IDENTIFYING THE CORRECT ARTICULATION POINT OF A QURANIC LETTERS OF THE THROAT (AL-HALQU) MAKHRAJ

Ahmad Al Baqir Bin Othman[1], Salmiah Ahmad[2, 3*], Khairayu Badron[1], Tareq M.K. Altalmas[3]

[1]Department of Electrical and Computer Engineering
Kulliyyah of Engineering,
International Islamic University Malaysia
[2]Department of Electrical Engineering,
College of Engineering and Technology,
Universiti of Doha for Science and Technology, Doha
[3]Department of Mechanical and Aerospace Engineering
Kulliyyah of Engineering,
International Islamic University Malaysia

Article history

Received: 22 Dec 2023

Received in revised form: 26 Dec 2023

Accepted: 26 Dec 2023

Published online: 29 Dec 2023

*Corresponding author: salmiah.ahmad@udst.edu.qa

ABSTRACT

Tajweed is an essential tool for a Muslim in reciting the Holy Quran as it ensures the recitation is properly done as what has been practiced by the Prophet Muhammad s.a.w and his companions. In Tajweed, mastering *makhraj*, or the Quranic letter point of articulation in pronunciation, is crucial for a Muslim. *Tajweed* is traditionally taught in "*Talaqqi*", where face-to-face sessions are commonly conducted with teachers, from early ages children. The fact is that the time and resources are limited especially in some areas in the world, where getting easy access towards a certified teacher for Quranic teaching and learning is a big issue. Considering this issue, with the enrichment of technology particularly in Artificial Intelligence (AI) nowadays, this limitation can be overcome. The conventional face-to-face learning can be supported with AI-based system for Quranic teaching and learning. There are few systems with regard to Quranic teaching and learning but most of them are focusing on memorization and recitation of the Quran rather than the *Tajweed*. This study presents the algorithm design, technique, and simulation of a Speech Recognition-based method to detect the correct articulation point (*Makhraj)* of the throat letters in the Quranic word. The study starts with a lots of data collection from experts in pronouncing the Quranic letters via developed platform and Apps called *SanaAlTajweed*, data augmentation, data pre-processing, features extraction using MFCC and deep learning model and validation using designed Graphical User Interface (GUI). Data was trained using an improved deep learning Convolutional Neural Network (CNN) classification model. Results shows that the algorithm was able to detect the letters produced at throat area, which are also known as *Izhar Halqi* letters. The rresults obtained have shown a promising achievement where the model has attained an accuracy of 89.39% on the test dataset with a loss of 0.4305. The outcomes may help to provide the most efficient way for

machines to analyse the Quranic recitation and translate it into a new model that will help human and computer to understand, interact, and improve.

***Keywords**: MFCC, CNN, Data Augmentation, Makhraj, Tajweed, Quran*

1.0 INTRODUCTION

The law of *Tajweed* governs the Quran recitation [1]. The author in [2] states that Muslims must observe Tajweed, the right pronunciation and intonation of the Quran, to ensure accurate recitation and retain the text's content. The author emphasises that Tajweed requires a lot of time to perfect and that traditional Tajweed teaching is face-to-face with tutors and limited in time. Automatic Speech Recognition in Quranic studies aids the teachers and allows everyone to learn Al-Quran independently [3]. Thus, Human-Computer Interaction based *Tajweed* learning is becoming more popular nowadays.

Makhraj or articulation points is a term used in Islamic linguistics to refer to the points of articulation of the Arabic alphabet's letters. It describes how to pronounce the letters correctly and is an important concept in Quranic teaching and learning. A *makhraj* of a letter refers to the portion of the mouth or throat that produces its sound, such as the lips, teeth, or throat. Scholars dispute on the precise number and names of the *makhraj*, but in general, there are five basic makhraj for the Quranic letters, which are empty space between mouth and throat (*Al-Jauf*), the throat (*Al-Halqu*), the tongue (*Al-Lisan*), between the two lips (*As-Syafatan*) and the nasal passage (*Al-Khaisyum*), as shown in Figure 1.

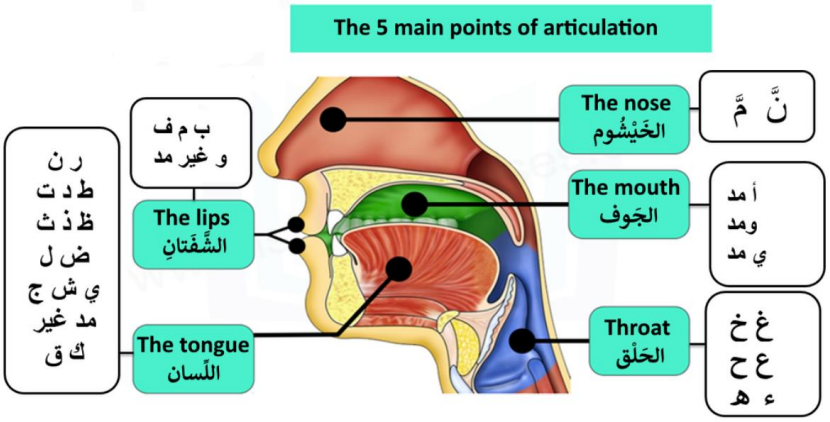


Figure 1: Classification of Letters Based on Its Articulation Point [3]

Human-computer interaction is the study of human-computer collaboration [4]. As new technology arises, human-computer interaction strategies have evolved over decades. The technology is essential for visually impaired individuals who cannot use a mouse and keyboard [5], via voice command. Additionally, it provides assistance to all users, making it a versatile solution for various needs.

The author in [4] describes voice recognition as converting speech signal to orthographic representation and transcribing it into text. Speech recognition improves Human-Computer

Interaction (HCI) by allowing computers to interpret human commands. Due to the necessity to automate simple human-machine interaction jobs, ASR technology is growing in popularity [1], which its common phase can be described in Figure 2.

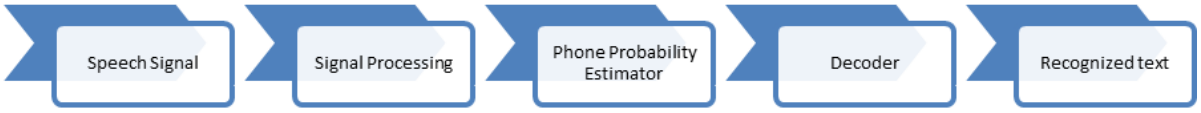


Figure 2: Common Phases of Speech Recognition [1]

Several studies have explored the speech recognition for *Quranic* recitation but most of the studies only focused on the memorization and the recitation without focusing on the *Tajweed* especially on the articulation points and they are mainly using the machine learning. The main design of speech recognition systems consists of three main approaches which are acoustic-phonetic approach, pattern-recognition approach, and artificial intelligence or deep learning approach [3]. Speech recognition using AI requires feature extraction, data and classification model for learning algorithm as shown in Figure 3. Pattern recognition and acoustic phonetic approaches are combined to create the AI or knowledge-based approach and some defines AI as neural networks, [6] and [7]. The authors emphasize that AI and simulations can improve human-computer communication and interaction.

Figure 1: Artificial Intelligence Approach Flow [8]

Speech recognition commonly uses MFCCs as features extraction to efficiently analyze the spectral characteristics and identify words and phrases, which is depicted in Figure 4. Speech recognition and other applications use its spectral representation of a sound. The anatomy of the tongue, teeth, and so on filters the sound emitted by humans, which determines which sounds are generated. We should be able to get a good approximation of the phoneme created if we could figure out what it looks like. The envelope of the short-time power spectrum displays the vocal structure, and the MFCCs' role is to appropriately reflect this envelope [9].

Fourier transforms decompose sound signals into their frequency components, creating MFCCs. The spectrum is then converted into coefficients that show the relative power of each frequency component at different time points. To improve the recognition accuracy, audio modelling and neural networks are often combined. The authors in [10] conducted Arabic speech recognition using MFCC features extraction to recognize letters of *sin* (س), *tsa* (ث), and *sya* (ش).

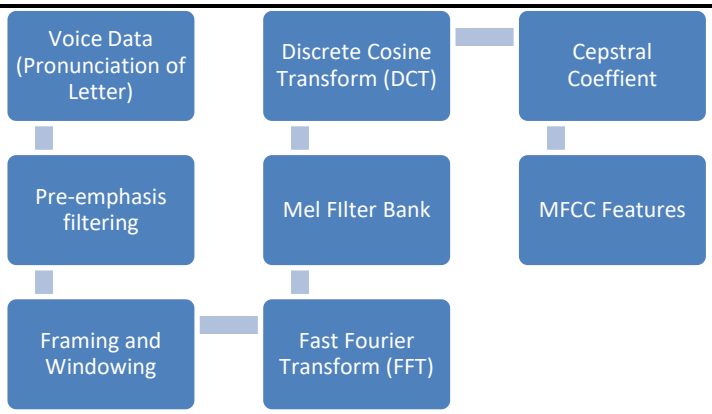


Figure 2: Block Diagram of MFCC Features Extraction [4]

A classification model is a type of algorithm that is used to predict a categorical label for a given input. One of the most widely used and well-known networks within Deep Learning is Convolutional Neural Network (CNN,) [11]. CNN are utilised to ensure that 2D input-data structures, such as image signals, are utilised to their full potential. This operation makes use of a very limited number of parameters, which not only makes the training procedure easier but also helps the network to function more quickly. CNN or ConvNets, are a subclass of neural networks that are most commonly employed in the context of image and speech recognition applications. The high dimensionality of images can be reduced by its built-in convolutional layer without any information being lost in the process. This method makes it possible to put the learning process into action even when there is no data that has been labelled available, [12].

The author in [13] made a study on real-time detection of *Mad Lazim Harfi Musyba* using a deep neural network method, CNN. The implementation of the method was facilitated using the TensorFlow GPU library. The results of the trials conducted using the Deep Convolutional Neural Network model indicated a detection performance of 9 verses with an average accuracy of 93.25 %. It can be concluded that the training data utilized in the CNN model is highly effective in identifying the *Mad Lazim Harfi Musyba* law.

This paper emphasizes on the identification of *Izhar Halqi* letters, which are produced in the throat area from a recorded continuous 28 letters Quranic Section 2 describes the methodology conducted throughout the process and the proposed technique, Section 3 highlights and discuss the results obtained using the proposed technique and Section 4 concludes the findings and recommends future actions accordingly.

2.0 METHODOLOGY

This section focuses on discussing the method that are utilized for the completion of the research. The detail explanations of the selection libraries, algorithm and software platform are also discussed in this section. The general methodology used in this paper is depicted in Figure 5 and Figure 6. Data collection was done to collect mass amount of data and in experimental setup, continued with data augmentation. Data pre-processing was done to the raw data to make

the data suitable for training our model. Next, feature extraction is determined to be used it for the classification model. In classification model step, the architecture of CNN Model is constructed and the pre-process data is used to train the classification model. Next, the trained model is tested and if the articulation point is correctly classified, we can proceed for the model and validation. If it is not correctly classified, hyperparameter of the model will be adjusted and the training model will be fed with more data. Finally, if the validation is satisfied, the model will be deployed for real time.

The proposed speech recognition system for Quranic recitation begins with specific collection of data to create a comprehensive dataset. The dataset is pre-processed before features extraction, classification model, and evaluation. The features extraction stage extracts data for classification model training. The classification model stage classifies speech samples into articulation points using a suitable model. Finally, the system is assessed using metrics to identify the areas for improvement. This algorithm aims to provide an efficient and accurate speech recognition system for specific *makhraj* of Quranic letters pronunciation.

2.1 Data Collection

The process of data collection for speech recognition of Quranic recitation is an important step in ensuring the accuracy and effectiveness of the recognition system. The first step in this process is to gather a dataset of audio samples of the Quranic letters' pronunciation. This dataset will be used to train the speech recognition model and will serve as the foundation for the system's performance.

One of the challenges in collecting this dataset is the potential for errors or variations in pronunciation that may be introduced through open public sampling. To mitigate this risk, it is important to take audio recording samples of the most fundamental block that makes up the Quranic recitation. Instead of collecting full recitations of the Quran, which can lead to complexity and variability, the focus should be on collecting audio for only *sukun* pronunciation technique. It is an effective method for ensuring the correct articulation of Quranic letters is by pronouncing the letters without vowel (*Sukoon*) and putting a Hamzah (ء) before it. This approach will help to identify where the letters are articulated from correctly in the mouth. This technique will help to reduce the window of uncertainty when dealing with uncontrolled public samples and ensure the accuracy of the speech recognition system. Only 28 combinations of Quranic alphabets as known as sukoon need to be recited by the reciter during the recording sessions for data collection, which can be seen in Table 1 and Table 2.

The information about respondents, including name, age, country of origin, Quranic education background, and gender, is collected through a survey conducted on a developed website called SanaAlTajweed. Those data used has been endorsed by certified Qari before proceeding to labelling.

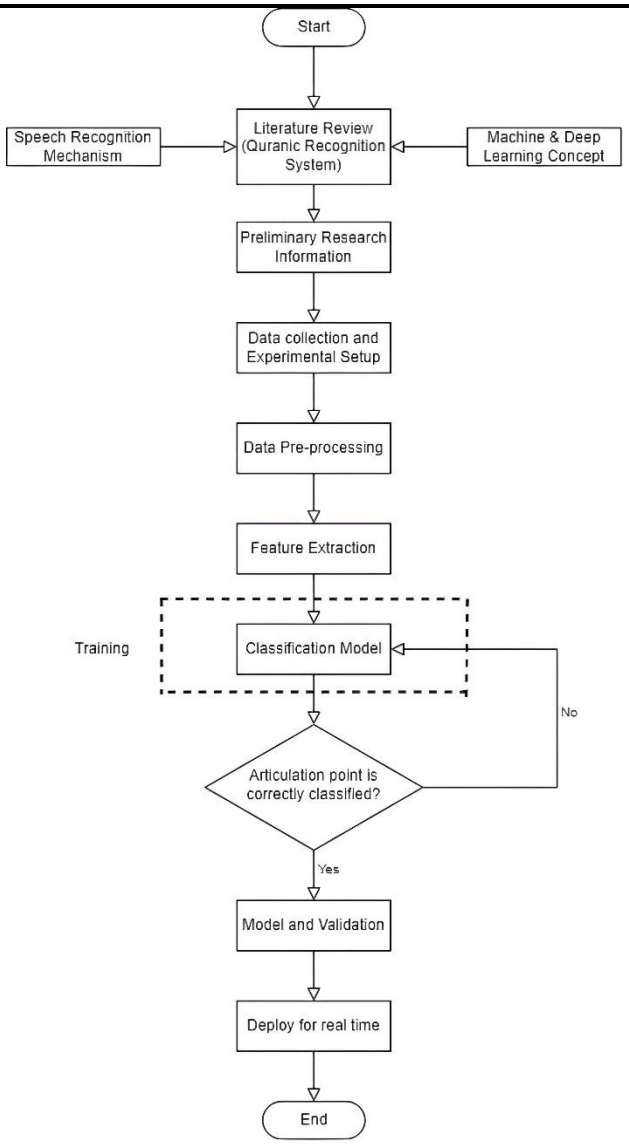


Figure 3: Flowchart of Methodology

Figure 4: Proposed Technique

Table 1: Table of 28 Combination of Letter [3]

Combination of Quranic letters	Pronunciation	Combination of Quranic letters	Pronunciation
اَءْ	a	أضْ	adh
أبْ	ab	أطْ	athd

أَتْ	at	أَظْ	azd
أَثْ	ath	أَعْ	a’
أَجْ	aj	أَغْ	agh
أَحْ	ah	أَفْ	af
أَخْ	akh	أَقْ	ak
أَدْ	ad	أَكْ	akk
أَذْ	az	أَلْ	al
أَرْ	ar	أَمْ	am
أَزْ	azz	أَنْ	an
أَسْ	as	أَهْ	ahh
أَشْ	ash	أَوْ	aww
أَصْ	asd	أَيْ	aii

Table 2: The Points of Articulation and Its Related Letters. [3]

The Place of articulation	**The letters**
The empty space in the mouth and throat	أ that is preceded by a letter with a *Fathah*. و with a *Sukoon* that is preceded by a letter with a *Dhammah*. ي with a *Sukoon* preceded by a letter with a *Kasrah*
The throat	Which it is included 3 levels; The deepest part of the throat (ء - هـ) The middle part of the throat (ح- ع) The closest part of the throat (closest to the mouth) (غ- خ)
The tongue	The deepest part of the tongue (ق - ك) Middle of the tongue (ش – ج - ي) The side of the tongue (ض - ل) The tip of the tongue (ر – ن – ت – د – ط – ظ – ذ - ث)
Interdental	س – ز – ص
The two lips	م – و – ف – ب

2.2 Labelling

Total of 70 reciters audio data samples that has been recorded and those were endorsed by certified Quranic Expert. After data was collected, each audio data is separated based on each letter and labelled using unique numbering. For example, for letter *Ba* (ب), holds the number of 2 based on the sequence of Quranic letters. There are 28 Quranic letters or *Hijaiyah* Characters. Thus, there will be 28 file that contains the data of each letter. For each data labels that have been separated according to their respective letters will be used later as the training

and testing data and for the deep learning model, a minimum number of 50 data for each label has been processed and carefully selected based on their audio clarity and accuracy in pronunciation.

2.3 Audio Data Augmentation

Since our data is somewhat limited, audio data augmentation is applied. Audio data augmentation refers to the process of applying various transformations or modifications to existing audio data to create new, synthetic examples. This technique is commonly used in deep learning tasks, such as speech recognition, music analysis, or sound classification, to increase the diversity and size of the training dataset. The goal of audio data augmentation is to expose the model to a wider range of variations and scenarios, making it more robust and capable of generalizing well to unseen data. By introducing the augmented examples during training, the model learns to be invariant to certain transformations or noise conditions, improving its performance in real-world scenarios. The process of audio data augmentation will multiply the amount of our data by modifying the current audio data. The modifications that are made to the audio data is time stretching, pitch shifting and adding the white noise.

2.4 Data Pre-Processing

The next important step is pre-processing of the data to prepare it for the training and testing phase of the system. This includes various tasks such as converting audio files into raw data, adjusting the sampling frequency, removing noise, and normalizing the data to ensure the dataset is ready for the model. These pre-processing steps are crucial in ensuring that the model can work with the data efficiently and effectively, leading to improved recognition results, as shown in Figure 7.

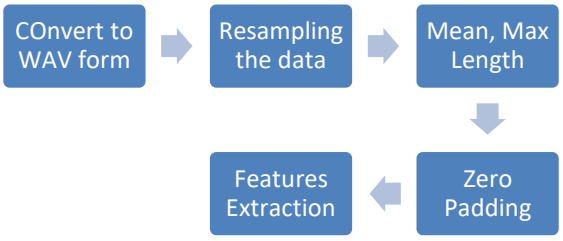
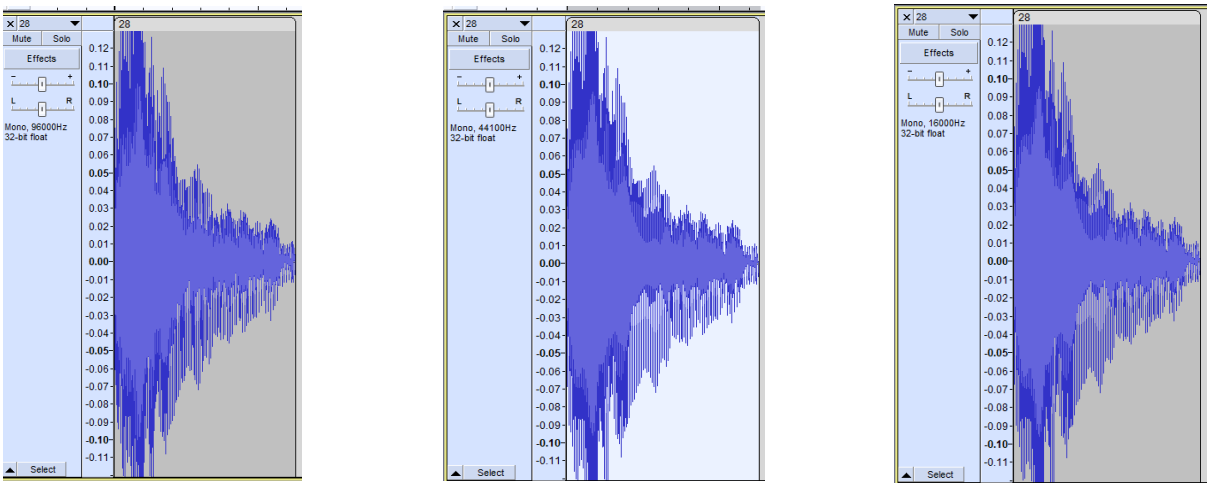


Figure 5: Flow of Data Pre-Processing

The raw data audio signal refers to the original audio data that is recorded, which can be in various formats such as MP3, WMA, or WAV. To ensure that the data can be efficiently processed by the software, it is necessary to convert the audio data into a raw format, such as the WAV file format, with the audio data is typically sampled at a frequency of 44.1kHz.

The purpose of this step is to ensure that the audio data is represented at a consistent frequency, this is important because different audio recordings may have been made at different sampling frequencies, and this could negatively impact the performance of the speech recognition system. By resampling the audio data to a consistent frequency, the speech recognition system will be able to process the data more accurately and efficiently. To achieve this step, TensorFlow_Io library is used. The frequency is set for resampling audio data to a

consistent frequency which in this project is 44.1 kHz and are optimized for speech recognition tasks, refer to Figure 8.



a) Before Sampling b) 44.1kHz Sampling Frequency c) 16kHz Sampling

Figure 8: Audio Signal Before and After Sampling

The data that has been obtained are in a large amount and different in playback time. To make sure the playback time for our training data is in constant time, mean, min and max of our data must be calculated. The mean of the data will give the average sample that our data had and from there the average of playback time can be calculated. The min and max will give the shortest playback time and longest playback time of our data respectively using the following equations.

$$Average\ Playback\ Time = \frac{Mean\ (kHz)}{Sample\ Rate} \quad (1)$$

$$Shortest\ Playback\ Time = \frac{Min\ (kHz)}{Sample\ Rate} \quad (2)$$

$$Max\ Playback\ Time = \frac{Max\ (kHz)}{Sample\ Rate} \quad (3)$$

Zero padding is a technique used in data preprocessing to handle inputs of varying lengths or sizes. It involves adding zeros to the beginning of a data sequence to make it conform to a desired fixed length. In this research, the length of the sequences is varied which from 0.26 second to 1 second. By adding zeros, the shorter sequences are extended to match the length of the longest sequence in the dataset or batch. This alignment allows the data to be processed more efficiently and in a consistent manner. The equation for zero padding is given in Eq. (4)

$$ZeroPadding = Playback\ Time\ (kHz) - Audio\ Sample\ Rate \quad (4)$$

2.5 Features Extraction

In this section, multiple types of features extraction are used to find the most suitable features

extraction. The feature extraction proposed are Mel Spectrogram, Spectrogram and MFCC. Among all, those features that produce more accuracy, and less losses will be chosen as the features for the letters.

a. Spectrogram

The audio waveform data is then converted into spectrogram data. We would need to convert the dataset from time domain signals into time-frequency domain signals to transform the waveform in the dataset from its current time domain format into a spectrogram type of data.

b. Mel Spectrogram

Mel spectrogram is a type of spectrogram that is generated by mapping the power spectral density (PSD) of a sound signal to the Mel frequency scale. The Mel-scale is a nonlinear scale that is based on the human perception of pitch, with higher resolution at lower frequencies and lower resolution at higher frequencies. By mapping the PSD to the Mel scale, the Mel spectrogram emphasizes the frequency regions that are most relevant to the human ear. The extracted sample is a tuple containing the Mel spectrogram of the audio signal and the corresponding label for the sample.

c. MFCC

Next method for feature extraction that are tested is Mel-Frequency Cepstral Coefficients (MFCCs). MFCC are a commonly used features in speech recognition. They are derived from the log power spectrum of the speech signal and capture the spectral envelope of the speech signal. MFCCs are based on the human auditory system and are designed to capture the characteristics of speech that are most relevant to the perception of speech.

For this project, MFCCs can be used to extract the spectral envelope of the speech signal and uses it as a feature for the recognition task. The spectral envelope captures the overall shape of the speech signal and was use to differentiate between different sounds and phonemes. For example, the spectral envelope of the sound "ب" will be different from the spectral envelope of the sound "ت". By extracting the MFCCs of the speech signal, we can use the spectral envelope as a feature to train a classifier to differentiate between the different sounds and phonemes. The matrix MFCC that contain the set of coefficients that capture the spectral envelope of our audio signal will be stored in the .json file.

2.6 CNN Classification Model

Convolutional Neural Networks (CNNs) are a type of deep learning model that are well-suited for image and signal processing tasks. They are composed of multiple layers of artificial neurons, which are trained to extract features from the input data, shown in Figure 9. CNNs are particularly useful for tasks involving visual data, such as image classification, object detection, and speech recognition.

In the context of this research, a CNN was used to extract the features from the speech signal that can be used to distinguish between different combinations of 2 letters. The CNN is

trained on a dataset of speech samples, where the correct articulation point of the combination of 2 letters is known, and then it can be used to predict the correct articulation point for new speech samples. Details algorithm is shown in Figure 10.

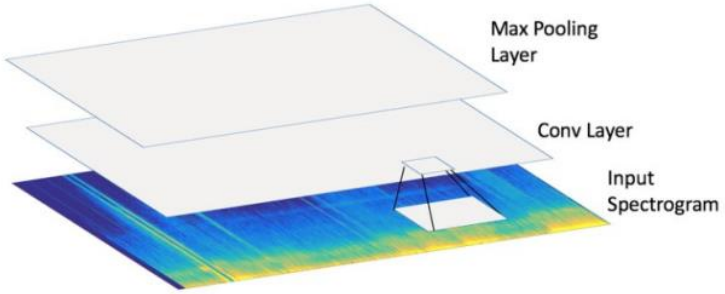


Figure 9: Explanation of Process of Spectrogram into Convolutional Layer

```
In [46]: model.summary()
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 770, 255, 32)      320

 max_pooling2d (MaxPooling2D  (None, 385, 127, 32)     0
 )

 conv2d_1 (Conv2D)           (None, 383, 125, 64)      18496

 max_pooling2d_1 (MaxPooling  (None, 191, 62, 64)      0
 2D)

 conv2d_2 (Conv2D)           (None, 189, 60, 64)       36928

 flatten (Flatten)           (None, 725760)            0

 dense (Dense)               (None, 128)               92897408

 dense_1 (Dense)             (None, 1)                 129

=================================================================
Total params: 92,953,281
Trainable params: 92,953,281
Non-trainable params: 0
_________________________________________________________________
```

Figure 10: Architecture and Summary of CNN Model

From the summary of CNN Model in Figure 10, we can see that the model has 11 layers in total, including 3 convolutional layers, 2 max pooling layers, 1 flatten layer, and 2 dense layers. The input to the model is a grayscale image of size (772, 257, 1). The first convolutional layer has 32 filters with a kernel size of (3, 3). The second convolutional layer has 64 filters with a kernel size of (3, 3). The third convolutional layer also has 64 filters with a kernel size of (3, 3). Each max pooling layer has a pool size of (2, 2). The first dense layer has 128 units, and the output layer has a single unit with a sigmoid activation function. The model has a total of 92,953,281 trainable parameters, which is a relatively large number. This indicates that the model is capable of learning complex patterns in the input data.

2.7 Evaluation

Evaluating the performance of a CNN-based model for identifying the correct articulation point in the Quranic letter's combination involves measuring how well the model is able to classify new speech samples based on their articulation points. It is important to evaluate the performance of the speech recognition system to ensure that it is accurately identifying the correct articulation points. Some evaluation that are done, using the following equations

$$Accuracy = \frac{Number\ of\ Correct\ Prediction}{Total\ Number\ of\ Prediction} \quad (5)$$

$$Precision = \frac{TP}{TP + FP} \quad (6)$$

$$Recall = \frac{TP}{TP + FN} \quad (7)$$

Where TP = True Positive, FP= False Positive, FN= False Negative.

3.0 RESULTS AND DISCUSSION

In this section, the analysis of the results from the model that have been developed in the methodology section are presented here. The data augmentation which involves changing the pitch, adding white noise, and time stretching audio samples are implemented to increase the amount of data and to make sure the model may adapt to changes in audio signal. The identification of the best features extraction technique for training a CNN model and a series of experiments were conducted to test multiple methods such as spectrograms, Mel-spectrograms, and Mel-Frequency Cepstral Coefficients (MFCCs). The CNN architecture, improvements that was made such as regularization, batch normalization, and dropout, and the model's performance on the training and test datasets are discussed. The results of predicting the shorter and longer duration of data recorded audio file also are discussed. For the experimentation analysis on this section, the identification was tested on the letters that belongs to the *Makhraj* of the throat (*Al-Halqu*), which has a total of 6 letters. This *Makhraj* prediction will reflect the overall performance of this research model accuracy.

3.1 Data Augmentation Result

Data augmentation is the process of artificially increasing the size and diversity of an audio dataset by applying various transformations or modifications to the existing audio samples, which is mainly done to increase the diversity and size of the training dataset, as can be seen in Figure 11.

a. Changing The Pitch Audio Sample by 1 Semitones

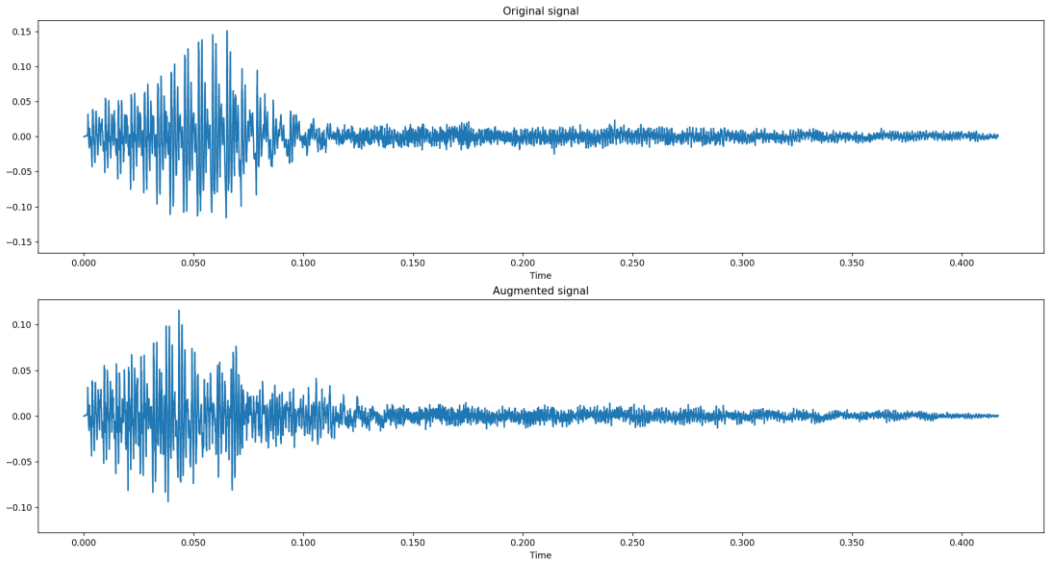


Figure 11: Comparison of Original Signal and After Changing the Pitch of ج

Figure 11 shows the original versus the augmented audio signal when changing the pitch by 1 semitone. The frequency of the audio wave increases and this results in a perceived higher pitch, as if the audio is being played on a slightly higher musical note. This is very important because Quranic pronunciation can be performed by individuals with different vocal characteristics and pitch ranges. By training the model on pitch-shifted samples, it learns to recognize recitation points regardless of the voice's specific pitch. This enhances the model's ability to handle variations in vocal styles and individual differences.

b. Adding White Noise by Factor Percentage of 0.1%

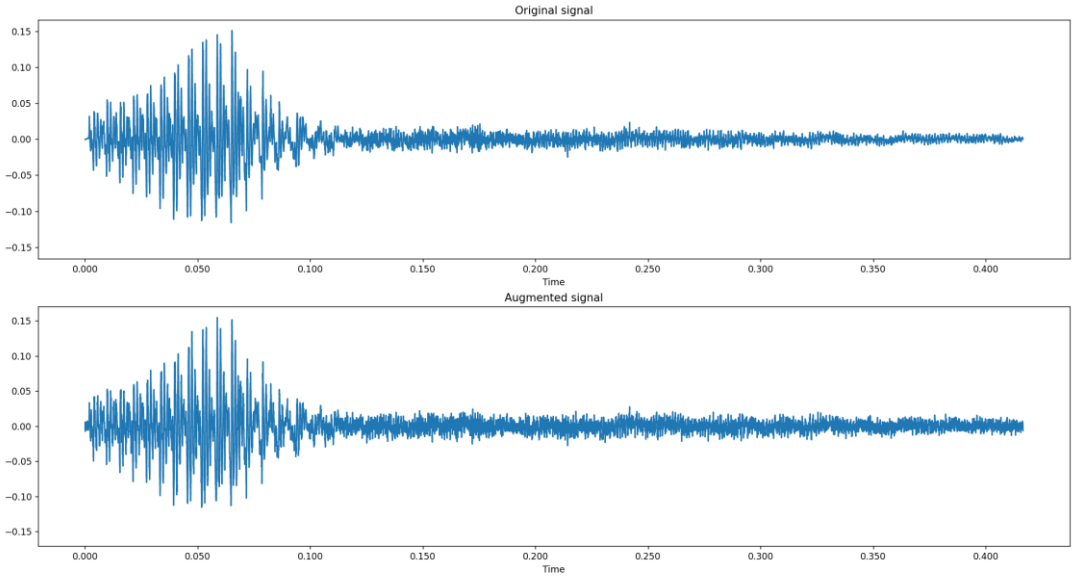


Figure 12: Comparison of Original Signal and After Adding White Noise of ج

As it can be seen on Figure 12, the audio signal of original and augmented is slightly difference. This is because adding white noise with percentage of 0.1 % means the amplitude of the added noise will be 0.1 % of the maximum amplitude of the original signal. White noise is a random signal contains equal energy at all frequencies. 0.1 % White noise will not change how the pronunciation of ح١ but only simulate the presence of ambient noise or environmental factors that can be encountered during real-world listening conditions.

c. Time Stretching with Rate of 0.8

Fig. 13 shows the difference between time stretching with rate of 0.8. When applying time stretching with a rate of 0.8 to an audio waveform, the duration of the audio wave will be increase by 20 % while maintaining the original pitch. This means that the audio will play back at a slower speed. The affect is it decreases the tempo, where it is slowing the playback speed, resulting in a longer duration for the audio. This can be useful because the pronunciation of ح١ is different for each people. Some of them pronounce it faster and some of them pronounce it slowly. This helps the training process so that the model can expose to wider range of variations.

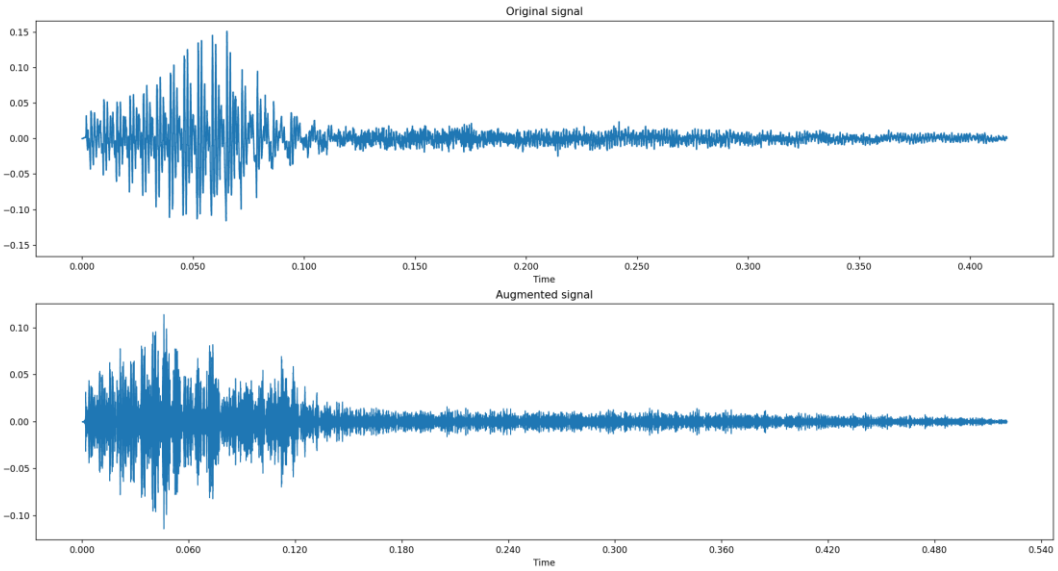


Figure 13: Comparison of Original Signal and After Time Stretching of ح١

3.2 Feature Extraction Result

A series of experiments had been conducted to test the multiple methods. Each feature extraction technique represented the audio data in a distinct way, such as Mel-frequency cepstral coefficients (MFCCs), spectrograms, or Mel-spectrograms, shown in Figure 14. The performance is compared based on losses, accuracy or precision to evaluate their effectiveness. Thus, through this comparative analysis, MFCC was determined to yield the best result with smaller losses and high accuracy.

a. Spectrogram

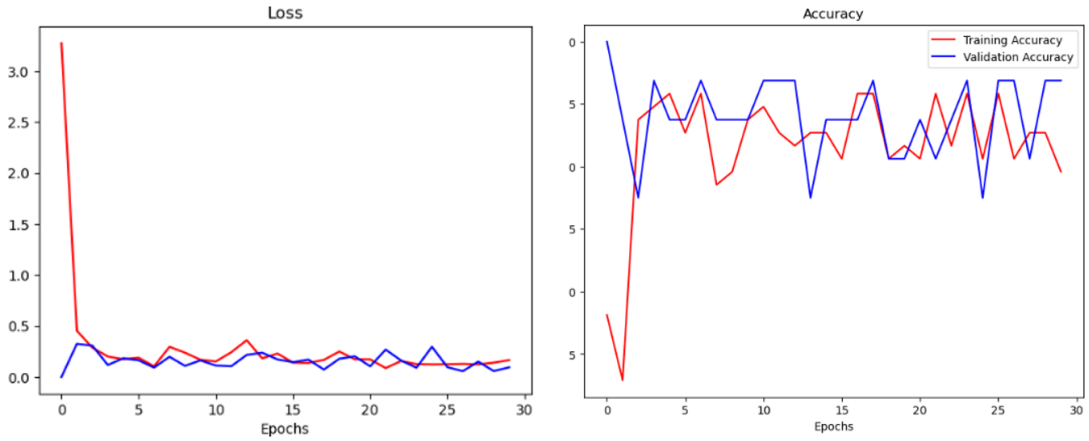


Figure 14: Loss and Accuracy for Spectrogram Features Extraction
(x-axis = No. Of Epochs, y-axis = Loss and Accuracy)

Based on the training loss, which is the red line from Figure 14, the loss is very low. It shows that the model is effectively fitting the training data and the model is learning from the training data and improving its performance on it. However, the validation loss is sometime higher than the loss. A high validation loss suggests that the model is not performing well on new, unseen data. It means that when the model is presented with examples from the validation set, it is not able to make accurate predictions, resulting in a larger discrepancy between the predicted outputs and the true targets. Based on Figure 14 also using spectrogram as features extraction gives low value of loss and high value of accuracy. But, from the graph, it also can be seen that both accuracy and loss are inconsistent throughout the epochs. This inconsistent training metrics can indicate that the model is not effectively learning the patterns in the data. As a result, the predictions made by the model will be unreliable or inconsistent as well. The model will struggle to generalize well to unseen data, leading to unpredictable or inaccurate predictions. Thus, it is not suitable to use it as features for these letters.

b. Mel-Spectrogram

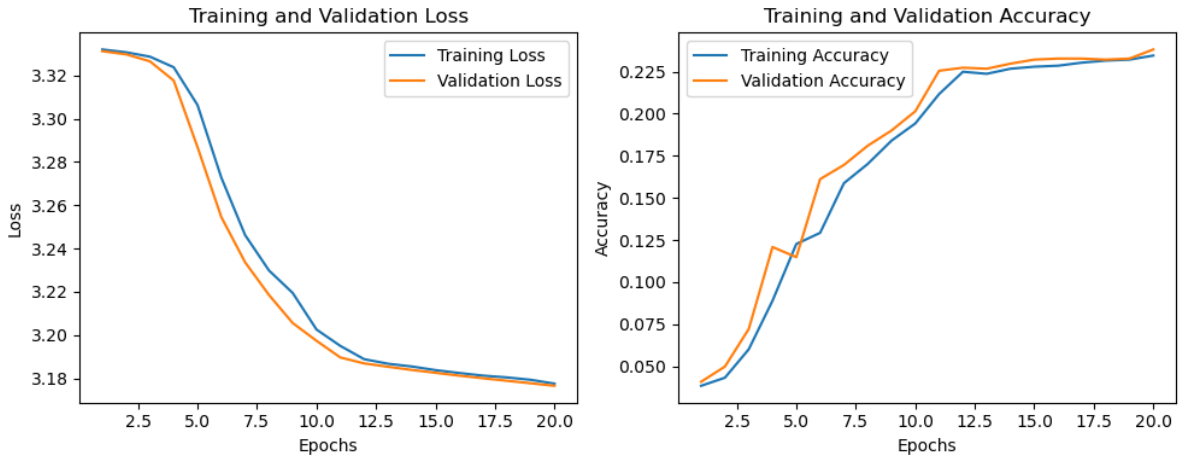


Figure 15: Loss and Accuracy for Mel-Spectrogram Features Extraction.

For Mel-Spectrogram features extraction, as shown in Figure 15, the loss is quite high which from 3.32 to 3.18. The accuracy also only at 0.225 which is very low. This indicates that the spectrogram does not adequately capture the discriminative features necessary for accurate classification or prediction. High losses also indicates that the model's predictions are less accurate and reliable. It implies that the model is struggling to capture the underlying patterns and information in the audio data, leading to limited predictive power. Low in validation accuracy also shows that the model is struggling to generalize well to unseen data. Low accuracy also suggest that the Mel-spectrogram features alone are not sufficient for capturing the complex patterns and discriminative information present in the audio data. It indicates that there may be other relevant features or representations that need to be considered to improve the model's performance.

c. MFCC

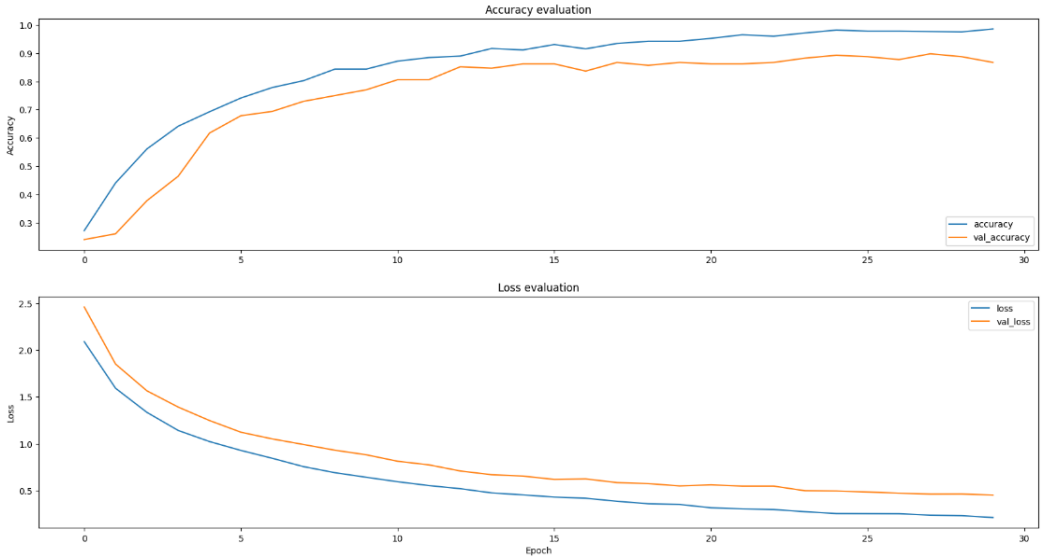


Figure 16: Loss and Accuracy for MFCC Features Extraction

For Figure 16, the accuracy is high and the losses is very low. High accuracy and validation accuracy indicate that the model is performing well and making accurate predictions on both the training and validation datasets. It suggests that the model has learned to capture the important patterns and features present in the MFCC representations. Models with high accuracy and validation accuracy are more likely to produce reliable and trustworthy predictions. This is especially important when this research where the accurate predictions are crucial because it is dealing with *Tajweed*. Lower losses and validation losses indicate that the model is effectively minimizing the discrepancy between predicted outputs and the true labels for both the training and validation data. This suggests that the model has good generalization capabilities and is not overfitting the training data. MFCC features are known to be robust to variations in the audio signals, such as changes in pitch, background noise, or speaker differences. The high accuracy and validation accuracy indicate that the model can effectively handle these variations and extract meaningful information from the MFCC representations.

3.3 CNN Architecture and Training Data Result

Figure 17 shows the improved CNN Model Architecture, as compared to Figure 10. CNN architecture in Figure 17 gives more accurate and capture important data. This architecture also prevent from overfitting happens, where overfitting is the model tends to 'memorize the data' instead of learning the data. The old model in Figure 10 is simpler with fewer parameters, which may make it faster to train but could potentially lead to lower performance or a higher risk of overfitting, depending on the complexity of the dataset and the improved models are more complex. The similarities between the 2 architectures are; both models use convolutional layers to extract from the input data and use MaxPooling2D layers to down sample the feature maps. Rectified Linear Activation Function (ReLU) also used as the activation functions for both architectures. Flatter layers are used to convert the 2D features maps into a 1D vector. Both models also have Dense Layer with ReLU and Sigmoid activation function for their final classification and Adam are used as the optimiser.

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 42, 11, 64)        640

batch_normalization (BatchN  (None, 42, 11, 64)        256
ormalization)

max_pooling2d (MaxPooling2D  (None, 21, 6, 64)         0
)

conv2d_1 (Conv2D)            (None, 19, 4, 32)         18464

batch_normalization_1 (Batc  (None, 19, 4, 32)         128
hNormalization)

max_pooling2d_1 (MaxPooling  (None, 10, 2, 32)         0
2D)

conv2d_2 (Conv2D)            (None, 9, 1, 32)          4128

batch_normalization_2 (Batc  (None, 9, 1, 32)          128
hNormalization)

max_pooling2d_2 (MaxPooling  (None, 5, 1, 32)          0
2D)

flatten (Flatten)            (None, 160)               0

dense (Dense)                (None, 64)                10304

dense_1 (Dense)              (None, 6)                 390

=================================================================
Total params: 34,438
Trainable params: 34,182
Non-trainable params: 256
_________________________________________________________________
```

Figure 17: Improved CNN Model

The difference between both architectures is explained below:

i. Input Shape:

For the old architecture, the input shape is directly specifying in the first Convolutional

layer. When the shape are directly defines, it will not be adaptable to different datasets without the need to modifying the code. For the improved version, a function called 'build_model' is used to takes the 'input_shape' as the parameter. With this function, it allows flexibility in specifying the input shape of the model based on the dataset.

ii. Regularization:

In older architecture of CNN in Figure 10, this architecture does not include any regularization techniques to the model. The regularization technique helps to prevent model from the risk of overfitting if the dataset is complex or limited in size. The improved version in Figure 18 applies L2 regularization to the convolutional layers by using '*kernel_regularizer=tf.keras.regularizers.l2(0.001)'*. This L2 regularization helps to add a penalty term to the loss function where it encourages the model to have smaller weight values. This regularization technique helps to prevent overfitting by reducing the complexity of the model.

iii. Batch Normalization

The older architecture does not have any batch normalization layers. This makes the model prone to training instability and inconsistency especially if the dataset has high variation or different scales across the features. While the improved version includes batch normalization layers using '*tf.keras.layers.BatchNormalization()*' after each of the convolutional layers. Batch normalization helps to normalize the activations of previous layer and help to accelerate and stabilize the training process. This normalization also acts as a form of regularization.

iv. Dropout

In the older version, the dropout layer was not included. Without the dropout, the model may be prone to overfitting especially in when the dataset is small, or the model capacity is high. In this research, the dataset is considered as small. Thus, dropout is added into the architecture. The improved version uses Dropout layer '*tf.keras.layers.Dropout(0.3)*' after the dense layers. Basically, dropout layers randomly set a fraction of input units to 0 during training. This helps to prevent overfitting by reduce the reliance of the model on specific neurons.

From Fig. 17, the model has 34,438 parameters, out of which 34,182 are trainable parameters and 256 are non-trainable parameters from the batch normalization layers. The first layer in the model is a 2D convolutional layer (Conv2D) with 64 filters of size 3x3. It applies a rectified linear unit (ReLU) activation function to introduce non-linearity. The output of this layer is then normalized using batch normalization (BatchNormalization), which helps to stabilize and accelerate the training process. A max pooling operation (MaxPooling2D) is applied to reduce the spatial dimensions of the feature maps while preserving the number of channels. The second layer is another 2D convolutional layer with 32 filters of size 3x3, followed by batch normalization and max pooling. These convolutional layers capture and learn spatial patterns in the data. The third layer is a 2D convolutional layer with 32 filters of size 2x2, along with batch normalization and max pooling. This layer further extracts features from the input data.

The output of the convolutional layers is then flattened into a 1D vector using the Flatten layer, which allows the subsequent fully connected layers to process the learned features. The next layer is a dense (Dense) layer with 64 units, which applies a ReLU activation function. This layer introduces non-linearity and learns high-level representations from the flattened features. Finally, the output layer is another dense layer with 6 units (equal to the number of classes), using a SoftMax activation function. This layer produces probability distributions over the classes, indicating the likelihood of the input belonging to each class.

The model is compiled with the Adam optimizer, which is an optimization algorithm that adapts the learning rate during training. The loss function used is the sparse categorical cross-entropy, suitable for multi-class classification tasks. The model is trained using the training data, with validation performed on a separate validation set. Early stopping is applied to monitor the accuracy and stop training if there is no improvement after a certain number of epochs.

The training progress is recorded in the history object, which contains information about the accuracy and loss of the model on the training and validation sets during each epoch. This information can be visualized using the plot_history function, which plots the accuracy and loss curves. After the training, the model is evaluated on the test set to measure its performance in terms of loss and accuracy. Additionally, a confusion matrix is generated to visualize the performance of the model in classifying the test samples. Finally, the trained model is saved for future use.

3.4 Epochs Cycles and Accuracy Relation

The value of the losses represents how well the model performs during training. It indicates the difference between the projected and actual outputs. A smaller loss number suggests that the predictions and actual values are more aligned.

Table 3 shows that the model starts with a high loss value and low accuracy in the first epoch. However, as training develops, both the loss and the accuracy eventually improve. The loss diminishes, showing that the model's predictions are growing more accurate. The percentage of successfully predicted samples in the training dataset is represented by the accuracy value. It assesses the model's overall performance. A higher accuracy means that the model predicts more accurately.

Table 3: Table of Accuracy and Loss of Each Epoch

Epoch	Loss	Accuracy	Epoch	Loss	Accuracy
1	2.4013	0.1972	16	0.4345	0.927
2	1.695	0.315	17	0.424	0.9281
3	1.4111	0.4981	18	0.3914	0.9219
4	1.2135	0.6236	19	0.3722	0.9334
5	1.0655	0.7081	20	0.3623	0.9347
6	0.9503	0.7721	21	0.3307	0.9475
7	0.8504	0.8003	22	0.3179	0.9488
8	0.7718	0.8027	23	0.3033	0.9475
9	0.7036	0.8643	24	0.3118	0.9526
10	0.6359	0.8643	25	0.2844	0.9577
11	0.5823	0.8822	26	0.2744	0.9577
12	0.5629	0.8873	27	0.2479	0.968
13	0.5107	0.9078	28	0.2327	0.9757
14	0.5004	0.8963	29	0.2277	0.9757
15	0.4616	0.9206	30	0.2281	0.9757

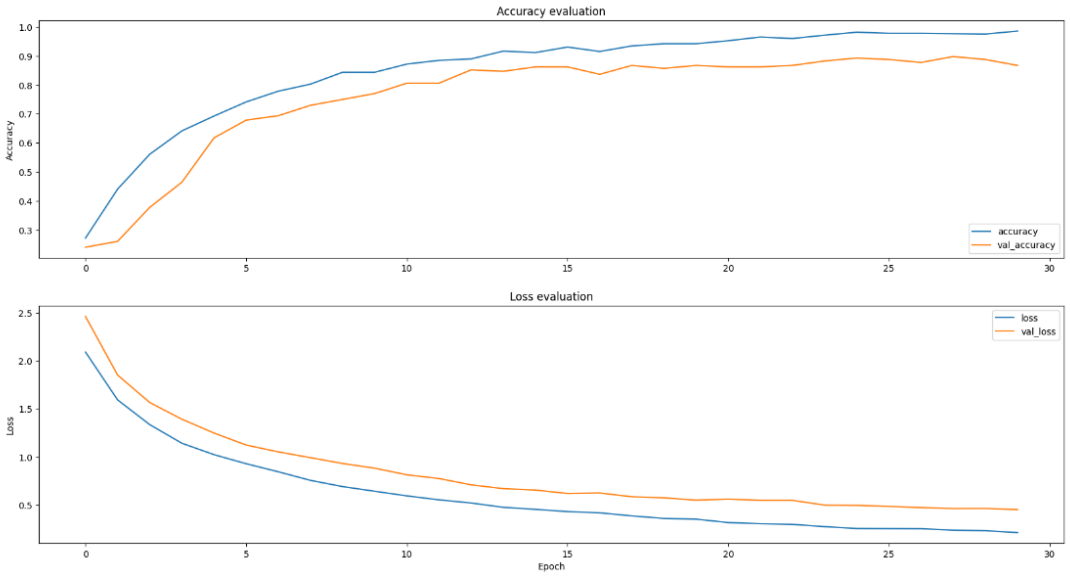


Figure 18: Accuracy and Loss Evaluation based On Number of Epochs
(x-axis = No. Of Epochs, y-axis = Loss and Accuracy)

Based on Figure 18, the model's loss and accuracy change over time as it learns from the training data. Concurrently, the accuracy rises, showing that the model is making fewer

mistakes. The model achieves a loss of 0.4305 and an accuracy of 89.39% on the test dataset towards the conclusion of training, indicating that the model is working reasonably well.

3.5 Confusion Matrix

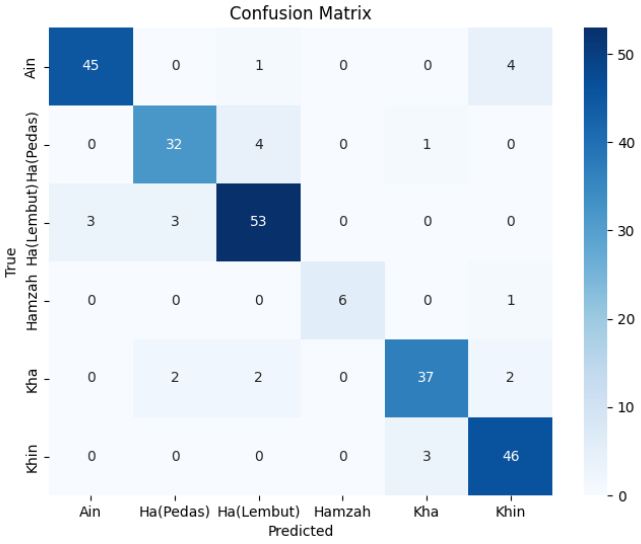


Figure 19: Confusion Matrix of Training Data

Based on the Figure 19 that shows the confusion matrix of the training data, if the 48 letter of اع was predicted, 45 out of 48 will be detected as اع which is true and 3 out of 48 will be detected as اح. As for 37 ,اح was predicted and 32 out of 37 was detected as اح while 3 was detected as اه and 2 was detected as اخ. As for اه, 60 was predicted and 53 out of 60 are detected as اه while 1 was detected as اع, 4 was detected as اح and 2 was detected as اخ. For اء, 6 was predicted and 6 out of 6 are detected as اء. For اخ, 41 predictions were made and 37 was detected as اخ while 1 is detected as اح and 3 was detected as اغ. Lastly for اغ, 53 predictions were made and 46 out of 53 are detected as اغ while 4 detected as اع, 1 detected as اء and 2 were detected as اخ.

3.6 Testing Result on Short Audio

Using the model that have been created, the model was tested with all the Quranic letter that belongs to *Makhraj* of the throat (*Al-Halqu*), The results are shown in the Figure 20 and Table 4.

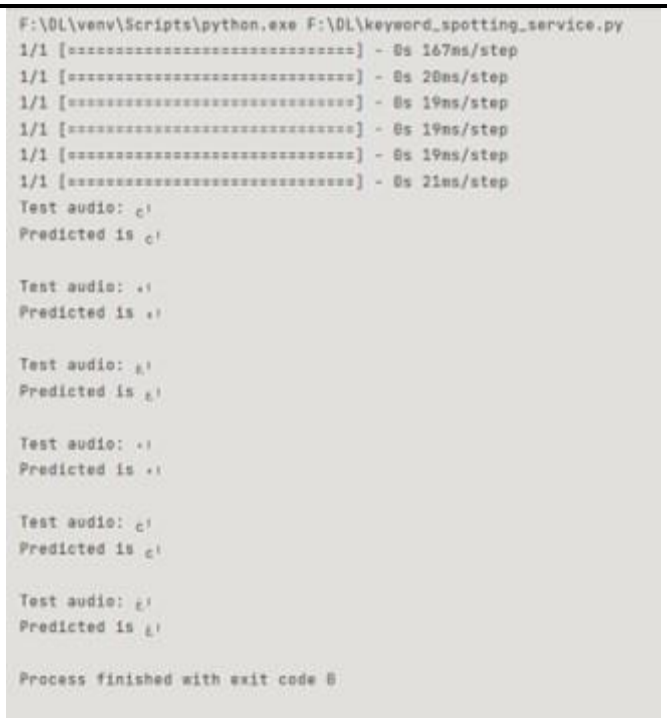


Figure 20: Predicted Result for Each of the throat (*Al-Halqu*) *Makhraj*

Table 4: Test and Predicted Result of Short Audio

Test	Predicted
اح	اح
اه	اه
اع	اع
اء	اء
اخ	اخ
اغ	اغ

3.7 Testing Result on Longer Data Recording

In this part, the process involves in creating a function for slicing a longer audio recording into smaller segments to predict a keyword. For each segment, the signal corresponding to the start and end time of the segments are extracted and pre-processed. The MFCCs are extracted from the signal segment, and a prediction is made based on the MFCCs. Each segment is predicted with the available dataset and if the segment contain keyword that are not in our data, it will return as 'Unknown Keyword'. The results are compiled as in Table 5.

Table 5: Test and result on longer recorded data

file_name	segment_number	start_time	end_time	predicted_keyword
AzhadNew.wav	1	0	2	اغ
AzhadNew.wav	2	2	4	اه
AzhadNew.wav	3	4	6	اخ
AzhadNew.wav	4	6	8	Unknown Keyword
AzhadNew.wav	5	8	10	اه
Husna.wav	1	0	2	اه
Husna.wav	2	2	4	اح
Husna.wav	3	4	6	Unknown Keyword
Husna.wav	4	6	8	اء
Husna.wav	5	8	10	Unknown Keyword
Husna.wav	6	10	12	اه
Husna.wav	7	12	14	اء
Husna.wav	8	14	16	Unknown Keyword
Husna.wav	9	16	18	اغ
Husna.wav	10	18	20	اغ
Husna.wav	11	20	22	Unknown Keyword
Husna.wav	12	22	24	اع
Husna.wav	13	24	26	Unknown Keyword
Husna.wav	14	26	28	اغ
Husna.wav	15	28	30	Unknown Keyword
Husna.wav	16	30	32	اغ
Husna.wav	17	32	34	Unknown Keyword

Based on Table 5, it shows the name of recording and the segment of each recording and also the predicted keyword. If the keyword could not be recognized, it returns the string of 'Unknown Keyword', There are many strings of 'Unknown Keyword' is because our model is only trained with 6 letter that belongs to group of *Makhraj* of the throat (*Al-Halqu*).

3.8 Graphical User Interface

A simple graphical user interface (GUI) was developed using tkinter library in PyCharm, which can be seen in Figure 21. Firstly, the GUI will first ask user to load the audio file that the user want to classify and proceed to preprocess the audio file before extracting the MFCC value. Then, the waveform and spectrogram of the audio file are extracted and will be used to display it in the GUI. The selected audio file will be classified using the model that has been trained before and will be predicted.

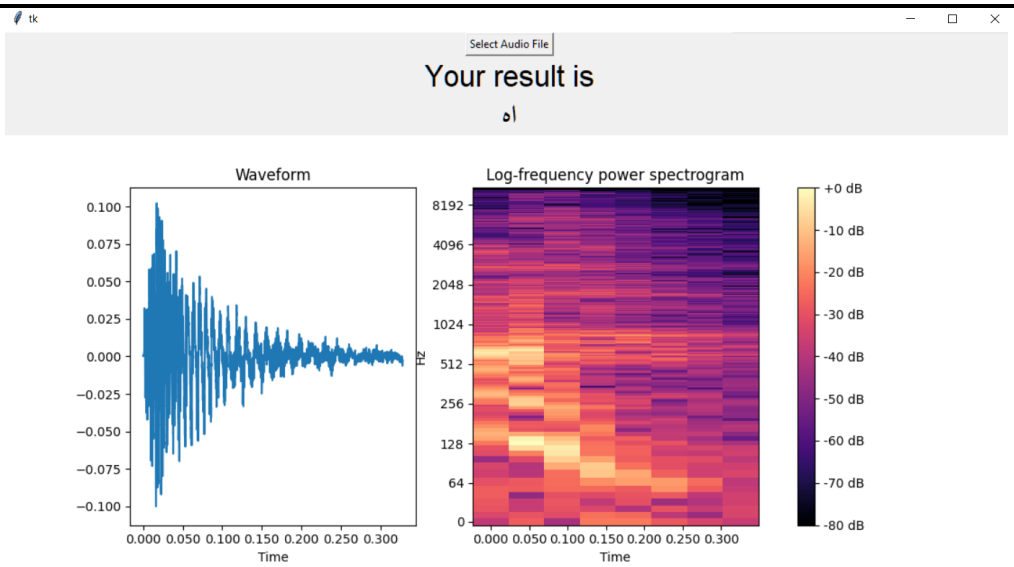


Figure 21: Result shown in GUI

Figure 21 shows the result of selected audio file. The audio file will be classified with our model that has been trained. The results will display the predicted output, the waveform of the audio file and the spectrogram of the audio file.

4.0 CONCLUSION

In this paper, data was collected through the https://www.sanaaltajwed.com/ and the data was endorsed by the certified Qari. The data that has been endorsed will be going through the standard processes for speech recognition including audio augmentation, data pre-processing features extraction using proven MFCC and an improved CNN deep learning model are used as the classification model with the loss of 0.4305 and an accuracy of 89.39%. The model was then was tested with shorter and longer recording of the Quranic letters. To conclude, this model is capable to predict the short recording more precise then to predict the longer recording of the Quranic letters.

REFERENCES

[1] Ridwan, T., & Majid, N. (2019, April). Development System for Recognize Tajweed in Qur'an using Automatic Speech Recognition. In *Proceedings of the 1st International Conference on Science and Technology for an Internet of Things.*

[2] Alagrami, A. M., & Eljazzar, M. M. (2020). Smartajweed automatic recognition of Arabic quranic recitation rules. *arXiv preprint arXiv:2101.04200*.

[3] Ahmad, S., Badruddin, S. N., Hashim, N. N., Embong, A. H., Altalmas, T. M., & Hasan, S. S. (2019, May). The Modeling of the Quranic Alphabets' Correct Pronunciation for Adults and Children Experts. In *2019 2nd International Conference on Computer Applications & Information Security (ICCAIS)*. 1-6.

[4] Ogundokun, R. O., Abikoye, O. C., Adegun, A. A., & Awotunde, J. B. (2020). Speech recognition system: Overview of the state-of-the-arts. *Int. J. Eng. Res. Technol*, 13, 384-392.

[5] Ibrahim, Y. A., Odiketa, J. C., & Ibiyemi, T. S. (2017). Preprocessing technique in automatic speech recognition for human computer interaction: an overview. *Ann Comput Sci Ser*, 15(1), 186-191.

[6] Haridas, A. V., Marimuthu, R., & Sivakumar, V. G. (2018). A critical review and analysis on techniques of speech recognition: The road ahead. *International Journal of Knowledge-Based and Intelligent Engineering Systems*, 22(1), 39-57.

[7] Singh, A. P., Nath, R., & Kumar, S. (2018, November). A survey: Speech recognition approaches and techniques. In *2018 5th IEEE Uttar Pradesh Section International Conference on Electrical, Electronics and Computer Engineering (UPCON)*. 1-4.

[8] Nassif, A. B., Shahin, I., Attili, I., Azzeh, M., & Shaalan, K. (2019). Speech recognition using deep neural networks: A systematic review. *IEEE access*, 7, 19143-19165.

[9] Khan, R. U., Qamar, A. M., & Hadwan, M. (2019). Quranic reciter recognition: a machine learning approach. *Advances in Science, Technology and Engineering Systems Journal*, 4(6), 173-176.

[10] Wahyuni, E. S. (2017, November). Arabic speech recognition using MFCC feature extraction and ANN classification. In *2017 2nd International conferences on Information Technology, Information Systems and Electrical Engineering (ICITISEE)*. 22-25.

[11] Yao, G., Lei, T., & Zhong, J. (2019). A review of convolutional-neural-network-based action recognition. *Pattern Recognition Letters*, 118, 14-22.

[12] Alzubaidi, L., Zhang, J., Humaidi, A. J., Al-Dujaili, A., Duan, Y., Al-Shamma, O., ... & Farhan, L. (2021). Review of deep learning: Concepts, CNN architectures, challenges, applications, future directions. *Journal of big Data*, 8, 1-74.

[13] Noeman, A., & Handayani, D. (2020, March). Detection of Mad Lazim Harfi Musyba Images Uses Convolutional Neural Network. In *IOP Conference Series: Materials Science and Engineering*. 771(1), 012030.