t,j}$, of each word w_t represent the hidden states. The problem can then be formulated as finding the best state sequence given the observation *W*. The solution of this problem is usually approximated using the Viterbi Algorithm VA [13].

Let us define $\phi(t,i)$ to be the probability of the most likely partial state sequence or path until time *t*, and ending at the i^{th} state (the i^{th} diacritized word corresponding to w_t.).
The algorithm proceeds in the following steps:

Step 1: Initialization

$$\phi(1,i) = P(v_{1,i} \mid w_1) \tag{5}$$

Step 2: Induction

Let k_t be the index of the word w_t in $\mathbf{L_U}$, i.e. $w_t = \mathbf{L_U}(k_t)$; and let $n_v(t)$ be the cardinal of the subset V_{k_t}, then $\phi(t,i)$ can be recursively obtained as follows

$$\phi(t,i) = \max_j \{\phi(t-1,j)P(v_{t,i} \mid v_{t-1,j}; w_{t-1}w_t)\}, \qquad j = 1,2,...,n_v(t); \text{and } t = 2,3,..M \tag{6}$$

$$U(t,i) = \arg\{\max_j \{\phi(t-1,j)P(v_{t,i} \mid v_{t-1,j}; w_{t-1}w_t)\}\}, \qquad j = 1,2,...,n_v(t); \text{and } t = 2,3,..M \tag{7}$$

Step 3: Best Path

$$U(M, i_{best}) = \arg\{\max_j \{\phi(M,j)\} \quad j = 1,2,...,n_v(M)\}\} \tag{8}$$

Step 4: Back Tracking

$$\begin{aligned} i_M &= i_{best} \\ d_t &= v_{t,i_t}, \text{ and } i_{t-1} = U(t,i_t); \text{ for } t = M, M-1,......2 \\ d_1 &= v_{1,i_1} \\ D &= d_1 d_2 ....d_M \end{aligned} \tag{9}$$

Several observations can help in simplifying the Viterbi recursions. First, we notice that the conditional probability appearing in (6) can be written as:

$$P(v_{t,i} \mid v_{t-1,j}; w_{t-1}w_t) = \frac{P(v_{t,i}v_{t-1,j} \mid w_{t-1}w_t)}{P(v_{t-1,j} \mid w_{t-1}w_t)} = \frac{P(v_{t,i}v_{t-1,j} \mid w_{t-1}w_t)}{P(v_{t-1,j} \mid w_{t-1})} \tag{10}$$

Since each vowelized word is mapped to one unvowelized base, this implies that $P(w_t \mid v_{t,j}) = 1$. We can then simplify further Equation (10) as follows:

$$P(v_{t,i} \mid v_{t-1,j}; w_{t-1}w_t) = \frac{P(v_{t,i} \mid v_{t-1,j})}{P(w_t \mid w_{t-1})} \tag{11}$$

Moreover, it is clear then the denominator of (11) is not part of the maximization in (6). This leads to a further simplification of the recursion (6) as follows:

$$\phi(t,i) = \max_j \{\phi(t-1,j)P(v_{t,i} \mid v_{t-1,j}\}, \qquad j = 1,2,...,n_v(t); \text{and } t = 2,3,..M \tag{12}$$

In other words, the evaluation of the recursion, (6) and (7), can be performed using the conditional probability of the vowelized words only. All the probabilities needed for computing the Viterbi recursion can be obtained from the statistics of training sets, and stored for on line Viterbi calculations.

3. THE TRAINING DATABASE

For testing purpose we started by a fully diacritized Arabic corpus. The corpus consists of 100 articles collected from magazines and news papers covering various subjects. The text was annually diacritized by Arabic language specialists. The corpus was developed by king Abdulaziz City of Science and technology, and is currently being expanded to include at least 50,000 sentences.

The total number of words in the corpus came to 102k words. The raw training transcript is first processed to remove numbers and special symbols. The Arabic letter extension character is also removed. All punctuation marks were replaced by one symbol. The transcript was manually checked to correct partially vowelized words, and a few inconsistencies in the diacritization styles. The net training data consists of about 804K characters. The vowelized vocabulary list consists of about 29,500 words. Tables of the frequencies of the vowelized and uvowelized vocabulary are also generated. A word is defined here to be any sequence of letters and diacritical marks delimited by the space character or a punctuation mark. The maximum number of voweled words for any undiacritized word was found to be 6.

Although several cycles of inspection and checking has been performed on the corpus, but the corpus is still far from the flawless state.
Two words $A = a_1 a_2 ... a_{m_a}$ and $B = b_1 b_2 .... b_{m_b}$ are considered identical in the regular sense if $R(A,B) = 1$, where $R(A,B)$ is defined as follows:

$$R(A,B) = \begin{cases} 1 \text{ if } m_a = m_b \text{ and } a_i = b_i \text{ for } i = 1,2,..., m_a \\ \quad 0 \quad \text{otherwise} \end{cases}$$

One problem with the above definition of word similarity is that the S*ukoon* diacritical mark does not consistently appear explicitly in the text. A metric is designed so that two words are still considered identical even if one of them is missing one or more S*ukoon*. Define the mapping $S(.) : \mathbf{L_V} \rightarrow \mathbf{L_V}$ which strips words from S*ukoon* diacritical marks. Let $S(A) = A^0$, and $S(B) = B^0$. Then two words *A* and *B* are said to be R^0 identical, $R^0(A,B)=1$, if $R(A^0,B^0)=1$. When generating $\mathbf{L_V}$, all words in $\mathbf{T_V}$ which are R^0 identical are represented by a single word in the vocabulary $\mathbf{L_V}$. The vocabulary word is selected or generated to be the one with all its S*ukoon* cases removed for efficient memory utilization. Finally, each word in the table $\mathbf{L_V}$ is mapped to its undiacritized base $\mathbf{L_U}$. A database is generated which contains the undiacritized word bases, lists of the corresponding diacritized words, and the counts of each diacritized word. A Matlab program is built to automate building of this database. This database is called "uvwowled words database". UVWDB. Next, a database is generated for each unique sequence of two words in the list file $\mathbf{T_V}$ together with its number of occurrences. In this case, two two-word sequences are considered identical if their corresponding words are R^0 identical. The database generated is called Bigram Database BIGDB. The BIGDB is constructed in such a way that for every voweled word in $\mathbf{L_V}$ it lists all its bigrams and their corresponding frequencies.

Another problem in creating the training databases is that there are many articles, and common short words are not fully diacritized. The missing of diacritical marks represents a serious problem. At the minimum it unnecessarily increases the size of the vocabulary, and creates ambiguity as it would not be clear if the missing diacritics is simple Sukoon or not. The problem is partially alleviated by creating a lexicon of exception cases of common words and articles which usually appear partially or totally undiacritized.

4. EXPERIMENTAL RESULTS

The method was first tested using randomly selected sentences from the training corpus. The word error rate came to less than 0.5%. In fact most of the errors were due to errors in the corpus itself, e.g. extraneous diacritics or missing diacritics. Next the algorithm was evaluated using a testing text from outside the training corpus. The algorithm in its current form can only generate diacritic marks if all the unvoweled words exist in its UVWDB. Accordingly, in the testing text, sentences containing unlisted words are excluded from the test. However, we are currently investigating two approaches to solve his problem. In the first approach we synthesize a word-level diacritization based on letter/diacritics statistics using again an HMM technique. In the second approach, the unknown unvowelized word is matched to a one of a given set of morphological patterns. Since many word sequences in the test set did not occur in the training corpus, we employed a simple bigram smoothing method together with deleted interpolation smoothing [13]. The parameters of the deleted interpolation smoothing were determined experimentally.

The word error rate came to about 5.5%. Analysis of the errors reveals the following classes of errors. The first class of errors turned out to be due to inconsistent representation of diacritical marks in the training data bases, missing diacritical marks, or extraneous diacritical marks. The majority of the repeated errors are caused mainly by a few articles and short words. The ambiguity in determining the proper form of these short words could hopefully be resolved by using higher order grams, and restricting some articles to a single diacritized form. The rest of the cases are more difficult to resolve and may require higher order grams or post processing stage of the resulting diacritized text using knowledge-based morph-syntax word correction.

The algorithm presented in this paper assumes as well that the input word sequence is totally undiacritized. However, in reality, the input text may contain partial diacritization. The algorithm needs to be modified to take into consideration the presence of partial diacritization to improve the efficiency of the algorithm and enhance its performance. Finally, the algorithm generates diacritized text with undetermined end case. The formal approach to resolve these end cases is to implement a syntax analyzer [9]. The syntax processing can be inserted as a post processing stage after the Viterbi algorithm.

5. CONCLUSION

The paper presents an HMM based method to solve the problem of generating the diacritical marks of the Arabic text. The basic form of the algorithm achieves a word error rate of about 5.5%. The use of higher order grams for frequent words with multiple voweled versions could lead to a substantial improvement in the performance. The algorithm needs as well a preprocessing stage to synthesize voweled forms for the unlisted words, and a post processing stage to generate the end cases.

6. ACKNOWLEDMENT

The authors would like to acknowledge King Abdulaziz City for Science and Technology and King Fahd University of Petroleum and Minerals for their support

7. REFERENCES

1) Moustafa Elshafei, Husni Al-Muhtaseb, Mansour Al-Ghamdi,"Techniques for High Quality Arabic Speech Synthesis", Information Sciences 140(3-4): 255-267 (2002).

2) M. Elshafei Ahmed, "Toward an Arabic Text-to-Speech System", in the special issue on Arabization, the Arabian Journal of Science and Engineering, Vol. 16, No. 4B, pp.565-583, October 1991.

3) Trost, Harald, "Recognition and Generation of Word Forms for Natural Language Understanding Systems. Integrating Two-Level Morphology and Feature Unification", Applied Artificial Intelligence, v 5, n 4, Oct-Dec, 1991, p 411-457.

4) Farghaly, A. and J. Snellart. "Intuitive Coding of the Arabic Lexicon", SYSTRAN, MT, Summit IX Workshop, Machine Translation for Semitic Languages: Issues and Approaches, Tuesday September 23, 2003, New Orleans, Louisiana, U.S.A.

5) El-Saadani, T. A. Hashish, M. A, "Semiautomatic Vowelization of Arabic Verbs ", Egyptian Computer... p88-93, Apr 1988.

6) D. Vergyri and K. Kirchhoff, "Automatic Diacritization of Arabic for Acoustic Modeling in Speech Recognition," in COLING Workshop on Arabic-script Based Languages, (Geneva, Switzerland), August 2004.

7) F. Debili and H. Achour "Voyellation Automatique de l'Arabe", Proc. of the workshop on Computation approaches to Semitic languages, COLING-ACL '98.

8) Beesley, K. 1998. "Arabic Finite-State Morphological Analysis and Generation", in COLING-96 Proceedings 1 : 89-94, Copenhagen.

9) Shatta, Usama, "A Systemic Functional Syntax Analyzer and Case-Marker Generator for Speech Acts in Arabic", International Conference for Statistics, Computer Science, Scientific & Social Applications. 19th. Cairo (EGY). Apr 9-14, 1994

10) Khoja, Shereen. APT: Arabic Part-of-Speech Tagger. Proceedings of the Student Workshop at the Second Meeting of the North American Chapter of the Association for Computational Linguistics (NAACL2001), Carnegie Mellon University, Pittsburgh, Pennsylvania. June 2001..

11) El-Barhamttoushi, H. M. Arabic Speech Lexical Engine, International Conference on Intelligent Computing & Information Systems. 1st. Cairo (EGY). Jun 24-26, 2002.

12) Gal, Ya'akov, "An HMM Approach to Vowel Restoration in Arabic and Hebrew", Proceedings of the Workshop on Computational Approaches to Semitic Languages, ACL, July 2002, Philadelphia.

13) X. Huang, A. Acero, and H. Hon, Spoken Language Processing, Prentice Hall PTR, New Jersey, 2001.