HAL
open science

Restoration of Arabic Diacritics Using a Multilevel Statistical Model

Mohamed Seghir Hadj Ameur, Youcef Moulahoum, Ahmed Guessoum

To cite this version:

Mohamed Seghir Hadj Ameur, Youcef Moulahoum, Ahmed Guessoum. Restoration of Arabic Diacritics Using a Multilevel Statistical Model. 5th International Conference on Computer Science and Its Applications (CIIA), May 2015, Saida, Algeria. pp.181-192, 10.1007/978-3-319-19578-0_15 . hal-01789976

HAL Id: hal-01789976

https://inria.hal.science/hal-01789976v1

Submitted on 11 May 2018

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Distributed under a Creative Commons Attribution 4.0 International License

Restoration of Arabic Diacritics Using a Multilevel Statistical Model

Mohamed seghir Hadj Ameur, Youcef Moulahoum, and Ahmed Guessoum

NLP, Machine Learning and Applications (TALAA) Group
Laboratory for Research in Artificial Intelligence(LRIA)
Department of Computer Science, University of Science and Technology Houari Boumediene (USTHB)
Bab-Ezzouar, Algiers, Algeria
{mohamedhadjameur,moulahoum.youcef}@gmail.com, aguessoum@usthb.dz

Abstract. Arabic texts are generally written without diacritics. This is the case for instance in newspapers, contemporary books, etc., which makes automatic processing of Arabic texts more difficult. When diacritical signs are present, Arabic script provides more information about the meanings of words and their pronunciation. Vocalization of Arabic texts is a complex task which may involve morphological, syntactic and semantic text processing.
In this paper, we present a new approach to restore Arabic diacritics using a statistical language model and dynamic programming. Our system is based on two models: a bi-gram-based model which is first used for vocalization and a 4-gram character-based model which is then used to handle the words that remain non vocalized (OOV words). Moreover, smoothing methods are used in order to handle the problem of unseen words. The optimal vocalized word sequence is selected using the Viterbi algorithm from Dynamic Programming.
Our approach represents an important contribution to the improvement of the performance of automatic Arabic vocalization. We have compared our results with some of the most efficient up-to-date vocalization systems; the experimental results show the high quality of our approach.

Keywords

Statistical Language Model, Arabic language, Hidden Markov Model, Automatic Vocalization, Dynamic Programming, Smoothing, Corpus, Viterbi Algorithm.

Introduction

Arabic texts are generally written without diacritical signs (newspapers, books, etc.) this does not pose a problem for people who have a certain mastery of Arabic since they can easily infer the diacritical signs from the context of the words. However, this can be problematical for non-native Arabic speakers. As a matter of fact, the absence of diacritical signs in words also makes their automatic

processing more difficult. Indeed, when diacritics are present, the Arabic script provides more information about words meanings and their pronunciations. As such, Arabic vocalization is used in order to increase the performance of many applications such as Arabic text-to-speech (TTS) [1,10] and speech recognition [16].

Arabic diacritics restoration (text vocalization) is the process of assigning Arabic diacritics such as fatha ("a" sound as in "apple"), damma ("oo" sound as in "book") and kasra ("i" sound as in "in") to a given text (or script). Arabic diacritical signs are represented in Table 1.

Diacritic	Example	Pronunciation
Fatha	ذهبَ	/t//a/
Damma	الطفلُ	/t//u/
Kasra	البيتِ	/t//i/
Tanween Damma	كِتابٌ	/t//un/
Tanween Kasra	كِتابٍ	/t//in/
Tanween Fatha	كِتابًا	/t//an/
Sukuun	الوقْتُ	/t/
Shadda	مُدَرِّسَة	/t//t/

Table 1. Arabic diacritical signs.

During the last few years, the statistical approaches have been proven to be more efficient in the tackling of different problems of natural language processing. For the vocalization problem more specifically, most of the recent work was based on statistical approaches which can be either purely statistical ones or hybrid methods that combine a statistical language model and some other treatments.

Hybrid methods, such as [7] which uses a morphological tagger or [4] which uses *AlKhalil Morpho Sys* [4], depend on the effectiveness (accuracy) of these morphological analysers and taggers. Purely statistical methods however do not have such a dependence. Recent works based on purely statistical methods have reported very interesting results. This is the case for [9] which uses only a word-based bigram language model and the work of [2] which uses a character-based 4-gram model.

In this paper, we aim to further improve the previous statistical Arabic text vocalization approaches used in [9] and [2] by proposing a new simple but efficient system that relies on a purely statistical language model coupled with dynamic programming which combines these two approaches; thus our vocalization system is based on two models: the first one is a bi-gram word-based model [9] which is first used for vocalization and the second one is a 4-gram character-based model [2] which is used to handle the words that remain non-vocalized (OOV words). Smoothing methods are used in order to handle the problem of unseen words; the optimal vocalized word sequence is selected using the Viterbi algorithm [12].

This paper is organized as follows: Section 2 gives an overview of the state of the art vocalization systems. Section 3 explains our approach to restoring Arabic diacritics using a statistical language model and dynamic programming. Section 4 presents our tests and experimental results. A conclusion of our work is given in Section 5.

Related Work

Vocalization approaches can be divided into two main categories: Rule-based and Statistical Approaches. During the last decade, the statistical approaches have widely been used in a variety of natural language processing applications which have proven their efficiency. For the vocalization problem, most of the recent work was based on statistical approaches. These statistical approaches can be classified into two categories: purely statistical methods, or hybrid methods that combine a statistical language model and some other treatments.

In terms of purely statistical methods, one may cite [6] where the authors presented a vocalization approach based on Hidden Markov Models (HMMs). The hidden states correspond to the vocalized words and each one of them has a single emission leading to a non vocalized word (an observed state). In [2], a similar approach was used but with a character-based 4-gram model (a sequence of 4 consecutive vocalized letters) instead of a word-based model. The most recent work based on purely statistical methods is [9] where its authors used a statistical bigram language model coupled with dynamic programming to choose the most likely sequence of diacrtics. They improved their own work in [8] by using a higher order n-gram statistical language model.

For the methods which use a hybrid approach, we can mention [7] whose authors developed a hybrid system which combines a statistical n-gram language model (where n equals 1, 2 or 3) combined with a morphological tagger. In a similar way, in [3] a statistical n-gram language model is also used along with morphological analysis using *AlKhalil Morpho Sys* [4]. In [15], an approach was proposed which combines lexical retrieval, bigram-based and SVM-statistical prioritized techniques. In [14], the authors proposed two methods: the first uses an n-gram statistical language model along with A^* lattice search while the second method attempts to segment each Arabic word into all its possible morphological constituents then proceed in a similar way as the first one. The authors reported that their second approach gives better results. Finally, in [17], a statistical classifier was proposed which is based on the maximum entropy principle, which uses the combination of a wide array of lexical, segment-based and part-of-speech tag features in order to select the best classification.

It turns out that Arabic text vocalization is not yet optimal as will be shown in Section 4.4. No system is currently good enough to restore diacritics with high enough a quality as to be able to build solid applications on it. For this reason, we have decided to dig deeper into this problem. This has led us to building a system which has given very encouraging results as will be shown in the sequel.

Arabic Text Vocalization Approach

In this section we will formally introduce the problem of Arabic text vocalization and present the different models generated in our system.

Formalizing the problem

Vocalization of Arabic text (or Restoration of Arabic Diacritics) is the process of assigning diacritical signs to each word in a given text or script.
This problem can be formalized as follows: given a sequence of non-vocalized words (or script) $W = w_1, w_2, ..., w_n$, the vocalization task is to find the best sequence of vocalized words $V = v_1, v_2, ..., v_n$ from all the possible vocalization sequences of W.

Assigning a score to each possible vocalized word sequence can be used to select the best vocalization from all the possible ones. This score can be calculated using the Chain rule:

$$P(W) = \Pi_{k=1}^{n} P(W_k | W_1^{k-1}) \tag{1}$$

By making the independence assumption (Markov assumption) for the n-grams model, instead of using the whole history (chain-rule) the n-gram model can approximate the history of a given word using just the last k words. The probability will thus be estimated as follows:

$$P(W_n | W_1^{n-1}) = P(W_n | W_{k-(n-1)}^{n-1}) \tag{2}$$

Using the Markov assumption in the case of a bi-gram language model, we will have:

$$P(W_n | W_1^{n-1}) = P(W_n | W_{n-1}) \tag{3}$$

$P(W_n | W_{n-1})$ is computed using the Maximum Likelihood Estimation (MLE):

$$P(W_n | W_{n-1}) = \frac{C(W_{i-1}, W_i)}{C(W_{i-1})} \tag{4}$$

where $C(W_{i-1}, W_i)$ and $C(W_{i-1})$ are the counts of the bi-gram $W_{i-1}W_i$ and the uni-gram W_{i-1} respectively.

Presentation of the vocalization system

This section presents the global structure of our vocalization system. The automatic vocalization of Arabic texts consists of two main phases: vocalization using a bi-gram word-based model followed, for the unresolved cases, by vocalization using a 4-gram character-based model. This is illustrated in Figure 1 and explained in more details in the following two subsections.

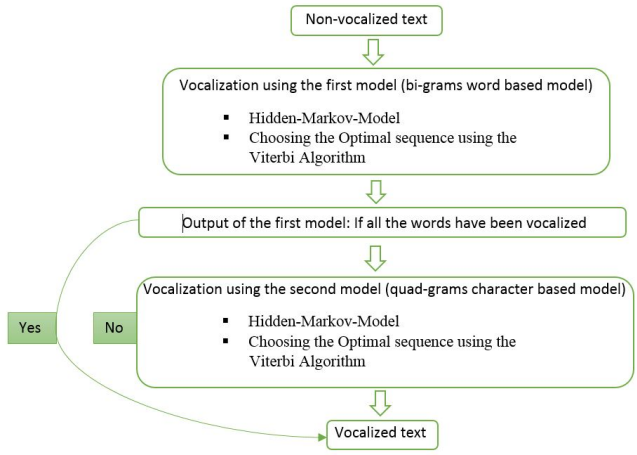

Fig. 1. The Vocalization System

We should point out that we have decided for our word-based model to restrict ourselves to bi-grams for computational efficiency reasons. Going for higher-order n-Gram models would indeed be costly in execution time in an application which should be as fast as possible to be integrate into larger, possibly online, applications. As to the 4-gram character-based model, this is due to the fact that we have analyzed that 4 letters are can be quite rich a background to allow for reasonably good diacritization, especially that this second model is used as a complement to the word-based one.

The first model

Our first model is a bi-gram-based language model. To illustrate it, let us consider the following non-vocalized sentence:

خلق الإنسان علمه البيان

The functioning of the first model can be summarized in the following steps:

The first step is to build a dictionary which associates for each non vocalized word all its possible vocalizations.

In the second step, a lattice is created for the non-vocalized sequence $W = w_1, w_2, ..., w_n$ which gives, for each non-vocalized word w_i, all its possible vocalizations from the dictionary. In order to simplify the explanation, our example considers only a subset of the possible vocalizations of each word (as shown in Figure 2). Given the assumption, the size of the subset of possible vocalizations would be $8 * 4 * 8 * 2 = 512$.

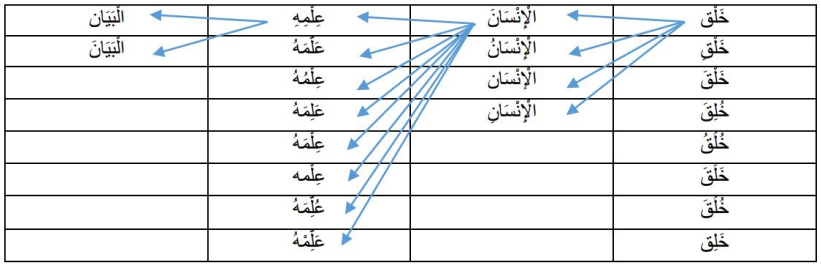

Fig. 2. Some of the possible vocalizations for a non-vocalized sentence.

The third step consists in associating to each possible vocalization (e.g. in Figure 2, each sentence among the 512 possible ones) a probability using the bi-gram language model:

$$P(W_n|W_1^{n-1}) = P(W_n|W_{n-1}) \tag{5}$$

For example one of the possible vocalizations is:

خَلَقَ الْإِنْسَانَ عَلَّمَهُ الْبَيَانَ

The probability of the above sentence is calculated as follows:

$$P\ (\text{خَلَقَ الْإِنْسَانَ عَلَّمَهُ الْبَيَانَ}) =$$

$$P\ (\text{خَلَقَ}) * P\ (\text{الْإِنْسَانَ} \mid \text{خَلَقَ}) * P\ (\text{عَلَّمَهُ} \mid \text{الْإِنْسَانَ}) * P\ (\text{الْبَيَانَ} \mid \text{عَلَّمَهُ})$$

Similarly, probabilities are assigned to all the possible vocalized word sequences.

The forth and final step is to find among all the possible vocalizations the one that has the highest probability.

$$v_1, \widehat{v_2, ...}, v_n = argmax(\Pi_{k=1}^{n} P(v_k|v_{k-1})) \tag{6}$$

The number of all possible vocalizations is very large as mentioned in Figure 2. Let N be the average number of all the possible vocalizations for each word and L the length of the non vocalized sequence (i.e. the number of words in it). The number of all possible vocalizations will then be N^L. Trying to find the best vocalization by a brute-force approach would have an exponential complexity ($O(N^L)$) and is clearly not efficient. An alternative is to use , the Viterbi algorithm, a Dynamic programming approach. To this end, let us present our Hidden Markov Model (HMM) which will be used as an input to the Viterbi algorithm to get the best vocalization.

Hidden Markov Model Our Hidden Markov Model (HMM) is defined by:

- A set of states which represent the vocalized words $v_1, v_2, ..., v_n$.
- A set of observations which represent the non-vocalized words $w_1, w_2, ..., w_n$.
- The transition matrix which contains the transition probabilities $P(v_i|v_{i-1})$

Generally each sequence in the HMM depends on two probabilities (transitions and emissions). In our model however, only the transition probabilities are considered.

The Viterbi Algorithm The best vocalized sequence is chosen from the HMM using the Viterbi algorithm which is a very efficient algorithm for selecting the best vocalization sequence [12]. The latter uses a recursive relation in which the probability of each node at a given level i is calculated according to its preceding level $i-1$ (see Figure 3).

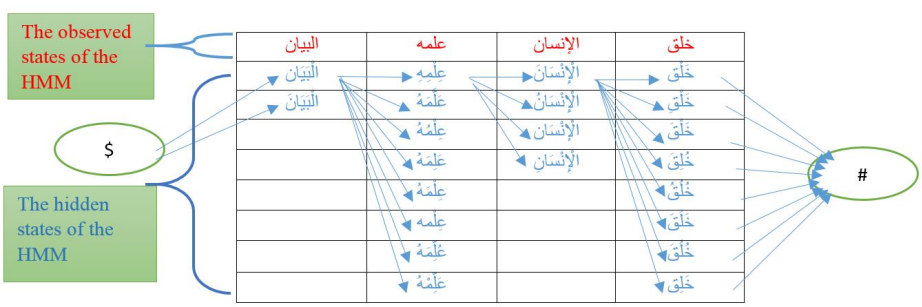

Fig. 3. Finding the optimal vocalized sequence using the Viterbi algorithm

As shown in Figure 3, the weight of each node of index (i, j) of level i is calculated from all the nodes of its preceding level $i-1$, using the following formula:

$$P(i,j) = max_{k=1,v_{i-1}}(P(i,j|i-1,k) * p(i-1,k)) \quad (7)$$

where i and j are the indexes of the line and colon, respectively, in the transition matrix.
v_{i-1} is the number of all possible vocalization for word $i-1$.
After calculating the weights of all nodes (in this forward-moving computation), while keeping track of the best nodes on this path, back-tracing is done in order to find the optimal path.

The Viterbi algorithm allows to efficiently solve the problem of selecting the best vocalization sequence.

In the next section, we introduce the smoothing method we use in order to handle the problem of unseen bigrams.

Handling the problem of unseen bi-grams The maximum likelihood estimation (MLE) is calculated using Equation 4. This equation assigns a null probability to any bi-gram that does not belong to the training corpus. According to Jurafsky and Martin ([11]), almost 99% of all the possible combinations of bi-grams may be missing from any given corpus. This is why the use of smoothing methods are necessary in order to avoid having null probabilities (as a result of the products of probabilities). This problem is handled by taking some of the probability mass from the existing n-grams and distributing it to the non-found n-grams.

Additive Smoothing: One of the simplest smoothing methods is Additive Smoothing [5] which assumes that each of the n-grams occurs one more time than its actual occurrence count. Thus we add one to all the n-grams. This yields in the case of bi-grams:

$$P_{add}(w_i|w_j) = \frac{K + c(w_i, w_j)}{(K * V) + c(w_i)} \tag{8}$$

where K is a constant between 0 and 1 and V is the size of the vocabulary.

Absolute discounting: Absolute discounting [11] is an interpolated smoothing method [13] which is obtained by discounting a constant D between 0 and 1 from each non-null probability mass. This yields in the case of bi-grams:

$$P_{abs}(w_i|w_j) = \frac{max(c(w_i, w_j) - D, 0)}{c(w_i)} + \frac{D}{c(w_i)} N_{1+}({w_i}^*) P_{abs}(w_j) \tag{9}$$

where $P_{abs}(w_j) = \frac{1}{V}$, V is the vocabulary size.
$N_{1+}({w_i}^*)$ is the number of all the words without repetition that follow w_i in the training corpus.

In the next section we explain our second model.

Letter-based Model

The second model is a 4-gram character-based model which is used to handle the words that remain non-vocalized (out of vocabulary words, OOV). The HMM used in this model is now introduced.

Letter-Based Hidden Markov Model Our HMM is defined by states that represent the vocalized letters and observations that represent the non-vocalized letters. It consists of:

- A set of states which represent the vocalized letters $q_1, q_2, ..., q_n$.
- A set of observations which represent the non-vocalized letters $l_1, l_2, ..., l_n$.
- The transition matrix which contains the transitions $P(q_i|q_{i-1}, q_{i-2}, q_{i-3})$.
- The emission matrix which contains $P(l_i|q_i)$.

This model is used in a similar way to the previous model. The same smoothing methods are used and the optimal path is selected using the Viterbi algorithm. This model has the capacity to vocalize any Arabic word; this is why it is used as a final step to ensure the complete vocalization of the non-vocalized script.

Implementation and Tests

In this section, we start by presenting the corpus we have used and some statistics relating to it. We then move to presenting the results of testing our implementation.
The source code of our vocalization system is available at `https://github.com/Ycfx/Arabic-Diacritizer` under the GNU General Public License (*GPL*).

Corpus construction

The largest part of our corpus is automatically retrieved from the site `http://www.al-islam.com/` using a URL-rule-based crawler. This site is an Islamic religious site that contains vocalized text about a number of subjects (Hadith, Commentaries of the Quran, etc.). A vocalized *Holy Quran* was also downloaded from `http://tanzil.net/` and added to the corpus. Each downloaded vocalized text goes through cleaning, tokenisation, and normalisation steps to finally yield a properly vocalized corpus. On the other hand, its non-vocalized version is obtained by simply deleting the diacritical signs from the vocalized corpus.

Corpus statistics

We have created a large Arabic corpus which contains more than 10 million words (tokens) 2. We have used 90% of the total words of the corpus for the training phase and the remaining 10% for the testing phase.

	Corpus
Sentences	799 470
Tokens	10 634 921
Types	379 429

Table 2. Corpus statistics.

Evaluation Measures

To measure the performance of the different vocalization systems we have used the Word Error Rate (*WER*) and the Diacritic Error Rate (*DER*) measures:

- *WER1*: the number of words vocalized wrongly by the system (taking into account the diacritic of the last letter of the word).
- *WER2*: the number of words vocalized wrongly by the system (not taking into account the diacritic of the last letter of the word).

- *DER1*: the number of characters vocalized wrongly by the system (taking into account the diacritic of the last letter).
- *DER2*: the number of characters vocalized wrongly by the system (not taking into account the diacritic of the last letter).

Results

In this section we will start by presenting the results obtained by changing the smoothing parameters, then we give a detailed comparison of our system with the state of the art vocalization systems.

Impact of the smoothing parameters on the vocalization system Table 3 shows the impact of changing the smoothing parameters on the vocalization system. The K and D smoothing parameters were investigated for additive and absolute discounting methods respectively.

Smoothing methods	WER1	WER2	DER1	DER2
Absolute discounting (K=1)	11.85	6.67	4.63	3.49
Absolute discounting (K=0.5)	11.57	6.30	4.34	3.23
Absolute discounting (K=0.1)	**11.53**	**6.28**	**4.30**	**3.18**
Additive smoothing (D=1)	16.87	9.49	8.10	6.86
Additive smoothing (D=0.5)	15.75	9.16	7.85	6.83
Additive smoothing (D=0.1)	15.41	9.05	7.77	6.83

Table 3. The impact of the smoothing parameters on the vocalization system.

The experimental results prove that the use of smoothing methods has a noticeable influence on the overall performance of the vocalization system. The best performance of our system was achieved using absolute discounting with $D = 0.1$, where the results we obtained were 11.53% in terms of Word Error Rate *(WER1)* and 6.28% when the case ending was ignored (*WER2*), and in terms of Diacritic Error Rate the results were 4.30% for *DER1* and 3.18% when ignoring the last diacritical mark (*DER2*). These results show that the Absolute Discounting method gives a better practical performance in comparison to Additive Smoothing.

Comparison of our System with the Different Vocalization Systems In order to evaluate the overall performance of our vocalization system, we have compared its performance to some of the most efficient implementations available today. However since these systems have not been tested on the same corpus, the conclusions should be taken with some caution. The results of the comparison are summarized in Table 4.

Vocalization Systems	WER1	WER2	DER1	DER2
Zitouni et al (2006) [17]	37	15	23	10
Habash et al (2007) [7]	14.9	4.8	5.5	2.2
Shaalan et al (2009) [15]	12.16	**3.78**	-	-
Rashwan et al (2011) [14]	12.5	3.8	**3.1**	**1.2**
Hifny et al 2013) [9]	12.5	7.4	-	-
Bebah et al (2014) [3]	21.11	9.93	7.37	3.75
Our system	**11.53**	6.28	4.30	3.18

Table 4. Comparing the performance of our system to those of some other vocalization systems.

When case ending (the last diacritical sign) is ignored *(WER2,DER2)*, the results were clearly better for all the compared systems, which is explained by the added difficulty when attempting to vocalize the last letter *(WER1,DER1)*. Our system gives the best result in terms of *WER1*, next to best on *DER1*, and its performance on the other measures is very close to the best results reported in the literature. That our system performs extremely well on *WER1* and *DER1* is indeed what we want; it shows that other systems have problems handling the last letter diacritic which is very crucial in Arabic.
Our system performance has thus been proven to be very competitive; It shows the effectiveness of the multilevel approach we have adopted for the vocalization problem.

Conclusion

We have presented in this paper a multilevel statistical vocalization model. Our system is based on two models: the first one is a bi-gram word-based model which is used first for vocalization and the second one is a 4-gram letter-based model which is used as a back-off, i.e. to handle the words that remain non-vocalized after the application of the first model. We have used smoothing methods to handle the problem of unseen words and the Viterbi algorithm to select the optimal path, i.e. best vocalization sequence, in the HMM. The results shows the efficiency of our vocalization system in comparison to other state-of-the-art systems.

Our system can be improved in several ways which we intend to explore:

- Our first model is based on bi-gram probabilities only; the use of n-gram models with $n > 2$ should yield better results.
- We can enrich the corpus by adding many more modern Arabic texts to it.
- In this work, only two smoothing methods have been used; using other smoothing methods could give better results.

References

1. Ahmed, M.E.: Toward an arabic text to speech system. The Arabian Journal of Science and Engineering 16(4B) (1991)

2. Alghamdi, M., Muzaffar, Z.: Kacst arabic diacritizer. In: The First International Symposium on Computers and Arabic Language. pp. 25–28 (2007)
3. Bebah, M., Amine, C., Azzeddine, M., Abdelhak, L.: Hybrid approaches for automatic vowelization of arabic texts. arXiv preprint arXiv:1410.2646 (2014)
4. Boudlal, A., Lakhouaja, A., Mazroui, A., Meziane, A., Bebah, M., Shoul, M.: Alkhalil morpho sys1: A morphosyntactic analysis system for arabic texts. In: International Arab Conference on Information Technology (2010)
5. Chen, S.F., Goodman, J.: An empirical study of smoothing techniques for language modeling. In: Proceedings of the 34th annual meeting on Association for Computational Linguistics. pp. 310–318. Association for Computational Linguistics (1996)
6. Gal, Y.: An hmm approach to vowel restoration in arabic and hebrew. In: Proceedings of the ACL-02 workshop on Computational approaches to semitic languages. pp. 1–7. Association for Computational Linguistics (2002)
7. Habash, N., Rambow, O.: Arabic diacritization through full morphological tagging. In: Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Companion Volume, Short Papers. pp. 53–56. Association for Computational Linguistics (2007)
8. Hifny, Y.: Higher order n gram language models for arabic diacritics restoration. In: Proceedings of the 12th Conference on Language Engineering, Cairo, Egypt (2012)
9. Hifny, Y.: Restoration of arabic diacritics using dynamic programming. In: Computer Engineering & Systems (ICCES), 2013 8th International Conference on. pp. 3–8. IEEE (2013)
10. Hifny, Y., Qurany, S., Hamid, S., Rashwan, M., Atiyya, M., Ragheb, A., Khallaaf, G.: An implementation for arabic text to speech system. In: The proceedings of the 4th Conference on Language Engineering (2004)
11. Jurafsky, D., Martin, J.H.: Speech and language processing. Pearson Education India (2000)
12. Neuhoff, D.L.: The viterbi algorithm as an aid in text recognition. Information Theory, IEEE Transactions on 21(2), 222–226 (1975)
13. Ney, H., Essen, U., Kneser, R.: On structuring probabilistic dependences in stochastic language modelling. Computer Speech & Language 8(1), 1–38 (1994)
14. Rashwan, M.A., Al-Badrashiny, M.A.S., Attia, M., Abdou, S.M., Rafea, A.: A stochastic arabic diacritizer based on a hybrid of factorized and unfactorized textual features. Audio, Speech, and Language Processing, IEEE Transactions on 19(1), 166–175 (2011)
15. Shaalan, K., Abo Bakr, H.M., Ziedan, I.: A hybrid approach for building arabic diacritizer. In: Proceedings of the EACL 2009 Workshop on Computational Approaches to Semitic Languages. pp. 27–35. Association for Computational Linguistics (2009)
16. Vergyri, D., Kirchhoff, K.: Automatic diacritization of arabic for acoustic modeling in speech recognition. In: Proceedings of the workshop on computational approaches to Arabic script-based languages. pp. 66–73. Association for Computational Linguistics (2004)
17. Zitouni, I., Sorensen, J.S., Sarikaya, R.: Maximum entropy based restoration of arabic diacritics. In: Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics. pp. 577–584. Association for Computational Linguistics (2006)