International Journal of Engineering & Technology, 7 [2.29] [2018] 742-746

International Journal of Engineering & Technology

Website: www.sciencepubco.com/index.php/IJET

Research paper

Arabic Part-of-Speech Tagger, an Approach Based on Neural Network Modelling

Rabab Ali Abumalloh[1]* , Hasan Muaidi Al-Serhan[2] , Othman Bin Ibrahim[3], Waheeb Abu-Ulbeh[4]

[1,,3,4]Faculty Of Computing, Universiti Teknologi Malaysia, 81310 UTM Johor Bahru, Malaysia
[1]Computer Department, Community College, Imam Abdulrahman Bin Faisal University, P.O.383, Qatif, Saudi Arabia
[2c] Ajloun National University, Information Technology, Ajloun − 26810, Jordan
**Corresponding Author E-Mail: Ramolloh@Uod.Edu.Sa*

Abstract

POS-tagging gained the interest of researchers in computational linguistics sciences in the recent years. Part-of-speech tagging systems assign the proper grammatical tag or morpho-syntactical category labels automatically to every word in the corpus per its appearance on the text. POS-tagging serves as a fundamental and preliminary step in linguistic analysis which can help in developing many natural language processing applications such as: word processing systems, spell checking systems, building dictionaries and in parsing systems. Arabic language gained the interest of researchers which led to increasing demand for Arabic natural language processing systems. Artificial neural networks has been applied in many applications such as speech recognition and part of speech prediction, but it is considered as a new approach in Part-of-speech tagging. In this research, we developed an Arabic POS-tagger using artificial neural network. A corpus of 20,620 words, which were manually assigned to the appropriate tags was developed and used to train the artificial neural network and to test the part of speech tagger systems' overall performance. The accuracy of the developed tagger reaches 89.04% using the testing dataset. While, it reaches 98.94% using the training dataset. By combining the two datasets, the accuracy rate for the whole system is 96.96%.

Keywords: *Part of speech tagging; Arabic tagger; artificial neural networks*

1. Introduction

This research addresses the problem of designing POS-tagger for Arabic language. Which can be used for developing other applications in natural language processing field. Computational linguistics is a discipline of artificial intelligence that deals with the logical modeling of natural language from a computational perspective. It combines two fields: computer science and natural languages [1]. Computational linguistics focuses on proving linguistics theories using computer [1]. Part of speech tagging can be defined as the process of automatically assigning or choosing the best matching part-of-speech or other syntactic class marker to all the included words in the corpus [2]. The previous definition suggests that the system will accept a string of words and a predefined tagset as inputs and assigns the best single tag for each word. Part of speech tagger can be developed using two main methodologies which are: a] rule-based methodology; [b] stochastic[probabilistic] methodology. These two methodologies are the basic of most of the current POS-tagging systems. To get higher success rates these two methodologies were combined in some of the existing systems.

Artificial neural networks, or neural computation networks could be defined as:" parallel computing systems which is biologically inspired, that composed of large number of interconnected simple processors with weights bound to the connections [3]. The basic idea of ANNs was derived from biological neural networks [4]. Biological neural networks consist of nerve cells or special biological cells that can process information. Nerve cells composed of a cell body [soma], and two out reaching branches: axons [transmitters] and dendrites [receivers] [3]. ANNs models were inspired from biological neural networks that uses the many desirable characteristics in the human brain, which include distributed representation and computation, massive parallelism and the ability to learn and make generalizations [3].

ANNs can be represented by a graph consists of the nodes which represents the artificial neurons and the directed weighted edges that represents the connections from the outputs to the inputs of the neurons. An ANN can be defined by four parameters: type of neurons, connectionist architecture, learning algorithm and recall algorithm [4]. ANNs can be categorized according to their connections into two types: fully connected, in which each neuron is connected to all other neurons in the network and partially connected, which does not present all the possible connections between the neurons in the network [4]. ANNs can also be categorized according to their connectionist architecture into: auto-associative and hetero-associative.

The limitations of the current Arabic tagging systems and the decency of the accuracies of the available systems have induced us to investigate a novel approach to build POS-tagger system for Arabic language based on artificial neural networks. Thus, the main research question of this study is summarized as follows:

Is it applicable to use artificial neural networks as a tool to build part-of-speech tagger system for Arabic language?

To answer the research question, backpropagation ANN was developed. The parameters of this ANN were optimized and tested based on experimentation. The technique that was used for the development of the part-of-speech tagger system which described in this research proved to be well suited to the Arabic language.

2. Literature Review

The need For Arabic language part of speech taggers in different natural language processing applications motivated the researchers to develop tagger systems based on different methods since 2000, but these researches didn't investigate the problem as extensively as in English language [1].

Khoja [1] developed the APT system [Automatic Arabic POS-Tagger]. This tagger combined two approaches: statistical and rule-based techniques [1]. The APT is considered as the first tagger system for Arabic language. The tagset used in APT consists of 131 tags which were derived from the BNC English tagset [1]. Khoja derived her initial tagset from the grammar of Arabic language. The APT achieved an accuracy of 86% [1].

Alqrainy developed part of speech tagger using the rule-based approach. The tagger is called AMT [Arabic Morphosyntactic Tagger]. The input of AMT is untagged raw partially-vocalized Arabic corpus. While, the output is the correct tags to each word in the corpus. The system didn't depend on manually tagged or untagged dictionary to generate the tagged corpus. The AMT consists of two rule components: pattern-based rules and lexical and contextual rules. The AMT system achieved an average accuracy of 91% [5].

3. Methodology/Materials

3.1. The Arabic Tagset

The Arabic tagset that has been compiled in this study is based on the grammar of Modern Standard Arabic which categorize the Arabic words into three classifications which are: verb, noun and particle. The main part of speech tags for Arabic language are presented in Figure 1. A set of tag symbols were generated to serve as a tagset in this study. Each part of speech tag is represented by three letters [T, S, G] where:

1. T: This letter is used to represents the main tag of the word: noun, verb or particle.
2. S: This letter is used to represents the sub-class of the main tag.
3. G: This letter represents the gender [masculine or feminine].

For example, the word [كاتبة] is represented by the tag [N1P2F3], where N1 represents the noun and P2 represents the sub class of the word which is patient noun and the gender of this noun is feminine which is represented by F3. For this study we needed a tagged corpus to be used in the training process, to the best of the authors knowledge there is no available free corpus for the public. Because of this, we developed our own tagged Arabic corpus. Studies show information regarding development of the corpus [6].

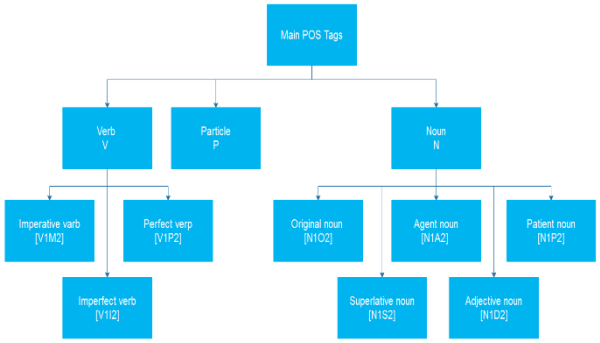

Fig. 1: The main part of speech tags for Arabic language

3.2. The Architecture of The Developed POS-Tagger System

POS-tagging process contains several stages which are: tokenizing process, cleaning process, preparation of the data and tagging process. The system is considered as a classification system, since the main task of the developed POS-tagger is to classify the Arabic words according to their tags. The architecture of the developed POS-tagger system is elaborated in Figure 2.

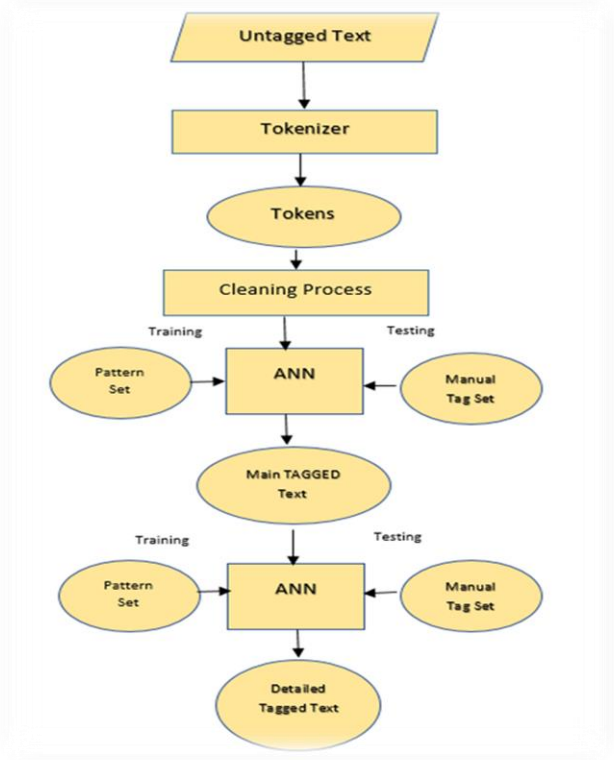

Fig. 2: The architecture of the developed POS-tagger system

We developed a tokenizer system to process the input text and to split it into separate tokens. Using the space character, the input text will be converted into tokens by the tokenizer using a simple program. An example of the tokenizing process is presented in Fig. 3. After the tokenizing process is done, there is a need for a cleaning process. Punctuation marks must be extracted from the input text. Otherwise, the punctuation marks will be considered as words. Numbers also were extracted from the input text. The input to the cleaner is a set of tokens included punctuation marks, numbers and the output is a set of more manageable tokens. The cleaning process is presented in Fig. 4.

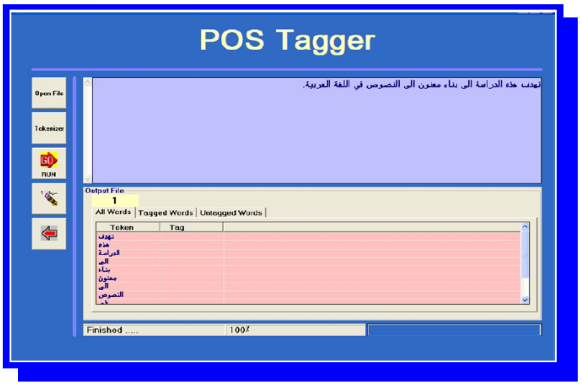

Fig. 3: Tokenization process

Fig. 4: Cleaning process

After the tokenizing and the cleaning processes, the final tokens should be prepared to be processed by the ANN. The preparation process is conducted using other two sub processes. These processes are: the coding and the normalization. The tokens here are a series of Arabic letters. Arabic letters were coded to numeric values using the method applied by [7] which is basically based on analyzing the Arabic affix letters' frequencies. An example of the coding method is presented in figure 5. After that, we applied the binary coded method to code the target vector. Each category in the tagset is linked to a vector of a binary number in which "1" represents the acceptance of the tag for the presented word and "0" means that the tag is not accepted for the given word. The length of the target vector depends on the number of different grammatical categories that were compiled in the developed annotated corpus. There are18 different tags in the annotated corpus, so the length of the target vector equals 18. Table 1 summarizes these tags and their corresponding target vectors.

Using the actual data as it is may lead the ANN architecture to saturation [7]. Also, there is a need for activation functions to normalize the inputs and targets to keep their values within a prespecified range. The normalization is the process of bringing the input data columns at the same interval. In our case, the input values will be normalized to the range [0...1] of real numbers. The normalization process is done using the premnmx function [8].This function is one of the built-in functions in MATLAB. The syntax of premnmx function is shown in Equation 1 [8].

$$[Xn, minX, max\ X] = premnmx(X) \quad (1)$$

Where:
- X: The input vector.
- Xn: The normalized input vector.
- minX: A vector containing minimums for each vector X.
- maxX: A vector containing maximums for each vector X.

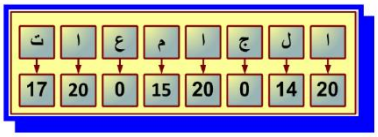

Fig. 5: Coding process of the word" الجامعات"

Table 1: Coding for the Target Vector

Code	Tag	Target Vector
1	[N1O2M3]	[000000000000000001]
2	[N1O2F3]	[000000000000000010]
3	[N1A2M3]	[000000000000000100]
4	[N1A2F3]	[000000000000001000]
5	[N1P2M3]	[000000000000010000]
6	[N1P2F3]	[000000000000100000]
7	[N1D2M3]	[000000000001000000]
8	[N1D2F3]	[000000000010000000]
9	[N1S2M3]	[000000000100000000]
10	[N1S2F3]	[000000001000000000]
11	[N1X2M3]	[000000010000000000]
12	[N1X2F3]	[000000100000000000]
13	[V1P2M3]	[000001000000000000]
14	[V1P2F3]	[000010000000000000]
15	[V1I2M3]	[000100000000000000]
16	[V1I2F3]	[001000000000000000]
17	[V1M2M3]	[010000000000000000]
18	[V1M2F3]	[100000000000000000]

3.2. The Topology of the Artificial Neural Network

The topology of the designed ANN which was used in this study is shown in figure 2. The number of neurons in the input layer is 12 which is considered as the maximum length of a valid word in Arabic language [7]. The number of neurons in the output layer depends mainly on the number of the classes that the ANN tries to classify. There is one hidden layer in the developed ANN. Using different number of neurons for the hidden layer is an important aspect for determining the best performance of the ANN. No simple rule exists that indicates how many hidden neurons are required for a specific problem [9]. Kasabov [4] suggests a formula to determine the number of neurons in the hidden layer. In this study, we followed Kasabovs' formula to determine the number of neurons in the hidden layer we found that the number of neurons in the hidden layer is 8 neurons [4]. The formula we used is shown in Equation 2.

$$HiddenSize = \frac{InputSize+OutputSize}{4} \quad (2)$$

Choosing an adequate value for the learning rate parameter has a significant effect over the ANN performance. Usually, the value should be within [0,1] interval of real values, but it is possible that this interval will be too large. Alternatively, choosing a small value for it within [0.05,0.25] of real values will be a good choice but it may cause a slow convergence to the solution. When the chosen value is too large, the process of learning is faster. For the present study, to choose the efficient value of the learning rate parameter, we conducted six independent running experiments randomly. Each experiment has its own learning rate value. The final results show that the optimum value for the learning rate is **0.1**. The chosen values for the learning ratio are presented in Table 2.

Table 2: Choosing the Learning Rate Parameter

Experiment	Learning rate	Training Size	Correctly Tagged Words	Ratio
1	0.7	5000	4710	94.2%
2	0.5	5000	4862	97.2%
3	0.35	5000	4888	97.8%
4	0.2	5000	4901	98%
5	0.1	5000	4967	99.3%
6	0.05	5000	4773	95.5%

After finalizing the network architecture including the number of the neurons at each layer, it is important to determine the different weights values that will lead to minimum error rate using the backpropagation training algorithm which will be used to train the developed network. We applied the method done by [10] to assign the connections weights that will reduce the error rate. In this method the input vector X = < X1, X2, ... , X12 > with its corresponding output Y = < Y1, Y2, ... , Y10 > were presented to the ANN as experimental data. The input to the artificial neural network consists of the collection of characters that form the Arabic words, while the generated output from the artificial neural network consists of the associated tags of the Arabic words. The error is calculated by computing the difference between the desired output and the actual output using the mean square error [MSE]. To change the weights of the artificial neural network and reduce the error rate the error was propagated backward. This process was repeated for a series of experiments until we reached an acceptable error rate. This summarizes how the learning was achieved in the proposed ANN. The backpropagation algorithm which was used to train the developed ANN has three main stages which are: initializing the weights, forward pass and backward pass.

Initializing the Weights: Set all the weights of the developed ANN to random numbers uniformly distributed inside a small range: $[-\frac{2.4}{Fi}, +\frac{2.4}{Fi})$,where **Fi** represents the total number of inputs to neuron **i** in the developed ANN. We initialized all the weights of the developed ANN as follows: the weights of the hidden layer are in the range: [−0.30, +0.30] and the weights of the output layer are in the range: [−0.13, +0.13].

The forward pass was done by applying the input vector **X** to the input layer and calculating the inputs and the outputs of the hidden layer using the result of the summation of multiplying the input values by the corresponding weights for each neuron in the hidden layer. The output for each neuron was calculated by applying the suitable activation function for the computed summation. In this study, the logistic activation function was used, because it gives graduation values between zero and one. After that, the inputs and the outputs for the output layer were calculated using the

summation of multiplying the output values from the hidden layer by their corresponding weights for each neuron in the output layer. The output for each neuron was calculated by applying the logistic activation function for the computed summation. The output here is the vector **O**. This vector represents the network output. Finally, the calculated network output **O** was compared against the desired output **Y**. If a difference existed, then the error is computed using equation 3.

$$Error = Desired\ Output - Network\ Output \quad (3)$$

In the backward pass, the adjusting of the weights for the output layer and the adjusting of the weights for the hidden layer were carried out.

The above two passes were repeated for each example in the training set. After all examples were trained, the mean square error [MSE] was computed using Equation 4. Where s represents the size of the training set.

$$MSE = \frac{1}{2}\sum_{p=1}^{s}(Y_p - O_p)^2 \quad (4)$$

When the computed [MSE] reached the threshold level, the training process will be terminated, and all the final weights will be saved. The threshold level used in this study is 10^{-5}.The main program that was used to build the system is Visual Basic programming language [VB]. In the training process and determining the final weight values we used MATLAB 'the Language of Technical Computing'. The input vectors and the desired outputs which were previously coded using the Visual Basic program were processed by MATLAB for the training process. The training process was completed by producing a text file that contained the final updated weights. These weights were returned to Visual Basic program to perform the testing and tagging process for different Arabic words in the corpus.

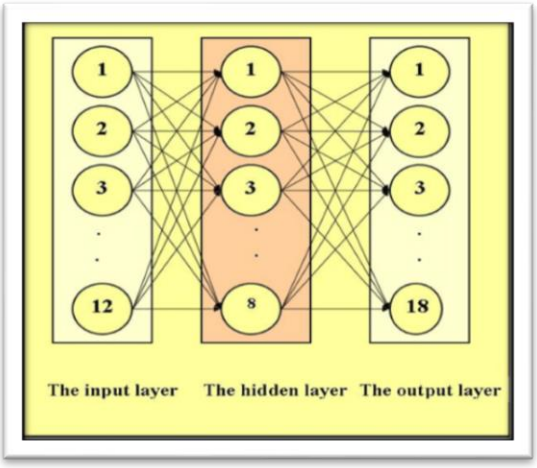

Fig. 6: The topology of the designed ANN

4. Results and findings

4.1. The training stage

We divided the complete dataset into two groups which were used for training and testing the artificial neural network. The data set of 20,620 Arabic tagged words were used to train and test the artificial neural network. Among these words we used 80% of the complete dataset which approximately contains 16,496 Arabic words to train the developed system while we used 20% of the complete dataset which contains 4,124 Arabic words to test the artificial neural network. The classification of the words among the training and the testing datasets was done in a random way to eliminate the bias. The developed ANN was trained by the backpropagation algorithm in a supervised learning paradigm. The developed ANN tagged 98.94% of the words in the training dataset correctly. These results indicate that the developed ANN is well trained. Figure 7 presents the results of the training process in a bar format.

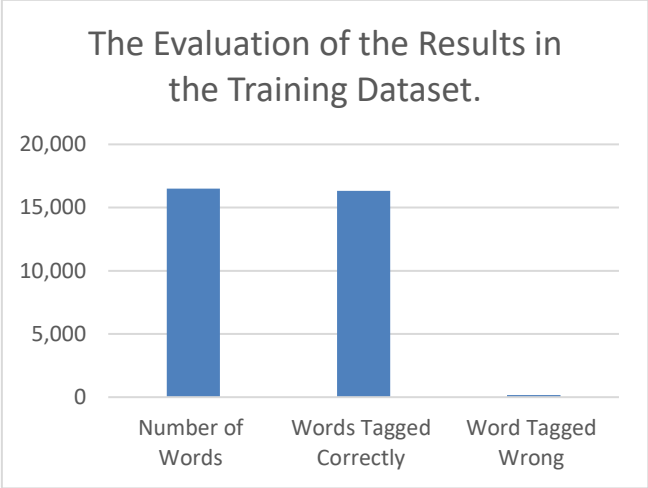

Fig. 7: The Results of the Training Stage

4.2. The Training Stage

To determine the accuracy of the developed system we conducted an experiment using the testing dataset which contains 4,124 Arabic words. These words are considered as a new words and unseen by the Artificial neural network. The accuracy of the artificial neural network can be measured using the success rate formula. The success rate measure was used because the developed tagger systems' main goal is to assign a single tag to each token, and this measure is suitable for this case. Success rate measure is expressed in Equation 5

$$S_R = \frac{C_T}{N} * 100\% \quad (5)$$

Where:

- S_R: The success rate.
- C_T: The number of the correctly tagged tokens.
- N: The size of the testing dataset.

The success rate resulted after conducting the experiment with the testing data set reached 89.04%. Figure8 shows the results of the experiment in a bar format. There is another measure of the performance of the tagger system which is the ambiguity rate that is used when more than one tag is assigned to the token. Since the target of the developed system is to assign one main tag to each token, this measure was not used.

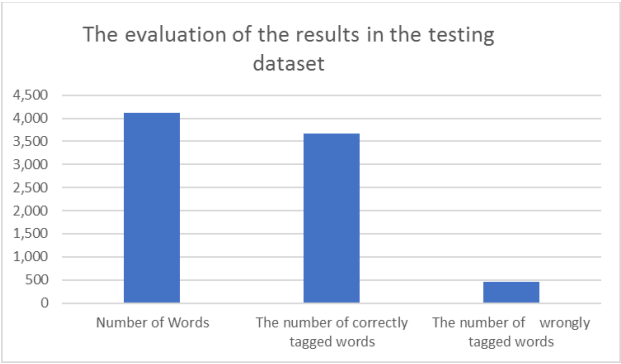

Fig. 8: The Results of the Testing Stage

5. Discussion of the Results

The overall accuracy of the developed tagger system reaches 98.94% using the training dataset. The overall accuracy of the developed system reaches 89.04% using the testing dataset. The accuracy rate for the whole system is 96.96%. Table 3 summarizes all these results. Figure 9 illustrates these results in a bar format. Table 3 shows

that there is a gap in success rate SR between the training dataset and the testing dataset. The success rate percentage reaches 98.94% for the training dataset, while it reaches 89.04% for the testing dataset. The reason behind this gap is that for the training dataset, all the words were seen before by the ANN. While, for the testing dataset, all the tested words were considered as new unseen words by the ANN. The above accuracy could be improved by increasing the number of the words in the training corpus. Also, particles could be removed from the testing corpus to increase the accuracy or it could be tagged separately and included within the training corpus. We can conclude that using the ANN to build the developed POS-tagger depends basically on the training process and the training data.

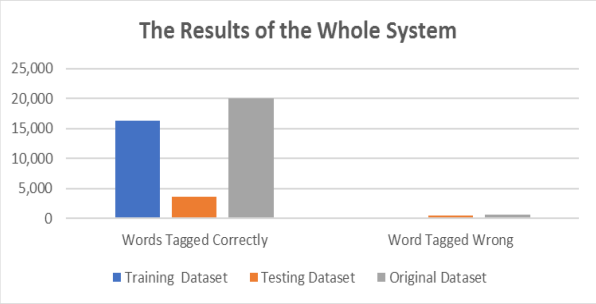

Fig. 9: The Results of the Whole System

Table 3: The Results of the Whole System

	Training Dataset	Testing Dataset	Original Dataset
Total number of the Words	16,496	4,124	20,620
The number of correctly tagged words	16,321	3,672	19,993
The number of wrongly tagged words	175	452	627
Success Rate SR	98.94%	89.04%	96.96%

6. Conclusion

The reason behind choosing the ANNs is that in the recent years they have been known as popular tools for research in the computational linguistics field. ANNs have many advantages over other approaches. One of these advantages is that ANNs can be efficient when the linguistic and the grammar rules are not known [in some cases] due to the complexity of the language itself. ANNs gather knowledge through learning or training. We can infer that the ANN is trained, if applying the input vectors of data will produce the expected outputs or at least consistent outputs. The knowledge gained through this process will be stored by the weights of connections between neurons. The training vectors [examples or patterns] will be presented by the ANN sequentially, and the weights will be adjusted to" remember" the knowledge represented by those vectors. The process of adjusting the weights is made accordingly with a pre-established procedure called training or learning algorithm. Generally, every input vector from the training set is presented to the network more than once. The training algorithm must determine the convergence of the weights to such values so that every input vector produces the expected output.

In this study, the backpropagation training algorithm was used to train the developed ANN. The ANN is the developed POS-tagger for Arabic language. All the needed parameters for the ANN were optimized during this study. In the existing literature, there are few researches in developing POS-taggers for Arabic language. This has motivated us to develop an Arabic part of speech system using artificial neural networks. The results of this study are encouraging and should give the researchers an insight for the future work in this area.

References

[1] Khoja S. APT : Arabic Part-Of-speech Tagger. Proceedings of the Student Workshop at NAACL. 2001:20--5.
[2] Jurafsky D, Martin JH. Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition. Speech and Language Processing An Introduction to Natural Language Processing Computational Linguistics and Speech Recognition. 1999;21:0-934.
[3] Larsen J. Introduction to Arti cial Neural Networks. 1999[November].
[4] Kasabov NK. Foundations of neural networks, fuzzy systems, and knowledge engineering. Engineering. 1996:581-.
[5] Alqrainy S. A morphological-syntactical analysis approach for Arabic textual tagging. 2008:1-2.
[6] Abumalloh RA, Al-Sarhan HM, Abu-Ulbeh W. Building Arabic corpus applied to part-of-speech tagging. Indian Journal of Science and Technology. 2016;9[46].
[7] Al-serhan HM, Montfort DE. Of Arabic Word Roots Extraction An Approach Based on. 2008.
[8] Demuth H. Neural Networks. Mathworks inc. 2006;19[1]:1-7.
[9] Al-Serhan HM, Muaidi H. Extraction of Arabic word roots: An Approach Based on Computational Model and Multi-Backpropagation Neural Networks. 2008.
[10] Lai S, Serra M. Concrete strength prediction by means of neural network. Construction and Building Materials. 1997;11[2]:93-8.