Asian Journal of Information Technology 5(7): 784-789, 2006

Design and Implement an Automatic Neural Tagger Based Arabic Language for NLP Applications

Jabar, H. Yousif, Tengku, M. and Tengku Sembok
Faculty of Information Science and Technology, Universiti Kebangsaan Malaysia,
43600 UKM Bangi, Selangor D. E., Malaysia

Abstract: To this day, various methods such as the statistical model, rule-based models and Support Vector Machine have been used to implement the POS tagger systems. However, these approaches require a large amount of data in order to adapt and implement the POS tagger. The neural approaches, on the other hand, only use lesser amount of data to perform the training and learning stages. The Arabic part of speech (POS) based multilayer perceptron is designed and implemented, while the Error back-propagation learning algorithm is used. The experiments have proven that not only the multilayer perceptron tagger is highly accurate (of 99.99%), it requires low processing time and a lesser amount of data to achieve the learning phase. A new coding scheme is investigated and implemented.

Key words: Arabic part-of-speech, multilayer perceptron, neural networks, information coding, NLP

INTRODUCTION

One of the aims of the Natural Language Processing (NLP) applications is to enable the computer to identify the text input and the meaning of each word used in a text. Hence, the POS tagger plays a crucial important part in most of the NLP applications, such as information extraction and information retrieval systems, machine translation, speech recognition as well as grammar spelling checkers. The POS is the classification of words according to their meanings and functions. Nevertheless, the accuracy of the POS tagging is determined by factors like ambiguous words and, phrases, unknown words as well as multi-part words. Neural networks are one of the most efficient techniques for learning from a scarce data. Therefore, the aim of this study is to design and implement an Arabic Part of Speech tagger (POS) based on the Multi-Layer Perceptron neural network (MLP).

The Arabic language processing has recently become the primary focus of research and commercial development because it has become the official language to a large number of populations[1,2]. In addition, any important Arabic NLP applications must include a speed POS tagging as one of its main tasks. The reliability of the POS tagger depends largely on its ability to address matters expeditious, expressive, inclusive, accurate and portable. The English POS tagger systems have been implemented using various methods such as "Rule-Based" model[3], Statistical model[4,5] and "Neural Network" model[6-10]. On the other hand, to perform the Arabic Pos tagger[4,11,12], the statistical models, rule-based models and Support Vector Machine (SVM) models that have been proposed.. However, in order to adapt and implement the POS tagger, these approaches require a vast amount of data. The neural approaches have been known to use only a little amount of data to perform both the training and learning stages. Moreover, the neural-based approaches are capable of consummating the associations (i.e., word-to-tag mappings) from a representative training data set and the approaches can also generalize the unseen exemplars[13].

MATERIALS AND METHODS

Arabic language overview: Arabic words are highly derivative and inflective, both in their nature and structure. Moreover, Arabic words are usually in the compound structures, which should syntactically be regarded as phrases rather than single words[14]. In addition, the orientation of the writing in Arabic is from right-to-left, which makes the Arabic writing to be distinctly distinguished from languages like English and Spanish or any other Latin-based alphabets,, etc. The Arabic words can be divided into three classes, namely Nouns, Verbs and Particles.

Corresponding Author: Jabar, H., Faculty of Information Science and Technology, Universiti Kebangsaan Malaysia, 43600 UKM Bangi, Selangor D. E., Malaysia

Noun: is a word that describes a person or thing, such as the word, "mdrsa" which means *school*.

VERB: is a word that describes actions. This class of words are in three forms, namely past, present and imperative.

Particle: is a word that is preceded or succeeded by a Noun or a Verb such as the prepositions, adverse, interjections and conjunctions.

Neural networks: The neural network, which is known as a powerful data-modeling tool, can capture and represent complex input/output relationships both in linear and non-linear forms. Scientist can adopt the neural network design theme in different applications with certain specific features exist in the method. Their main features are such as the massive parallelism, uniformity, distribution representation and computation, learning ability, trainability, generalization ability as well as adaptivity[15,16]. The Neural networks can be effectively used or applied in various areas like data classifications, resource allocation and scheduling, database mining, speech production and recognition, pattern recognition, etc.

One of the most common neural models used is the multilayer perceptron (MLP). It is known as a supervised network as it requires a desired input/output in order to learn. This type of network is aimed to engender a model that can correctly map the input into the output using the previous knowledge, while at the same time it is able to perform tagging task in a much lesser processing time.

Related work: One of the most renowned work was done by Schmid[10] In his experiment, Schmid successfully demonstrated that a Net-Tagger with a context window of three preceding words and two succeeding words trained on a large corpus called 'Penn Treebank Corpus' performed considerably well as compared to the "statistical" approaches based on "trigram" model and "Hidden Markov model" (HMM).

Similarly, Sejnowski and Rosenberg[17] were the first to introduce the idea of sliding window approach for implementing context in a multilayer perceptrons. In the NETtalk system, the approach was demonstrated and presented in a sequence of letters.

Marques and Pereira[9] adopted a similar approach in[18], but they use small training sets for Portuguese. In addition, they also explored the use of Elman's net but with worse results.

Qing Ma[8], on the other hand, used a two-layer perceptron but the network outputs were later corrected using the Brill's transformation rule approach[5,6] and the context window was dynamically sized.

Q. Ma *et al.*[6,7] created a neuron tagger with variable size context window and assayed it on the Chinese[6] and Thai corpora[7].

MLP CONFIGURATION AND DESIGN

One of the most widely implemented neural network topologies is the MLP. Its most discriminated functions take any shape as required by the input data clusters. It has also been proven that the MLP achieves the performance of the maximum of a posteriori receiver (i.e., considered as the optimal form a classification point of view) when the weights are properly normalized and the output classes are normalized to (zero value or one value)[8]. In this work, the MLP neural network with the error back-propagation learning algorithm is used[19]. The back-propagation rule propagates the errors throughout the network and allows adaptation of the hidden PEs. The description of the process is summarized below. The error correction learning works is the response at PEi at the iteration n, yi(n) and the desired response di(n) for a given input pattern an instantaneous error ei(n) is defined by:

$$e_i(n) = d_i(n) - y_i(n) \quad (1)$$

By correcting the present value of the weight, the theory of gradient descent learning is used to adapt each weight in the network as follows:

$$w_{ij}(n+1) = w_{ij}(n) + \eta\delta_i(n) + x_j(n) \quad (2)$$

Where ei (n) at the output PE computes the local error δ_i (n), it can be computed as a weighted sum of errors at the internal PEs. The constant step size is η.

Momentum learning is achieved when an improvement to the straight gradient descent in the sense that a memory term (the past increment to the weight) is used to speed up and stabilize convergence. Therefore, the equation to update the weights in momentum learning can be illustrated by:

$$w_{ij}(n+1) = w_{ij}(n) + \eta\delta_i(n) + x_j(n) + a(w_{ij}(n) - w_{ij}(n-1)) \quad (3)$$

The a is the momentum value that is set between 0.1 and 0.9.

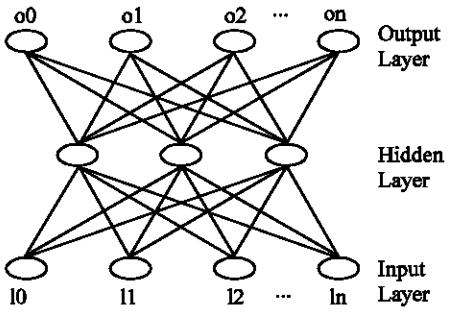

Fig. 1: MLP neural network

For this purpose, the multilayer perceptron neural network was used as depicted in Fig. 1. The MLP has one hidden layer, in which it consists of 13 PEs as the input and output each, with the maximum number of epochs as 1000. A transfer function known as the SigmoidAxon is implemented in the hidden and output layers. It serves as both bias and Sigmoid functions to each neuron in the layer. The range of each neuron in the layer is squashed to between -1 and 1. These nonlinear elements enable the network to make soft decisions. The momentum learning rule is adopted with constant step size =1 at the momentum rate = 0.7.

Each word marked as w from the training data set is encoded as an n-element vector IN = (b1,b2,b3,….,bn), where n is the total number of words. bj is the prior probability that the word w resembled to the tag pos_j and is computed from the training data as given in Eq. 4 and 5. If the word w appears in the training data, the bj will be computed as follows:

$$b_j = C(pos_j, w) / C(w) \quad (4)$$

Where C (pos_j, w) is the number of occurrences of w tagged as posj in the training data, C (w) is the number of occurrences of w in the training data. Otherwise, the b_j will be illustrated as below if the word w does not appear in the training data.

$$b_j = \begin{cases} \frac{1}{N(w)} \text{if pos is a candidate} \\ 0 \text{ otherwise} \end{cases} \quad (5)$$

N (w) represents the number of possible POS tags that can be assigned to the word w. Each n-element probability vector as encoded in the Eq.5.

The output pattern OUT is defined as follows:

$$OUT = (o_1, o_2, \ldots o_n) \quad (6)$$

The output at each node is a function of weighted sum of inputs at the node.

$$op_i = f(\sum_j w_{ji} \quad x_j) \quad (7)$$

Where x_j is the output from j^{th} node of the previous layer.

The network learns the word-tag mappings as a complex function:

$$F(\text{targetword, context}) = \text{tag} \quad (8)$$

The context refers to the set of words in its immediate neighborhood of the target word and the weights of the network serves as the parameters for the function F.

THE CODING SCHEME

In the neural network, several approaches have been made available for the purpose of coding information[20]. Thus, the preference in the means of transforming the data into binary vectors has a great influence in obtaining its success as choosing too many dimensions will require too many parameters and more data points. Whereas, in choosing a few dimensions, you may loose some information that can be essential to identify a particular feature (pattern, class, etc.).

In coding, the input data is converted into a suitable form, which the network can identify and use. Every word in a sentence is associated with a bit-vector, i.e., the size of which is equal to the number of different grammatical categories (for parts of speech) in a specific language. One of such coding approaches is as follows: for a given word, a 1 in a vector field corresponds to a category, admissible for that particular word, while a 0 corresponds to an inadmissible part of speech. The following correspondence between bit-vectors and grammatical categories is used. Nevertheless, this approach requires the use of many memory locations, which filled, with zeros and only one significant location has one value. This has become one disadvantage of this approach.

In contrast, the proposed coding scheme introduces an efficient encoding method, which uses a small number of memory locations and several significant bits, which can help to reduce the training time required as well as increase the chance of getting good results. The procedures of the coding are summarized as below:

- Read the input text (XML or text file) and extract the main information (drops the headers). The tokenization stage is finished when the main information was separated into words.

Table 1: The computed values of numeric data, binary data and probability

Word	Numeric data	Binary data	probability
ةالرابع	1449	10110101001	0.1875
ةالفنية	1311	10100011111	0.0625
ةالتحف	1253	10011100101	0.125
هبهذ	866	1101100010	0.0625
بالإعجا	1442	10110100010	0.125
فرط	646	1010000110	0.0625
مذ	455	111000111	0.0625

- Compute the probability of each word in the text to be used next in the input vector as one of the significant bit.
- Words are converted into numerical data format. This numerical data is later encoded into a binary form that requires lesser memory locations as compared to the old encoding methods.
- The category names (output vector) are converted into a binary data, which has several significant bits, by the use of algorithm.

For instance, suppose we have the following words as input text. The computed values of numeric data, binary data and probability are elucidated in Table 1.

RESULTS AND DISCUSSION

Section 6 represents the POS tagging experiments and the result of the Arabic language as performed by using the multilayer perceptron neural network. The MLP network is designed and implemented using the NeuroSolutions package. The experiments conducted involved the use of the Arabic tagset and Arabic corpus, consisted of 50 000 words[21].

Furthermore, the researcher did take into consideration the segmentation of a sentence into words using an innovation method, which it takes the XML file or text file as input and extract the important information as output. The objective of this study is to automated the problem of POS tagger and identify the correct POS tag for each word in the input text.

There are three categories involved in the input data sets; the training data, cross validation data and test data. While the network is being trained with the training set, the cross validation computes the error in a test data sets at the same time.

The networks performance can be measured using various techniques[22]. The "Mean Squared Error" MSE is usually used for this purpose and is two times the average cost. It is computed as follows:

$$MSE = \frac{\sum_{j=0}^{P} \sum_{i=0}^{N} (d_{ij} - yi_j)^2}{NP} \quad (9)$$

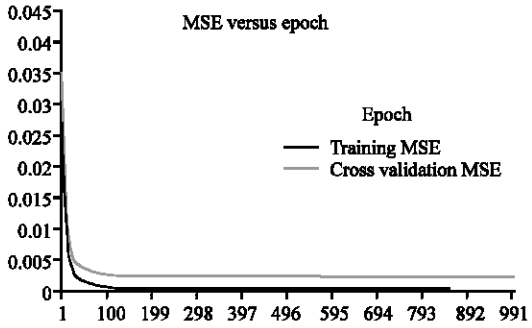

Fig. 2: Final MSE versus epochs

Table 2: Minimum and final MSE

Best networks	Training	Cross validation
Epoch #	1000	1000
Minimum MSE	0.000109798	0.002132578
Final MSE	0.000109798	0.002132578

Where, P is the number of output processing elements,

N is the number of exemplars in the data set.

y_{ij} is the network output for exemplar i at the processing element j.

d_{ij} is the desired output for exemplar i at the processing element j.

The size of the mean square error (MSE) can be used to determine how well the network output fits the desired output Since the beginning of the run, the lowest cost reported by the Error Criterion is called the minimum MSE. The stop criteria for supervised training are usually based on the mean squared error (MSE). Most often, the training is set to terminate when the MSE drops to some threshold. Another approach is to terminate when the change in the error between epochs is less than some threshold. The result of the final MSE versus Epoch is depicted in Fig. 2.

The minimum and final MSE values of the network output are depicted in Table 2.

In order to carefully determine the effect channels that each of the network inputs has on the network output, a crucial consideration must be taken. This provides feedback as to which input channels are the most significant. This will reduce the size of the network, which in turn reduces the complexity and the training times.

Sensitivity analysis is a method for extracting the cause and effect relationship between the inputs and outputs of the network. The basic idea is that the inputs to the network are shifted slightly and the corresponding change in the output is reported as either a percentage or a raw difference.

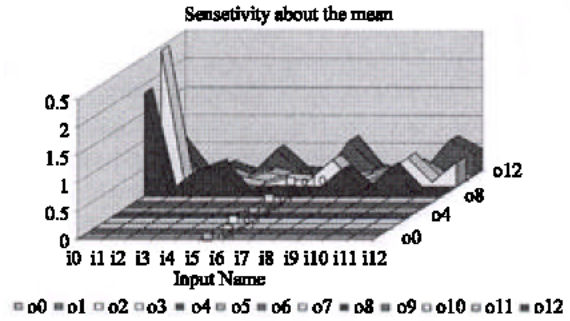

Fig. 3: The output of sensitivity test

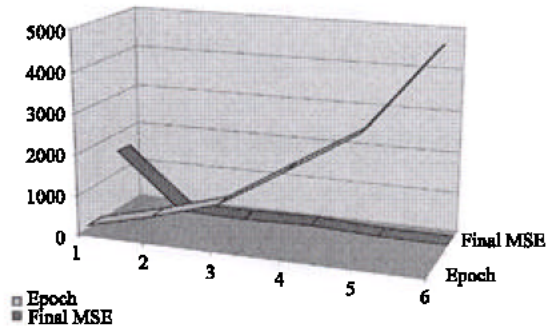

Fig. 4: Final MSE with different epochs

Table 3: Final MSE with different epochs

Epochs	Final MSE
100	0.00150795
500	0.00012305
1000	0.0001098
2000	0.00010621
3000	0.00006795
5000	0.00005495

The sensitivity test shows that the most significant output channels in our network are O8, O9, O10, O11 and O12. The output of the sensitivity test is illustrated in Fig. 3.

For the purpose of the network stability, the average of the MSE with different epochs is computed. Figure 4 illustrates the average of the MSE for the six runs and their epitomized results are shown in Table 3.

CONCLUSION

A careful comparison study is required, as the variables used in this study do not match those in the studies undertaken previously. However, comparing the MLP-tagger with other existing taggers is a difficult task because its accuracy relies on numerous parameters such as the language complication (ambiguous words and phrases), language type (English, Arabic, Chinese, etc.), training data magnitude, tag-set size and the evaluation measurement criteria. The tag-set size has a great impact on the tagging process. Table 4 summarizes the comparative information.

Table 4: The comparison results

	Schmid (21)	NLP (1)	Khoja (9)	Diab M (4)	Our tagger
Method	NN	NN	Rule-base and statistical	SVM-Tree Bank	RNN
Language	English	English	Arabic	Arabic	Arabic
Corpus size	$4.5x10^b$	$0.156622x\ 10^b$	$0.05x10^b$	4519 sentences	$0.05x10^b$
Training data size	44.4%	85%	100%	80%	10%
Tag size	48	48	131	19	131
Accuracy	96.22%	90.40%	90%	95.49%	99.99%

The proposed tagger has succeeds to identify and recognized all the tagging categories including the Punctuations category. When a small Arabic data set was used for the training purposes, this tagger had a tagging accuracy of 99.99%.

The study demonstrated that the MLP tagger could solve the problems in automatic tagging the Arabic POS. This new approach is proven to be highly accurate with a lesser processing time and higher speed of word tagging.

FUTURE WORK

More emphasis on solving the problems of manual pre- tagging should be taken into consideration in the future studies. To perform a fully automatic text tagging, an unsupervised learning approach should be implemented.

REFERENCES

1. Al-Sulaiti's and L. Corpus, 2005. Last visit 0n 14-5-2005, http://www.comp.leeds.ac.uk/latifa/ research. htm.
2. Mahtab, N. and C. Khalid, 2005. Survey on Arabic language resources and tools in the Mediterranean countries. Last access on 20-02-2005. http://www.nemlar.org/ Publications/Nemlarreport-ind-needs_web.pdf.
3. Brill, E., 1992. A simple rule-based part-of-speech tagger. In Proceedings of ANLP-92, 3rd Conference on Applied Natural Language Processing, Trento, IT, pp: 152-155.
4. Diab, M., K. Hacioglu and D. Jurafsky, 2004. Automatic tagging of Arabic text: From Raw Text to Base Phrase Chunks, in Proceedings of HLT-NAACL-04.

5. Brill, E., 1994. Some advances in transformation based part of speech tagging. In Proceedings of the Twelfth International Conference on Artificial Intelligence (AAAI-94), Seattle, WA.
6. Ma, Q., M. Sun and H. Isahara, 1998.A multi-neuro tagger applied in chinese texts. In Proceedings of 1998 International conference on Chinese information processing, Beijing, pp: 200-207.
7. Ma, Q., K. Uchimoto, M. Murata and H. Isahara, 1999. Elastic neural networks for part of speech tagging. In Proceedings of IJCNN'99, Washington, DC. pp: 2991-2996.
8. Ma, Q. and M. Masaki, 1999. Part of speech tagging with mixed approaches of neural networks and transformation rules, NLPRS'99 Workshop on Natural Language Processing and Neural Networks, Beijing, China.
9. Marques, N.C. and P.L. Gabriel, 1996. Using neural nets for portuguese part-of-speech tagging, Proceedings of the Fifth International Conference on The Cognitive Science of Natural Language Processing, Dublin City University, Ireland.
10. Schmid, H., 1999. Part-of-speech tagging with neural networks. In Proceedings of COLING-94, Kyoto, Japan, pp: 172-176.
11. Beesley, K., 2001. Finite-state morphological analysis and generation of arabic at xerox research: Status and Plans, ACL, Arabic NLP Workshop, Toulouse.
12. Khoja, S. 2001. APT: Arabic part-of-speech tagger. In Proceedings of the Student Workshop at the Second Meeting of the North American Chapter of the Association for Computational Linguistics (NAACL2001), Carnegie Mellon University, Pennsylvania.
13. Ahmed, 2002. Application of multilayer perceptron network for tagging parts-of-speech, proceedings of the Language Engineering Conference (LEC'02),IEEE.
14. Attia, M., 2000. A Large-scale computational processor of The arabic morphology and applications., MSc. thesis, Dept. of Computer Engineering, Faculty of Engineering, Cairo University.
15. Lippman, R., 1987. An introduction to computing with neural nets. IEEE Trans. ASSP Magazine, pp: 4-22.
16. Makhoul, J., 1992. Pattern recognition properties of neural networks. Proc. IEEE Workshop on Neural Networks for Signal Processing, pp: 173-187.
17. Sejnowski, T. and C. Rosenberg, 1987. Parallel networks that learn to pronounce English text. Complex Systems, pp: 145-168.
18. Elman, J.L., 1990. Finding structure in time, Cognitive Science, pp: 179-211.
19. Rumelhart, D.E., G.E. Hinton and R.J. Williams, 1986. Learning Internal Representations by Error Propagation. In D.E. Rumelhart and J.L. McClelland, editors, Parallel Distributed Processing, volume 1, Cambridge, USA, 1986. MIT-Press, pp: 318-362.
20. Vlaseva, S., 1999. Neural network modeling for Part-Of-Speech disambiguation on bulgarian sentences, Computational Linguistics and Represented Knowledge (CLaRK), International Summer School, University of Tübingen, Germany.
21. Khoja, S., R. Garside and K. Gerry, 2001. An Arabic tagset for the morphosyntactic tagging of arabic, Corpus Linguistics, Lancaster University, Lancaster, UK,
22. Pierce, D., 2003. Cost-effective machine learning strategies for shallow parsing. Ph.D. thesis, Cornell University.