Egyptian Informatics Journal (2016) 17, 305–314

Cairo University

Egyptian Informatics Journal

www.elsevier.com/locate/eij
www.sciencedirect.com

Building CMU Sphinx language model for the Holy Quran using simplified Arabic phonemes

Mohamed Yassine El Amrani [a,b,*], M.M. Hafizur Rahman [b], Mohamed Ridza Wahiddin [b], Asadullah Shah [b]

[a] Computer Science and Engineering Department, Jubail University College, Saudi Arabia
[b] Computer Science Department, Kulliah of Information Communication Technology, International Islamic University Malaysia, Malaysia

Received 28 August 2015; accepted 24 April 2016
Available online 7 June 2016

KEYWORDS

Automatic speech recognition;
Holy Quran recognition;
Human voice

Abstract This paper investigates the use of a simplified set of Arabic phonemes in an Arabic Speech Recognition system applied to Holy Quran. The CMU Sphinx 4 was used to train and evaluate a language model for the *Hafs* narration of the Holy Quran. The building of the language model was done using a simplified list of Arabic phonemes instead of the mainly used Romanized set in order to simplify the process of generating the language model. The experiments resulted in very low Word Error Rate (WER) reaching 1.5% while using a very small set of audio files during the training phase when using all the audio data for both the training and the testing phases. However, when using 90% and 80% of the training data, the WER obtained was respectively 50.0% and 55.7%.

1. Introduction

Muslims all over the world practice, at least, five prayers daily involving Holy Quran recitations. This makes the Holy Quran recitation and memorization at the center of every Muslim. Since any memorization process requires a continuous review and the best way to have one's recitation reviewed and verified is to involve another person, preferably an expert in Holy Quran recitations, who follows one's recitation and correct inaccuracies. When this option is not available, each reciter has to continuously check, after each couple of *Ayat* ("sentences"), if his recitation was correct. Doing so, the reciter needs to look at the Holy Book and verify what was

* Corresponding author at: Computer Science and Engineering Department, Jubail University College, Saudi Arabia.
E-mail addresses: amranim@ucj.edu.sa (M.Y. El Amrani), hafizur@iium.edu.my (M.M.H. Rahman), mridza@iium.edu.my (M.R. Wahiddin), asadullah@iium.edu.my (A. Shah).

Peer review under responsibility of Faculty of Computers and Information, Cairo University.

Production and hosting by Elsevier

http://dx.doi.org/10.1016/j.eij.2016.04.002

recited while trying to avoid getting a hint on the next *Ayah* (“sentence”) which will undermine his verification.

The advances in Automatic Speech Recognition (ASR) ought to allow any Holy Quran reciter to independently check the accuracy of his recitation. The wide availability of devices equipped with microphones and powerful computing capabilities constitutes a great potential for using ASR systems in assisting in checking recitations. Moreover, much research has been done in order to apply the ASR technology to the Arabic language and to the Holy Quran. Many Automatic Arabic Speech Recognition (AASR) systems were developed in order to improve the recognition accuracy of the Arabic language in general and the Holy Quran specifically. Many of those systems were based on one of the most actively investigated speech recognition frameworks which is developed by the Carnegie Melon University (CMU). The CMU Sphinx is a speaker-independent statistical set of tools that are flexible enough to be used for any language.

In this paper, we assessed the performance of the CMU Sphinx trained language models using simplified phonemes consisting of Arabic diacritic letters, instead of the commonly used Romanized phonemes, and detailed the processes and experiments introduced in [1]. The rest of this paper is organized as follows: The use of Arabic Speech Recognition Systems applied to the Holy Quran is discussed in Section 2. The different processing and experimental setups are detailed in Section 3. The results of the different experiments are discussed in Section 4. Finally, the conclusion and future research are discussed in Section 5.

2. Automatic Arabic Speech Recognition (AASR)

The advances of AASR systems should allow anyone to autonomously and easily verify his/her Holy Quran recitation. However, the majority of those systems using Sphinx tools were facing the complexity of using Romanized phonemes, representing pronunciation sounds, in order to accurately train the language model. In this paper, the focus will be on investigating the performance of simplified Arabic phonemes, instead of the Romanized ones, for the construction of a robust language model for the Holy Quran recitation verification. These Arabic phonemes will be automatically generated based on the Arabic and Tajweed rules along with the required data to train the language model using the CMU Sphinx tools.

Speech recognition systems are very popular since they allow a more intuitive and natural way for humans to interact with devices and machines. However, applying ASR techniques to the Arabic language is still in its early stages when compared to ASR for English. There are various challenges that are facing any project related to ASR such as stuttering, coughing, false starts, dis-fluency, pitch, and repetitions but have moreover the challenges that arise from the pace of speech as described in [2]. In the context of the Holy Quran, elongations are another challenge for accurate recognition since some *Ayat* allows different lengths for the pronunciation of some letters of a word, which leads to the same word being pronounced differently even by the same speaker and still be a valid pronunciation that needs to be correctly recognized.

Nevertheless, there are many tools that can be used for the AASR and can be applied to the Holy Quran. One of the most actively developed frameworks is the CMU Sphinx which is a statistical speaker-independent set of tools using the Hidden Markov Models (HMM) which provide a statistical language model using unigrams, bigrams, and trigrams probabilities. An introduction to CMU Sphinx 4 for the Arabic Speech Recognition is provided in [3–5]. This statistical approach has the advantage of allowing speaker-independent models to be trained for continuous speech recognition of any language.

The idea of applying the AASR techniques to the Holy Quran is not new and the Holy Quran is the subject of many research projects as seen in [6]. Many researchers have been using CMU Sphinx tools for Arabic Speech Recognition as well as for the Holy Quran as found in [3,4,7–10]. Other different projects focused on improving the correct pronunciation of letters as in [11], the recognition of Tajweed rules, Arabic pronunciation rules specific to Holy Quran recitation, as in [12,13], the creation of a virtual learning systems as found in [14,15], and detecting the mistakes of students’ recitations discussed in [16].

However, Romanized phonemes are commonly used in order to represent the pronounced letters and words even for AASR systems as in the previously mentioned research ([7–9,11–13,15–19]). This use of Romanized phonemes for Arabic is unnatural, very difficult to read, and time-consuming as mentioned in [5]. Even if it obtained good results, it makes the process of preparing and generating the set of phonemes used in the phonetic transcription of the audio files used in the training of the language model very complex to manage and highly time-consuming. Very few researchers have investigated using a list of phonemes composed of Arabic letters as in [5,10,19–21] which are more natural and better adapted to be used with Arabic words. While some have investigated the use of Arabic phonemes, very few tried to apply it to the Holy Quran recitation verification. However, researchers in [10] found that Romanized phonemes outperformed the Arabic phonemes when building a language model for one Holy Quran chapter (112) while others demonstrated as in [5] that Arabic phonemes can obtain good recognition rates for different types of corpora. However, its performance was very poor on the Holy Quran corpus by achieving a WER of 46%. However, neither specific details were given about the number of chapters that were used in their experiments, nor on the number of different reciters (speakers) used to train their language model. In light of these last two most relevant research projects involving the use of Arabic phones, it appears that mixed results were obtained and further investigation should allow a better evaluation of the performance of the use of the Arabic phoneme in an accurate recognition of the Holy Quran recitations.

3. Language model for the Holy Quran

Certainly, the use of new technologies such as AASR in order to help Muslims in their memorization and recitation review of the Holy Quran has a potential for a bright future. One of the widely available tools that have the potential to achieve this goal is the CMU Sphinx. It is a robust and flexible Open-Source framework that can be adapted for the Holy Quran. However, in order to obtain the needed high recognition accuracy, extra care needs to be put in the generation of the language model for the *Hafs* narration of the Holy Quran since

it is the critical phase of this process. This narration is the most commonly used in the Muslim world and the vast majority of available audio data are using this narration.

In order to build the language model for the Holy Quran, it is crucial to have a vast amount of training data. There are many recordings of renowned Holy Quran reciters that will need to be downloaded and organized into different folders for each speaker. These audio files need to contain one *Ayah* in each file in order to simplify the automated generation of the transcription file required by the Sphinx 4 trainer. Each audio filename is identified using the number of the Surah ("chapter") and the number of the *Ayah* along with an unique identifier since many recordings of the same *Ayah* are to be gathered. In addition, the Holy Quran text of the *Ayat* needs to be stored in a database in order to facilitate its retrieval along with its association with the different audio files by different speakers. At this stage, we need to have a program that will automate the generation of all the required files for the Sphinx 4 trainer. The Arabic and Tajweed rules will be implemented in the program using a rule-based component in order to generate the simplified list of Arabic phonemes by using one diacritical Arabic letter. Then, the training would be carried out using different training scenarios and parameters. At first, only a small set of Holy Quran chapters should be used since the training phase and programming of all the Arabic and Tajweed rules are time-consuming in both programming and processing. This small set of Holy Quran chapters is used to validate the performance of the language model by using different training scenarios with different parameters and audio files to compute the Word Error Rate (WER).

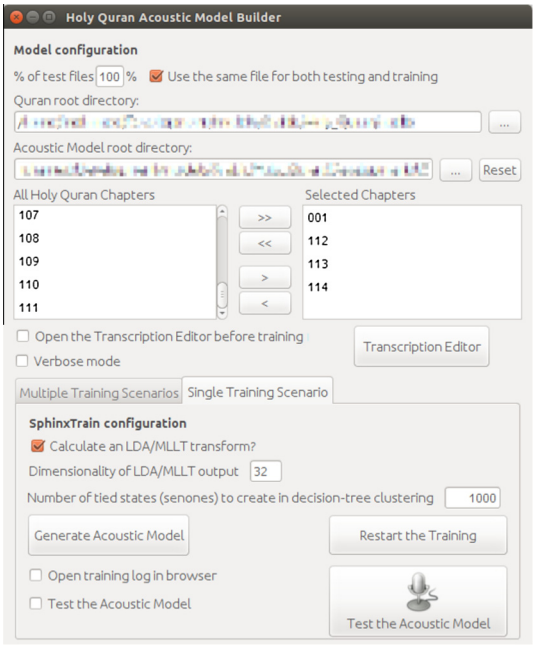

Figure 1 Main window for automatic generation and training of the language model for the Holy Quran.

Once the performance of this simplification is confirmed, the next step would be to generalize this process to include more chapters of the Holy Quran to ultimately generate the language model for all the Holy Quran chapters. This involves generating the set of simplified Arabic phonemes respecting all Arabic and Tajweed rules along with gathering a large number of recitation recordings to use for the training. At this stage, each audio file for each *Ayah* might need to be manually checked in order to see whether it needs some processing such as locating and tagging all pauses in the recitation in the middle of the *Ayah*. This involves correctly determining whether

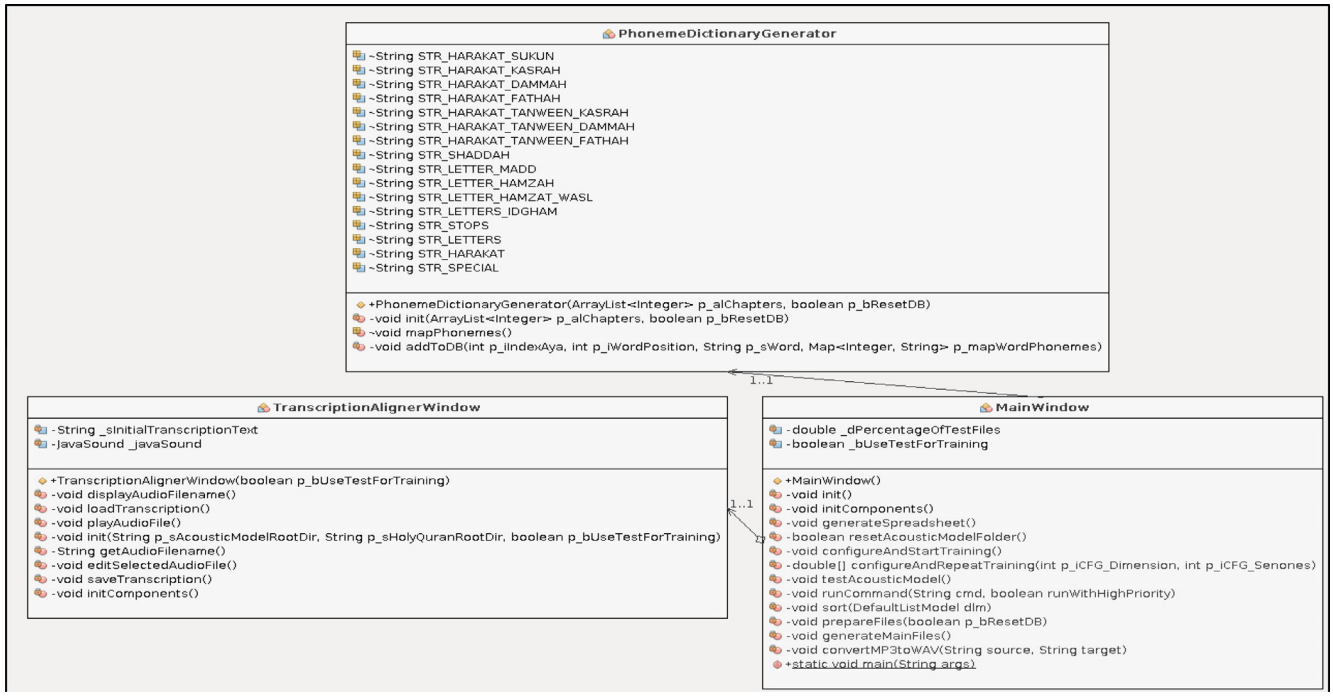

Figure 2 Summarized class diagram for the main three components of the language model builder.

there were some repeated words after the pause in the middle of the *Ayah* which needs to be transcribed accordingly. Subsequently, different training scenarios will need to be conducted using different training parameters in order to find the optimal simplification of the Arabic set of phonemes.

4. Experimental setup

In order to validate the potential performance of the language model using simplified Arabic phonemes, a Java program was created as shown in Fig. 1. The program allows selecting the Holy Quran chapters to use for the training as well as setting the training parameters and is populating the database with the phonemes present in each word of each *Ayah*. The program has three main components: the main window, the transcription editor, and the phoneme generator as in Fig. 2. The main window constitutes the link between to other two components and gathers the data required for the training. After setting the different parameters in the *MainWindow*, the list of phonemes for each word based on its position in an *Ayah* is extracted and saved in a database by the *PhonemeDictionaryGenerator*. The *TranscriptionAlignerWindow* is used to edit and correct transcription of the *Ayat* in order to take into consideration pauses in some recitations.

Among the different parameters that need to be entered in the developed program, two training parameters affect greatly the performance of the language model: the number of Senones (or tied states) and the dimension of the Gaussian Mixtures as discussed in [8,15] for the training of the Holy Quran language model. While the dimension affects the phonemes context dependency, the more Senones a model has, the more precise is the discrimination of the sounds and less is the recognition of unseen speech.

The actual version incorporated only a few Arabic and Tajweed rules since it was tested only on 4 Holy Quran chapters: 001, 112, 113, and 114. The choice of those chapters was motivated by the fact that they are very short in terms of the length of the *Ayat* and it has a very small set of Arabic and Tajweed rules that needs to be implemented. The developed program automatically generates the transcription file, the phonetic dictionary and the set of simplified Arabic phonemes respecting the applicable Arabic and Tajweed rules. The generated data are stored in a MySQL database, as in Fig. 3, and are used to prepare the files respecting the different formats required by the Sphinx 4 tools. Different training scenarios with different parameters were conducted and the performance of the language model for each scenario was saved in a spreadsheet file for further analysis.

The following sections will detail the different types of data generated by the program in order to build a language model using the CMU Sphinx 4 training tools: the transcription file, the phonetic dictionary, the list of phonemes, and the recordings of Holy Quran recitations.

<s> إِلَٰهِ النَّاسِ </s> (114003_id0000)
<s> إِيَّاكَ نَعْبُدُ وَإِيَّاكَ نَسْتَعِينُ </s> (001005_id0001)
<s> بِسْمِ اللَّهِ الرَّحْمَٰنِ الرَّحِيمِ </s> (001001_id0002)
<s> وَلَمْ يَكُنْ لَهُ كُفُوًا أَحَدٌ </s> (112004_id0003)
<s> الرَّحْمَٰنِ الرَّحِيمِ </s> (001003_id0004)
<s> قُلْ هُوَ اللَّهُ أَحَدٌ </s> (112001_id0005)
...

Figure 4 Sample of a transcription file for the chapters 001, 112, 113, and 114 of the Holy Quran.

4.1. Transcription file

The transcription file is initially generated automatically using the Holy Quran from the database along with the audio recitations of the Holy Quran. Each line holds the Arabic text contained in the audio files between the delimiters "<s>" and

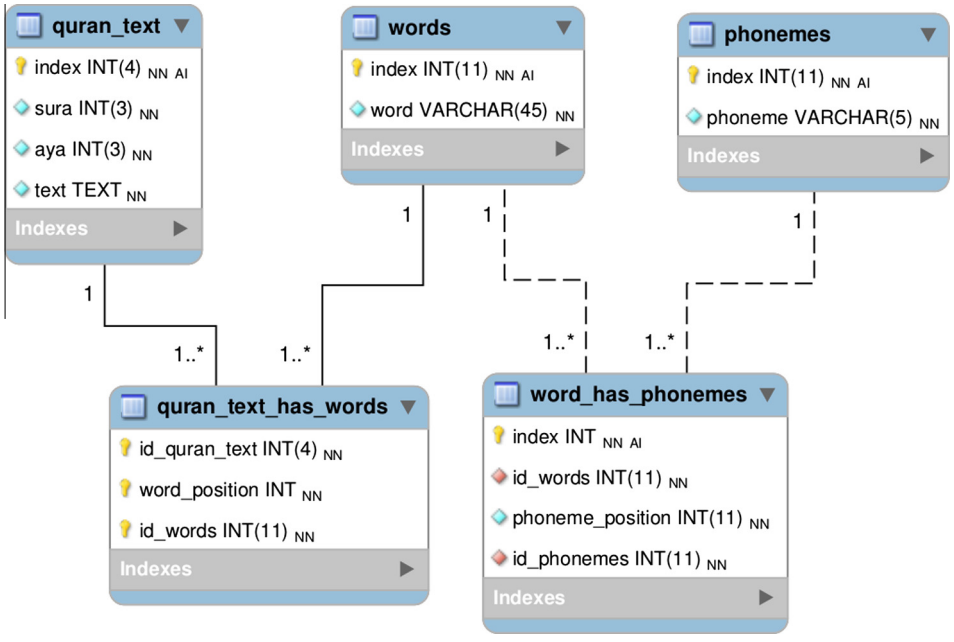

Figure 3 Entity-relationship diagram of the MySQL database for the Holy Quran language model generation.

"</s>" representing the beginning and ending silence utterances, along with the audio filename as shown in Fig. 4. Since some audio files where an adult reciter has a child reciting after him, an automated processing was added in order to tag the silence between recitations with "<sil>", representing the silence between the two recitations. In some very few cases, manual updates of the transcription were necessary in order to tag the silences when a reciter pauses his recitation in the middle of a long *Ayah*.

4.2. Phonetic dictionary

The phonetic dictionary file contains the list of automatically generated phonemes that represent a word as it is recited as shown in Table 1. Each phoneme is using the simplified Arabic representation of an Arabic letter with its diacritics. Some words might have different pronunciations depending on its position in the *Ayah* and they have a number between parentheses in order to differentiate between them. In the case of the selected Holy Quran Chapters, only three words have two entries in the phonetic dictionary.

4.3. Arabic phonemes

The set of Arabic phonemes generated for the phonetic dictionary reached 80 and is listed in Table 2. Each phoneme consists of a diacritic Arabic letter with the exception of three non-diacritic Arabic letters. The Arabic letters "ا", and "ي", and "و" are the only letters without diacritics that were used for representing the elongations of a letter with Fathah "◌َ", Kasrah "◌ِ", and Dammah "◌ُ" respectively. The emphasis symbol (Shaddah: "◌ّ") can be also found with a diacritic symbol in order to represent the emphasis on the pronunciation of a letter. Finally, the Hamzah "ء" has been used to represent all its different forms (ا أ إ ئ ؤ ء) depending on its diacritic or position in the sentence. The main advantage of the use of this simplified representation of phonemes is to be able to easily verifying the correctness of the representation of the pronunciation of any word while being able to capture all the different Arabic and Tajweed rules present in the selected chapters, which leads to a faster language model building for the whole Holy Quran.

4.4. Training audio files

The audio files used for building the language model were downloaded from [22,23] and needed to be re-sampled and converted to 16 kHz, 16 bit, mono files in MS WAV format as required by the Sphinx 4 trainer (version 1.08). Furthermore, some manual processing had to be done such as trimming the beginning and the end of the audio files to match the *Ayah* since it might include some utterance not present in the transcription from the Holy Quran or because of the existence of long silences that needed to be removed to avoid being considered by the trainer. The

Table 1 Phonetic transcription for the chapters 001, 112, 113, and 114 of the Holy Quran.

1.	بِسْمِ	بِ سْ مِ	35.	يَلِدْ	يَ لِ دْ
2.	اللَّهِ	لَّ هِ	36.	وَلَمْ	وَ لَ مْ
3.	الرَّحْمَٰنِ	رَّ حْ مَ ا نِ	37.	يُولَدْ	يُ و لَ دْ
4.	الرَّحْمَٰنِ(2)	ءَ رَّ حْ مَ ا نِ	38.	يَكُنْ	يَ كُ
5.	الرَّحِيمِ	رَّ حِ ي مْ	39.	لَهُ	لَّ هُ
6.	الْحَمْدُ	ءَ لْ حَ مْ دُ	40.	كُفُوًا	كُ فُ وَ نْ
7.	لِلَّهِ	لِ لَّ هِ	41.	أَعُوذُ	ءَ عُ و ذُ
8.	رَبِّ	رَ بِّ	42.	بِرَبِّ	بِ رَ بِّ
9.	الْعَالَمِينَ	لْ عَ ا لَ مِ ي نْ	43.	الْفَلَقِ	لْ فَ لَ قْ
10.	مَالِكِ	مَ ا لِ كِ	44.	مِنْ	مِ نْ
11.	يَوْمِ	يَ وْ مِ	45.	شَرِّ	شَ رِّ
12.	الدِّينِ	دِّ ي نْ	46.	مَا	مَ ا
13.	إِيَّاكَ	ءِ يَّ ا كَ	47.	خَلَقَ	خَ لَ قْ
14.	نَعْبُدُ	نَ عْ بُ دُ	48.	وَمِنْ	وَ مِ نْ
15.	وَإِيَّاكَ	وَ ءِ يَّ ا كَ	49.	غَاسِقٍ	غَ ا سِ قِ نْ
16.	نَسْتَعِينُ	نَ سْ تَ عِ ي نْ	50.	إِذَا	ءِ ذَ ا
17.	اهْدِنَا	ءِ هْ دِ نَ	51.	وَقَبَ	وَ قَ بْ
18.	الصِّرَاطَ	صِّ رَ ا طَ	52.	النَّفَّاثَاتِ	نَّ فَّ ا ثَ ا تِ
19.	الْمُسْتَقِيمَ	لْ مُ سْ تَ قِ ي مْ	53.	فِي	فِ
20.	صِرَاطَ	صِ رَ ا طَ	54.	فِي(2)	فِ ي
21.	الَّذِينَ	لَّ ذِ ي نَ	55.	الْعُقَدِ	لْ عُ قَ دْ
22.	أَنْعَمْتَ	ءَ نْ عَ مْ تَ	56.	حَاسِدٍ	حَ ا سِ دِ نْ
23.	عَلَيْهِمْ	عَ لَ يْ هِ مْ	57.	حَسَدَ	حَ سَ دْ
24.	غَيْرِ	غَ يْ رِ	58.	النَّاسِ	نَّ ا سْ
25.	الْمَغْضُوبِ	لْ مَ غْ ضُ و بِ	59.	مَلِكِ	مَ لِ كِ
26.	وَلَا	وَ لَ	60.	إِلَٰهِ	ءِ لَ ا هِ
27.	الضَّالِّينَ	ضَّ ا لِّ ي نْ	61.	الْوَسْوَاسِ	لْ وَ سْ وَ ا سِ
28.	قُلْ	قُ لْ	62.	الْخَنَّاسِ	لْ خَ نَّ ا سْ
29.	هُوَ	هُ وَ	63.	الَّذِي	ءَ لَّ ذِ ي
30.	اللَّهُ	لَّ هُ	64.	يُوَسْوِسُ	يُ وَ سْ وِ سُ
31.	اللَّهُ(2)	ءَ لَّ هُ	65.	صُدُورِ	صُ دُ و رِ
32.	أَحَدٌ	ءَ حَ دْ	66.	مِنَ	مِ نَ
33.	الصَّمَدُ	صَّ مَ دْ	67.	الْجِنَّةِ	لْ جِ نَّ تِ
34.	لَمْ	لَ مْ	68.	وَالنَّاسِ	وَ نَّ ا سْ

transcription editor module of the developed Java program allows to edit the audio and the transcription files before the starting the training and is shown in Fig. 5. Two types of updates to the audio files can be made: temporary and permanent. The temporary changes are made to the converted files from the root folder containing the original audio files. If the changes need to be saved permanently, the file from the root folder will be updated. This allows assessing the performance of the built language model based on the temporarily updated files before deciding to make the changes permanent.

The recordings from a total number of 22 different renowned reciters, as listed in Table 3, were used as the training data from which all the required files were generated. Among those reciters, only one was found to have a recording

Table 2 List of Arabic phonemes generated from the chapters 001, 112, 113, and 114 of the Holy Quran.

1	ءَ	17	دِ	33	صِ	49	قَ	65	نَ
2	ءِ	18	دّ	34	صَّ	50	قُ	66	نِ
3	ا	19	دْ	35	صٍ	51	قِ	67	نَّ
4	بُ	20	ذَ	36	ضُ	52	قْ	68	نْ
5	بِ	21	ذُ	37	ضَّ	53	كَ	69	هُ
6	بِّ	22	ذِ	38	طَ	54	كُ	70	هِ
7	بْ	23	رَ	39	عَ	55	كِ	71	هْ
8	تَ	24	رِ	40	عُ	56	لَ	72	و
9	تِ	25	رَّ	41	عِ	57	لِ	73	وَ
10	ثَ	26	رٍّ	42	عْ	58	لَّ	74	وِ
11	جِ	27	سَ	43	غَ	59	لٌ	75	وْ
12	حَ	28	سُ	44	غْ	60	لْ	76	ي
13	ح	29	سِ	45	فَ	61	مَ	77	يَ
14	حْ	30	سْ	46	فُ	62	مُ	78	يُ
15	خَ	31	شَ	47	فِ	63	مِ	79	يَّ
16	دُ	32	صُ	48	فَّ	64	مْ	80	يْ

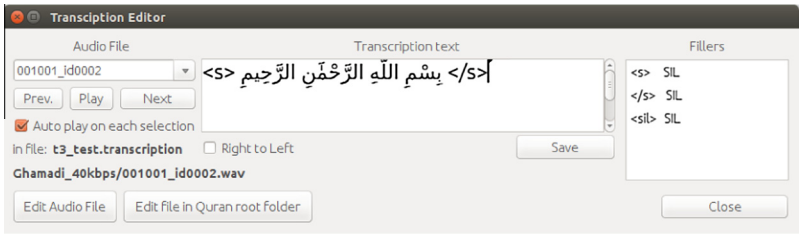

Figure 5 Transcription editor is used when the transcription of the audio file needs to be edited manually.

Table 3 MP3 Holy Quran files from renowned reciters, with the original sampling rates, that had to be resampled to 16 kHz and were used to train the language model for the chapters 001, 112, 113, and 114.

1. Abdul Basit Murattal @ 40kbps
2. Abdullaah 3awwaad Al-Juhaynee @ 128kbps
3. Abdullah Basfar @ 32kbps
4. Abu Bakr Ash-Shaatree @ 64kbps
5 .Ahmed ibn Ali al-Ajamy @ 64kbps
6. Banna @ 32kbps
7. Ghamadi @ 40kbps
8. Hani Rifai @ 192kbps
9. Hudhaify @ 32kbps
10. Husary @ 40kbps
11. Ibrahim Akhdar @ 32kbps
12. Minshawy Murattal @ 48kbps
13. Mohammad al Tablaway @ 64kbps
14. Mostafa Ismail @ 128kbps
15. Muhammad Ayyoub @ 64kbps
16. Muhammad Jibreel @ 64kbps
17. Muhsin Al Qasim @ 192kbps
18. Nasser Alqatami @ 128kbps
19. Saood ash-Shuraym @ 64kbps
20. Tunaiji @ 64kbps
21. Yaser Salamah @ 128kbps

of a boy child repeating the recited *Ayah* by the reciter "Khaleefa". While this was not originally planned, this case was taken into consideration when generating the transcription file in order to reflect the repeating content of the repeated *Ayah* to match the content of the audio file. It is worth mentioning that, at the time of the experimentations, the content of all the audio files used reached only 40 min approximately and that the Sphinx trainer reported, as a warning, that this is a small set of training data.

5. Results and discussion

All experiments were performed using a single computer with the following specifications: Intel® Core™ i7-4702MQ CPU @ 2.20 GHz × 8, 16 GB of RAM, Ubuntu 14.04.1 LTS 64 bits.

The minimum WER was achieved 20 times with a dimension set to 32 with a percentage of 51% for the different 39 Senones settings. While it reached 18% (7/39) with the dimension set to 38 and 24 as well, and 8% (3/39) with a dimension of 26, the remaining 5% is distributed on the remaining (2/39) of one from each dimension: 30 and 34.

All experiments were completed using the audio recitations from the 22 different male reciters and 1 boy child for the selected 4 Holy Quran chapters resulting in 65 words and 80 phonemes. Since the audio training data is small, all of the recordings were used for both the training and the testing at first and another set of experiments did not use all the audio files for both the training and testing in order to assess the built language model with unseen audio data.

The WER is automatically computed by the Sphinx tools based on the testing data. However, when all the audio files were used for both the training and the testing, the WER is computed only on audio files seen during the training. Thus, the results obtained are strongly coupled to the training data. Nevertheless, this will allow assessing the potential perfor-

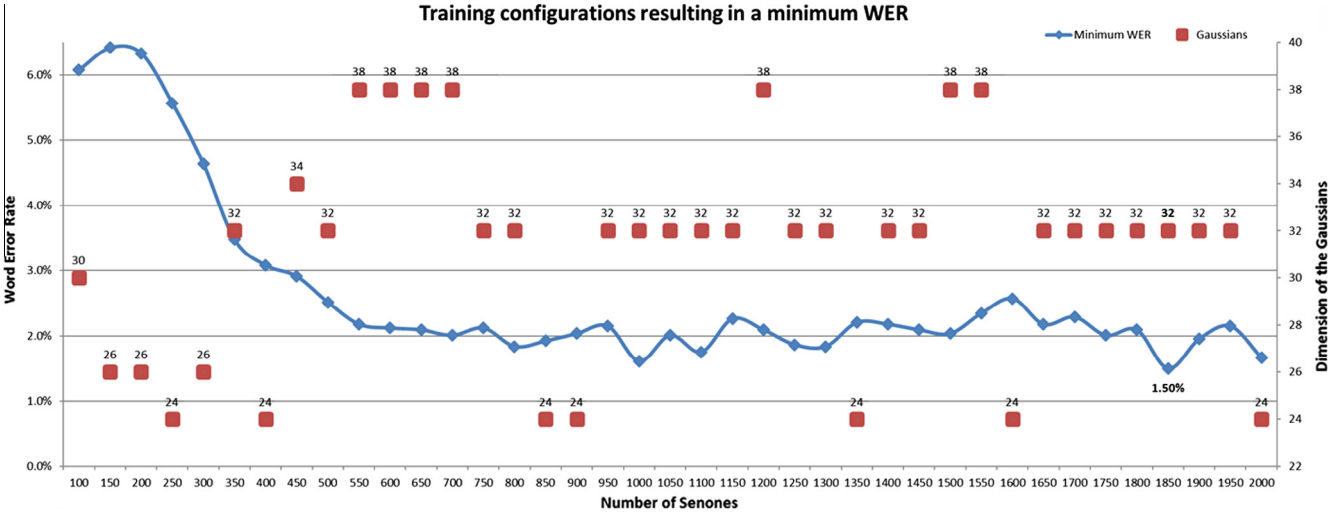

Figure 6 Settings leading to the minimum WER for the Holy Quran chapters 001, 112, 113, and 114 for each training scenario.

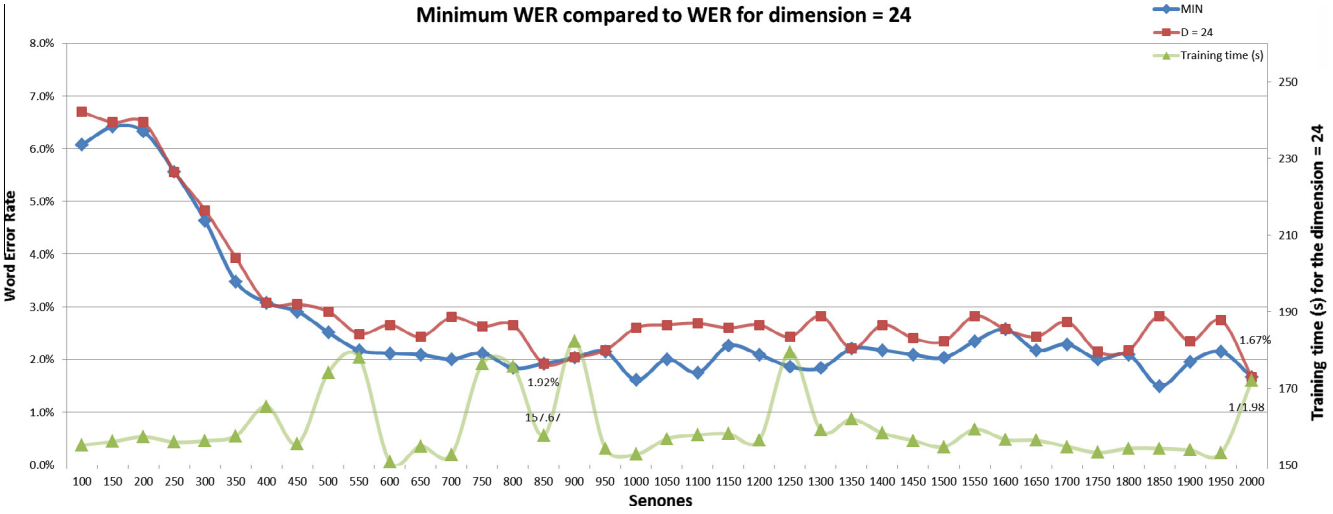

Figure 7 Comparing the minimum value of the WER of each training configuration to the results of the training with the dimension fixed to 24.

mance of using simplified Arabic phonemes for training the language model for the Holy Quran.

5.1. Training the model using all data

The training scenarios were generated by varying the number of Senones from 100 to 2000 with an increment of 50. For each number of Senones used, the dimension of the Gaussians was set to a value from 2 to 64 with an increment of two (2). Finally, the training was repeated twice for each training scenario. The repetition of the training was made since in many occurrences, the same configuration led to different WERs. The average of the performance of the training is calculated using the Word Error Rate (WER) provided by the Sphinx tool during the testing phase immediately after completing the training. The minimum of the computed average of all WERs for each number of Senones along with the dimension of the Gaussians that obtained that minimum is shown in Fig. 6. It shows that setting the dimension to 32 is the optimal dimension of Gaussians for many different settings of Senones and the lowest WER reached 1.5% and was obtained with a number of Senones set to 1850. Two interesting performances were reached when the Senones were set to 1000 and 2000 which resulted in a WER of 1.62% and 1.67% respectively.

These results help to realize that this phoneme simplification using diacritic Arabic letters has the potential of attaining promising results since the WER obtained in many experimental settings was less than 2% with a minimum of 1.5%. This confirms that it is possible to achieve very good word recognition accuracy with simplified Arabic phonemes. Even if most of the minimum WER were obtained when the dimension of the Gaussian mixtures was set to 32, the minimal WER was similarly reached with a setting of the Gaussians to 24 and 38. By comparing the WER resulting from each of those

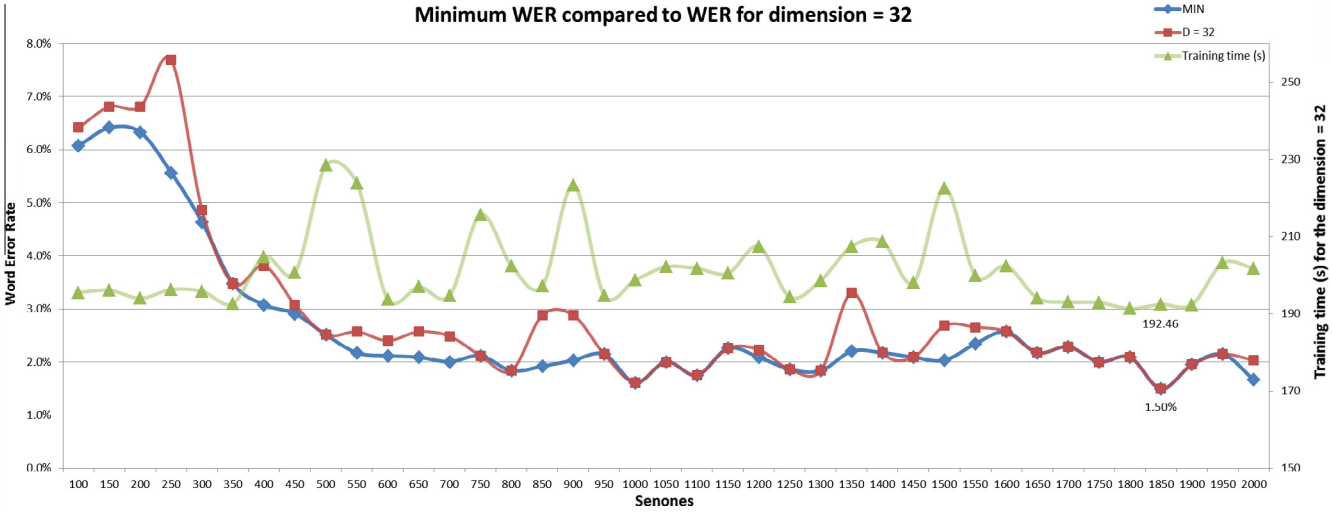

Figure 8 Comparing the minimum value of the WER of each training configuration to the results of the training with the dimension fixed to 32.

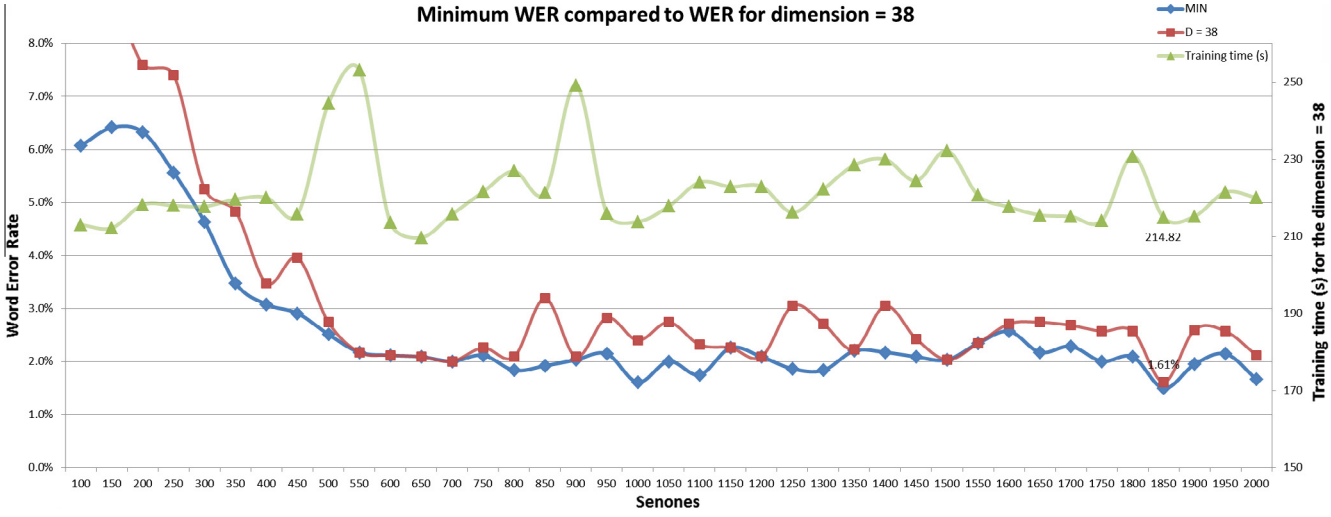

Figure 9 Comparing the minimum value of the WER of each training configuration to the results of the training with the dimension fixed to 38.

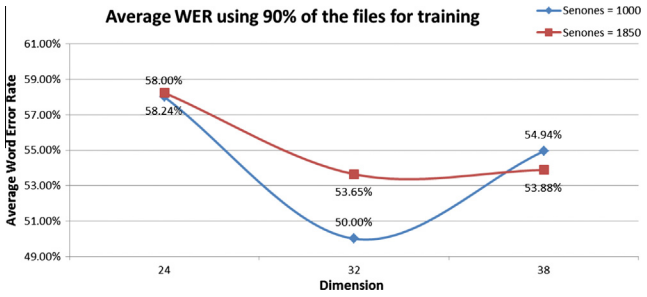

Figure 10 Average of WER obtained while repeating the training 5 times using 90% of all audio files for training and the remaining 10% for testing, with configurations of Senones being 1000 and 1850, and with dimensions set to 24, 32, and 38.

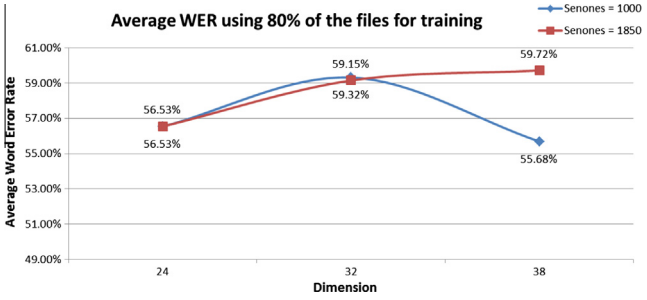

Figure 11 Average of WER obtained while repeating the training 5 times using 80% of all audio files for training and the remaining 20% for testing, with configurations of Senones being 1000 and 1850, and with dimensions set to 24, 32, and 38.

settings with the minimum WER, it appears that their performance is close to the minimum WER as seen in Figs. 7–9.

The optimal dimension of the Gaussians was found to be 32; however, when setting it to 24, the obtained performance was not very far from the minimum WER as shown in Fig. 7. Using that setting of the Gaussians led to obtaining a WER of 1.67% when the Senones were set to 2000 with a difference of only 0.17% from the minimal WER of 1.5%. However, the duration of the training for that setting was only 171.98 s which is the shortest training time for all minimal values of WER and it took almost 12% less time to train the same language model with a competitive performance. With the dimension of Gaussians set to 32, it is clear that the training time for each number of Senones has increased compared to the setting of 24 and the minimum WER is obtained after a training time of 192.46 s as in Fig. 8. It is important to notice that the difference in performance between these two settings is very small which might suggest setting initially the dimension of the Gaussians to 24 when training the language model on a large set of audio data to quickly anticipate the optimal WER before using the quasi-optimal value of 32. Furthermore, when setting the dimension of the Gaussians to 38, the training time increased as compared to the previous values as seen in Fig. 9. The minimal WER of 1.61% is very close to the minimal WER but it was obtained after a training time of 214.82 s with the number of Senones set to 1850.

It is worth mentioning that there are 2 peaks in the training time curve that are found to be common to each one of those top 3 dimensions which are found when the Senones were set to 550 and 900 as seen in Figs. 7–9. More experiments should be carried out in order to accurately understand and interpret these patterns.

Finally, it is very important to remember that all these results were obtained using all the audio files for both the training and the testing of the trained model. Using this experimental setting did not test the model on unseen audio data. This resulted in a model dependent on the trained audio files. This led to more experimental settings to be carried out. In order to assess the trained language model on unseen data, some of the audio files need to be excluded from the training phase and used only during the testing phase.

5.2. Testing the model on unseen data

In order to assess the performance of the language model on audio files unseen during the training phase, different experiments were set by varying the percentage of unseen audio files during the training. Instead of always using 100% of the files for both the training and the testing, audio files were separated into two groups: one for the training and the other for the testing. The audio files were randomly separated into either to be used for the training of the language model or for its testing when the resulting WER is computed.

Based on results from the previous experiments and in order to train the language model quickly, different training scenarios were performed using the number of Senones set to 1000 and 1850 with the dimension of the Gaussians to be 24, 32, or 38. These values were used based on the results of previous experiments when they allowed obtaining very low WERs. Each training was repeated five times and the average of all resulting WERs was computed and is shown in Figs. 10 and 11.

The first experimental setting used 90% of the files for the training and the remaining 10% for the testing phase and the results are shown in Fig. 10. This resulted in a sharp increase of the WER reaching as high as 58.24% when the number of Senones was set to 1850 and the dimension of the Gaussians set to 24. The minimal WER was 50% obtained with 1000 Senones and 32 for the dimension of the Gaussians. This trend is confirmed by the second experimental setting when the training data were reduced to 80% of all audio files. The WER computed while testing the built language model resulted in higher WERs ranging between 55.68% and 59.72% as in Fig. 11. The minimum WER (55.68%) was reached with the number of Senones set to 1000 and 38 for the Gaussians.

It is clear that the language model performed poorly when tested on unseen audio files and this is mainly due to insufficient training data since when 100% of the training data was used, and it constituted around 40 min of audio data which made the Sphinx tools raise warnings about this small set of data. When only the training data were reduced to 90% and then 80% of all the audio files, it resulted in a data set of only 36 and 32 min respectively. However, the purpose of these experiments was assessing the potential performance of simplified Arabic phonemes for training the language model for the Holy Quran and it showed great potential.

6. Conclusion

The results from these different experiments while building the language model for the Holy Quran using simplified Arabic phonemes with a small amount of training data showed that it is possible to obtain potential good recognition accuracy provided that enough training data are used. Preliminary results have shown the potential of a simplification using Arabic phonemes in order to facilitate and automate the generation of the language model. However, more work needs to be done in terms of generalizing the process to ultimately cover all chapters of the Holy Quran. Additionally, overcoming the challenges of limited resources and processing power of mobile devices and implement a mobile application for the continuous recitation verification of the Holy Quran will increase its use.

Finally, this project can lead to future work on building the language model for different known narrations of the Holy Quran such as *Warsh* and *Qalun*. Furthermore, this simplification of the language model has the potential to be applied to any Arabic corpora which open the door to a wider number of applications in which Romanized phonemes were commonly used to develop the language model.

References

[1] El Amrani MY, Rahman MMH, Wahiddin MR, Shah A. Towards using CMU Sphinx tools for the Holy Quran recitation verification. In: 3rd Int Conf Islam Appl Comput Sci Technol, Konya, Turkey.

[2] Anusuya MA, Katti SK. Speech recognition by Machine: a review. Int J Comput Sci Inf Secur 2009;6.

[3] Lamere P, Kwok P, Gouvea E, Raj B, Singh R, Walker W, et al. The CMU SPHINX-4 speech recognition system. IEEE Intl Conf Acoust Speech Signal Process (ICASSP 2003), Hong Kong, vol. 1. p. 2–5.

[4] Walker W, Lamere P, Kwok P, Raj B, Singh R, Gouvea E, et al. Sphinx-4: a flexible open source framework for speech recognition. SMLI 2004:1–9, doi: 10.1.1.91.4704.

[5] Hyassat H, Abu Zitar R. Arabic Speech Recognition using SPHINX engine. Int J Speech Technol 2006;9:133–50. http://dx.doi.org/10.1007/s10772-008-9009-1.

[6] Muhammad WM, Muhammad R, Muhammad A, Martinez-Enriquez AM. Voice content matching system for quran readers. In: Proc Spec Sess – 9th Mex Int Conf Artif Intell Adv Artif Intell Appl MICAI 2010. IEEE; 2010. p. 148–53. http://dx.doi.org/10.1109/MICAI.2010.11.

[7] Satori H, Harti M, Chenfour N. Introduction to Arabic Speech Recognition using CMUSphinx system. In: Proc Inf Commun Technol Int Symp.

[8] Abu Shariah MAM, Ainon RN, Zainuddin R, Elshafei M, Khalifa OO. Natural speaker-independent Arabic Speech Recognition System based on Hidden Markov Models using Sphinx tools. In: Int Conf Comput Commun Eng. IEEE; 2010. p. 1–6. http://dx.doi.org/10.1109/ICCCE.2010.5556829.

[9] Elshafei M, Al-muhtaseb H, Al-ghamdi M. Speaker-independent natural Arabic Speech Recognition System. In: Int Conf Intell Syst. Bahrain; 2008.

[10] Hafeez AH, Mohiuddin K, Ahmed S. Speaker-dependent live Quranic verses recitation recognition system using Sphinx-4 framework. In: 17th IEEE Int Multi Top Conf 2014. IEEE; 2014. p. 333–7. http://dx.doi.org/10.1109/INMIC.2014.7097361.

[11] Arshad NW, Sukri SM, Muhammad LN, Ahmad H, Hamid Rosyati, Naim F, et al. Makhraj recognition for Al-Quran recitation using MFCC. Int J Intell Inf Process 2013;4:45–53. http://dx.doi.org/10.4156/ijiip.vol4.issue2.5.

[12] Ibrahim NJ, Zulkifli MY, Razak Z. Improve design for automated Tajweed checking rules engine of Quranic verse recitation: a review. QURANICA-Int J Quranic Res 2011;1:39–50.

[13] Ibrahim NJ, Idris MYI, Razak Z, Rahman NNA. Automated tajweed checking rules engine for Quranic learning. Multicult Educ Technol J 2013;7:275–87. http://dx.doi.org/10.1108/METJ-03-2013-0012.

[14] Mohamed SAE, Hassanin AS, Ben Othman MT. Virtual Learning System (Miqra ' ah) for Quran recitations for sighted and blind students. J Softw Eng Appl 2014:195–205.

[15] Yekache Y, Kouninef B, Mekelleche Y, Mohamed S. Building Quranic reader voice interface using sphinx toolkit. J Am Sci 2013;9:473–9.

[16] Ahsiah I, Noor NM, Idris MYI. Tajweed checking system to support recitation. In: 2013 Int Conf Adv Comput Sci Inf Syst. IEEE; 2013. p. 189–93. http://dx.doi.org/10.1109/ICACSIS.2013.6761574.

[17] Ali M, Elshafei M, Al-Ghamdi M, Al-Muhtaseb H. Arabic phonetic dictionaries for speech recognition. J Inf Technol Res 2009;2:67–80. http://dx.doi.org/10.4018/jitr.2009062905.

[18] Abro B, Naqvi AB, Hussain A. Qur'an recognition for the purpose of memorisation using Speech Recognition technique. In: 2012 15th Int Multitopic Conf. IEEE; 2012. p. 30–4. http://dx.doi.org/10.1109/INMIC.2012.6511440.

[19] Brierley C, Sawalha M, Heselwood B, Atwell ES. A verified Arabic-IPA mapping for Arabic transcription technology, informed by Quranic recitation, traditional Arabic Linguistics, and modern phonetics. J Semit Stud 2016;61(1):157–86. http://dx.doi.org/10.1093/jss/fgv035.

[20] Yekache Y, Mekelleche Y, Kouninef B. Towards Quranic reader controlled by speech. Int J Adv Comput Sci Appl 2012;2:134–7.

[21] Abu Shariah MAM, Ainon RN, Zainuddin R, Khalifa OO, Elshafei M. Phonetically rich and balanced Arabic speech corpus: an overview. In: Int Conf Comput Commun Eng ICCCE'10. p. 11–3. http://dx.doi.org/10.1109/ICCCE.2010.5556832.

[22] Verse By Verse Quran Files n.d. <http://www.everyayah.com/data/status.php> [accessed March 25, 2016].

[23] Download Quran Text – Tanzil n.d. <http://tanzil.net/download/> [accessed March 25, 2016].