International Journal of Computer Engineering Research Vol. 2(2), pp. 28-33, March 2011
Available online at http://www.academicjournals.org/IJCER
ISSN 2141- 6494 ©2011 Academic Journals

Full Length Research Paper

Hidden markov model based Arabic morphological analyzer

A. F. Alajmi*, E. M. Saad and M. H. Awadalla

Communication and Electronics Department, Faculty of Engineering, Helwan University, Egypt.

Accepted 24 February, 2011

Natural language processing tasks includes summarization, machine translation, question understanding, part of speech tagging, etc. In order to achieve those tasks, a proper language representation must be defined. Roots and stems are considered as representations for some of those systems. A word needs to be processed to extract its root or stem. This paper presents a new technique that extracts word weights, by stripping of prefixes and suffixes from a given word. This technique is based on Hidden Markov Model (HMM). A path from a start state to the end state represents a word, each state constitute letters of a word. States are prefixes, weights, and suffixes. The best selected path should have the highest likelihood of a word. The approach results in a promising 95% performance.

Key words: Natural language processing, morphology, hidden markov model, stem.

INTRODUCTION

The Arabic word is characterized by a well defined letters organization. Words are originated from sections of 3 letters called tri-root, or 4 letters called quad-root, which is the basic block of a word. Furthermore, different forms of words with possibly different meanings are generated from those roots based on well established morphological rules, which are called weights. Thus, by detecting those weights, a word can be reversed into its original root. Over 300 weights represent all forms of an Arabic word, but adding prefixes and suffixes complicate the detection of a word root.

Features are the basis of a text processing system, and in our case, those features are words in a given text. The word by itself does not provide a good representation of a text due to its inflation. Therefore, segmentation of a surface word - word which appears in a text- is a must in order to assure a more efficient text processing system. Thus, further processing of a word is needed to produce better features. One way is to use a stem, which is a result of stripping prefixes, suffixes, and infixes from a word and thus provides better representation. Stem, sometimes referred to as the root, has a drawback of grouping words with possibly different meaning under one root that will affect the accuracy of the outcome of such a system. Another way of presenting a word is the words' weight, which is extracted by stripping prefixes and affixes. The process will minimize the number of features and, preserve the meaning of the word.

In this paper, a new statistical approach is presented based on Hidden Markov Model to extract words' weights and roots. This approach identifies three segments of a given surface word - word in a text. A word is represented by different states. States in the model are divided into three segments. The first segment represents prefixes, the second segment represents the weights that the word belongs to, and the third segment represents the suffixes which a word might be attached to. Word may or may not have a prefix or a suffix. A set of states (path) represents a word, where each letter of the input word is represented by a single state. Furthermore, the extraction of Arabic word weights may lead to word type (noun, verb) detection. Weights may represent nouns, verbs, or both. It will be shown that our approach will detect over 90% of word type, and 95% for weight extraction.

As far as we know, there were no works done on the

*Corresponding author. E-mail: om_mo3ath@yahoo.com, alajmi@ieee.org, alajmi@acm.org. Tel: +96599203181.

extraction of a word weight. Most of the research focuses on the root, and stem detection. Deferent techniques were used to extract roots, or stems. Most are rule based, and few are statistical based. The presented technique is considered as a morphological analyzer, which will serve as a weight extractor, a root and stem extractor, a word type identifier. It can also be used to convert a word into its singular state by weights conversion rules (for example, مسلماتto مسلمة).

The next section presents some of the previous works about other morphological systems developed. Following this, we describe the Hidden Markov Model for weight extraction. Finally, we present the results of our system and conclude with a list of future improvements identified as a result of the evaluation.

PREVIOUS WORK

Various morphological systems were developed in literatures. Almost all the system focused on extracting roots or stems. Morphological systems are categorized as statistical driven methods (Al-Sahmsi and Guessoum, 2006; Mohamed et al., 2009; Ahmed and Nürnberger, 2007; Sinane et al., 2008), machine translation driven methods (Chen and Gey, 2002) and rule based methods (El-Hajar et al., 2010; Larkey et al., 2002, 2005; Buckwalter, 2002; Al-Ameed et al., 2005; Khoja and Garside,1999; Darwish, 2002).

A Hidden Markov Model Based part of speech approach was introduced in the works of Al-Sahmsi and Guessoum (2006). It uses HMM to resolve Arabic text POS (Part of Speech) tagging ambiguity through the use of a statistical language model developed from Arabic corpus. The paper presents the characteristics of the Arabic language and the POS tag set that has been selected. It then introduces the methodology followed to develop the HMM for Arabic. For the POS-tagging problem, observation sequence is a sequence of words. The transition probabilities are obtained from the trigram model and the emission probabilities are obtained from the lexical trigram model. The states of the HMM model are the POS tags. A training corpus of Arabic news articles has first been stemmed using the Buckwalter's stemmer, and then, tagged manually with proposed tag set. Then, a trigram language model was built for the tagged training corpus. The trigram language model computes lexical probabilities. Then, the POS tag sequences was obtained from the training corpus and created a trigram Arabic language model based on the POS tag corpus. Next, lexical and contextual probabilities were used to determine the HMM model's parameters as follows: contextual probabilities were transition probabilities and lexical probabilities were the emission probabilities. Once matrices A and B were computed, search needs to be performed to find the POS tag sequence that maximizes the product of the lexical and contextual probabilities. The proposed HMM POS tagger achieved a performance of 97%.

El Hajar et al. (2010) combine morphological analysis with Hidden Markov Model (HMM) and rely on the Arabic sentence structure to produce Arabic Part-Of-Speech Tagging. The morphological analysis is used to reduce the size of the tags lexicon by segmenting Arabic words in their prefixes, stems, and suffixes due to the fact that Arabic is a derivational language. HMM is used to represent the Arabic sentence structure in a hierarchical manner. Each tag in this system is used to represent a possible state of HMM and the transitions between tags (states) are governed by the syntax of the sentence. A corpus is manually tagged and then used for training and testing this system. Experiments conducted on the data set have given a recognition rate of 96%.

Arabic stemming algorithms can be classified, according to the desired level of analysis (El-Hajar et al., 2010), as either stem-based or root-based algorithms. Stem-based algorithms, remove prefixes and suffixes from Arabic words, while root-based algorithms reduce stems to roots. Light stemming refers to the process of stripping off a small set of prefixes and/or suffixes without trying to deal with infixes.

One light stemmer is Larkey et al. (2002), who used a predefined list of prefixes and suffixes to produce a prefix/stem/suffix form. The maximum number of prefixes it can remove is 3, and the maximum number of letters in a suffix is 2. Thus, it fails to remove prefixes that have more than three letters long and suffixes that have more than two letters long. Larkey et al. (2005) revisited the light stemmers and developed another one called light10 that exploits the possibility of having more prefixes and suffixes in the list.

Another light stemmer introduced in Buckwalter (2002), returns all valid segmentations based on the fact that an Arabic prefix length can go from zero to four letters, and the stem can consist of one or more letters, and the suffix can consist of zero to six letters. It returns stems rather than roots. It is based on a set of lexicons of Arabic stems, prefixes, and suffixes, with truth tables indicating their legal combinations. The three dictionaries list possible prefixes, Arabic stems, and possible suffixes. The three compatibility tables indicate compatible prefix/ stem category pairs, compatible prefix/suffix category pairs, and compatible stem/suffix category pairs.

Al-Ameed et al. (2005) is based on the elimination of the Arabic character "و" if it is the beginning of the word, of specific list of prefixes and the suffixes. This stemmer is not dictionary driven, so it cannot apply a criterion that an affix can be removed only if what remains is an existing Arabic word. The stemmers work blindly on words even if they are not found in a word list. It attempts to remove strings which would be found reliably as affixes far more often than they would be found as the beginning or end of an Arabic stem without affixes. The light stemmers do not remove any string that would be considered Arabic prefixes by itself.

Khoja and Garside (1999) presented a simple morphological analyzer, where layers of prefixes and suffixes are removed, then a list of patterns and roots are checked to determine whether the remainder could be a known root with a known pattern applied. If so, it returns the root. Otherwise, it returns the original word, unmodified. This system also removes terms that are found on a list of 168 Arabic stop words.

Taghva et al. (2005) introduced a stemmer without a root dictionary. It uses a similar approach to extract roots as Khoja's approach, but without using a root dictionary or lexicon, and performs as well as a light stemmer. This method is based on the elimination of several sets of affixes, and on the application of several patterns. This method does not use any dictionary to extract the Arabic root. To implement this algorithm, they have defined several sets of the affixes, D diacritic. P3 P2 P1 prefix of three, two, and one letter. And S3 S2 S1, suffix of three, two, one letter, and several sets of pattern models of four, five and six letters. Furthermore, a three, four, five letters roots Models were defined.

Chen and Gey (2002) developed two Arabic stemmers and an Arabic stop list at TREC 2001. The two researchers created a machine translation (MT) based stemmer and a light stemmer. The stemmer based on translation was relied on the idea of translating the Arabic word to the English, after removing English stop words, then, extract the base word in English, then translate this word in Arabic to the root for example: أطفالنا (our children), remove "our" is a word, أطفالنا is apparent that in relation to "child". So أطفالنا is related to طفل. The light stemmer (Chen and Gey, 2002) was called Berkeley which shares many of prefixes and suffixes that should be removed with the light stemmers developed by Larkey et al. (2002) and the one developed by Darwish (2002). They identified other sets of prefixes and suffixes. They start by counting the words that begin with a given prefix, and the number of words ending with the given suffixes. At the end, the prefixes that must be removed are identified: 19 three-letters, 14 two-letters, and 3 one letter, and the suffixes: 18 two-letters, 4 one letter. To remove the prefixes and suffixes in the predefined sets, each algorithm proposes their own rules.

A statistical method which belongs to the "N-gram" class was developed by Ahmed and Nürnberger (2007) and Sinane et al. (2008). An n-gram is a subsequence of n letters from a given word to predict the next letter in such a sequence. It is based on the concept of words similarity or dissimilarity. Two words are considered similar if they have several common substrings of N letters. Two words are considered dissimilar if they do not have common different substrings of N characters. N-gram was implemented with bi-gram N=2 and tri-gram N=3. Similarity or dissimilarity statistical coefficients are calculated between the processing word and a list of roots are extracted from a dictionary to extract the root of a word. The roots that have the highest for similar or lowest for dissimilar coefficient are named as probable roots.

THE PROPOSED APPROACH

Hidden Markov Model is one of the most important machine learning models in speech and language processing (Jurafsky and Martin, 2000). HMM is a probabilistic sequence classifier, given a sequence of units (in our case letters) and its job is to compute the probability distribution over possible labels and choose the best label sequence.

The Hidden Markov Model is a finite set of states, and a set of transitions between states that are taken based on the input observations. Each of which is associated with a probability distribution (Lawrence, 1989). Weights are augmented; where each transition is associated with a probability of how likely state a transit to state b. Transitions among the states are governed by a set of probabilities called transition probabilities. In a particular state, an outcome or observation can be generated, according to the associated probability distribution. It is only the outcome, not the state visible to an external observer and therefore, states are ``hidden'' to the outside; hence the name Hidden Markov Model (Lawrence, 1989).

A Markov chain is a special case of a weighted auto-maton in which the input sequence uniquely determines which states, sequence will go through. In our case the sequence represents a word.

Weight extraction

Hidden Markov Model is used to extract Arabic word weights. HMM is represented by a set of states and a set of transitions from one state to another. A given word is tested through the model by using states as the letters of the word, and the transition from start state 0 to end state will represent the full word. The model will output the path which yields the highest probability. There are two probability matrices, the state transition probability matrix, and the emission probability matrix. State transition matrix will provide the probability of going from state i to state j. Furthermore, the emission probability matrix will provide the probability of emitting an observation in a given state i, observations are the alphabets of the Arabic language plus a special character called “Shadda” “شدة”, a total of 31 observation is considered.

Elements of the proposed Hidden Markov Model are:

A set of N states $S = s_1 s_2 ... s_N$ representing the number of states of the model, each state represent one letter of a word, and a path from state s_i to s_j represent a word. N = 172 states.

A transition probability matrix A. $A = a_{11} a_{12} ... a_{nn}$, where a_{ij} represents the probability of moving from state i to state j, $A = \{a_{ij}\}$. That is going from one letter to the next in a given word.

A sequence of K observation $O = o_1 o_2 ... o_k$ each drawn from the vocabulary $V = v_1, v_2, ... v_V$, V represents Arabic letters plus some special letters. The number of observation symbols in the alphabet, M =31.

A sequence of emission probabilities $E = e_i(o_k)$, each sequence expresses the probability of an observation o_k being generated from a state i.

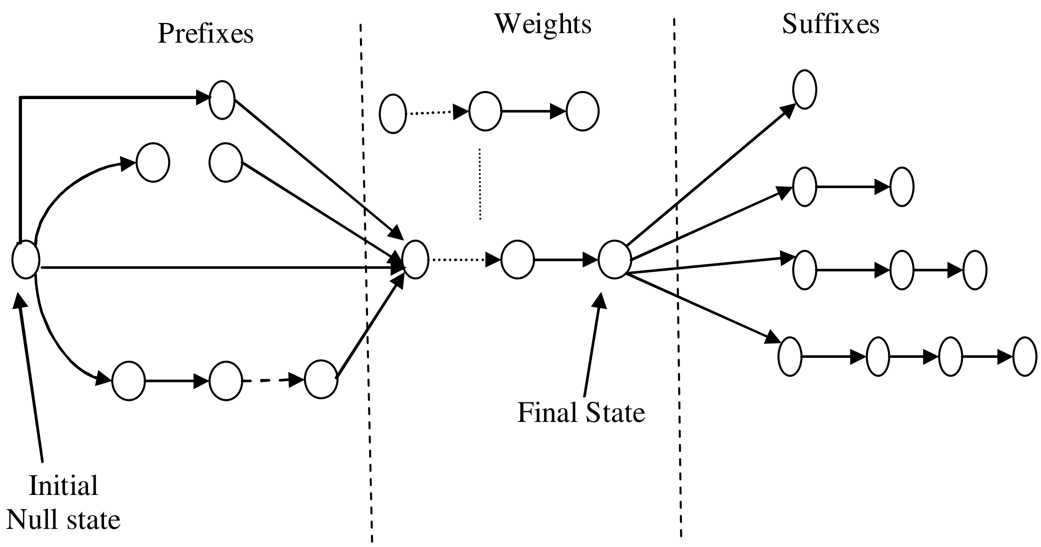

Figure 1. Proposed model.

Table 1. Example of words decoding

Word	Prefix	Weight	suffix	length	Weight
Fsayakfekahma	Will be	Enough	For them	10	FAL
Muslim	-	Muslim	they	6	mFAL
Yalaabn	ya	play	they	5	FAL
Aljamee	the	all	-	6	FAeL

A special (start) state and a final (end) state S_0, S_F which are not associated with observations. The proposed model has one null initial state and multiple end states. End state can be the last state of any valid weight states, or a suffix state.

For example, a word “مسلمون” (Muslims) has six letters, adding a null state, it should start with state 0 and goes up to six states depending on A (transition probability), and E (emission probability). There could be multiple correct paths for the word, but the only one with the highest probability will be accepted as a valid solution. In the case of our example, a path of states, 0, 8, 15, 16, 17, 170, 171, state 0, as a starting null state, state 8 will represent the letter “” and it is considered as a part of the prefix states group, and it only prefixes a noun, so the word will be identified as a noun. State 15 up to 17 represent the weight “F3L” (فعل=سلم) and it is also the root of the word. States 170, and 171 (ون) are the suffixes of the word, and it is special for plurals. Other words are found in the same way (Figure 1).

First, we define the number of states S in the system. A total of 172 states were identified as prefixes, weights, or suffixes. Prefixes are represented by 15 states. States are logically divided into three groups that identify the set of prefixes state group, the set of weights states group, and the set of suffixes states group. Weights are represented by 82 states, and suffixes are represented by 75 states. We start with one initial null state, and multiple end states. An end state is the last state of any valid end state of a weight, or a suffix state. Observations are 28 Arabic letters added to it shadah (شدّة), Alef maqsora (ى) and Taa (ة), and we distinguish between Alef and Hamza (أ and ا). A total of 31 observations is embedded. Figure 1 shows the proposed model design.

A word may or may not have a prefix. Prefixes are of length up to 7, for example the word (وبالاستخدام). The word has a prefix of length 7. A word without prefixes or suffixes could be of size 3 - 4 - 5 - 6 - 7 with infixes. A word may or may not have a suffix. Suffixes could be of length up to 4. For example, the word (فعاليّات) has a suffix of length four letters.

For example, a word (اجتمعنا) will have one prefix, and two suffixes, leaving 4 letters to represent the pattern (افتعل), which is a verb in the past tense (as shown in Table 1). Hidden Markov Model is characterized by three problems, the evaluation problem, the decoding problem, and the learning problem. Evaluation is also referred to as computing the likelihood, given an HMM $\lambda = (A, E)$ and a sequence of letters $O = o_1, o_2, ..., o_k$, find the pro-bability that the word letter are generated by the model, $p\{O \mid \lambda\}$. Forward algorithm (Jurafsky and Martin, 2000) is used to compute such likelihood.

Furthermore, decoding will discover the best hidden state sequence (S) in the model that produces the word. Given a word is represented by letters $O = o_1, o_2, ..., o_k$ and HMM $\lambda = (A, E)$. The discovery of the hidden sequence depends upon the way most likely state sequence is defined. It can be interpreted as a search in a graph whose nodes are formed by the states of the HMM in each of the time instant $k, 1 \le k \le K$. Viterbi algorithm (Jurafsky and Martin, 2000) solves this problem where the whole state sequence with the maximum likelihood is found.

In addition to evaluation and decoding, the learning problem is needed to extract model parameters from a training set. Learning is defined as, given a model λ and a sequence of letters (a word) $O = o_1, o_2, ..., o_k$, how should we adjust the model parameters in order to maximize $p\{O \mid \lambda\}$, that is to learn the HMM parameters A, and E.

The input to the learning algorithm would be unlabeled sequence of observations O (letters) and a vocabulary of potential hidden states S which simply means the word and the correct path of states it should have. Standard algorithms for HMM training are Forward-backward, and Baum Welch algorithm. The Algorithm will train both the transition probabilities, A, and the Emission Probability, E, of the HMM. Generally, the learning problem is how to adjust the HMM parameters, so that the given set of observations (words) is represented by the model in the best way for the weight-root extraction system. The Forward-Backward Algorithm was used to train our system.

Word type detecting

The proposed system can extract the word type (Verb-Noun) depending on different criteria. The detection of a word type (N, V) may depend on any of the following: Prefixes; suffixes; weights; word preceding the word in question (particles).

Some prefixes are attached only to nouns (for example, ال), others may only precede verbs (for example, ي). The same concept follows the suffixes attached to nouns only (for example, ات), and other attached to verbs only (for example, نّ). If word type was not detected by prefixes and suffixes then we check for the extracted word's and suffixes then we check for the extracted word's weight. Weights are either belonging to nouns, or verbs, or common between them. For example, the word (احتبس) has the weight (افتعل) which is a verb; whereas, the word (مركب) has the weight (مفعل) which is a noun. An example of common weight is (فاعل). Words preceding the word in question may detect a word type. For example, words like (في) only precede nouns and words such as (لم) only comes before verbs. Those preceding words are considered as stop words in a text processing system. Over 90% of word types can be detected by the given method. Weights might also help in part-of-speech tagging.

Bi-gram word model

Hidden Markov Model will provide the most probable path for the given sequence of letters that represents a word. The relation between two consecutive letters is not preserved by the model. Therefore, a bi-gram model was constructed from the training words to preserve the letter to letter succession. This is done because of a problem detected upon testing the decoding phase of the HMM. A word which begins and ends with letters that has a high possibility of being a prefix or a suffix can be interpreted wrongly by the system. For example, the word (نشرت) begins with a letter (ن) which can be a prefix and ends with the letter (ت) which can be a suffix. The correct path is to consider the last letter as a suffix, but the system may consider wrongly the first letter as a prefix. To prevent this, Two, 28 × 28 matrices were constructed with Arabic alphabetic as the rows and the columns of the matrix. The value is considered as the probability of going from letter A to letter B in the beginning of the word for the first matrix, and the probability of going from letter A to letter B at the end of the word for the second matrix. It was found that, having the sequence (نش) as the first two letters of a word is more probable (14%) than having the sequence (رت) as the last two letters of the word (2%).

EXPERIMENT AND RESULTS

About 15 million words were used to train the model. Those words constitute all possible different forms that a word could have. Words were generated by the aid of Arabic dictionary (Ar-Rhazi, 1989; Al-Asmar, 2009). Based on word root, and possible weights for those roots, different forms of a word were generated. The generated words were attached to different prefixes and suffixes following Arabic morphological rules. The following are the procedure to produce the Hidden Markov Model parameters: Collect words' roots and patterns for those roots from Arabic dictionaries; generate different forms of a word using morphological rules; add suffixes and prefixes to resulting words; use the final result to train the model using forward-Backward algorithm.

The result is two matrices, one for state transition probability, and the other for observation emission probability. State transition matrix will provide the probability of going from state si to state sj. Emission probability matrix will provide the probability of emitting a letter E in state si. These matrices are used as inputs for the Viterbi algorithm to decode a given word. The algorithm was altered to give all possible paths, and not only the one with the highest probability. In order to extract the correct path, further rules have to be applied, which are: End states must not be before the last state of any valid weight (pattern); prefix and suffix matching table must be applied. For example prefix "ي" does not match suffix "ت"; check the Bi-Gram generated matrices probability if the first and the last letters of the word are probable prefix and suffix.

The words were decoded using Veterbi Algorithm. Those words (training set) were extracted from different documents. The following are the processing procedure of a text in order to extract weights and roots: Tokenizing words and eliminating all punctuation; Hamza must all be normalized to one shape "أ"; altered veterbi algorithm is used to decode the words, and find all possible paths; apply the weights correctness rules, and prefix-suffix matching table; select the path with highest probability; states which belongs to the weights' states are identified, thus, extract the root. Table 2 shows an example of the text decoding.

Testing of 50, randomly selected documents from the internet, shows an average count of 400 words after tokenization. The results were compared manually against Arabic dictionary to compare between the correct and the outcome of the system. The presented approach achieved a promising accuracy of 95%. It was found that 2% of error is due to spelling mistake.

CONCLUSION

Arabic is a highly inflected language. The wide range of word forms and the large variety of prefixes and suffixes complicate the extraction of precise features for a text

Table 2. Test result.

Input string	State transition	P of emission	P of state transition	Expected
She Eats	0-3-37-38-39-40-41	0.000616	0.006813	Ttfaal
They Feel	0-5-7-8-9-102-103-	0.0001037160	0.0029909289	Yafalon
Increase	0-2-25-26-27-28	0.000001	0.011867	Eftaal

processing system. Therefore, a preprocessing technique is needed to unify similar words into a single feature before further processing of the text. Root is aimed at finding the base letters which represent a word in a dictionary and, stemming simply refers to stripping prefixes and suffixes from a word. Also, the root may represent a group of words that may have different meaning such as the word (مجتمع=community) and the word (جامع=Mosque) that belongs to the same root (جمع), but has different meaning.

Arabic words are structured in well known patterns called weights thus "weights" are selected in this paper as a feature of an Arabic text. Weights are closer to stems, except some of the prefixes that belong to the weight which will not be removed. The presented approach is based on Hidden Markov Model. Each state in a model is considered as a letter of a word, a word is represented by consecutive states, from start to end. Two questions to be answered are; what is the likelihood of a path for a word, and what is the probability of emitting the letter of a word in a given state of the path. The model was trained with a collection of words extracted from Arabic dictionaries and, it was ensured that words constitute a verity of prefixes, suffixes, and weights. Different rules have to be applied before selecting the path with highest probability. States are distinguished as prefixes or the weight or suffixes from the selected path. Testing the system with different documents which belongs to different categories, a 95 % correctness was accomplished by this paper. The weights of document's words were manually checked against Arabic dictionary to compare with the extracted result.

In the future, we aim at studying the reduction of the extracted weights by grouping weights with similar meaning and different states (single, plural, past, present) into unified ones. Furthermore, the selected features will be tested in text processing task (for example, clustering) against the root features for a comparison.

REFERENCES

Ahmed F, Nürnberger A (2007). N-grams Conflation Approach for Arabic, ACM SIGIR Conference, Amsterdam.

Al-Ameed H, Al-Ketbi S, Al-Kaabi K, Al-Shebli K, Al-Shamsi N, Al-Nuaimi N, Al-Muhairi S (2005) Arabic Light Stemmer: A new Enhanced Approach. The Second International Conference on Innovations in Information Technology (IIT'05).

Al-Asmar R (2009). The Detailed Lexicon in Morphology. Scientific book publisher. (Arabic Book)

Al-Sahmsi F, Guessoum A (2006). A hidden Markov Model – Based POS Tagger for Arabic. 8es Journees internationals d'Analyse statistique des Donnees Textuelles.

Ar-Rhazi MB (1989). Mukhtar Us-Sihah, Librairie du Liban. (Arabic Book)

Buckwalter T (2002). Buckwalter Arabic Morphological Analyzer. the Linguistic Data Consortium, University of Pennsylvania.

Chen A, Gey F (2002). Building an Arabic stemmer for information retrieval.

Darwish K (2002), Building a shallow Arabic Morphological Analyzer in one day, Proceedings of the ACL-02 workshop on Computational approaches to semitic languages, pp.1-8, July 11, Philadelphia, Pennsylvania

El-Hajar A, Hajar M, Zreik K (2010). A System for Evaluation of Arabic Root Extraction Methods. fifth international Conference on Internet and Web Applications and Services.

Jurafsky D, Martin JH (2000). Speech and Language Processing An Introduction to Natural Language Processing Computational Linguistics and Speech Recognition, Volume: 21, Prentice Hall.

Khoja S, Garside R (1999). Stemming Arabic Text. Technical report, Lancaster University, Lancaster, U.K.

Larkey L, Ballesteros L, Connell M (2005). Light Stemming for Arabic IR Arabic Computational Morphology: Knowledge-based and Empirical Methods, A.Soudi, A. van en Bosch, and Neumann, G., Editors. Kluwer/Springer's serieson Text, Speech, and Language Technology.

Larkey LS, Ballesteros L, Connel ME (2002). Improving Stemming for Arabic Information Retrieval: Light Stemming and Co-occurrence Analysis. Proc. of the 25th annual international ACM SIGIR conference on Research and development in information retrieval, pp. 275 – 282.

Lawrence RR (1989). A tutorial on Hidden Markov Models and selected applications in speech recognition, Proceed. IEEE, 77: 2.

Mohamed El-Hadj, Al-Sughayeir IA, Al-Ansari AM (2009). Arabic Part of Speech Tagging Using the Sentence Structure. 2^{nd} international Conference on Arabic Language Resources & Tools. Cairo.

Sinane M, Rammal M, Zreik K (2008). Arabic documents classification using N-gram, Conference ICHSL6, Toulouse.

Taghva K, Elkoury R, Coombs J (2005). Arabic Stemming without a root dictionary.