JKAU: Eng. Sci., **Vol. 20** No.1, pp: 29-43 (2009 A.D. / 1430 A.H.)

A Simple Time Alignment Algorithm for Spoken Arabic Digit Recognition

Yousef Ajami Alotaibi

Computer Eng. Dept., College of Computer & Information Sciences, King Saud University, P.O. Box 57168, Riyadh 11574, Saudi Arabia
yalotaibi@ccis.ksu.edu.sa

Abstract. The problem associated with spectral sequence comparison for speech comes from the fact that different acoustic renditions, or tokens, of the same speech utterance are seldom realized at the same speed across the entire utterance. In this paper a simple and effective time alignment was introduced for spoken Arabic digit recognition systems. We meant with simplicity here not only in its need for low computational power, but also simplicity to understand, to implement, and to explain to others. While high power computers are available today, time alignment algorithms, such as dynamic time warping algorithm and hidden Markov models need relatively high CPU time, which should be reserved for other complicated tasks. This algorithm has a high accuracy rate considering the very limited number of frames taken from input utterances to be used in training or testing. An artificial neural network based speech recognition system was designed and tested with automatic Arabic digit recognition to test this time alignment algorithm. The system is an isolated whole word speech recognizer and it was implemented in a multi-speaker mode (*i.e.*, the same set of speakers was used in both the training and testing phases). During recognition process, digitized speech was cleaned of noise, then the signal was pre-emphasized and it was windowed and blocked by Hamming window, the time alignment algorithm was used to compensate for the differences in the utterance length and misalignments between phonemes. Frames features were extracted using MFCC coefficients to reduce the amount of the information in the input signal. Finally, the neural network classified the unknown digit. This recognition system achieved 99.48% correct digit recognition while using only seven frames in the time alignment algorithm.

Keywords: Alignment, ANN, Arabic, Speech, Recognition, Digits.

1. Introduction

1.1 Arabic Language

Arabic is a Semitic language, and it is one of the oldest languages in the world today. It is the fifth widely used language nowadays. Arabic is the first language in the Arab world and Arabic alphabet is used in several other languages, such as Persian and Urdu[1, 2].

Standard Arabic has 34 phonemes, of which six are vowels, and 28 are consonants[3]. A phoneme is the smallest unit of sound that indicates a difference in meaning, word, or sentence. The Arabic language has fewer vowels than English: three long and three short vowels, while American English has at least twelve vowels[4]. Arabic phonemes contain two distinctive classes, which are named pharyngeal and emphatic phonemes. These two classes can be found only in Semitic languages like Hebrew[3, 5].

1.2 Spoken Digits Recognition

Automatic recognition of spoken digits is one of the challenging tasks in the field of computer speech recognition. A spoken digit recognition process is needed in many applications that use numbers as input.

Arabic has seen a relatively limited number of research efforts compared to other languages such as English and Japanese. A short survey of literature, will confirm the lack of the research papers and references on Arabic speech recognition in general. In fact a very limited number of Arabic digit recognition papers is available to the public. Some research has been conducted on Arabic digit recognition. In 1985, Hagos[6] and Abdullah[7] separately reported Arabic digit recognizers. Hagos designed a speaker-independent Arabic digit recognizer that used template matching for input utterances. His system is based on the LPC parameters for feature extraction and log likelihood ratio for similarity measurements. Abdullah developed another Arabic digits recognizer that used positive-slope and zero-crossing duration as its feature extraction algorithm. He reported a 97% accuracy rate. Both systems of these are isolated-word recognizers in which template matching is used.

The Arabic digits zero to nine (sĕfr, wâ-hĕd, ‘aâth-nāyn, thâ-lă-thâh, ‘aâr-bâ-‘aâh, khâm-sâh, sĕt-tâh, sûb-‘aâh, thâ-mă-nĕ-yah, and tĕs-

âh) are polysyllabic words except the first one, zero, which is a monosyllable[3]. The allowed syllables in Arabic language are: CV, CVC, and CVCC where V indicates a (long or short) vowel while C indicates a consonant. Arabic utterances can only start with a consonant[3]. Table 1 shows the ten Arabic digits along with the way to pronounce them and the number and types of syllables in every spoken digit.

Table 1. Arabic digits.

Digit	Arabic writing	Pronunciation	Syllables	IPA representation	No. of syllables
1	و احد	w â-hĕd	CV-CV	wa :-?id	2
2	أثنيـن	‘aâth-nāyn	CVC-CV	?aθ-ni:n	2
3	ثلاثه	thâ-lă-thâh	C -CV-CVC	θa-la-θah	3
4	أربـعه	‘aâr-bâ-‘aâh	CV -CV-CVC	?ar-ba-ʕah	3
5	خمسه	khâm-sâh	CVC-CV	xam-sah	2
6	سته	sĕt-tâh	CVC-CV	sit-tah	2
7	سبـعه	sûb-‘aâh	CVC-CV	sab-ʕah	2
8	ثمانيه	thâ-mă-nĕ-yah	CV-C -C -	θa-ma-ni-jah	4
9	تسعه	tĕs-âh	CVC-	tis-ʕah	2
0	صفر	sĕf	CVC	şif	1

1.3 Neural Networks

Artificial neural networks (ANNs) have for many years been thought to hold the potential of achieving human-like performance in the automatic speech recognition. These models are composed of many nonlinear computational elements operating in parallel in patterns similar to the biological neural networks[8]. ANN has been used extensively in speech recognition during the past two decades. The most beneficial characteristics of ANNs for solving speech recognition problems are fault tolerance and nonlinear properties[9].

ANN models are distinguished by network topology, node characteristic, and training or learning rules. One of the important models for neural networks are multilayer perceptrons (MLPs), which are a feed-forward network with zero, one, or more hidden layers of nodes between the input and output nodes[8]. The capabilities of the MLP stem from the nonlinearities used with its nodes. Any MLP network must consist of one input layer (not computational, but source nodes), one output layer (computational nodes), and zero or more hidden layers (computational

nodes) depending on network sophistication and the application requirements[9].

1.4 Time Alignment and Normalization

The same word spoken by different speakers, or the same speaker at different times, is often of different time duration. An utterance (digit in our case), which is to be recognized is not a steady sound but is complex and involves a sequence of short-time acoustic representations (phonemes)[10]. The pattern-comparison technique for speech recognition must be able to compare sequences of acoustic features[11].

When comparing different tokens of the same utterance, speech rate variation as well as duration variation should not contribute to the (linguistic) dissimilarity score. Thus there is a need to normalize speaking rate fluctuations in order for the utterance comparison to be meaningful before a recognition decision can be made. The need for time alignment arises not only because different utterances of the same word will generally be of different durations, but also because phonemes within words will also be of different durations across the utterance[12, 13].

The popular time alignment and normalization scheme used with the pattern-comparison technique is dynamic time warping (DTW)[10]. This technique allows parts of a word to be stretched or compressed differently than other parts. In fact this is a non-linear time alignment technique, which is widely used and based on dynamic programming[10]. This method results in minimum cost or shortest path problems[14]. Unfortunately this technique needs a very expensive CPU computation and is not economic for small vocabulary recognition systems such as digit recognition systems[1, 13]. Also this technique is very complicated if used with ANNs.

In this paper, a simpler and effective time alignment algorithm was introduced and tested. A system based on an ANN system was designed and used in conjunction with spoken Arabic digits to investigate this new algorithm. This study concentrated on multi-speaker mode using artificial neural networks, namely multilayer perceptrins (MLPs) with the concentration on these ten words as isolated utterances.

2. Experimental Framework

2.1 System Overview

A complete speech recognition system based on neural networks was developed to carry out the goals of this research. This system was partitioned into several modules according to their functionality as shown in Fig. 1. First is the digital signal processing front-end module, whose functions are speech acquisition through a microphone, filtering, and sampling. A band-pass filter with cut-off frequencies of 100Hz and 4.8kHz was used to filter the speech signal before processing. The sampling rate was set to 10kHz with 16-bit resolution for all recorded speech tokens.

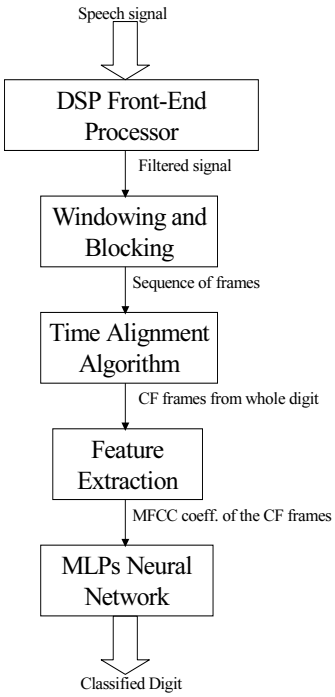

Fig. 1. System block diagram.

Due to the concentration on the main goals of this research, a manual endpoint detection method was also used to separate speech from silent portions of the signal. Endpoint detection is considered as a big and challenging task in digital speech processing which made us to select manual methods. Endpoint detection also located the beginning and end

points of the spoken word (digit)[15]. After that, blocking speech into frames module partitioned the whole speech utterance to a number of frames depending on our window and utterance duration. A time alignment algorithm was applied to a sequence of frames as detailed in the following subsection.

Linear predictive coding (LPC) coefficients were computed for sequential frames 64 points (6.4 ms) apart. In each case, a 256-point Hamming window was used to select the data points to be analyzed[16]. The linear predictive coding module calculates ten mel-frequency cepstrum coefficients (MFCCs), with LPC order (p=10), for each frame in the spoken utterance, thus 11 MFCC coefficients were extracted from each frame. For MFCC computations, 20 triangular band-pass filters were considered in the feature extraction subsystem as in[17].

A fully connected feed-forward multilayer perceptron (MLP) network was used to recognize the unknown spoken digit. All MLP neurons used logistic non-linearities and a back-propagation training algorithm[9]. The network consisted of 143 nodes in the input layer (source nodes or the beginning layer). The number of nodes in this layer depends on the number of MFCC coefficients for every frame and the number of considered frames in the whole token that is currently applied to the input layer.

The MLP network contains two hidden layers with 40 nodes in the first hidden layer and 15 nodes in the second hidden layer. The output layer consists of 10 neurons. This implies that the network consists of a total of four layers, namely, input, first hidden, second hidden, and finally output layer with a number of nodes equal to 143, 40, 15, and 10, respectively. Each neuron in the output layer should be *on* or *off* depending on the applied digit in the input layer. For the normal and intended situation, only one node should be *on* while all others should give an *off* state if the applied utterance is one of the ten Arabic digits, otherwise, all neurons should give an *off* state.

2.2 The Proposed Time Alignment Algorithm

The time alignment algorithm is very simple and straightforward; yet, it achieved a high accuracy rate when used with an Arabic digit recognition system. To explain this algorithm, consider the following

parameters: (number of) considered frames (CF), start point (SP), and end point (EP). These parameters must be adjusted before running the algorithm. CF should take only positive integer grater than 2, while SP and EP should take values between 0 and 1 inclusive.

CF is the number of frames that would be considered by the time alignment algorithm and the rest of utterance's frames are totally ignored. Start point (SP) is used to compute the first frame to be considered by the system. Similarly, end point (EP) is used to compute the last one. To identify the first frame in a given utterance, we multiply SP by the total number of frames contained in the whole utterance under processing and, likewise, we multiply EP by total frames to get last frame. In both cases we must round multiplication outcomes to the nearest integer. SP must have a value less than EP.

To illustrate, if SP=0 and EP=1 then the first and last frames in the given utterance are the same as the first and last frames to be considered by the alignment algorithm. After computing the first and last frames to be picked up by the algorithm, the system must get CF-2 frames evenly distributed between the two previously located frames using EP and SP, hence we have CF total of frames. Figure 2 shows an example of the algorithm with specific values to its parameters. In this example we assumed that CF=5, SP=0.05, EP=0.95, and total number of frames in the considered utterance is 80. Depending on the algorithm, the first frame to be taken from the utterance is the fourth one (*i.e.*, 80×0.05) and the last one is the seventy-sixth (*i.e.*, 80×0.95). The remaining three frames should be evenly located between the first and last frames; specifically we have to consider the twenty-second, the fortieth, and fifty-eighth frames. This algorithm is outlined in Fig. 3 in the form of a flowchart.

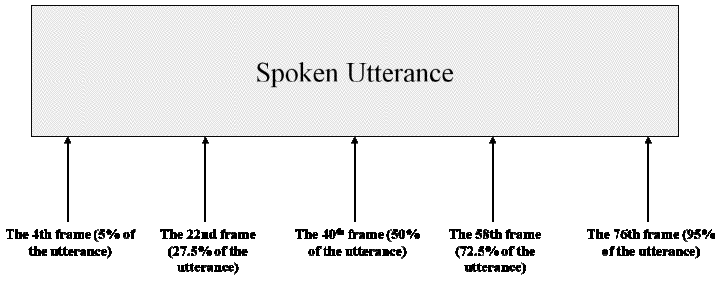

Fig. 2. Illustration example (with CF=5, SP=0.05, and EP=0.95).

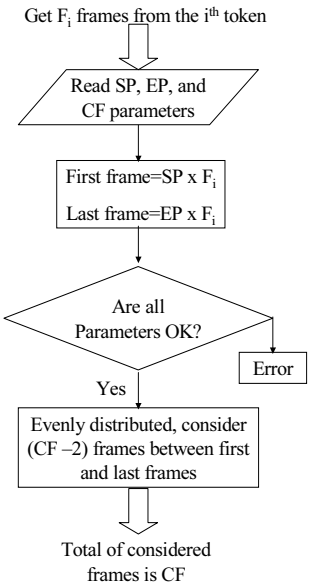

Fig. 3. Time alignment algorithm flowchart.

One of the drawback of this algorithm is that its accuracy is dependent on an accurate endpoint detection scheme applied to utterances. False endpoints detection may cause misleading results to this time alignment algorithm. This algorithm uses a linear expansion or compression to input patterns (utterances) and we have the fact those vowels and other voiced phonemes are more variable in duration, more than stop or plosive consonants; nevertheless, it demonstrated excellent results with Arabic digit recognition.

2.3 Database

An in-house database was created from all ten Arabic digits. Seventeen male Arabic native speakers were asked to record all digits ten times. Hence, the database consisted of 10 repetitions of every digit produced by each speaker for a total of 1,700 tokens. All samples for a given speaker were recorded in one session. During the recording session, each utterance was played back to ensure that the entire digit was included in the recorded signal. All the 1,700 (10 digits × 10 repetitions × 17 speakers) tokens were used for training and testing phases.

3. Results

3.1 System Overall Performance

To train the system, the first and second repetitions of each digit that were uttered by all speakers were used. Thus, the total tokens considered for training was 340 (17 speakers × 2 repetitions × 10 digits). For testing mode, tokens 3 through 10 were used in recognition phase (testing mode), which gave total of 1,360 tokens.

Table 2 shows the accuracy of the system for digits individually in addition to the system's overall accuracy for SP= 0.05, EP=0.95, and CF=9. Depending on testing database set, the system attempted to recognize 136 tokens for every digit where the total number of tokens was 1,360. The overall system performance was 99.48%, which is reasonably high. The system failed to recognize only 7 tokens out of the 1,360 total tokens. Digits 1, 5, 6, and 9 got 100% recognition rate. Even though, the database size was small (only the ten spoken Arabic digits), the system showed a high performance regardless of dialect variations and the fact that we considered multi-speaker mode in contrast to speaker-dependent (one speaker only trains and uses the system) mode. In addition to this, our time-alignment algorithm was very simple and straightforward.

Table 2. Confusion matrix.

	one	**two**	**three**	**four**	**five**	**six**	**seven**	**eight**	**nine**	**zero**	**(%)**
one	136	–	–	–	–	–	–	–	–	–	100
two	1	135	–	–	–	–	–	–	–	–	99.3
three	–	–	134	1	–	1	–	–	–	–	98.5
four	–	1	–	135	–	–	–	–	–	–	99.3
five	–	–	–	–	136	–	–	–	–	–	100
six	–	–	–	–	–	136	–	–	–	–	100
seven	–	–	–	1	–	–	135	–	–	–	99.3
eight	–	1	–	–	–	–	–	135	–	–	99.3
nine	–	–	–	–	–	–	–	–	136	–	100
zero	1	–	–	–	–	–	–	–	–	135	99.3
Total											99.5

3.2 Time Alignment Algorithm Evaluation

In general the algorithm gave a high performance level in recognizing Arabic digits. The performance exceeds 99% for values of CF=8 with the appropriate values of SP and EP as can be seen from Fig.

4. For all different values of CF, the algorithm gave higher performances in the case of (SP=0, EP=1) and (SP=0.6, EP=0.94) and gave lower performances in the case of (SP=0.24, EP=0.76). This implies that most of first and last quarters of Arabic digits are important for comparisons during system training and testing tokens among different digits. At the beginning of training the ANN, free parameters (weights and biases) are initialized with small random values, hence with the same parameters and different run times we get different system performances (due to randomness of free parameters) but the differences are very small.

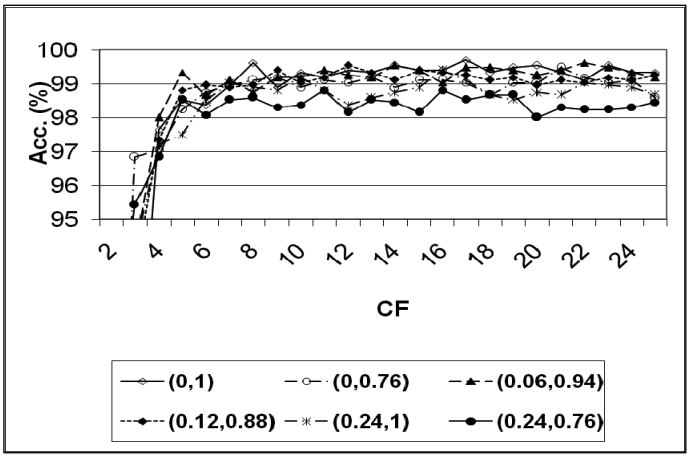

Fig. 4. System accuracy for different CF and various (SP,EP).

If we select CF greater than 8, SP less than 0.1, and EP greater than 0.9 we will end up with greater than 99% system performance, however, if we increase CF to more than 10, the increment of improvement (if it is not zero), is so small that it does not justify using such. This can be seen in Fig. 3, 4 & 5. It has been noticed that the first quarter of Arabic digits have more importance than their last quarter because SP affect the performance sharply if it is decreased more than 0.15, as can be seen from Fig. 5 and 6. In other words, ignoring large parts of the first quarter degrades the performance of the system more than is caused by the ignoring more parts of the last quarter. For good performance, SP value should be less than 0.1 and EP should be greater than 0.9. This means we must discard less than 20% as total from both ends.

This proposed algorithm is the fastest one if compared to other algorithms such as DTW based ones[1, 18]. This algorithm requires significantly less computation and is simpler to implement and to

understand if compared to the popular DTW related algorithms. Programming wise, the CPU time required for this algorithm is just the time required to create and search an array with length equal to number of frames in the processed utterance as discussed above. On the other hand, the CPU time required for DTW based algorithm is a function dependent on two-dimensional matrix with lengths equal to both the width of the reference and the width of the test utterances as discussed in[18]. This used DTW means that the CPU will take a considerable computation time for long utterances.

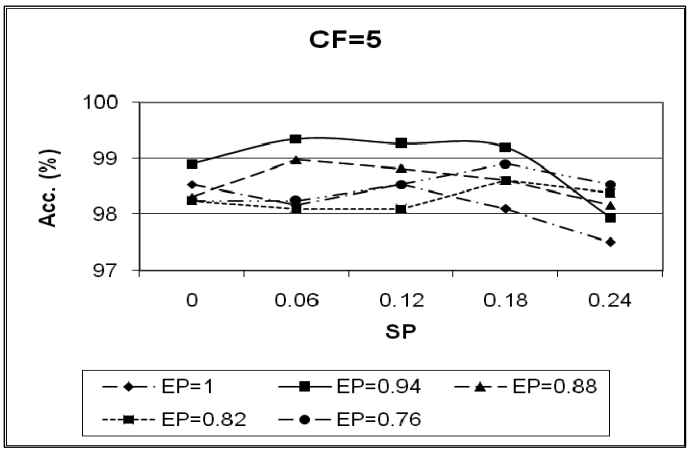

Fig. 5. System accuracy for CF=5 and different (SP,EP) values.

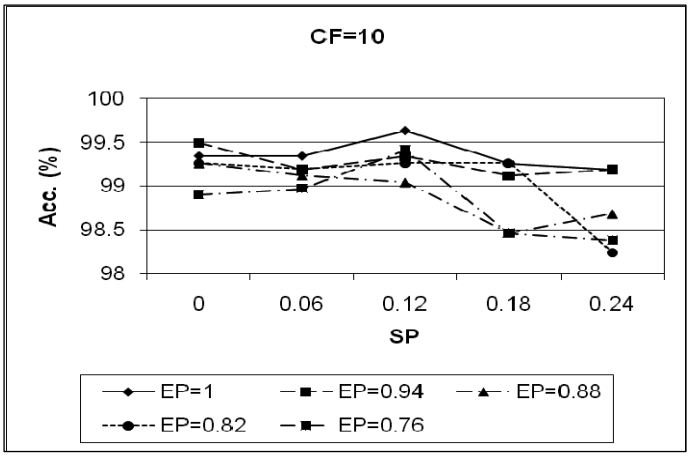

Fig. 6. System accuracy for CF=10 and different (SP,EP) values.

4. Discussions and Conclusion

This paper discussed a new simple time alignment algorithm for an Arabic spoken digit recognizer. There is a need to emphasize some points in this research. First, the research discussed the time alignment algorithm not neural networks nor spoken Arabic digit characteristics. Second, the high system performance and effectiveness gained is the best among the current found in literature in spoken Arabic digit recognition. Third, this high performance with the new proposed time alignment algorithm may be linked to some characteristics of Arabic digits such as the high degree of orthogonality between each pair of them. This issue will be investigated in upcoming research, which will depend on content of language phonemes and spectrograms of Arabic digits.

Fourth, someone will notice the limitation of the database used in the training and testing phases compared to databases used with languages of industrial countries such as English and Japanese, nevertheless we used a multi-speaker mode. Fifth, the algorithm assumes a perfect endpoint algorithm because the initial and terminal of any utterance carry distinguishing and valuable information for the recognition process. Sixth, in Fig. 1 we can move the time alignment subsystem before the front-end signal (DSP) subsystem, hence we can reduce the amount of CPU time needed for the DSP subsystem to minimum because this subsystem will manipulate only CF frames instead of the whole utterance. This concept is logical and easily realizable because unused frame can be discarded from the front-end subsystem after discovering them by the aid of this novel alignment algorithm. Therefore more benefits may be gained by saving more system resources.

To conclude, this algorithm was tested on a MLP neural network based recognizer. The system consists of a DSP front-end processor module, a blocking and framing module, a time alignment module, a feature extraction module, and an MLP network. A database of 1,700 tokens was created for 17 Arabic native speakers. The overall system performance was 99.49% with CF set to 9. This algorithm demonstrated a high performance with Arabic digit recognition and an accuracy above 99% for CF equal or greater than 7 and appropriate values of SP and EP. Increasing a CF to more than 10 results in insignificant gain other than higher use of CPU time and memory. If we ignore more than 10% from the beginning of the utterance or more than 10% from the end of the

utterance then the performance degrades accordingly, but the effect on performance caused by the beginning side is more than that of the end side.

References

[1] **Kirchhoff, K., Bilmes, J., Das, S., Duta, N., Egan, M., Gang J., Feng H., Henderson, J., Daben L., Noamany, M., Schone, P., Schwartz, R.** and **Vergyri, D.,** "Novel Speech Recognition Models for Arabic: Johns-Hopkins University Summer Research Workshop 2002," Final Report, *http://www.clsp.jhu.edu/ws02.html*, accessed on 19 Mar. (2008).

[2] **Al-Zabibi, M.**, "An Acoustic-phonetic Approach in Automatic Arabic Speech Recognition," *The British Library in Association with UMI* (1990).

[3] **Alkhouli, M.,** "*Alaswaat Alaghawaiyah*," Daar Alfalah, Jordan (1990) (in Arabic).

[4] **Deller, J., Proakis, J.** and **Hansen, J.**, "*Discrete-Time Processing of Speech Signal*," Macmillan (1993).

[5] **Elshafei, M.,** "Toward an Arabic Text-to -Speech System," *The Arabian Journal for Science and Engineering,* **16**(4B): 565-83, Oct. (1991).

[6] **Hagos, E**., "*Implementation of an Isolated Word Recognition System*," UMI Dissertation Service (1985).

[7] **Abdulah, W.** and **Abdul-Karim, M.**, "Real-time Spoken Arabic Recognizer," *Int. J. Electronics*, **59**(5): 645-648(1984).

[8] **Lippmann, R.**, "Review of Neural Networks for Speech Recognition," *Neural Computation*: 1-38, MIT press (1989).

[9] **Haykin, S.**, "*Neural Networks: A Comprehensive Foundation*," 2*nd* Ed., Prentice Hall (1999).

[10] **Gangashetty, S.**, **Csekhar, S.** and **Yegnanarayana, B.,** "Extraction of Fixed Dimension Patterns from Varying Duration Segments of Consonant-Vowel Utterances", *Proceedings of the International Conference on Intelligent Sensing and Information Processing* (2004).

[11] **Rowden**, **C.**, "*Speech Processing*," McGraw-Hill,1992.

[12] **Rabiner, L.** and **Juang, B.,** "*Fundamentals of Speech Recognition*," Prentice Hall 1993.

[13] **Watanabe, K.** and **Sugiyama, M.**, "Automatic Caption Generation for Video Data Time Alignment between Caption and Acoustic Signal", *IEEE Third Workshop on Multimedia Signal Processing,* Sept. (1999).

[14] **Sakoe, S.** and **Chiba, S.**, "Dynamic Programming Algorithm Optimization for Spoken Word Recognition", *IEEE Transactions on Acoustic, Speech, and Signal Processing*, **26**: 43-49(1978).

[15] **Rabiner, L.** and **Samber; M.**, "An Algorithm for Determining the Endpoints of Isolated Utterances," *The Bell System Technical Journal*, **54**(2): 297-315(1975).

[16] **Nocerino, N.**, **Soong F., Rabiner, L.** and **Klatt, D.,** "Comparative Study of Several Distortion Measures for Speech Recognition," *Speech Communication*, **4**: 317-31(1985).

[17] **Davis, S.** and **Mermelstein P.**, "Comparison of Parametric Representations for Monosyllabic Word Recognition in Continuously Spoken Sentences," *IEEE Trans. on Acoustic, Speech, and Signal Processing*, Vol. ASSP-28, No. 4, Aug. (1980).

[18] **Yfantis, E. A.**, **Lazarakis, T.** and **Angelopoulos, A.**, "On Time Alignment and Metric Algorithms for Speech Recognition," *Proceedings of IEEE International Conference on Information Intelligence and Systems*, Oct. (1999).

خوارزمي مبسط للاصطفاف الزمني للتعرف الآلي على الأرقام العربية المنطوقة

يوسف عجمي العتيبي

قسم هندسة الحاسب، كلية علوم الحاسب والمعلومات، جامعة الملك سعود، ص. ب. ٥٧١٦٨ الرياض ١١٥٧٤ – المملكة العربية السعودية

المستخلص. المشكلة الملازمة لمقارنة سلسلة طيفية من الكلام يأتي من حقيقة اختلاف طول الكلمات المنطوقة، ومواقع، وطول الفونيمات لنفس الكلمة، والمتكلم، وصعوبة توافقها إضافة إلى اختلافها الفونتكي. يعرض في هذا البحث خوارزمي مبسط للاصطفاف الزمني للتعرف الآلي على الأرقام العربية المنطوقة. يقصد بكلمة مبسط هنا أي أنه يحتاج إلى زمن أقل للمعالج لإنجاز الحسابات المطلوبة، وكذلك فيه شيء من السهولة للفهم، والتصميم، والتطبيق. مع أن الحاسبات السريعة متوفرة الآن، إلا أن الطرق المعتادة مثل خوارزم الالتفاف الزمني المتغير. يحتاج نموذج ماركوف الخفي إلى جهد كبير لمعالج الكمبيوتر، وهذا صعب جدا، وبخاصة في حالة التطبيقات الخاصة بالزمن الحقيقي. هذا الخوارزمي أعطى دقة عالية مع استخدام عدد محدود من الإطارات الزمنية المأخوذة من الكلمة المنطوقة لاستخدامها في مرحلة التدريب أو الاختبار. يعتمد النظام على الشبكات العصبية الاصطناعية، وذلك للاختبار، والتعرف الآلي على الأرقام العربية المنطوقة، وذلك لعمل فحص، واختبار لهذا الخوارزمي المبسط للاصطفاف الزمني. يتعامل النظام مع الكلمات المنطوقة في معزل عن بعضها، وأخذ كل كلمة بمجملها كاملة كوحدة كاملة للتعرف

عليها آليا، وذلك لمجموعة محددة، ومعروفة مسبقا من المتكلمين. عملية معالجة الكلمة المنطوقة تمر بعدد من العمليات الأساسية في أي نظام مشابه مثل عمليات استخلاص الصفات المميزة للكلام، وعمل إطارات من الإشارة، والاصطفاف الزمني، وغيرها من العمليات الضرورية لإزالة أية بيانات في الإشارة غير ضرورية. الصفات المميزة للكلام اعتمدت على مميزات عوامل (MFCC). بلغت دقة التعرف لهذا النظام باستخدام هذا الخوارزم للاصطفاف ٩٩,٤٨٪، وذلك عند استخدام سبعة إطارات فقط من الرقم المنطوق.