Building Quranic reader voice interface using sphinx toolkit

[1]Yacine Yekache, [1]Belkacem Kouninef, [1]Yekhlef Mekelleche, [2]Senouci Mohamed

[1]Institut National des Télécommunications et des Technologies de l'Information et de la Communication
INTTIC, Laboratoire LaRATIC – Oran, ALGERIA
[2]Departement Informatique & Mathematique, Université Oran, ALGERIA
Corresponding author: yyekache@ito.dz

Abstract: In this paper we investigated the building of a quranic reader controlled by speech. This system is based on open source CMU Sphinx toolkit, which represents an HMM speech recognition toolkit built for English language, and tuned by us to support Arabic. For this purpose, we have collected a speech corpus called "Quranic Reader Command and Control Corpus" QRCCC from several speakers using web Java applet to train the HMM acoustic model. The performances of this model were tested by varying the training parameters "the number of Gaussians Mixtures and Senones" using Pocket Sphinx decoder. The model with the best parameters was chosen to be integrated in a demo application built using Sphinx-4 to perform recognition.
[Yekkache Y, Kouninef B, Mekelleche Y. Senouci M. **Building Quranic reader voice interface using sphinx toolkit.** *J Am Sci* 2013;9(11):473-479]. (ISSN: 1545-1003). http://www.jofamericanscience.org. 60

Keywords: arabic speech recognition; quranic reader; speech corpus; HMM; acoustic model.

1. Introduction

Communication is an essential part of human life. If communication is disturbed or impossible, the consequences are loneliness and isolation. It is well known that speech plays a key role in communication and it explains why humans also want to have speech as a means of communication/interaction, In fact, speech communication with computers, PCs, and household appliances is envisioned to be the dominant human-machine interface in the near future. In spite of the tangible success of this technology for the English language and other languages, unfortunately there is a lack of these applications for Arabic language. For this purpose we decided to contribute by developing a command and control application which is "a Quranic reader controlled by speech" using CMU Sphinx toolkits.In this work we will show the theory behind this type of applications and describe the process in which the system should be built, and explains components used to perform this process

The most dominant approach for ASR system is the statistical approach Hidden Markov Model (HMM), trained on corpora that contain speech resource from a large number of speakers to achieve acceptable performance[13]; unfortunately there is a lack of this corpus for Arabic language. In this work we collected a new corpus called Quranic reader command and control which we used to create an acoustic model using "sphinx train", then tune the model parameters to achieve good accuracy.

2. Quranic reader and arabic language

Quranic reader

Quran is the central religious text of Islam, which is the verbatim word of God and the Final Testament, following the Old and New Testaments. It is regarded widely as the finest piece of literature in the Arabic language. The Quran consists of 114 chapters of varying lengths, each known as a sura. Chapters are classed as Meccan or Medinan, depending on when (before or after Hijra) the verses were revealed. Chapter titles are derived from a name or quality discussed in the text, or from the first letters or words of the sura.

There is a crosscutting division into 30 parts of roughly equal division, ajza, each containing two units called ahzab, each of which is divided into four parts (rub 'al-ahzab).

The Quran is the muslims way of life and the guidance from Allah for that every muslim should read, listen and memorize it; nowadays there are computer tools used for this purpose, the interaction with this tools is by using a mouse or a keyboard but in some situation it is difficult to use them for example when driving a car or for blind person; so our goal is to create a Quranic reader controlled by speech.

Arabic language

Quran is revealed in Arabic for that it is the official language of 23 countries and has many different, geographically distributed spoken varieties, some of which are mutually unintelligible. Modern Standard Arabic (MSA) is widely taught in schools, universities, and used in workplaces, government and the media.

Standard Arabic has basically 34 phonemes, of which six are vowels, and 28 are consonants. A phoneme is the smallest element of speech units that makes a difference in the meaning of a word, or a sentence. The correspondence between writing and

pronunciation in MSA falls somewhere between that of languages such as Spanish and Finnish, which have an almost one-to-one mapping between letters and sounds,and languages such as English and French, which exhibit a more complex letter-to-sound mapping[1].

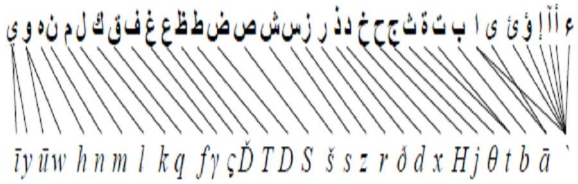


Figure1: letter to sound mapping in Arabic

3. Data collection and processing

The creation of acoustic model pass through two steps, the first is data collection (corpus) and the second is training the model using this data,
The following elements were required to train a new acoustic model:

- Audio data with recorded speech;
- Transcription of each audio file;
- Dictionary with phonetic representations of all words appearing in the transcriptions;
- List of phonemes (and sounds) appearing in the transcriptions

Speech collection

The Quranic reader should recognize 114 suras name's, famous reciters names and control words to function properly (see table1), The amount of audio data required to properly train the model depends on the type of the model. For a simple command-and-control one-speaker system the amount of data can be fairly low. For multi-speaker systems the amount of required audio increases and this is our case.

For this purpose we developed a java applet using a source code powered by voxforge, and we host it in a web server at the internet see figure2.

This applet permits to collect speech easily from a variety of speakers at worldwide scale using social networks and forums, ; their personal profile includes information like name, gender, age, and other information like environmental condition of recording (for example: class room condition, sources of noise like fan, generator sound etc) Technical details of device (pc, microphone specification) are also registered. each speaker recorded all the words two times

Audio files were recorded using sampling rate of 16KHZ and 16 bits per sample. Each file has been named using this convention: speakername-commandID.wav [6, 9] for example a file with the name yacine-001 mean that this file is recorded by yacine and it contain the Fatiha word.

Figure2: java applet used to collect speech from the web

Table 1. Words used in the corpus

Suras names	الفاتحة البقرة آل عمران	001 002 003 …. 114
Reciters names	الغامدي السديسي العجمي الحذيفي	115 116 117 118
Control	تلاوة / Recitation إنهاء / End توقف / stop إستمر / continue تكرار /repeat تحفيظ Memorize تفسير / Interpretation إنتقل / switch بحث / search سورة / sura آية / Verse حزب / groupe جزء / section نفذ / execute	119 120 121 122 123 124 125 126 127 128 129 130 131 132
Arabic digit	صفر /zero واحد/one .. تسعة/nine	133 134 …. 142

Preprocessing

A part of The collected speech data was preprocessed manually using audacity software to confirm that the speakers said the right words and also to remove excessive noise and silence, this step is very hard and time consuming so we decide to reduce the vocabulary size of our system by using only 33 suras names then the total number of word trained will be 60 words. These audio files was divided into three sets, the first is composed of 1800 files from 15male and 15 female used to train the acoustic model, the second composed of 360 files from the same speakers and the third is composed of 360 files from other speakers. The last two sets are used to test the performance of the acoustic model.

Transcription and fileID files

The second step is the transcription of the training set and the test set of the collected audio files; any error in the transcription will mislead the training process later. The transcription process is done using a java program developed by us to generate a primary transcription files which should be tuned manually, that is, we listen to the recording then we match exactly what we hear into text even the silence or the noise should be represented in the transcription. Our program generates also a file which contain audio file ID without extension with reference to the root folder.

Sphinx toolkit accept only ASCII symbol so mapping from Arabic phoneme to ASCII representation should be done. We used the mapping in table 2 [5].

Transcription file format:
<s> ELFAETIHT </s> (AbderrahmaneAR0001)
<s>ELBAEQAARAAH </s>
(AbderrahmaneAR0002)
<s>EAELIHAIUHMRAAN </s>
(AbderrahmaneAR0003)
<s> YUWNAES </s> (AbderrahmaneAR0010)
<s> HUWD </s> (AbderrahmaneAR0011)

<s>:Starting silence
</s> :Closing silence
ELFAETIHT : the ASCII phonetic transcription of the Arabic word الفاتحة
AbderrahmaneAR0001: audio file name containing the word

Table 2: Arabic phoneme to ASCII mapping

Phoneme	Letter	Phoneme	Letter
/AE/	ـَ (Fatha)	/KH/	خ (Khah)
/AE:/	بَا	/D/	د (Dal)
/AA/	ـَا	/DH/	ذ (Thal)
/AH/	ـَة	/R/	ر (Reh)
/UH/	ـُ (Damma)	/Z/	ز (Zain)
/UW/	دُو	/S/	س (Seen)
/UX/	ـُن	/SH/	ش (Sheen)
/IH/	ـِ (Kasra)	/SS/	ص (Sad)
/IY/	فِي	/DD/	ض (Dad)
/IX/	ـِف	/TT/	ط (Tah)
/AW/	لَو	/DH2/	ظ (Thah)
/AY/	ضَيف	/AI/	ع (Ain)
/UN/	ـُ نجي	/GH/	غ (Ghain)
/AN/	ـَم	/F/	ف (Feh)
/IN/	ـِ سما	/V/	ف فيزا
/E/	ء (Hamza)	/Q/	ق (Qaf)
/B/	ب (Beh)	/K/	ك (Kaf)
/T/	ت (Teh)	/L/	ل (Lam)
/TH/	ث (Theh)	/M/	م (Meem)
/JH/	جيم فصحة (Jeem)	/N/	ن (Noon)
/G/	جيم مصرية	/H/	ه (Heh)
/ZH/	جيم معطشة	/W/	و (Waw)
/HH/	ح (Hah)	/Y/	ي (Yeh)

Phonetic dictionary

In this step we mapped each word in the vocabulary to a sequence of sound units representing pronunciation; that it contained all words with all possible variants of their pronunciation, to take into account pronunciation variability, caused by various speaking manners and the specificity of Arabic. Careful preparation of phonetic dictionary prevents from incorrect association of a phoneme with audio parameters of a different phoneme which would effect in decreasing the model's accuracy.

File format

ELFAETIHT	E L F AE: T IH HH AA H
ELBAEQAARAAHT	E L B AE Q AA R AA H
EAELIHAIUHMRAAN	E AE: L IH AI UH M R AA: N
YUWNAES	Y UW N AE S
HUWD	H UW D
YUWSUHF	Y UW S UH F

List of phoneme

This is a file which contain all the acoustic units that we want to train model for, The SPHINX does not permit us to have units other than those in our dictionaries. All units in the dictionary must be listed here. In other words, phone list must have exactly the same units used in your dictionaries, no more and no less. The file has one phone in each line, no duplicity is allowed.

File format:

E
L
F
AE:
T
IH
HH
H
B

Filler Dictionary File

The filler dictionary contains the filler words e.g. the words for mentioning silence and special sounds like cough, breath, chair creaking, door closing etc. We have defined the non-speech utterances i.e. the start of utterance silence <s>, the end of utterance silence <\s> and the middle of utterance silence <sil> in the filler file. we have mapped them all to the same phone SIL, which models silence or the background noise

File format:

<s> SIL
</s> SIL
<sil> SIL

Language model file

This file is created using lmtool which is a web based tool that allows users to quickly compile text-based components needed for using an ASR decoder. To do this, a corpus is needed, which in this case means a set of utterances that is expected for recognition system to be able to handle. The resulting file is in "arpa" format which is standard in speech recognition research. It lists 1, 2-and 3-grams along with their likelihood.

4. Build Acoustic Model with Sphinx Train

SphinxTrain is the acoustic training environment for CMU Sphinx (for Sphinx2, Sphinx3 and Sphinx4), It is a suite of programs, script and documentation for building acoustic models from data for the Sphinx suite of recognition engines.

It is not possible to proceed to the recognition without having an acoustic model, which is necessary to compare the data coming from Front End. This model should be prepared using Sphinx Train tool.

To create the acoustic model we need as input the recorded speech, transcription, dictionary and phoneme files to produce the acoustic model. Much of Sphinx Train's configuration and setup uses scripts written in the programming language Perl.

First MFCC features are extracted from the audio training data set. Each recording can be transformed into a sequence of feature vectors using the front-end executable provided with the SPHINX training package, then using this features the HMM model is trained.

The training process consists of three phases. Each phase consists of three stages (model definition, model initialization, and model training) and makes use of the output of its previous phase [2,7,8,11,12].

- Context-independent phase (CI):

In this phase the main topology for the HMMs is created. The topology of an HMM specifies the possible state transitions in the acoustic model, the default is to allow each state to loop back and move to the next state our model has three emitting states a nd a simple left-to-right topology. The entry and exit states are provided to make it easy to join model together. The exit state of one phone model can be merged with the entry state of another to form a composite HMM. This allows phone model to be joined together to form word..

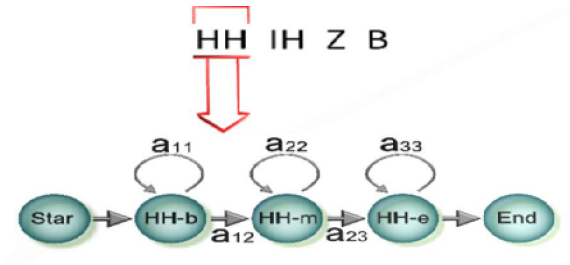


Figure3: HMM context-independent phone model

We assume a parametric form for the density, in which the parameters must be estimated with an iterative solution. In our case a density is represented as a weighted sum or mixture of Gaussians densities.

$$p(o_t / q_t = i) = \sum \frac{w_{ij}}{\left|2\pi\sigma_{ij}\right|^{d/2}} e^{-\frac{1}{2}(O-\mu_{ij})^T \sigma_{ij}^{-1}(O-\mu_{ij})}$$

with the means, variances, and mixture weights to be learnt from the training data using Baum-Welch re-estimation algorithm.

- Untied context-dependant phase (CD)

During the second phase, Arabic phonemes and phones are further refined into Context-Dependent tri-phones. The HMM model is now built for each tri-phone, where it has a separate model for each left and right context for each phoneme and phone. As a result of the second phase, tri-phones are added to the HMM set.

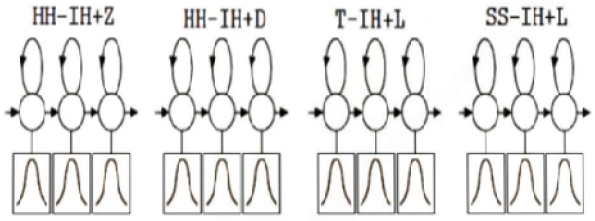
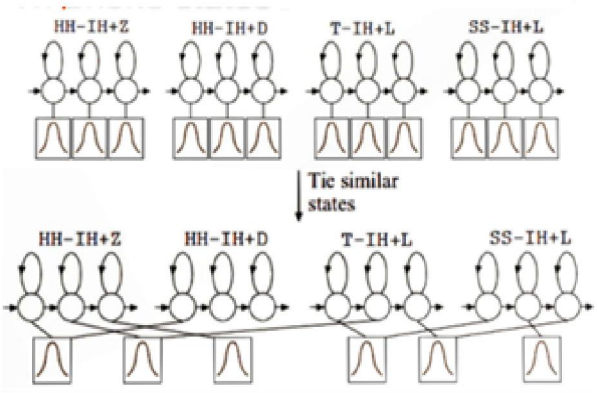


Figure 4: IH+ context dependant phone modling

- Tied context-dependant phase

This phase aims to improve the performance of the model generated by the previous phase by tying some states of the HMMs when the observation density, is known to be the same in two or more states.. These tied states are called senones. The process of creating these senones involves building some decision trees based on some "linguistic questions.



5. Acoustic models performance evaluation

Several parameters can be tuned when training the acoustic model like the number of HMM states, senones and the number of Gaussians mixture. We choose to use a topology with 3 emitting states and we trained several acoustic models by varying the number of senones and the number of Gaussians mixture, each model was tested by Pocket Sphinx using the two test sets (the first contain different utterances from the same speakers who trained the model and the second set from other speakers).

Tables bellow show different performance for each pair of parameters:

Table 3: decoding result for each model using test set1

Test set 1(398.25 seconds speech from trained speakers)			
Senones number	Number of Gaussian	Accuracy	Decoding time
500	1	71.6%	2.15s
	2	76.9%	2.69s
	4	86.1%	4.25s
	8	87.2%	8.75s
	16	87.5%	13.04s
	32	71.3%	19.63s
1000	1	78.3	2.54s
	2	83.6	3.31s
	4	89.2	4.64s
	8	91.1	9.06s
	16	73.9	12.76s
	32	25.8	18.39s
2000	1	78.3	2.56s
	2	83.6	3.33s
	4	89.2	4.67s
	8	91.1	10.01s
	16	73.9	12.50s
	32	25.8	18.01s

Table4: decoding result for each model using test set

Test set 2(215.61 seconds speech from other speakers)			
Senones number	Number of Gaussian	Accuracy	Speed
500	1	76.1%	1.11s
	2	81.7%	1.28s
	4	85.6%	2.00s
	8	86.1%	4.08s
	16	82.8%	6.96s
	32	62.2%	10.78s
1000	1	80%	1.24s
	2	85.6%	1.63s
	4	87.2%	2.43s
	8	82.8%	4.48s
	16	53.3%	6.59s
	32	10%	10.60s
2000	1	80%	1.33s
	2	85.6%	1.70s
	4	87.2%	2.48s
	8	82.8%	4.52s
	16	53.3%	6.32s
	32	10%	10.05s

Effect of Gaussian mixture number on accuracy

Figure (6) and figure (7) shows the number of Gaussians effect versus accuracy.

For every curve, there are an optimal number of mixtures. This illustrates the importance of trainability, it is critical to assure that there are

enough samples for training an increased number of features or parameters. Increasing the number of features or parameters beyond a certain point is likely to be counterproductive. So the number of Gaussians depends on amount of training data, if we have important amount of training, we can increase the number of Gaussians.

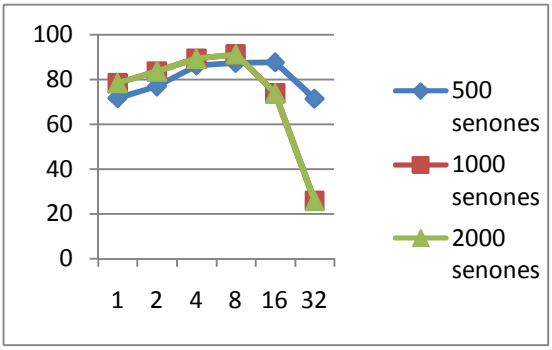


Figure 6: number of Gaussians and senones versus accuracy in test set 1

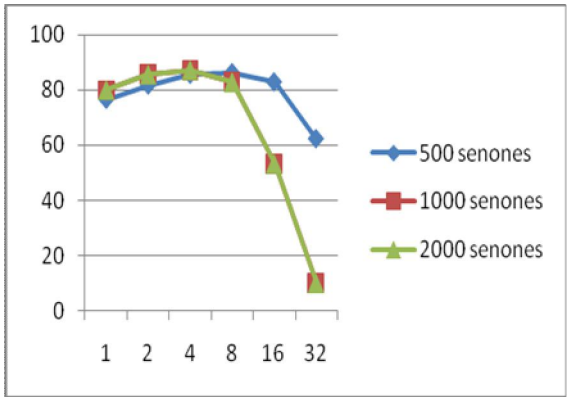


Figure7: number of Gaussians and senones versus accuracy in test set 2

Effect of senones on accuracy:

Parameter tying is used when the observation density, is known to be the same in two or more states. Such cases occur often in characterizing speech sounds. In our example, we see that by increasing the number of senones we have better accuracy when the number of Gaussians is well chosen, because we have insufficient training data to estimate reliably a large number of model parameters

We can see also that the result when using 1000 senones or 2000 senones is the same, because in our vocabulary, we haven't more than 1000 states. We have noticed that, in some cases for example when the number of Gaussians is 16 and 32, increasing the number of senones leads to very bad performance especially for test set 2 because we have more parameters to extract but we have insufficient amount of training data.

Hence, the model with large number of senones is more robust but less precise, in contrast with a model with less number of senones.

Effect of Gaussain and senones number on decoding time:

We notice that the decoding time increases by increasing the number of Gaussians because in each state, we have more parameters to take on consideration when decoding.

When increasing the senones number the behavior is not the same because we have more distribution calculus (time loss) but we will not ask linguistic questions to find which phoneme is to be replaced (time gain) for this reason the decoding time is unpredictable.

Finally, the model chosen by us to be implemented in sphinx-4 based on the results is the model trained with 4 Gaussians and 1000 senones which have a good recognition accuracy in both tests, 89.2% for test set1 and 87.2% for test set2.

6. Building application with Sphinx-4

The Sphinx-4 architecture has been designed with a high degree of flexibility and modularity. Each labeled element in Figure (8) represents a module that can be easily replaced, allowing researchers to experiment with different module implementations without needing to modify other portions of the system. The main blocks in Sphinx-4 architecture are frontend, decoder and Linguist. [3]

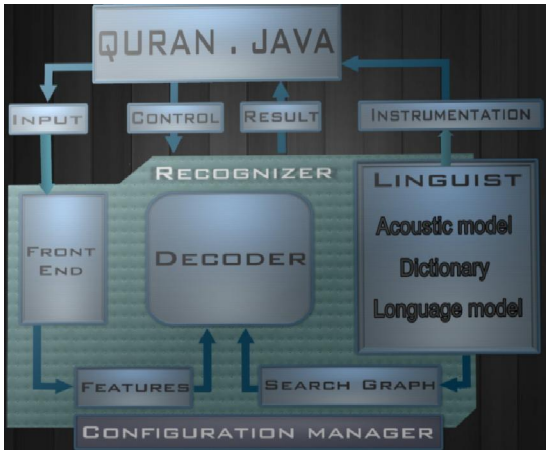
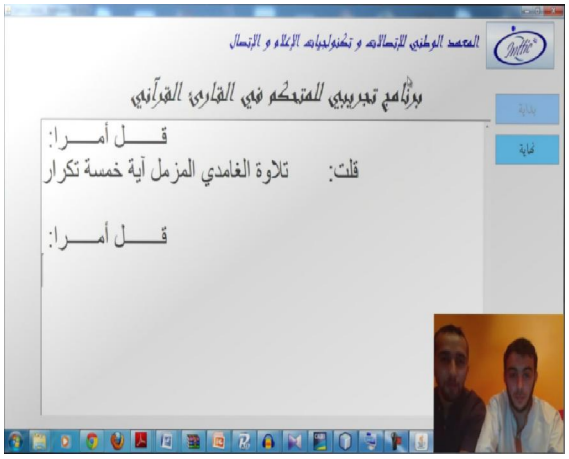


Figure 8: Spinx4 architecture

To build an application with Sphinx-4, the most important modification was done in the linguistic and acoustic part. so using the output of the training process we build a JAR file of the acoustic model then we integrated the language model file created in data collection step, after that we developed a demo application called Quaran.java to interact with a

system, recognize the input speech from the microphone and display the word that have been said [4,5,10].

The result was a multi speaker speech recognizer system able to recognize Quranic reader command and control words; as shown in figure (9) where I said the firsts 3 words and my friend the seconds 3words and the application was able to recognize them without error.



Conclusion

In this paper we reported the process of designing multi-speaker task oriented continuous speech recognition system for Arabic based on CMU Sphinx toolkit to be used in the voice interface of a Quranic reader. The stages of collecting data and training the acoustic model was described in detail, where we used a java applet distributed in the web to collect a large amount of speech, a part of this speech was processed to build several acoustics models by varying the number of Gaussians mixture and senones.

The best model among them which is trained with 4 Gaussians mixtures and 1000 senones and achieved good recognition accuracy in both tests, 89.2% for trained set and 87.2% for untrained set, was chosen to be implemented in sphinx4 and build Quranic reader voice interface.

Acknowledgements:
Authors are grateful to respondents of this study

Corresponding Author:
Yacine Yekkache
Institut National des Télécommunications et des Technologies de l'Information et de la Communication, INTTIC Laboratory LaRATIC, Oran, ALGERIA
E-mail: yyekache@ito.dz

References

1. W.Holmes, M.Huckvale "why have hmms been so successful for automatic speech recognition and how might they be improved" From: Speech Hearing and Language - Work in Progress, Phonetics, UCL, 1994.
2. Nizar Y. Habash, "Introduction to arabic natural language processing" 2010 by Morgan & Claypool.
3. Artur Janicki, Dariusz Wawer "Automatic Speech Recognition for Polish In a Computer Game Interface" Proceedings of the Federated Conference on Computer Science and Information Systems pp. 711–716.
4. Alotaibi Y., Alghamdi M., and Alotaiby F., "Using a Telephony Saudi Accented Arabic Corpus in Automatic Recognition of Spoken Arabic Digits," in Proceedings of 4th International Symposium on Image/Video Communications over Fixed and Mobile Networks, Spain, pp. 43-60, 2008.
5. M. Ali, M. Elshafei, M. Alghamdi, H. Almuhtaseb, and A. Al-Najjar, "Generation of Arabic Phonetic Dictionaries for Speech Recognition," IEEE Proceedings of the International Conference on Innovations in Information Technology, UAE, pp. 59 – 63, 2010.
6. J. L.R. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition",Proc. IEEE 77(2), 257–286 (1989).
7. Jacob Benesty, M. Mohan Sondhi, Yiteng Huang "Springer handbook of speech processing" partE p 540.
8. Alotaibi Y., "Comparative Study of ANN and HMM to Arabic Digits Recognition Systems," Journal of King Abdulaziz University: Engineering Sciences, vol. 19, no. 1, pp. 43-59, 2010.
9. Azmi M. and Tolba H., "Syllable-Based Automatic Arabic Speech Recognition in Different Conditions of Noise," IEEE Proceedings of the 9th International Conference on Signal Processing, China, pp. 601-604, 2008.
10. Nofal M., Abdel-Raheem E., El Henawy H., and Abdel Kader N., "Acoustic Training System for Speaker Independent Continuous Arabic Speech Recognition System," in Proceedings of the 4th IEEE International Symposium on Signal Processing and Information Technology, Italy, pp. 200-203, 2008.
11. Willie Walker, Paul Lamere, Philip Kwok, Bhiksha Raj, Rita Singh, Evandro Gouvea, Peter Wolf, Joe Woelfel, "Sphinx-4: A Flexible Open Source Framework for Speech Recognition,". SMLI TR2004-0811 c2004 SUN MICROSYSTEMS INC.
12. H. Hyassat, and R. Abu Zitar, "Arabic speech recognitionusing SPHINX engine," International Journal of SpeechTechnology, Springer, pp. 133-150,2008.
13. Satori H., Harti M., and Chenfour N., "Arabic Speech Recognition System Based on CMUSphinx," in Proceedings of IEE International Symposium on Computational Intelligence and Intelligent Informatics, Morocco, pp. 31-35, 2009.

11/2/2013