

# **SELF-LEARNING TECHNIQUES FOR ARABIC SPEECH SEGMENTATION AND RECOGNITION**

BY

**AHMED HAMDI ABO ABSA**

A Dissertation Presented to the  
DEANSHIP OF GRADUATE STUDIES

**KING FAHD UNIVERSITY OF PETROLEUM & MINERALS**

DHAHRAN, SAUDI ARABIA

1963 ١٣٨٣

In Partial Fulfillment of the  
Requirements for the Degree of

## **DOCTOR OF PHILOSOPHY**

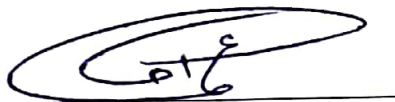
In

**ELECTRICAL ENGINEERING**

**February 2018**

KING FAHD UNIVERSITY OF PETROLEUM & MINERALS  
DHAHRAN- 31261, SAUDI ARABIA  
**DEANSHIP OF GRADUATE STUDIES**

This thesis, written by **Ahmed Hamdi Abo absa** under the direction of his thesis advisor and approved by his thesis committee, has been presented and accepted by the Dean of Graduate Studies, in partial fulfillment of the requirements for the degree of **DOCTOR OF PHILOSOPHY IN ELECTRICAL ENGINEERING.**



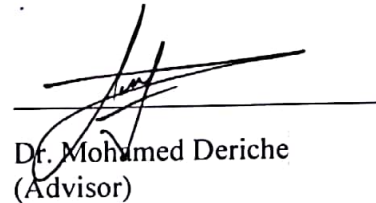
Dr. Ali Ahmad Al-Shaikhi  
Department Chairman



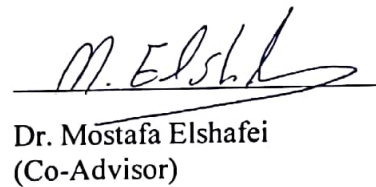
Dr. Salam A. Zummo  
Dean of Graduate Studies



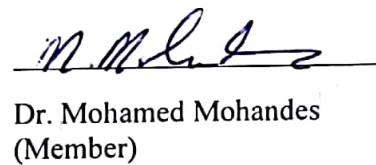
17/5/18  
Date



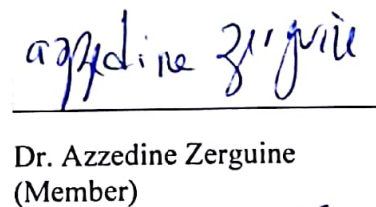
Dr. Mohamed Deriche  
(Advisor)



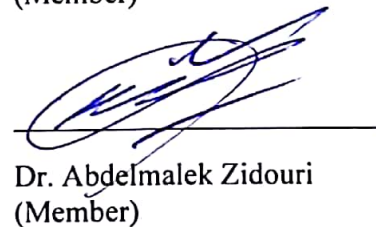
Dr. Mostafa Elshafei  
(Co-Advisor)



Dr. Mohamed Mohandes  
(Member)



Dr. Azzedine Zerguine  
(Member)



Dr. Abdelmalek Zidouri  
(Member)

© Ahmed Hamdi Abo absa

2018

|This dissertation is dedicated to my parents, my wife, my children and all family |

## ACKNOWLEDGMENTS

Praise is to Allah by whose grace good deeds are completed.

First, I would thank Almighty Allah for guiding me through making this dissertation and enabling my mind and body to fulfill it. I would thank Prophet Mohammad, Peace Be Upon Him, for transferring the message of Allah so that we can worship Him as he commands.

I would also give my thanks to my dear parents whom provided me with their sincere prayers for guidance in this work. I would also thank my precious wife and my sons Hamdi and Omar, in addition to the my newborn daughter Sahar for their patience during my PhD study period.

As I also thank my supervisor and academic father, Dr. Mohammad Derich who taught me the fundamentals of signal and speech processing and allowed me to attend all of the courses he taught. I have benefitted a lot from his excellent teaching skills, let alone his marvelous academic writing skills that I experienced when writing our papers.

As I also give my thanks to Prof. Mustafa AlShafei, the expert in Arabic speech processing who guided me to many concepts related to my works. I would also thank Dr. Mohammad Almuhandes who had introduced me to my supervisor, Dr. Derich. As I also thank Prof. Azzaddin Zerguin who taught me Adaptive Filtering. He was in deed a distinguished professor and I have learned a lot from him. I also want to thank Dr. Abdelmalek Zidouri for providing me with access to the Digital Signal Processing Lab.

I am also thankful to my friend and brother, Dr. Yousef Elarian for his help in many life issues. As I also provide my thanks to my neighbors: Husam Suwad (Abu Issa) for being a kind neighbor especially in the start of my studies, and to Ahmad Ghunaim, Saed Shirafy and Ala Talahma for all their helps.

I am also thankful to those who helped me in my dissertation, especially Dr. Qadri Mayyala, Mr. Mohammad Yahya, Dr. Mohammad Amro, Mr. Mohammad Samara, Mr. Mohammad Qannan, Dr. Irfan Ahmad and Dr. Mohammed Qurashi.

I also appreciate the Palestinian community for their endless help, especially Mr. Wasim Abu Alown and Raed Al-Masri. As I also express my deep appreciation to the “KFUPM Families” for the weekly meetings and familiar help, especially during my wives pregnancy and birth-giving.

|

# TABLE OF CONTENTS

ACKNOWLEDGMENTS .....	V
TABLE OF CONTENTS .....	VII
LIST OF TABLES.....	XI
LIST OF FIGURES.....	XII
LIST OF ABBREVIATIONS.....	XIV
ABSTRACT .....	XVII
ملخص الرسالة .....	XIX
<b>1 CHAPTER 1 INTRODUCTION .....</b>	<b>1</b>
1.1 Introduction .....	1
1.2 Overview of speech recognition.....	4
1.3 Research Motivation.....	8
1.4 Research Objectives .....	10
1.5 Thesis Organization.....	11
1.6 Thesis Contributions .....	12
<b>2 CHAPTER 2 LITERATURE REVIEW .....</b>	<b>14</b>
2.1 Introduction .....	14
2.2 Human Speech Segmentation and Recognition.....	15
2.3 Review of Segmentation Methods .....	17
2.3.1 Introduction .....	17
2.3.2 Explicit Segmentation .....	19
2.3.3 Implicit Segmentation .....	22

2.3.4	Arabic Speech Segmentation .....	25
<b>3</b>	<b>CHAPTER 3 HYBRID AUTOMATIC SPEECH SEGMENTATION USING MULTIPLE FEATURES AND A GENETIC ALGORITHM.....</b>	<b>29</b>
3.1	Introduction .....	29
3.2	The Proposed Automatic Speech Segmentation Algorithm .....	32
3.2.1	Pre-processing .....	33
3.2.2	Feature Extraction for Speech Segmentation.....	35
3.2.3	Speech Segments Detection .....	40
3.2.4	Post Processing of Detected Segments .....	41
3.2.5	Robust boundary prediction using fusion techniques .....	43
3.2.6	Segmentation using Genetic Algorithms.....	46
3.3	Performance Evaluation using the F-measure.....	49
3.4	Experimental Results .....	52
3.4	Conclusion .....	61
<b>4</b>	<b>CHAPTER 4 ENHANCEMENT SPEECH SEGMENTATION USING ENSEMBLE-BASE CLASSIFIERS AND A HIERARCHICAL TREE STRUCTURE .....</b>	<b>63</b>
4.1	Introduction .....	63
4.2	The system Architecture .....	66
4.3	Feature Extraction.....	67
4.3.1	The Energy Feature.....	68
4.3.2	Pitch Feature .....	68
4.3.3	Formant Frequencies.....	69
4.3.4	The Mel-Frequency Cepstrum Features .....	70
4.3.5	The Discrete Wavelet Transform .....	72



4.4	Dimension Reduction using PCA .....	76
4.5	Hierarchical tree classification .....	84
4.6	Individual Classifiers .....	85
4.6.1	K-Nearest Neighbor (KNN).....	85
4.6.2	The Multi-Layer Perceptron (MLP) Network .....	89
4.6.3	The Support Vector Machines (SVM) Algorithm .....	91
4.7	Ensemble Classifiers.....	96
4.8	Experimental Results .....	101
4.9	Conclusion .....	109
<b>5</b>	<b>CHAPTER 5 SPEECH SEGMENT UNIT CLASSIFICATION USING DEEP NEURAL NETWORKS.....</b>	<b>111</b>
5.1	Introduction.....	111
5.2	The system Architecture .....	113
5.3	K-means Clustering .....	113
5.4	Determining the optimal number of clusters .....	115
5.4.1	The Silhouette Index.....	115
5.4.2	The Elbow Method .....	117
5.5	Deep learning for speech analysis .....	118
5.5.1	Autoencoder .....	122
5.5.1.1	Manifold training with autoencoder .....	124
5.5.2	Softmax activation function .....	130
5.6	Experimental Results .....	135
5.7	Conclusion .....	141
<b>6</b>	<b>CHAPTER 6 CONCLUSION AND FUTURE WORK.....</b>	<b>143</b>
6.1	Conclusion .....	143

6.2 Future work .....	145
<b>REFERENCES.....</b>	<b>148</b>
<b>VITAE</b>	<b>162</b>

## LIST OF TABLES

Table 3.1 Arabic consonants [125] .....	32
Table 3.2 Precision, Recall, and F-measure, with different segmentation methods .....	53
Table 3.3 F-measure for different frame sizes with a fixed minimum number of frames/segment unit .....	55
Table 3.4: F-measure for different values of minimum frames/segment unit with a fixed frame width .....	56
Table 3.5: F-measure values after using GA for Energy, ZCR and Entropy features with different threshold types .....	58
Table 3.6: F-measure values for Basic Fusion technique with differnt $\Delta$ error tolerance .....	59
Table 3.7: F-measure values for linear regresion fusion technique with different error range .....	60
Table 4.1 The confusion matrix for the first stage in our HTR system (CV vs. CVC) ...	104
Table 4.2 Confusion matrix for second stage recognition in CVs .....	104
Table 4.3 Confusion matrix for second stage recognition in CVCs .....	105
Table 4.4 Average accuracy of Ensemble Classifiers with and without HTR .....	107
Table 4.5 Accuracy, precision, recall and F-measure using the second stage under CVs .....	108
Table 4.6 Accuracy, precision, recall and F-measure using the second stage under CVCs .....	109
Table 5.1 - Performance with different setups of the DNN .....	139
Table 5.2 - A comparison results between MLP and DNN .....	141

## LIST OF FIGURES

Figure 2.1: Summary of automatic speech segmentation techniques .....	28
Figure 3.1: Flow chart for the proposed automatic speech segmentation algorithm .....	33
Figure 3.2: a) Speech signal before and after pre-emphasis filter b) Spectrum of hamming window .....	34
Figure 3.3: Short time energy profile of a given speech signal .....	36
Figure 3.4: Short time zero crossing rate of the speech signal .....	38
Figure 3.5: Entropy of a typical speech signal.....	40
Figure 3.6: a) Automatic speech segmentation before post processing, b) after post processing using the energy feature, c) Using ZCR feature, d) Using the entropy feature.....	43
Figure 3.7: Block diagram for fusing multiple segmentation engines [31] .....	44
Figure 3.8: Block diagram of Genetic Algorithms .....	49
Figure 4.1: Flow Chart for the Proposed Arabic Speech Recognition System.....	67
Figure 4.2: MFCC parameter estimation .....	71
Figure 4.3: a) Daubechies-5 and b) Meyer wavelets. ....	73
Figure 4.4: The wavelet packet frequency bands decomposition with seven levels .....	75
Figure 4.5: Scatter plot of first and second of PCA-LDA projections.....	81
Figure 4.6: Histograms and simulated pdfs for the top feature from PCA-LDA projections .....	83
Figure 4.7: The proposed segment unit classification tree .....	85

Figure 4.8: An example for assigning a new vector to one of the classes using the KNN classifier .....	86
Figure 4.9: Typical standard of Multilayer perceptron network.....	90
Figure 4.10: SVM widen the margin between the two classes of data that are needed to be classified. ....	95
Figure 4.11: Classifier combination methods used by information level. ....	98
Figure 4.12: The proposed segment unit classification tree .....	102
Figure 4.12: The Overall Average Accuracy of Individual Classifiers with and without HTR.....	106
Figure. 5.1: Flow Chart for the Proposed Quran Speech Recognition System.....	113
Figure 5.2: Results for optimal number of clustering using the silhouette method.....	117
Figure 5.3: The percentage of variance explained for our data .....	118
Figure 5.4: Autoencoder structure .....	122
Figure 5.5: Variational autoencoder [122].....	125
Figure 5.6: Convergence of NN training (a), Zoomed region (b).....	136
Figure 5.7: Convergence of the NN with different number of hidden layers .....	137
Figure 5.8: a) Mesh plot of the confusion matrix b) Zoom the mesh plot.....	140

## LIST OF ABBREVIATIONS

ASR	Automatic Speech Recognition
IPA	International Phonetic Alphabet
ZCR	Zero Crossing Rate
HMM	Hidden Markov Model
MFCC	Mel Frequency Cepstral Coefficients
PLP	perceptual linear prediction coefficients
FFT	Fast Fourier Transform
ASR	Automatic Speech Recognition
DNNS	Deep Learning Neural Networks
TTS	text to speech
LPC	linear predictive coding
LVCSR	large vocabulary continuous speech recognition
GMM	Gaussian mixture model
TDRT	time-dependent recurrent trend
STFT	short time Fourier transform
ALCR	average level crossing rate
TIMIT	Texas Instruments Massachusetts Institute of Technology
GLR	the Brandt Generalized Likelihood Ratio
C/V	consonant and vowel
PSD	the power spectral density
K-NN	K-Nearest Neighbor
MLP	Multi-Layer Perceptron

NN	neural networks
CGD	Conjugate Gradient Descent
GD	gradient descent
SVM	Support Vector Machines
FNNs	Feed-forward neural networks
BP	Backpropagation
DBNs	Deep Belief Networks
SESM	The sparse encoding symmetric machine
PCA	Principle Component Analysis
LVASR	Large Vocabulary Automatic Speech Recognition
RTRL	real-time recurrent learning
BPTT	back-propagation through time
EKF	Extended Kalman Filtering techniques
LSTM	Long Short term Memory
ESN	Echo-State-Network
LM	language models
FD	False detections
FR	False rejection
FA	False Alarm
MD	Missed Detection
KACST	King Abdulaziz City for Science and Technology
PRC	Precision
RCL	Recall
HTRs	Hierarchical Tree Recognition Algorithms
DFT	The Discrete Fourier Transform

DCT	the Discrete Cosine Transform
DWT	the Discrete Wavelet Transform
LDA	Linear Discriminant Analysis
WCSS	Within-Cluster Sum of Squares

|



## **ABSTRACT**

Full Name : Ahmed Hamdi Abo absa  
Thesis Title : Self-Learning Techniques for Arabic Speech Segmentation and Recognition  
Major Field : Electrical Engineering  
Date of Degree : February 2018

Speech is the most natural form of human communication. Major achievements have been made in developing systems that automatically recognize human speech and respond (or take action) accordingly. An important preprocessing step in speech recognition systems, which plays a key role not only in recognition but also in a variety of other speech applications, is segmentation. Such a preprocessing step is important in identifying high level semantics of speech sounds including syllables, consonants, vowels, phones, dialects, ...etc. We show in particular, that such a step is crucial in properly analyzing Quranic (Muslim Holy Book) recitation. There are basically two general approaches used for speech segmentation, namely implicit and explicit approaches. Explicit segmentation uses a bottom up process and is based on the concept of fixed size speech frames. Such a framework has been heavily used in automatic speech recognition (ASR) systems based on the conventional Hidden Markov Model (HMM). The varying frame size or sample by sample approaches are mainly used in implicit segmentation approaches which are based on the detection of spectral distortions. The main objective of this thesis is to develop a comprehensive hybrid speech analysis system which includes robust segmentation and accurate classification for Arabic with particular focus on Quran. We develop a new framework that takes a recitation of the holy Quran as

input then creates speech segments by using a number of related special features. In addition to the traditional features, we introduce a new entropy based feature and show its relevance to the segmentation task. We then develop an approach for combining speech frames into what we call speech units using an optimization step. After obtaining the speech segment units, we develop our own framework for categorizing Quran speech segment units into a dictionary of around 22 super-classes with each covering around 30 sub-classes. For classification, we implement a number of basic classifiers including KNN, MLP, and SVM. We then test the performance of an ensemble-based classifier with very promising results. More importantly, and given the large number of classes, a Deep Neural Network (DNN) architecture was used for robust classification. We used the Autoencoder DNN model for recognizing the segment unit class with excellent results. We show that the DNN is better suited for more complex features where the inclusion of additional parameters and layers can better capture feature discrimination across a large number of classes. In summary, the dissertation provides a suite of new approaches for automatic Quran recitation systems including: robust segmentation, an optimum set of segment units, and finally a classification stage using DNN. |

## ملخص الرسالة

الاسم الكامل: أحمد حمدي أبو عبسة

عنوان الرسالة: تقنيات التعلم الذاتي في تقطيع الكلام العربي والتعرف عليها

التخصص: الهندسة الكهربائية

تاريخ الدرجة العلمية: ٢٨/٠٢/٢٠١٨ م

يعتبر الكلام من أهم وسائل التواصل بين البشر. ومن التطبيقات المهمة في هذا المجال هو كيف يستطيع النظام أو الآلة من خلال الكلام التعرف على مفهوم الصوت واتخاذ القرار المناسب. إن من أحد الطرق المهمة في التعرف على الكلام استخدام تقنية تقطيع الصوت إلى مقاطع. هذه المقاطع من الممكن أن تكون كلمات أو مقاطع لفظية أو مقاطع صوتية. ومن خلال الدراسات السابقة من الناحية العملية فإن هذه المقاطع لها أهمية كبيرة في التعرف على قراءة القرآن.

ولعمل التقطيع في الصوت يوجد نوعين: التقطيع الضمني والتقطيع التصريحي. في التقطيع التصريحي تعتمد على معرفة اللغة مسبقاً قبل التقطيع ويقوم على مقاطع صوتية ثابتة الزمن حيث أن أغلب الدراسات في هذا النوع تعتمد على نظام ماركوف الخفي. بينما في التقطيع الضمني يعتمد بشكل رئيسي على التغير في إشارة الصوت دون الاعتماد على زمن ثابت للمقطع. إن الهدف الرئيسي من هذه الرسالة هو تطوير نظام التقطيع الآلي والتعرف على هذه المقاطع اللفظية في القرآن الكريم. في هذه الرسالة قمنا بتطوير نظام جديد بحيث يتم إدخال الآية القرآنية ومن ثم يتم تقطيع الآية إلى مقاطع لفظية خاصة بالقرآن الكريم وذلك عن طريق استخدام ملامح وميزات متخصصة بالقرآن الكريم. بالإضافة إلى ذلك قمنا باستخدام خاصية مقياس العشوائية في الصوت والتي تعتبر أول مرة يتم فيها استخدام هذه الخاصية في القرآن الكريم. كما قمنا بعمل دمج بين الخصائص الأولية والحصول على المتغيرات الأمثلية عن طريق استخدام خوارزمية الجينات الوراثية والتي من خلالها تم الحصول على نسبة ممتازة جداً في النتائج.

كما قمنا في الرسالة بعمل طريقتين للتعرف على المقاطع الصوتية: الطريقة الأولى باستخدام خاصية الشجرة والطريقة الثانية استخدام تقنية الشبكة العصبية العميقة. في طريقة الشجرة قمنا باختيار ٢٢ صنف رئيسي بناءً على قواعد التجويد والتي تغطي ما يقارب ٣٠ صنف فرعي. كما قمنا في هذا النوع باستخدام ثلاث مصنفات (مصنف متعدد المستقبلات (الخلايا) ومصنف شعاع الدعم الآلي ومصنف ك-الجار الأقرب) وقمنا بعمل دمج بين هذه المصنفات وحصلنا على نتيجة جيد جداً. بينما في الطريقة الثانية قمنا باستخدام خوارزمية الشبكة العصبية العميقة وذلك بسبب كبر عدد الأصناف في المقاطع اللفظية القرآنية وتم الحصول على نتيجة ممتازة جداً.

إجمالاً في هذه الرسالة قمنا بعمل تقنيات جديدة في الصوت القرآني من حيث التقطيع والحصول على المقاطع الأمثلية ومن ثم التعرف على هذه المقاطع اللفظية.

# CHAPTER 1

## INTRODUCTION

### 1.1 Introduction

Speech is the most natural form of human communication. More particularly, there has been a major interest in developing advanced algorithms and systems for recognizing human speech and responding accordingly. Embedding systems with such capabilities in machines, has intrigued researchers for many decades. Many successes have been achieved but much more is needed for the future. This is especially true with the advances made in the context aware communication systems.

One particular application of interest is speech recognition. Automatic Speech Recognition (ASR) is defined as the process of converting audio waves (speech acoustic signals) to corresponding words or other linguistic units based on specific algorithms [1]. Automatic speech recognition appears in many applications and IT-solutions for industrial and civil areas such as: hands free operations, mobile voice application, human computer interaction, automatic translation, hearing aids for handicap, automatic dictation and simplified man-machine communication (via-voice systems). In speech recognition, it is easy to recognize isolated words, but the challenge is to recognize continuous speech. The performance of speech recognition systems can be affected by several conditions including: vocabulary size, speaker dependency, and noisy

environments, any others. The recognition performance increases when using small vocabulary and speaker-dependent conditions, while using large vocabulary and speaker-independent scenarios, the performance can significantly decrease.

One of the most fundamental preprocessing tasks before recognition is segmentation. While, we generally use fixed frame length in speech processing to preserve stationarity, the reality is that human voice comes out as a concatenation of speech units rather than frames. For this reason, it is important to develop robust algorithms to segment speech signals into fundamental units rather than frames.

Speech Segmentation is defined as the process of determining the boundaries between words, syllables, and phonemes of a certain language. Speech segmentation can be considered as a sub-problem for several speech processing applications as it is widely used in speech synthesis, speech recognition, speech compression...etc. Hence, an effective method of speech segmentation is essential especially when dealing with continuous speech.

Automatic speech segmentation is very important nowadays as it offers several advantages over manual segmentation. The manual segmentation is a traditional method usually performed by an expert phonetician based on careful listening and visual judgments to determine the boundaries of each phoneme or segment unit. Hence, manual segmentation is time consuming, tedious, and the results are very subjective because every expert phonetician have his/her own method of determining segmentation boundaries. Moreover, it is quite unpractical to carry manual segmentation with large datasets.

In general, there are two approaches used for automatic speech segmentation although some researchers may name them differently. Some researchers use the terms blind segmentation and aided segmentation [2], while others use the term implicit and explicit segmentations [3]. The implicit segmentation is based on splitting incoming utterances into segments without any linguistic information, while the explicit segmentation splits the utterance as in a transcription. However, the authors in [4], showed that implicit segmentation is a fundamental task in human speech development and language acquisition, and is considered more accurate as it specifically determines the start and the end of linguistic units.

The Arabic language has been used for more than 4000 years. It is the language of the Holy Quran, which is the main reference for all Muslims. It is worth noting that the Arabic letters are distinguished by their beautiful patterns. It is also one of the main languages written from right towards left. The Arabic letters are cursively written with most of these constructed from connecting portions of successive carved fragments with various curvatures [5].

The standard Arabic language has basically 35 phonemes; 6 vowels and 29 consonants. Standard Arabic language phonemes can be classified into two categories, vowels and consonants. The syllables in Arabic are based on the contrastive components that are contained in its structure. The successive contrastive elements within syllable boundaries are made up of segmental phonemes of the language. Each syllable has a main part which stands out with prominence. This part is referred to, here, as the “nucleus” of the syllable. The remaining components are referred to as ‘marginal factors’. The vowel always forms the syllable nucleus (e.g. *اَ*), while the consonants represent the

marginal phonemes in the syllable structure (e.g. ب) [5]. The number of syllables in an utterance is identical to the number of vowels[6].

The diversity in both spoken and written Arabic language brings with its major challenges. These challenges are especially reflected when reading Quran. The concept of combining syllables, stressing others, stopping somewhere else, is such a powerful one in highlighting the beauty and impact of the Quran. But with this beauty, brings with it major challenges.

## **1.2 Overview of speech recognition**

Speech recognition can be basically classified into two modes. The first is isolated word recognition; in which the words are surrounded by clear silence i.e. well known boundaries. The second is continuous speech recognition. The second mode is more difficult than isolated word recognition as the word boundaries are difficult to detect. A word may be uttered differently from one speaker to another due to the differences in dialects, gender and age; words may also be uttered in different ways by the same speaker as a result of emotion and illness. Temporal variability, due to differences in speaking rates, is easier to handle than acoustic variability introduced as a result of different pronunciations, accents, volumes, etc. [7]. In addition to speaker variability, all speech recognition systems are also affected by changes in the environment. The environment can introduce corruption to the speech signal due to background noise, microphone characteristics, and transmission channels...etc [8].

Speech recognition systems can be further classified as being either a speaker dependent or a speaker independent systems. A speaker dependent recognition system can extract uttered information from a specific speaker, or range of speakers, whose acoustic features have been previously obtained from a training speech database. This type of system is called a ‘Closed-set’ speech recognition system as the training dataset contains uttered information for all speakers. An ‘Open-set’ speaker independent speech recognition system is one where there is no uttered information for the recognized speaker contained within the training dataset.

Speech recognition can be achieved at a variety of levels of speech (Phone/ Phoneme/ grapheme, syllable, word, phrase, etc.). A phoneme is generally considered as “the smallest meaningful contrastive unit in the phonology of a language” [9]. Phonemes are defined by “minimal pairs” which produce a change of meaning if any of the phonemes is changed. Thus, phonemes are specific to a particular language. A phoneme can also be singular or can consist of a set of phones. While a phone is a single unit of speech sound, allophones are all possible spoken sounds that are used to pronounce a single phoneme. The human brain is presumed to perceive a given set of allophones as a particular phoneme [2]. If a phoneme is defined as the smallest unit of sound which can differentiate meaning, then a grapheme can be defined as the smallest unit in the writing system of any language that can differentiate meaning [10]. A grapheme can be a symbol or a letter. Sound–letter correspondence refers to the relationship between sound (or phoneme) and letter (or grapheme). A phonetic transcription system, like the International Phonetic Alphabet (IPA), transcribes the pronunciation of a language in a standard form. While a phonemic transcription system usually disregards all allophonic



differences and represents these using the same grapheme; it is also known as a representation of phonemic structure. However, graphemic systems use one-to-one letter to sound (phoneme to grapheme) rules for each word to generate a pronunciation dictionary [10].

Phonemes play a major role in most current continuous speech recognition systems. They are generally categorized into two main groups: consonants and vowels. A definition of vowels and consonants is stated in [11] as: “Vowels are produced without obstructing air flow through the vocal tract, while consonants involve significant obstruction, creating a noisier sound with weaker amplitude.” In the Arabic language, consonants are further categorized into four classes. These are: voiced and unvoiced stops (e.g. ب, ق respectively), voiced and unvoiced fricatives (e.g. ذ, ف respectively), nasal (e.g. م) and the trill (e.g. ر) and lateral (e.g. ل) classes. The fifth class covers long and short vowels [12].

Speech segmentation is a real challenge for continuous speech recognition systems. For limited vocabulary isolated word recognition, the problem can be easily solved by determining the correct boundaries of the isolated words and rejecting the artefacts of speech such as noise and intra-word stops. With regard to large vocabulary continuous speech boundary detection, on other hand, the problem becomes much more difficult because of the intra-word silences and other artefacts. These problems are tackled by applying robust speech boundary detection algorithms.

Speech activity detection algorithms can be applied on pre-emphasized speech signals to detect silence/speech boundaries. The most common methods used for end

point detection are the Energy and the Zero Crossing Rate (ZCR) features [13]. Typically, an adaptive threshold is applied, based on the characteristics of the energy, in order to differentiate between the background noise and the speech segments. However, such algorithms are very sensitive to the amplitude of the speech signal, such that the energy of the signal affects the classification results. This is especially a problem in noisy environments. Recently, a new end point detection algorithm has been proposed that uses Entropy contrast [14]. This algorithm uses the entropy feature rather than the energy of the signal. The calculation of the entropy is applied in the time domain. The authors show that the entropy is less sensitive to changes in amplitude of the speech signal.

Despite the use of the above techniques, current speech segmentation techniques do still introduce errors into the segmentation process. An alternative is to perform the recognition process without prior segmentation. This method includes the ‘silence’ as a phoneme, and the network is trained to recognize "silence" in the same manner as other patterns.

The phonemic representation of a given word is used in most speech recognizers to identify it. Thus, the availability of speech data that is time-aligned and labelled at phonemic level is a fundamental requirement for building speech recognition systems. Time-aligned phonemic labels can either be manually produced by expert human labelers or automatically produced using mathematical models. Though manually-aligned data is considered more accurate than automatically-aligned data [15], it is very time consuming to use it with large speech corpora. The most common method for automatic speech alignment is then so-called “forced-alignment.” The most common method for forced

alignment is based on building a phonetic recognizer using a Hidden Markov Model (HMM) [16][17][18].

Irrespective of whether the signal is segmented or not, all speech recognition systems use a feature extraction stage as an initial processing stage. The most common feature extraction techniques for speech are Mel Frequency Cepstral Coefficients (MFCC) [19], Fast Fourier Transform (FFT) [3] and Perceptual Linear Prediction coefficients (PLP) [20]. Typically, most ASR systems represent speech signals with the robust Mel Frequency Cepstral Coefficients (MFCCs).

### **1.3 Research Motivation**

There are two general problems in Automatic Speech Recognition (ASR); there are acoustic modeling, and time modeling problems. The acoustic modeling problem deals with signal representation while time modeling deals with speech temporal representation. The major challenge is to find the optimal set of features that best represent the spectral characteristics of speech, and track these with time. Some of the common spectral analysis techniques include FFT, LPC, PLP and MFCC using frame based analysis.

The problem in speech recognition can be reformulated as finding the best match between the estimated features and the different classes considered for the given application. There are, in general, three approaches in speech recognition; these are: Acoustic phonetic approaches, statistical pattern recognition approaches, and artificial

intelligence approaches [21]. Co-articulation adds another level of difficulty for the case of continuous speech where adjacent acoustic segments influence each other; which does not happen in isolated speech [22].

The process of finding the start and the end points in continuous speech segmentation is also a very challenging one. This task is not simple as the case of isolated word recognition. Manual speech segmentation is very tedious, time consuming and error prone. Furthermore, even if an expert phonetician performs the segmentation, the decisions are subjective and not reproducible.

Human adult can automatically, effortlessly and unconsciously, segment speech in real time. However, the task of automatic speech segmentation by machines is inherently an extremely difficult and non-trivial one [22][23]. Finding gaps or breaks or even silent segments is not as easy as the case of printed text. The co-articulation effect mentioned earlier in continuous speech makes the segmentation task even harder as speech discontinuities can't be easily identified [24]. Furthermore, there are no fixed cues or clear properties for the segmentation points. The only cue which is not fixed to any specific acoustic property is the abrupt change unit speech signal indicating the start of a new acoustic segment such as a phoneme, syllable or word. However, these kinds of abrupt changes can also occur because of noise and anywhere within the speech when there are spectral distortions.

Continuous speech often needs to be segmented into basic or fundamental phonetic units for recognition and synthesis. The syllable classifications of the form: Consonant-Vowel (CV) and Vowel-Consonant (VC) segmentation has traditionally been used, but is

unfortunately too simplistic [25]. If we look at the specific application of Quran reading in Arabic, for example, we have also an added level of complexity due to merging of consonants / vowels, as an example, in Arabic language when we say (من يعلم), in reading the Quran it will be like (مِيعَل).

Compared to other western and eastern language, the Arabic language has attracted lesser attention among researchers worldwide. Most of the reported studies conducted on Arabic language focused on basic vowels representation. A limited number of research studies have been carried out on more elaborate modeling and syllable classification such as consonants, pause, nasal...etc. [26]. Some of the most cited works focusing on Arabic language will be discussed in the next chapter.

## **1.4 Research Objectives**

The main objective of this thesis is to develop a new framework for Arabic Speech Segmentation and Recognition with special application to Quran recitation. In particular, the main objectives are:

- Develop algorithms for speech segmentation from the Holy Quran recitation to enable enhanced automated speech classification.
- Develop a new framework for representing short speech units that can enhance speech recognition tasks. The focus will be on statistical techniques combined with an information theoretic feature selection approach.

- Investigate the optimum set of Quran speech units that cover comprehensively the different recitation rules used in the Holy Quran.
- Develop a Quran speech database which includes a segmented and labeled verse-by-verse Quran recitation by different reciters.

## 1.5 Thesis Organization

This thesis is organized as follows:

- Chapter 2 provides a general overview of current methods for automatic speech segmentation and recognition systems. The different types of segmentations are first presented. Previous Arabic speech segmentation and recognition techniques are then presented. The review of classification techniques used in our work are also presented. This is followed by a discussion on ensemble based methods for combining classifiers. Finally, we briefly outline a deep learning neural network based approach for the recognition of the Arabic speech.
- Chapter 3 The developed Quran database used in this dissertation is described in details. Then, we briefly describe the speech segmentation and feature extraction techniques that we developed. A fusion technique of individual segmentation engines is introduced and investigated for speech segmentation. Also, an optimization algorithm is applied for finding the refined boundaries of segment units.
- Chapter 4 provides a description of the two new techniques used to enhance the Quran classification. The first technique is the hierarchical tree recognition based

on Tajweed recognition. The second technique is an Ensemble-based classifier combining several individual classifiers.

- Chapter 5 briefly describes the Deep Learning Neural Networks (DNNS) based model for Quran speech classification. The experiments and the results of using the Autoencoder algorithm for isolated speech segment unit recognition are discussed. A deep neural network approach for creating Quran acoustic models is finally presented.
- Chapter 6 provides a conclusion of the work undertaken as well as a discussion on future research directions on continuous Quran speech classification.

## **1.6 Thesis Contributions**

The major contributions of this thesis are:

- A comprehensive Quran database that is manually timed and syllabically labelled. This labeled database uses the KACST database for 10 professional reciters.
- New features are introduced for representing Quran speech recitation with good classification results.
- A new fusion technique of individual classifiers is introduced and used for optimal Quran speech segmentation.
- A hierarchical tree recognition approach is developed to solve the problem of speech classification with a large number of classes.
- An ensemble-based technique is introduced for the first time for enhancing Quran classification systems.

- We discuss a new approach for finding the optimal number of segment units useful for Quran recitation.
- Deep Learning Neural Networks (DNNs) acoustic models are discussed and used for Quran classification with superb results. |



## CHAPTER 2

### LITERATURE REVIEW

#### 2.1 Introduction

The basic concept of segmentation is to divide something or some object into discrete entities [27]. Speech segmentation can be used to divide continuous or connected speech into smaller discrete units that have different acoustic properties from their neighboring segments [2]. The discrete segments can be words, syllables or phoneme where segmentation may take place depending upon the purpose of the segmentation. Therefore, word segmentation is the process of dividing a continuous or connected speech into meaningful words.

Truly accurate segmentation can only be achieved manually [28]. However, manual segmentation is time consuming, expensive and may introduce human subjectivity in carrying out the segmentation [3] [29] [28]. Therefore, there is a need to develop good automatic segmentation. Numerous attempts were made in finding good algorithms for speech segmentation with satisfactory robust from the recognition point of view [30].

There are basically two general approaches for speech segmentation: implicit and explicit approaches [31] [4], also known as blind and aid approaches [2]. These approaches were implemented using different methods and processes. Some methods used bottom up process while others used top down processes. The processes themselves can be further categorized in term of using window processing frame which can be

classified as fixed framed as used in explicit segmentation or non- fixed framed or sample by sample as used in implicit segmentation based on the detection of spectral distortions. The developed methods are also different in terms of their focus in the post processing stage. Some segmentation methods, developed for text to speech (TTS) applications, require segmentation up to granular acoustic levels while others at word level segmentation mostly focus on automatic speech recognition (ASR). Based on the application of the segmentation process, the evaluation methods are also different. However, the general evaluation criteria are mainly match or hit, insertion and omission in comparison to some reference segmentation points. Segmentation works that include recognition use word error rate and recognition accuracy in algorithm evaluation.

## **2.2 Human Speech Segmentation and Recognition**

Human can understand conversational speech easily. In conversational speech, words utterances are spoken continuously and as a native speaker, a person generally can segment words in conversational speech easily without haste. However, when listening to speech in unfamiliar language, people experience difficulties hearing and detecting where one word ends and another begins. Most people assume that the speaker of that language speaks more rapidly than they do [32]. The situation is quite similar to speech understanding by infants. In the early months that of life, infants cannot segment words. However, it is only after seven to ten months, infants start to learn differentiating between words familiar to them [33], [32]. These facts show that word segmentation is usually

carried by humans for familiar words. Nevertheless, it is still not clear as to how exactly humans perceive these words [2].

Generally, the human process of speech understanding starts by discretely segmenting the words and collectively perceiving the meaning of the sentence. It is still debatable as to whether humans actually recognize words in a sentence via bottom up or a top down approach or perception. The bottom up perception is when the word is recognized from understanding the combination of smaller acoustics information like phonemes and syllables; which means segmentation starts with smaller to bigger acoustic models from phonemes to word. On the other hand, top down perception detects and recognizes word first, then the smaller acoustic information follows, which means that the segmentation of the word comes first, then the perception of smaller acoustic models from word to phonemes follows.

Based on the bottom-up perception, when a word is uttered, lower level acoustic models like phonemes or syllables are perceived by humans who then make sense of the combination of the acoustic models as the words in their memory. If the combination of the acoustic models leads to an unknown word, i.e. the word does not exist in the memory, then the person will not be able to recognize it. It will be like the situation where a person listens to a language not familiar to him. In contrast to the bottom-up perception, the top down approach stipulates that low level acoustic model is too fragile and small to be perceived as a meaningful unit. Therefore, it implies that humans only recognize word as a meaningful acoustic signal. It is only from the word, smaller acoustic models like phonemes and syllables that can be broken and perceived. Similarly, if the word is not in the person memory then the person may not be able to segment it or

understand it. Both approaches still follow the fact that humans can segment or recognize only words familiar to them.

As a brief conclusion, there are some important points to learn for human segmentation and perception. Speech segmentation and perception is possible only with familiar words whether perceived as top down or bottom up approaches. The top-down approach stipulates that phonemes are too small to be perceived as meaningful acoustic unit by humans before word perception. It is believed that humans perceive words first then smaller acoustic units within these words. While the bottom-up process starts perception via smaller acoustic models before the perception of words. Both processes still state that humans segment only words they are familiar with [32].

## **2.3 Review of Segmentation Methods**

### **2.3.1 Introduction**

Automatic speech segmentation is not an easy task as spoken speech is generally continuous and does not contain any reliable acoustic model for the blank spaces between words as in printed text [34]. The co-articulation effect in speech signals makes segmentation task even harder as it does not clearly mark any discontinuity in the speech signal [24] [35]. Due to co-articulation effect, the phoneme boundaries start to appear well within the previous phonemes [36]. In other words, the boundary of the following phoneme is within the frame of previous phoneme. To make the problem worse, there is no fixed cue or fixed properties indicating segmentation points. The only cue that is not

fixed to any specific acoustic properties is the abrupt changes of speech signal indicating new acoustic information such as a phoneme, a syllable or a word. Unfortunately, abrupt changes also occur due to noise, which may come from multiple sources.

Earlier work in speech processing treats segmentation as the task of start and end point detection as in isolated speech. The approach uses specific properties of the words in detecting the segmentation points. The authors in [22] were among the early researchers attempting to achieve segmentation and recognition of read mode connected digit utterances in room quality environment using the cues based on the specific digit properties. They used energy signal contour as digit properties with zero crossing rate (ZCR), Linear Predictive Coding (LPC) coefficients and error to detect voice and unvoiced signal in speech utterance indicating the segmentation points. They evaluated their performance based on the recognition rate of digits assuming that digits can be recognized hence the segmentation point is correct. Although they were able to obtain 87% digit recognition rate, in comparison to isolated digit recognition, there was 6% degradation in recognition accuracy. They concluded that degradations were due to co-articulation effects that exist in connected digits but not in isolated digits. The approach used [21] was a direct attempt finding the segmentation points the authors tried to find in an implicit way. Since then, many attempts have been discussed for the task of speech segmentation but most of the approaches avoid dealing with the segmentation problem directly or implicitly; instead they prefer to statically model the transitions between phonetic units or explicitly perform segmentation [37].

Automatic speech segmentation approaches can be divided into two major categories: implicit and explicit segmentations. Explicit segmentation algorithms are linguistically

constrained to a known phonetic sequence, while implicit segmentation requires no prior knowledge of the corresponding phoneme sequence [31]. Both approaches have their advantages and disadvantages in achieving segmentation. We will discuss previous studies carried under both strategies.

### **2.3.2 Explicit Segmentation**

In explicit segmentation, fixed size frames are used. The technique uses force alignment based on a given transcription sequence, leading to nearly no insertion as the number of segmentation marks equals that of the reference marks in the transcription [38]. This also leads to fewer omissions depending upon the given "time tolerance" with respect to the reference segmentation points. However, the explicit fixed size frames segmentation approaches require training of the acoustic information prior to segmentation. Furthermore, the boundaries between phoneme frames require delicate selection processes leading to poor match when small time tolerances are used. The co-articulation effects occur at phoneme level making segmentation at the phoneme boundaries an impossible task [35].

The explicit segmentation follows the human bottom up perception where word perception is understood via combinations of recognized phonemes and thus segmentation starts with phonemes. The most popular method in explicit segmentation is based on the Hidden Markov Model (HMM) [39][40][28][29][38][17].

The HMM uses fixed size windows of duration 10 ms to 50 ms. Speech is considered to be static within short periods of few milliseconds. Fixed size overlapped windows are used in automatic speech recognition. Heuristically speaking, when frames are used in segmentation, there will be a possible gap between the phoneme boundaries and the segmentation points. Since phonemes are small and slight, boundary selection in determining the segmentation points is a difficult task.

Realizing the limitation of the HMM method with forced alignment, most improved methods use fusion with segmentation refinement algorithms. In [17], the author used Fuzzy logic and Neural Networks for boundary refinement with the HMM. He compared the two refining methods and concluded that Neural Networks give better boundary refinement compared to Fuzzy logic. The evaluation of the work was based on the percentage of correct improvement in comparison to manual segmentation for a given time tolerance, and the segmentation performance was better when increasing the time tolerance. The experimental data used consistent of 80 sentences from Castilian Spanish uttered by one speaker.

In [40], the authors compared the segmentation performance of HMMs with different parameter refinement algorithms. They compared the Viterbi and Forward Backward algorithms and concluded that the Forward Backward algorithm with context dependent performs better than the Viterbi algorithm. The data used come from a large vocabulary continuous speech recognition (LVCSR) Dutch corpus. The evaluation method was based on segmentation deviation from the true segmentation points. The smallest deviation was 0.7% using "read aloud" and clean corpus while the deviation result using corpus with dialectical pronunciation increased to 7.5% at 100 ms tolerance.

In [39], the authors refined the HMM detected boundaries using a statistical model to match speech corpora. They used an English dataset and achieving a 93% hit rate.

In [38], the author compared the standard HMM and refined HMM using a Gaussian Mixture Model (GMM) at each segmented frame boundary. The results of the refined HMM outperformed the standard HMM. The performance used was based on the accuracy of the segmentation which was formulated from standard parameter performance matches, insertions and omissions. The accuracy of the refined HMM was 85% at 20 ms.

In [41], the authors improved the standard HMM based segmentation using a nonlinear dynamical method based on time-dependent recurrent trend (TDRT) for phoneme boundary adjustment. The work was tested using 38 utterances consisting of 380 sentences excerpted from the TIMIT database, achieving a 95.7% phoneme hit rate at a 30ms time tolerance.

Although most researchers tried to improve segmentation by enhancing boundary selection at phoneme level, the HMM still exhibits some disadvantages, discussed earlier, are summarized here:

- The reference patterns have to be generated before the method can be used [3].
- The reference patterns used to describe each frame do not always fit well the utterances, and do not account for all variabilities occurring in natural speech [3].
- HMMs based technique focus more on correct identification of the sequence of the phonemes, and not on accurately placing the phone boundaries [39].



- The HMM based segmentation model uses the Viterbi algorithm over the complete sequence of the input vector which is unsuitable for real time ASR application [42].
- The co-articulation effects present at the phoneme level, make the segmentation at the phoneme boundaries just like the HMM segmentation, challenge task [35].
- HMM based segmentation requires extensive training data [36], and has a high computational load.

### **2.3.3 Implicit Segmentation**

Under the implicit segmentation framework, segmentation is achieved without prior knowledge of the acoustic information on the phonetic transcription. Implicit segmentation basically uses the general properties of a signal for detecting significant spectral changes. According to [4], implicit segmentation is a fundamental task in human speech development and language acquisition, and it is considered to be more accurate and performs well as it specifically determines the start and end of the linguistic unit using no phonetic information [3]. Implicit segmentation is desirable as its implementation gives fewer misdetections or omissions [43]. A concept which disregards insertion points from a non-fixed windowed approach was discussed in by Brandt's algorithm which gives better segmentation results compared to the refined HMM [38]. However, since the approaches are linguistically unconstrained, they are expected to introduce numerous insertions.

Like the HMM based explicit approaches, some implicit methods were also applied with fixed size frames [3] [44] [45] [46] while others use variable size frames [47] [38]. As such, some implicit methods may also exhibit similar limitations to the ones in explicit methods.

In [3], The author introduced a hybrid method between implicit and explicit segmentations. The segmentation points were detected via spectral distortion between framed segments using the log amplitudes of the LPC then combined using an explicit approach that matches the reference spectral pattern. The author evaluated the performance via hit comparison to the reference phoneme. Using 90 randomly chosen words from the German, English and Dutch languages, which consist of 789 phonemes, the segmentation resulted in 96% hit rate at 30 ms.

In [42], the authors worked on a segmentation algorithm for robust speech detection. They used a statistical approach on the smoothed Short Time Fourier Transform (STFT) with non-parametric estimation of the background noise. Their experiments used speech in noisy environment and they applied word recognition based on segmentation points in the noiseless environment. The performance of the segmentation algorithm gave a WER of as low as 24%.

In [44], the focus was on syllable for segmentation. The authors used a minimum phase group delay approach derived from the short-term energy function. They tested the algorithm using the Switchboard and OGI-MLTS corpuses. The result reported was 85% match at 40ms time tolerance.

In [36], the authors proposed a segmentation method using the average level crossing rate (ALCR) to detect changes in the signal in the temporal domain. The proposed method gave 85% segment match on 100 sentences extracted from TIMIT (Texas Instruments Massachusetts Institute of Technology) speech corpus.

In [46], the authors proposed a segmentation method based on log energy and the ZCR as cues for the Malay utterances with different noise level. The authors claimed that nearly 90% of the words can be segmented currently. The evaluation was based on visual detection and playback.

In [48], the authors proposed an algorithm for automatic segmentation for the Indian language voiced speech by finding the entropy of the speech data which is placed for better performance. The results using the entropy based method reflected good performance than the energy-based techniques.

Some researchers also used different statistical approaches for automatic speech segmentation. For example, in [47], the authors introduced statistical methods that were based on sample by sample processing, unlike the previously mentioned methods which used frame based overlapping blocks. The authors used statistical approaches namely, the Brandt Generalized Likelihood Ratio (GLR), Divergence algorithm and Pulse Method. The author concluded the results with a detailed explanation on the usage of the methods on phonemic state and claimed that the statistical method able to detect phoneme with greater accuracy at voice and unvoiced part. The experiments used a French speech database, and out of 1534 phonemes only 12 omissions occurred.

In [38], the authors used the Brandt GLR methods to compare segmentation performance with the refined HMM, and concluded that ideal Brandt GLR method, which did not count insertions achieved better performance than the refined HMM method by more than 10% in accuracy. on experiments carried on music classification, it was found that the divergence algorithm performed better than the Brandt GLR algorithm [23].

Segmentation by implicit approaches like the divergence algorithm and the Brandt GLR method rely on the detection of signal spectral distortions which may occur at voiced and unvoiced transitions. This can lead to false segmentation points detection or insertion points. The only means to control these false segmentation points is via using the thresholds based methods on or/and parameters. Nevertheless, according to [49], these statistical methods are sensitive to the threshold value, and tuning the threshold and the parameters to reduce insertions can unfortunately lead to an increase in omissions.

### **2.3.4 Arabic Speech Segmentation**

Given the peculiarities and richness of the Arabic language, we have also seen a number of systems developed specially for Arabic speech segmentation. In [50], the authors proposed an implicit algorithm for Arabic speech consonant and vowel (C/V) segmentation. The proposed algorithm does not need special linguistic information; it is totally dependent upon a basic wavelet transformation, and spectral analysis to detect transients that happen in C/V. They applied the algorithm on 20 Arabic words recorded six times. They achieved 88.3% accuracy or C/V segmentation. In [51], the authors

developed a phoneme segmentation algorithm using the Fast Fourier Transform (FFT) based spectrogram. They achieved an overall segmentation accuracy of 95.39%. In [52], the authors developed a basic speech segmentation for recognition. They used a number of different cues with the best results achieved using the power spectral density (PSD) and the zero crossing rate (ZCR). They achieved 89% accuracy for eight different speakers.

In [53], the authors discussed an algorithm for vowel identification that uses formant frequencies. The investigation was carried based on Holy Quran Tajweed rules. The system showed up to 90% average accuracy on continuous speech files of 1000 vowels. Some researchers also applied statistical techniques (borrowed for other languages) on Arabic language. For example, in [54], the authors applied the principles of eigenvalues and eigenvectors for the first time in segmentation. A success rate of 80% was achieved for a small database of 50 spoken words. Also in [5], the authors evaluated the maxima and minima points of the energy curve and some other statistical measures. The average segmentation accuracy achieved across several speakers was 85.4%.

In [55], the author used a supervised Arabic speech segmentation based on the phoneme level. The author used the Short Time Fourier Transform to predict the phoneme boundaries based the locations of main energy changes in frequency over time. The segmentation accuracy achieved was 81%.

In summary, as mentioned earlier, there are two categories in automatic speech segmentation: Explicit and Implicit segmentations. The main difference between these is the knowledge of the transcription of the language before the automatic segmentation.

Also, we note that explicit segmentation mainly depends upon the classification algorithms used to refine the segmentation parts, but the implicit segmentation depends on the feature extraction algorithms to detect the changes in speech. The techniques were used before in explicit and implicit segmentations are summarized in Figure 2.1. The shadow boxes in Figure 2.1, represent the techniques that were used before in Arabic language.

In the next chapters, we will propose new techniques for automatic Arabic speech segmentation and classification, especially for Quran recitation.

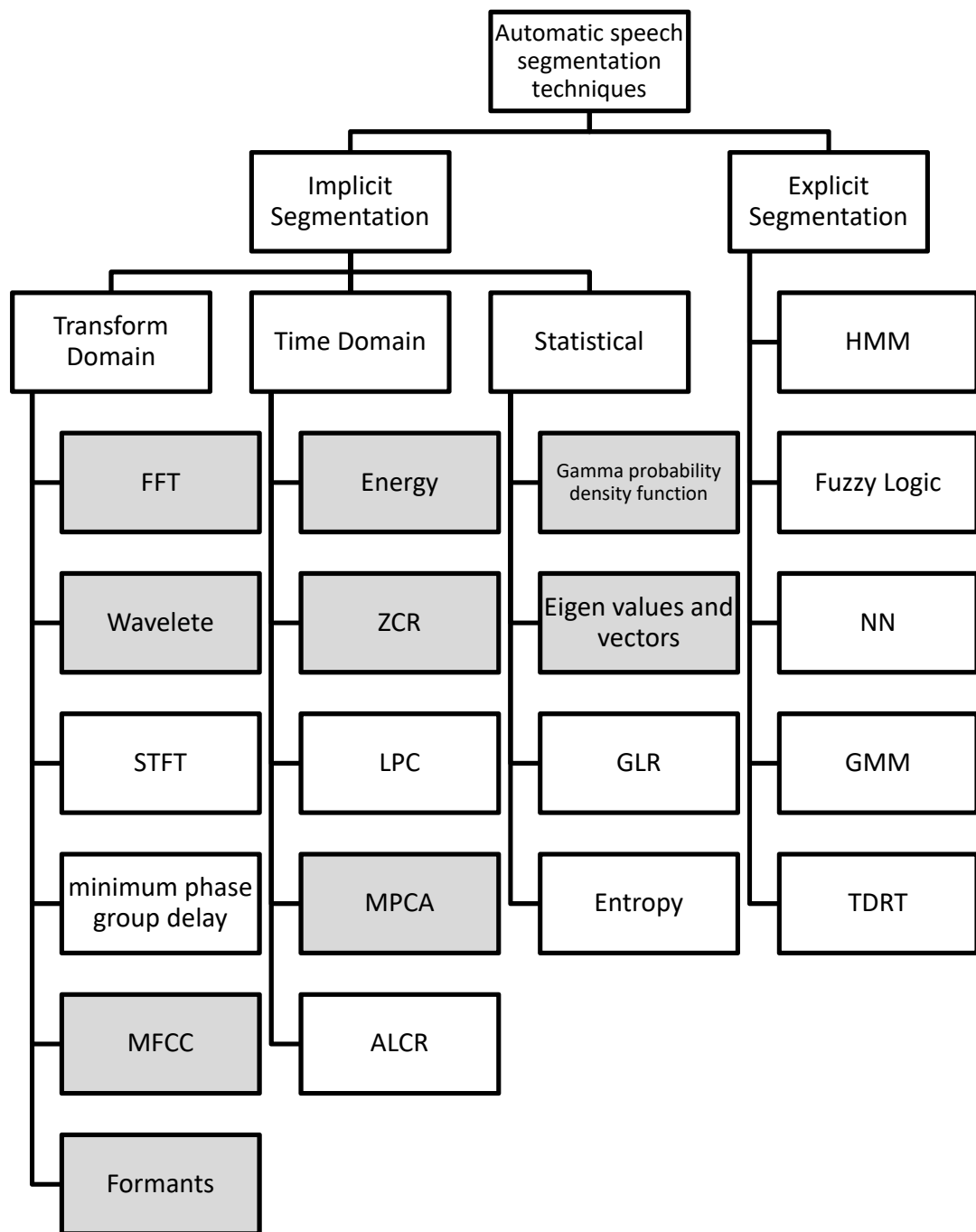


Figure 2.1. Summary of automatic speech segmentation techniques

# **CHAPTER 3**

## **HYBRID AUTOMATIC SPEECH SEGMENTATION**

### **USING MULTIPLE FEATURES AND A GENETIC**

### **ALGORITHM**

#### **3.1 Introduction**

Automatic Speech Recognition (ASR) is defined as the process of converting audio waves (speech acoustic signals) into the corresponding words or other linguistic units based on certain algorithms [1]. Automatic speech recognition is cornerstone to many applications and IT-solutions for industrial and civil purposes including: hands free operation, mobile voice applications, human computer interaction, automatic translation, hearing aids for the handicaps, automatic dictation, and simplified man-machine communication (via-voice systems), to mention a few. In speech recognition for human machine interaction, it is easy to recognize isolated words, but the challenge is to recognize continuous speech. The performance of speech recognition systems in human machine interaction is highly affected by several features such as vocabulary size, speaker dependency, and different type of noise [2] [3].

One of the most fundamental preprocessing tasks before recognition is segmentation. While, we generally use fixed frame length in speech processing to



preserve stationarity, the reality is that human voice comes out as a concatenation of speech units rather than frames. For this reason, it is important to develop robust algorithms to segment speech signals into fundamental units rather than frames.

In simple terms, speech segmentation can be defined as the process of determining the boundaries between words, syllables, and phonemes, of speech. Speech segmentation can be considered as a fundamental sub-problem for numerous speech processing systems including synthesis, recognition, ...etc. Hence, robust methods for speech segmentation are crucial in speech processing.

Naturally, automatic speech segmentation offers several advantages over manual segmentation. Manual segmentation is time consuming, tedious, and the results are very subjective as every expert phonetician has his/her own method for performing the segmentation task. Moreover, it is quite impractical to do manual segmentations with large datasets especially for real time applications.

In general, there are two approaches used for automatic speech segmentation: implicit and explicit segmentation [3]. Implicit segmentation splits incoming utterances into segments without any linguistic information while explicit segmentation splits the utterance as in a transcription scenario. Both approaches need to consider the nature of the language itself in using certain technique. Here, our focus is the Arabic language.

The Arabic language has been used for more than 4000 years. It is one of the main languages written from right towards left. The standard Arabic language has basically 35 phonemes; 6 vowels and 29 consonants as shown in Table 3.1. Standard Arabic language phonemes can be classified into two main categories, vowels like (اَ, اِ, اُ,

ِ) and consonants like (ب, ج, د, ...etc.). The syllables in Arabic are based on the contrastive components that are contained in its structure. The successive contrastive elements within a syllable boundary are made up of segmental phonemes of the language. Each syllable has a main part that stands out with prominence. This part is referred to, here, as the “nucleus” of the syllable. The remaining components are referred to as ‘marginal factors’. The vowel always forms the syllable nucleus while the consonant represents the marginal phonemes in the syllable structure. The number of syllables in a given utterance is identical to the number of vowels [6] as an example, the versa ( كلا سيعلمون), has six syllables: CVC CV CV CVC CV CVC, which is the same number of vowels in the versa. There are 5 different type of syllables in Arabic language: CV like (مَ, سَ), CVC like (قَدَ, يَزَ), CVV like (حَيَّ, قُو, تَا), CVVC like (نُونِ, مِيمِ, قَافِ) and CVCC like (خُسْرَ, تَبَّ, عَصْرَ).

The diversity in both spoken and written Arabic language brings with it major challenges. These challenges are especially reflected when reading Quran. While accurate reading is fundamentally based on accurate Arabic pronunciation, it is in nature much richer in semantics and in the way it sounds. The concept of combining syllables, stressing others, stopping somewhere else, is such a powerful one in highlighting the beauty and the impact of Quran. Much more combinations of vowels and consonants exist while some other are not allowed, as an example, (من يعمل) will be as (مَيِّعِل), also (اركب معنا) will be as (اركَمَّعنا). To handle these challenges, we propose an automatic speech segmentation algorithm focusing specifically on Quran recitation. The details are represented in the next section.

Table 3.1 Arabic consonants [125]

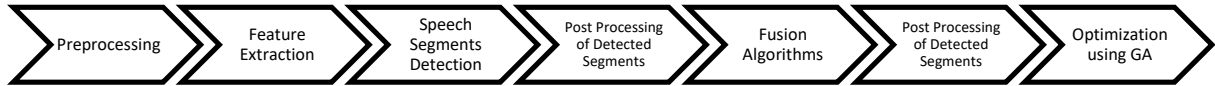
	Arabic Script	International phonetic script		Arabic Script	International phonetic script
1.	ب	B	15.	ي	Y
2.	ت	T	16.	ز	Z
3.	ث	Th	17.	ج	J
4.	د	D	18.	ر	R
5.	ذ	Dh	19.	غ	Gh
6.	س	S	20.	ح	Ha
7.	ش	Sh	21.	ع	3
8.	ف	F	22.	خ	Kh
9.	ك	K	23.	ق	Q
10.	ل	L	24.	ء	،
11.	م	M	25.	ص	Sv
12.	ن	N	26.	ط	Tv
13.	ه	H	27.	ض	Dv
14.	و	W	28.	ظ	Dhv

### 3.2 The Proposed Automatic Speech Segmentation Algorithm

In this work, we aim at developing an implicit automatic speech segmentation algorithm especially dedicated for Quran recitation. One major challenge is that we didn't

have a dataset that has a full phoneme description for Quran, and also there are several phonemes for Quran not found in the Arabic language. For example, when we say (مَنْ لَّهُمْ) (disconnected words), in Quran these two words are read as (مَلَّهْم) (one connected word). This concept is called (إدغام) and specific for Quran recitation but not found when reading standard Arabic. This is one of many instances where standard segmentation techniques used for other languages or Arabic don't apply for Quran recitation.

To track such linguistic richness, we propose a robust blind speech segmentation technique in this work. The proposed system is organized around the following steps (see Figure. 3.1): Preprocessing, Feature extraction, Speech Segments Detection, Post Processing of detected segments, and Optimization using a Genetic Algorithm (GA). Each of these steps will now be discussed in detail.



**Figure 3.1** Flow chart for the proposed automatic speech segmentation algorithm

### 3.2.1 Pre-processing

Since we are proposing an approach for automatic speech segmentation, mainly focusing on Quran recitation, we have initially opted to use in our experiment the comprehensive public KACST database[126]. The database was developed with full phoneme description of the Arabic language. In this database, the acquired speech is sampled at a frequency of 44.1 kHz. Sampled speech is then preprocessed to yield frames

of acoustic observations from which features are extracted. In order to the flatten speech spectrum, a pre-emphasis filter of the form  $(1 - 0.95z^{-1})$  was used before windowing and further analysis. The filter was then followed by windowing using a hamming window to divide the speech signal into a sequence of frames, each frame with a length of 1100 samples (25 ms) , so that each frame can be analyzed separately while preserving stationarity. The speech signal before and after pre-emphasis filter is shown in Fig. 3.2.

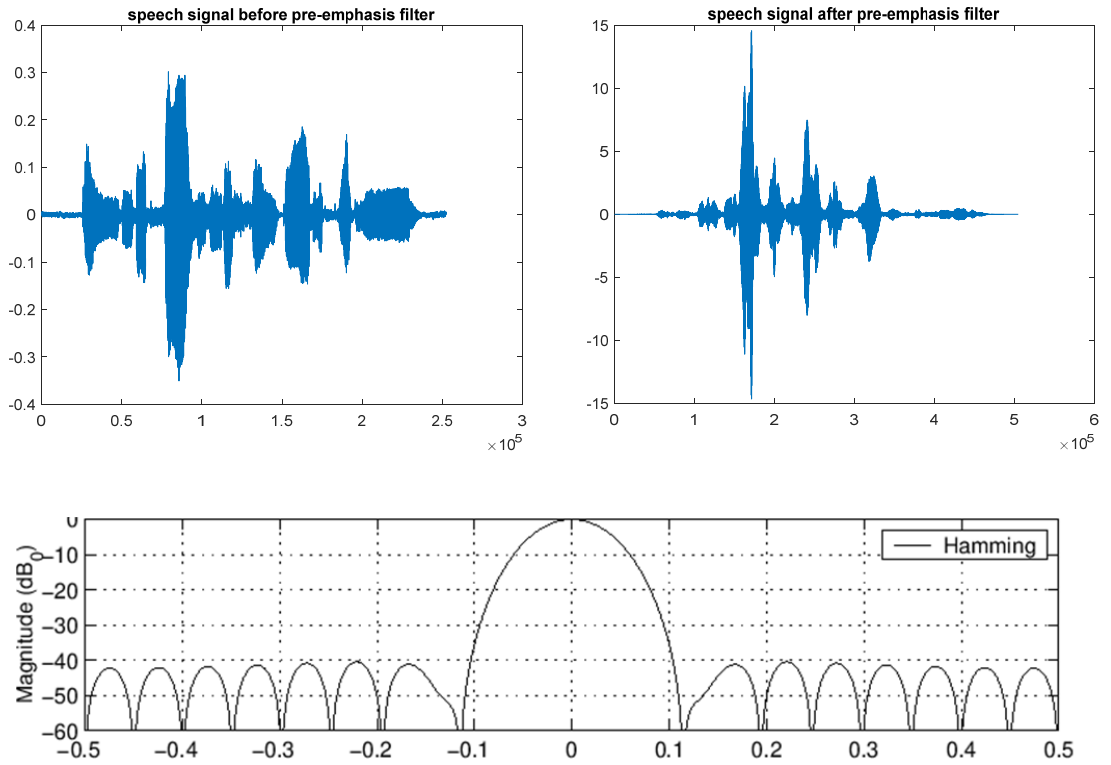


Figure 3.2: a) Speech signal before and after pre-emphasis filter b) Spectrum of hamming window

### **3.2.2 Feature Extraction for Speech Segmentation**

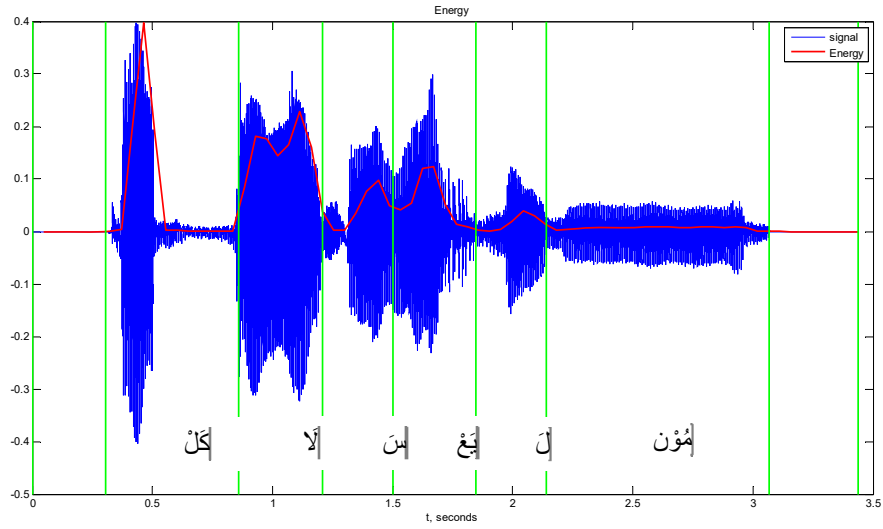
The main block of automatic segmentation systems is feature extraction; we need to extract robust features from speech data, which directly reflect the nature of speech and have strong discriminatory power to identify consonant and vowels. To reduce the complexity, we used three individual features: Energy, ZCR, and Entropy. The short term energy feature is a very important variable in automatic speech segmentation. It can distinguish between consonant and vowel syllables in speech. The short term zero crossing rate feature is another variable used to distinguish between voiced and unvoiced syllables. Here, we introduce a new feature for Arabic speech. More specifically, we use the entropy feature which mainly characterizes the degree of complexity in speech, and other physiological signals. In what follows, we will briefly describe these features and their importance with respect to automatic segmentation.

#### **a) The Short-Time Signal Energy Feature**

The short-term energy is one of the most important features used in automatic speech segmentation. It is an excellent characteristic for discriminating between consonant and vowel syllables, as it has a high value for vowel syllables, and a low value for consonant syllables, as shown with the red color in Figure 3.3. The short-term energy is determined by dividing the signal into frames using windowing, then for each frame, the squares of the sample values are averaged over the frame [127]. The short-time energy value for a given speech frame of length  $N$  is defined as:

$$E_n = \frac{1}{N} \sum_{m=1}^N [x(m)w(n-m)]^2 \quad (3.1)$$

Where  $x(m)$  is the discrete-time audio signal,  $n$  is time index of the short-time energy, and  $w(m)$  is a window function of length  $N$ . In practical setups, the Hamming window is generally used, which is discontinuous (slam to zero) at endpoints and side lobes are closer to equal ripple. The energy profile for the utterance (كلا سيعلمون) is shown in red in Figure 3.3. From Figure 3.3, we observe some segment units have high energy values like the segment unit (لا) and some of them have low energy values like segment unit (مون).



**Figure 3.3** Short time energy profile of a given speech signal

### **b) The Short-Time Average Zero-Crossing Rate Feature**

The short-term zero crossing rate is also an important feature in automatic speech segmentation. This feature calculates the number of zeros crossing in a given frame. It is an excellent characteristic for discriminating between voiced and unvoiced syllables, as it has a low value for voiced syllables, and a high value for unvoiced syllables as represented with the red color in Figure 3.4. The short-term zero crossing rate is determined by dividing the signal into frames using windowing, then, for each frame, the number of zeros crossing is calculated [128]. The short-time average zero-crossing rate is defined as:

$$Z_n = \frac{1}{2} \sum_{m=1}^N |sgn(x(m)) - sgn(x(m-1))| w(n-m) \quad (3.2)$$

Where,  $sgn(x(m))$  is given by:

$$sgn(x(m)) = \begin{cases} 1, & x(m) \geq 0 \\ -1, & x(m) < 0 \end{cases} \quad (3.3)$$

The ZCR profile for the utterance (كلا سيعلمون) is shown in red in Figure 3.4.



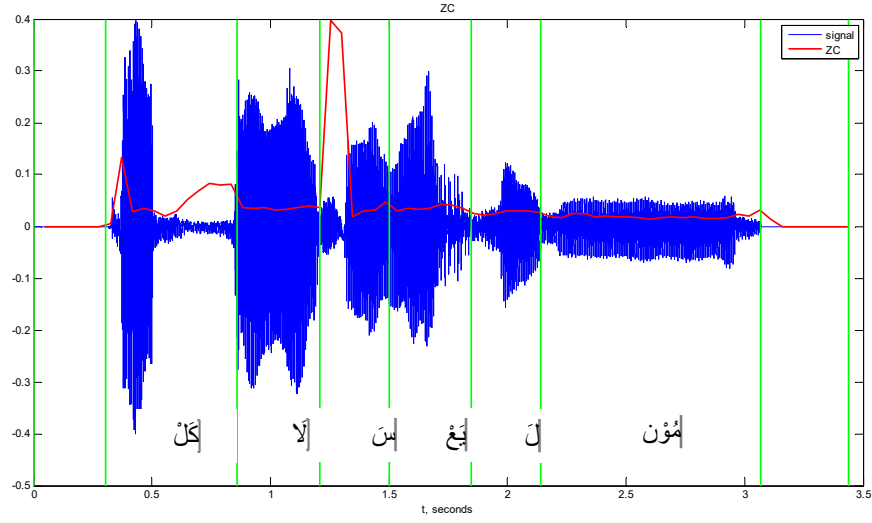


Figure 3.4 Short time zero crossing rate of the speech signal

### c) The Entropy Feature

The notion of entropy has been used to characterize the degree of complexity or chaos in speech as well as in other physiological signals [6]. In order to determine the probability distribution within each individual frame, a histogram with  $N$  bins is constructed. The histogram is then normalized to satisfy the statistical properties of the cdf. Selection of the number of bins ( $N$ ) for the histogram is a trade-off between sensitivity and computational load. The entropy for each frame is computed as follows [5].

$$H = -\sum_{i=1}^N p(x_i) \log_{10} p(x_i) \quad (3.4)$$

Where  $x_i = \{x_1, x_2, \dots, x_N\}$  is a quantized frame sample, and  $p(x_i)$  is a probability of a quantized frame sample  $x_i$  belonging to the frame interval. The relation between entropy

and signal processing is based on the hypothesis that a noise (white noise) is a projection of a system in thermodynamic equilibrium into a signal. As a result, the noise is supposed to have the highest entropy value while speech sounds (and mainly periodic sounds like e.g. vowels) have significantly lower entropy values since such signals are more organized and require an extra energy (or effort) to be produced in such an organized manner. Recently, speech segmentation using entropy has been included as a robust feature for automatic speech recognition [10]. It is useful in distinguishing between segment units based on tracking the consonant/vowel sequence in a given segment unit. However for clarity of presentation, we assume that we have the entropy profile  $\xi$  for the complete speech data available, where

$$\xi = [H_1 H_2 H_3 \dots H_m] \quad (3.5)$$

where we assume  $m$  represents the total frames in the incoming speech. The entropy values for the utterance (كلا سيعلمون) are shown in red in Figure 3.5. We notice in Figure 3.5, the entropy in the consonant region has a higher value than in the vowel region.

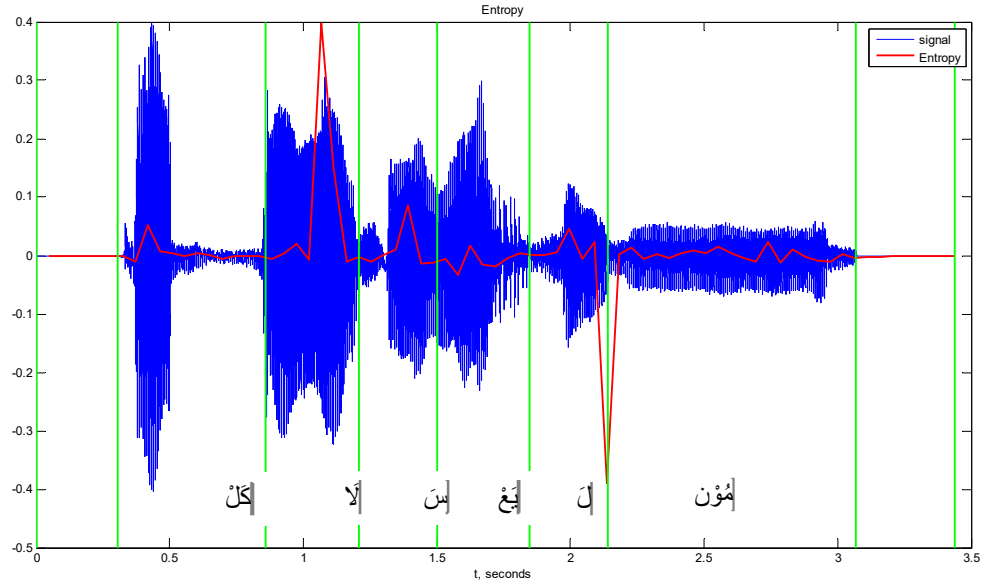


Figure 3.5 Entropy of a typical speech signal

### 3.2.3 Speech Segments Detection

After estimating the different features from the speech signal, thresholding is applied in order to detect the speech syllable segments. We used common statistical measures as thresholds such as median [36], mean [129] and mode. Each of these statistical measures has its own strengths and weaknesses. The most traditional measure used is the mean. While the mean fits well when representing mono-modal symmetric data, however, it is very sensitive to extreme values and outliers. Another measure related to the mean especially for mono-modal distributions is the mode. This measure fails when the distribution exhibits more than one mode and is not theoretically well behaved. Drawing for its power in image enhancement and segmentation, the median is seen as a qualitative average, not affected by outliers. The median exhibits a number of other advantages even though it is computationally more expensive to compute and its

statistically modeling is a challenging task [130],[40]. For the sake of completeness, we discuss the performance of all three measures in segmentation (see section 3.4 on experimental results).

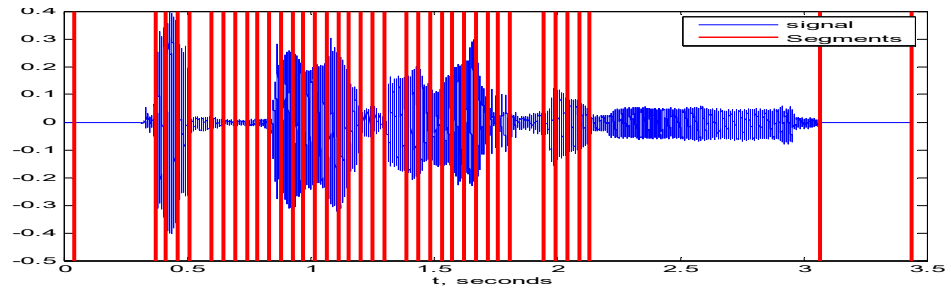
### 3.2.4 Post Processing of Detected Segments

While the results using simple thresholding are reasonable, we noticed that all features when used with simple thresholding result in a non-negligible number of false boundaries. Most transition errors happen because of noise/silence frames, or across two short segment units. To reduce the number of false boundaries, we optimize the length of resulting segments. When a given segment is below a certain length, we consider it to be part of the previous segment and add it to that segment. The rationale for such a step is that humans do not generate very small duration sounds (segments) as speech segments have to achieve a certain minimum duration [135].

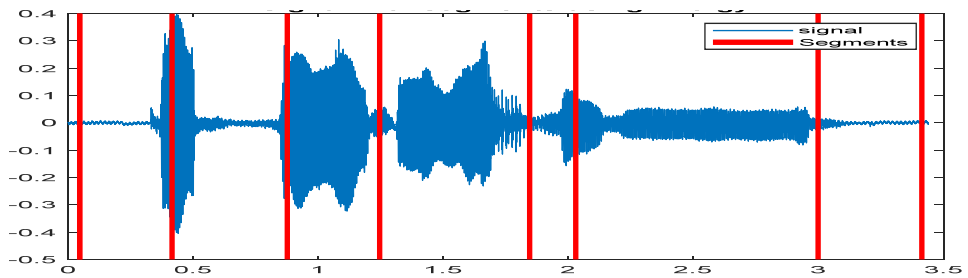
For Quran recitation, in particular, we have some words that very long like (أنلزمكموها) (فأسقيناكموها) .... etc. Such words cover at least 4-5 segment units. In general, the length of most segment units in the Quran, is between 2 to 8 phonemes in one segment unit [136]. Experimentally, the smallest phonemes were found to be stops phonemes like (أ, ب, ج, ق, ط, and د) with phoneme length between 7 ms and 10 ms [136]. The longest phonemes in the Arabic language are shown to be the nasal phonemes like (م, ن) with phoneme lengths of up to 24 ms. In Quran, the longest phonemes are shown to include up to six successive long vowels (we denote it here as V6), which results in segments of 100 ms in length [136].

Based on the above criteria, we introduced the following rule: If the segment unit is less than a certain minimum number of frames, then this segment unit is merged with the previous segment unit, and so on. The detected speech segments using all the individual features before and after post processing are shown in Figure 3.6 (a).

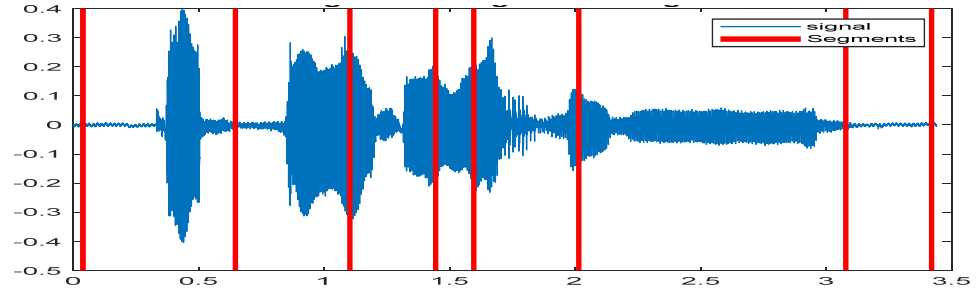
We notice in Figure 3.6, that for the boundary results obtained from the individual segmentation engines, some are inserted and some are missed. To solve this problem, we will use fusion techniques to enhance the segmentation accuracy. More details will be explained in the next section.



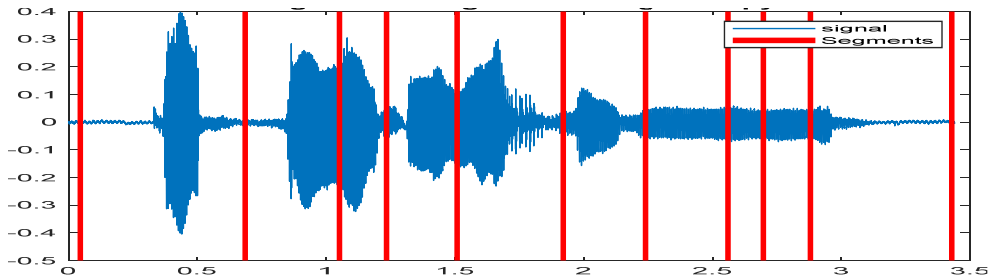
(a)



(b)



(c)



(d)

Figure 3.6: a) Automatic speech segmentation before post processing, b) after post processing using the energy feature, c) Using ZCR feature, d) Using the entropy feature

### 3.2.5 Robust boundary prediction using fusion techniques

The block diagram of the proposed regression fusion scheme for combining different segmentation engines is presented in Figure 3.7. This is a general fusion scheme and is independent of the individual segmentation engines used.

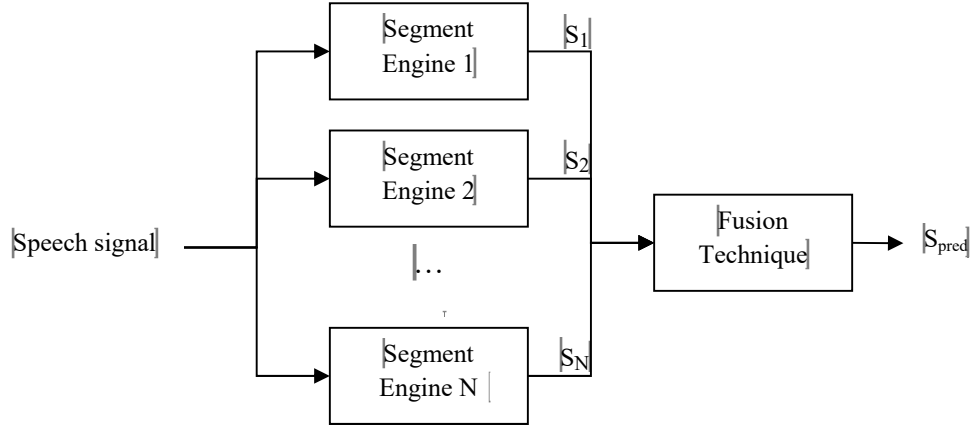


Figure 3.7 Block diagram for fusing multiple segmentation engines [31]

Let us define a set of  $N$  predicted syllable transition positions,  $\{S_i\}$ , where  $1 \leq i \leq N$ , as being the outcomes of  $N$  different segmentation engines. These engines produce syllable boundary predictions that are assumed to be independent of each other. As an example of fusion, the individual predictions can be combined using a regression function,  $f(\cdot)$ , to create a new syllable transition predicted positions where  $S_{fus} = f(S_1, S_2, \dots, S_N)$ . The parameters of the fusion function are adjusted by minimizing an error function between the true and the predicted syllable transition positions over the training dataset. For the true syllable transition positions, we consider the manually annotated labels of the syllable boundaries available in the speech database used for training [31].

However, using simple regression, was proven to lead many errors in the boundary positions. To complement the simplicity of the regression formulation, we investigated more advanced approaches. Here, we introduce a new approach for fusion of segmentation results using a new optimization framework based Genetic Algorithms. For

the sake of completeness, we will briefly outline the basic fusion approaches with optimization. Then, we discuss the basic regression based approach, followed by our proposed approach.

For basic fusion, we used the initial segmentation results from energy, zero crossing rate, and the entropy features as input to the system. The results are concatenated into a 3-dimensional vector, then in a final segmentation step, a rule based on the minimum number of frames to form a syllable is implemented. In addition to this simplistic approach, a number of more advanced approaches have also been investigated. These include linear regression, neural regression, SVM regression,...etc. [31]. In our work, we started by a linear regression on the individual segmentation techniques.

For the linear regression, the segmentation results from the individual features are weighted and summed as in the following equation:

$$\min_{\{w_j\}} S = \sum_{i=1}^M (S_{real}(i) - [w_0 + \sum_{j=1}^N w_j * S_j(i)])^2 \quad (3.6)$$

Where M is the number of boundary segmentations, N is the number of segmentation engines and  $w_j$  are estimated using a least squares minimization approach over the training data.  $S_j(i)$  is determined as:

$$S_j(k^*) = \min_k |S_{real}(i) - S_j(k)|, \quad k = 1, 2, \dots, K \quad (3.7)$$



Where  $K$  is the total number of boundaries in each utterance for each automatic segmentation engine ( $S_j$ ) and  $k^*$  are the selected boundaries obtained from the automatic segmentation engine and closer to the manual boundaries.

While implementing and testing the above regression algorithm, we faced two main challenges. First, we needed to specify the frame size in advance. Second, the minimum number of frames per syllable is also a challenge as it needs to be specified a priori. To handle the two challenges above, we develop here a new optimization architecture that automatically determines the optimal frame size and the minimum number of frames per syllable. The optimization scheme, implemented here, was based on a simple Genetic Algorithm.

### **3.2.6 Segmentation using Genetic Algorithms**

Genetic algorithms form a special class of optimization schemes. The concept is based on a set of candidate solutions to a given problem where we start with a set of random solutions. Each candidate set is typically an ordered fixed-length array of values (called 'alleles') for different attributes ('genes'). The set of alleles for a given gene is a group of possible values that the gene can possibly take. So, in building a GA for a specific problem, the first task is to decide on how to represent the possible solutions. Genetic Algorithms are different from other optimization and search methods; as these work with a coding of the parameters, not the parameters themselves. The search in GA depends upon a population of points, not a single one. Also, GAs use the objective

function itself, not derivatives or other auxiliary knowledge based on probabilistic/deterministic characterization.

In Figure 3.8, we briefly outline the main steps of a basic Genetic algorithm [48]. These are listed below:

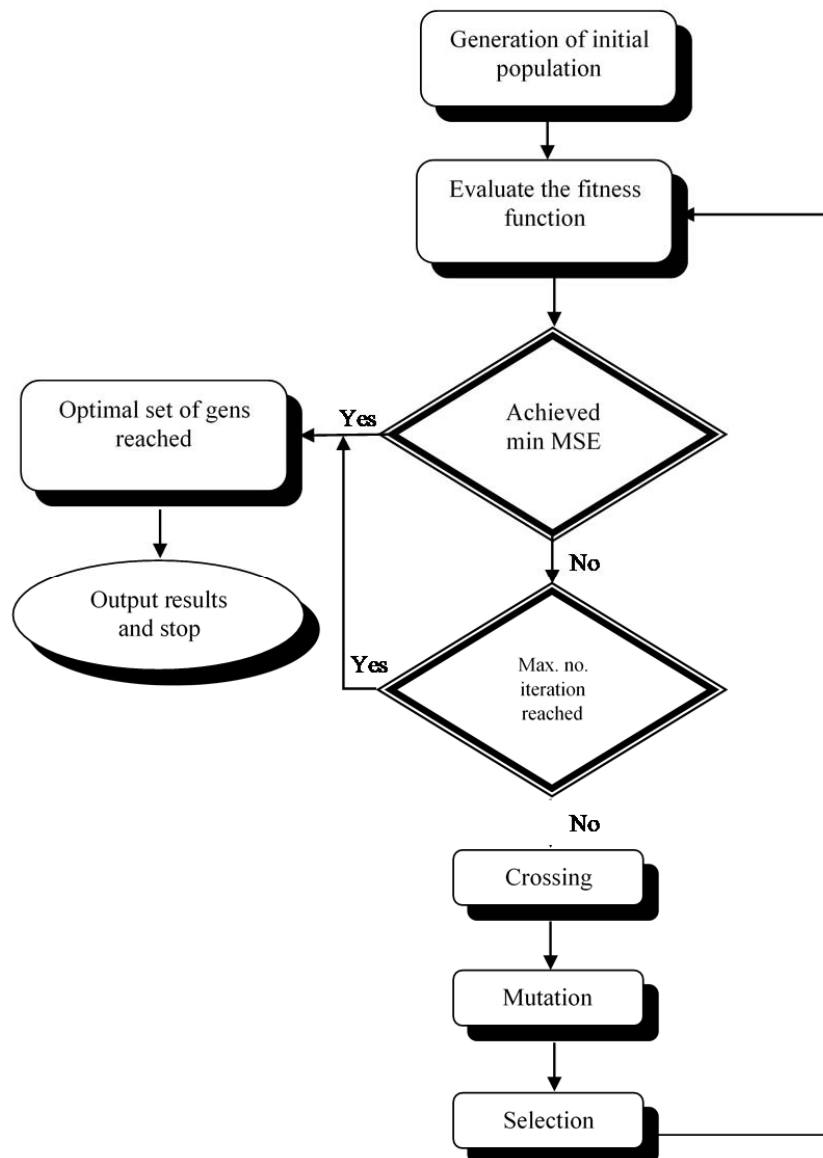
1. Initialization: A set of candidate solutions is randomly generated, then the algorithm goes through iterations till convergence.
2. Evaluation: Using some predefined problem-specific measure of fitness, a value is allocated to each measure of the solution set. This measure is called the candidate's fitness function.
3. Selection: Select pairs of candidate solutions from the current generation to be used for breeding. This may be achieved in either a random or a stochastic manner.
4. Breeding: Produce new individuals by using genetic operators on the individuals chosen in the selection step. There are two main kinds of operators: Recombination and Mutation [48]. Here, we used mutation.
5. Population update. The set is altered, typically by choosing to remove some or all of the individuals in the existing generation, and replacing these with the individuals produced in the breeding step (step 4). The new population thus produced becomes the current generation, and we start the iterative process again.

For our specific application, the initial population is generated using a uniform probability density function (PDF) with a size twice that of the number of segments

obtained from the fusion step [137]. To focus on the application of interest, we only discuss here the cost function and leave out the details of the algorithm. The objective function we minimize here is the mean squared error between the real and predicted boundary segmentations based on the F-measure value as:

$$\min_{\substack{\{frame\ size,\ \\ no.\ frames/segment\ unit\}}} S = \sum_{i=1}^M (S_{real}(i) - \gamma(i))^2 \quad (3.7)$$

Where  $M$  is the number of segment units, and  $\gamma$  is predicted boundary segmentations from the fusion methods. The details of the F-measure function are described in the next section.



**Figure 3.8 Block diagram of Genetic Algorithms**

### **3.3 Performance Evaluation using the F-measure**

For typical pattern classification problems, a number of measures have been introduced to quantify the overall performance of the developed algorithms. For classification tasks, the terms true positives, true negatives, false positives, and false

negatives are used to summarize the performance of the classifier being evaluated. The terms positive and negative refer to the classifier's prediction, and the terms true and false refer to whether the prediction matches the observation. If we define P as the positive instances (the test detects whatever it was designed to look for, e.g.: allergy test detects the presence of an allergic reaction), and N as the negative instances (that is a test result is negative for a healthcare). The precision and the recall measures are formulated as:

$$Precision = \frac{t_p}{t_p + f_p} \quad (3.8)$$

$$Recall = \frac{t_p}{t_p + f_n} \quad (3.9)$$

Where  $t_p$  is the number of the true positive (correct boundaries),  $f_p$  is the number of the false positive (wrong boundaries) and  $f_n$  is the number of false negative (missing boundaries). In our case, precision is describes the likelihood of how often algorithm identifies a correct boundary whenever a boundary is detected. Where recall represents the rate of correctly detected boundaries. These measures are used to compare the detected and real change points in the corpus under investigation. The most important measures used, are FD and FR which are calculated below:

$$FD = \frac{\text{number of false detections}}{\text{total amount of detections}} \quad (3.10)$$

$$FR = \frac{\text{number of missed detections}}{\text{total amount of true change points}} \quad (3.11)$$

where, FD (False detections) is number of points which are not real change points in the reference corpus; but, are detected by the system as change points. These points are called False Alarm (FA). Total number of detections is the total number of points detected by the system as change points. The missed detection is number of points which are real change points in the reference corpus but, are not detected by the system as change points. These points are called Missed Detection (MD). Total number of true change points is total number of points correctly detected by the system as change points. To determine the overall quality of the automatic segmentation method, the F measure is used to consider the impact of both FD and FR as:

$$F = \frac{2*(1-FD)*(1-FR)}{2-FD-FR} \quad (3.12)$$

In the next section, we discuss our experimental results based on the proposed work as discussed above.

### 3.4 Experimental Results

To test the performance of the proposed segmentation algorithms, we needed a database that is manually segmented covering Quran recitation. Since an appropriate speech corpus of the Quranic sounds was not available online, we used the Quran database from the Research Center at King Abdulaziz City for Science and Technology (KACST), Saudi Arabia. This database covers speech data from expert reciters memorizing the Quran and includes manual phoneme segmentation based on the phonetic of the Arabic language. The sampling frequency of the recorded data is 44.1 kHz, and the total number of sound files was 5935 with an average of about 594 files per reciter, and 50 minutes of duration per reciter.

In our work, we updated the two reciters from this database by introducing manual syllable segmentation according the Arabic language rules. In Arabic language, there are five types of syllable structures: CV, CVV, CVC, CVVC and CVCC. We resegmented the data according to these syllable structures manually and used these in our experiments. We used the resulting manual syllable segmentations as a reference base, to evaluate our proposed methods for automatic speech segmentation. All of our experiments were based on frames and the segmentation accuracy is measured in terms of the percentage of predicted boundaries within a tolerance ( $\Delta$ ) of zero frame,  $\pm$  one frame and  $\pm$  two frames from the manually annotated boundary labels. Zero frame tolerance means no difference between manual and predicted automatic segmentation, while one frame means  $\pm$  one frame error tolerance difference between manual and predicted

automatic segmentation. The two frames means  $\pm$  two frames error tolerance difference between manual and predicted automatic segmentation.

First, we started with a standard experimental setup. We used 80% of the data for training, and the remaining 20% for testing. The frame width was 25 ms, and the minimum number of frames for any segment unit was 4 with the  $\Delta$  being equal to 2 frames. We used the median as a threshold over the six following segmentation methods: Energy, Entropy, ZCR, Basic fusion, Majority voting and Linear regression. For basic fusion, we combined all the results from all the individual automatic segmentation engines, then we sorted the combination results as boundaries for each utterance. The results are shown in Table 3.2, where Precision (PRC), Recall (RCL), and F-measure are estimated as discussed in section 4 above.

**Table 3.2 Precision, Recall, and F-measure, with different segmentation methods**

Segmentation Technique	PRC	RCL	F-measure
Energy	75.0	66.7	70.6
Entropy	72.7	88.9	80.0
ZCR	100	44.4	61.5
Basic Fusion	81.8	100	90.0
Majority Voting	63.6	77.78	70.0
Linear regression	66.7	66.7	66.7



We note from the table that the entropy measure provides the best segmentation results compared to other individual measures, however the entropy is good for distinguishing between segment units based on the tracking of the consonant/vowel sequences in a given segment unit. Moreover, we see that simple fusion, which is rule-based (basic fusion) provides good results without the need to using linear regression.

From the above, we noted two main challenges: the frame width, and the minimum number of frames per segment unit. To consider these challenges, we discuss here two scenarios: a compromise between frame width and number of frames to form a syllable for each segmentation technique, and an enhancement of the segmentation results using optimization techniques.

Under the first scenario, we compared the results of each segmentation algorithm against manual segmentation. We tested different values of frame size with a minimum number of frames of 2 to create a segment unit. The F-measure results are presented in Table 3.3. It is clear that the performance changes with the frame size, however, such a change is not monotonic. For instance, with the entropy feature, the F-measure increases with the frame size then decreases again after a certain width. As such, defining a certain frame width is not expected to provide optimal segmentation results. Hence, as we will see below, a sequel optimization step is needed to obtain the best possible segmentation results.

Table 3.3 F-measure for different frame sizes with a fixed minimum number of frames/segment unit

Segmentation Technique \ Frame width size	15 ms	25 ms	35 ms	45 ms	55 ms
Energy feature	57.1	66.7	66.7	70.0	<b>80.0</b>
Entropy feature	38.1	42.4	38.5	<b>80.0</b>	66.7
ZCR feature	42.1	37.5	40.0	53.3	<b>66.7</b>
Basic Fusion	40.0	53.3	58.3	<b>81.8</b>	80.0
Linear Regression	66.7	55.6	66.7	<b>77.8</b>	76.3

In contrast to the scenario above, we fixed the frame size, and considered different values for the minimum number of frames/segment unit. The results are shown in Table 3.4. Once again, it is clear that, the performance changes with the minimum number of frames, in non-monotonic fashion. Both results in Tables 3.3 and 3.4, show the need to develop optimization techniques with respect to the minimum number of frames/segment unit and the optimal frame width.

**Table 3.4: F-measure for different values of minimum frames/segment unit with a fixed frame width**

<b>Segmentation Technique</b> \ <b>Min. Number of Frames</b>	<b>2</b>	<b>3</b>	<b>4</b>	<b>5</b>
<b>Energy</b>	66.7	66.7	70.0	<b>77.8</b>
<b>Entropy</b>	<b>43.8</b>	38.5	42.1	37.5
<b>ZCR</b>	<b>42.1</b>	40.0	40.0	40.0
<b>Basic Fusion</b>	48.5	58.3	<b>66.7</b>	58.8
<b>Linear Regression</b>	<b>88.9</b>	66.7	55.6	44.4

To tackle the challenges mentioned above, we propose to use an optimization technique to predict both the frame size and the minimum number of frames/segment unit. Here, we use a genetic algorithm to find the best prediction boundaries over the six segmentation methods. Genetic Algorithms have a different specification than other optimization and search methods; as these work with a coding of the parameter set, not the parameters themselves. Moreover, GAs use the objective function, not its derivatives or other auxiliary knowledge based probabilistic/deterministic characterization. In our work, the genetic algorithms solve a problem by, roughly, generating, changing, and evaluating candidate solutions to the problem of interest. A candidate solution to a problem is called a chromosome, and a chromosome is usually a bit string or some other encoding or representation of a solution. Initially, a random population of such

chromosomes is generated. Changing chromosomes is done by mutation and/or crossover operators, while chromosomes (candidate solutions) are evaluated by way of a domain dependent fitness function, which first decodes the chromosome, then evaluates its optimality, as a solution to the particular problem being addressed.

The model parameters of GAs such as population size, reinsertion rate, migration rate, are chosen empirically. The objective function is the mean squared error between the real boundaries and the predicted boundaries for the six methods discussed above.

In our work, the number of observations (utterances) was 1128 (for two reciters), and the setup optimization parameters were as follows: frame size is between 25 ms and 50 ms, and the minimum number of frames/segment unit is between 2 and 10. The GA is applied to obtain the segment unit boundaries. Under this scenario, we consider three cases:

Case 1: Optimizing the individual segmentations (Energy, Entropy and ZCR), where the optimization parameters are: frame size and the minimum number of frames/segment unit. The F-measure results after applying the optimization technique are shown in Tables 3.5. For each individual segmentation algorithm, we used different statistical threshold measures (median, mean, and mode) to find the best among these measures. Our results show that the median threshold measure leads to the best results which concurs with previous work [129].

**Table 3.5: F-measure values after using GA for Energy, ZCR and Entropy features with different threshold types**

<b>Individual segmentation engine</b>	<b>Threshold method</b>	<b>Frame width</b>	<b>Min Syllable length</b>	<b>F-measure value</b>
<b>Energy</b>	Median	2034.2	3.7	77.8
	Mean	1860.3	4.5	77.8
	Mode	1881.8	4.1	57.1
<b>ZCR</b>	Median	1675.5	2.2	71.4
	Mean	1779.3	2.3	70.6
	Mode	1329.2	5.3	50.0
<b>Entropy</b>	Median	2015.9	2.8	84.2
	Mean	2164.1	3.9	80.0
	Mode	2108.1	3.1	82.1

Case 2: Optimizing the rule-based basic fusion segmentation. Here, the parameters are: frame size, minimum number of frames/segment unit for each individual segmentation, and the minimum number of frames/segment units after applying the rule-based basic fusion. The optimization is performed over five parameters. The F-measure results are shown in Table 3.6, where we considered three possible values for the  $\Delta$  frame. We note that, when we optimize the rule-based basic fusion, the result come closer to 100% with an enhancement of at least 16% compared to the individual segmentation methods.

**Table 3.6: F-measure values for Basic Fusion technique with different  $\Delta$  error tolerance**

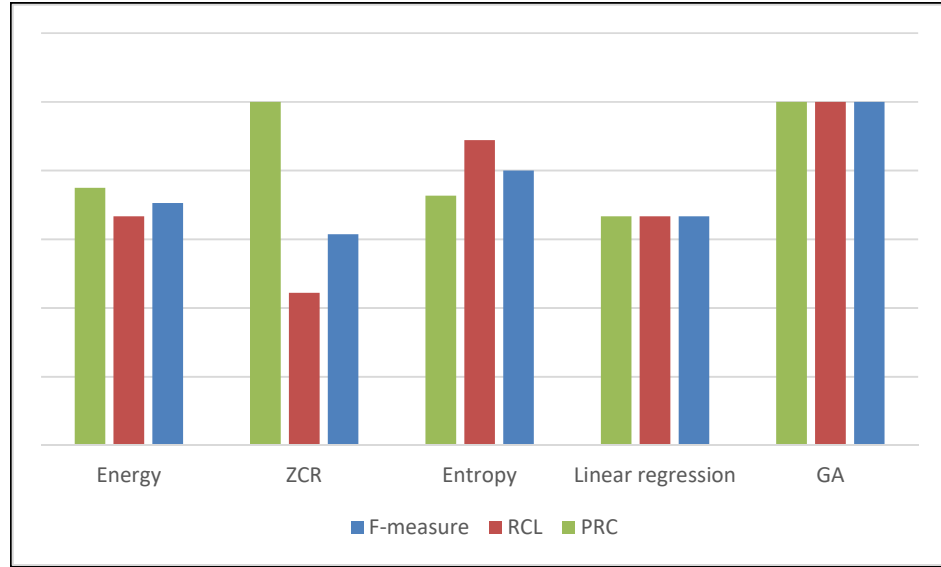
$\Delta$ frame	Frame width	Min Syllable length in Energy	Min Syllable length in ZC	Min Syllable length in Entropy	Min Syllable length according to rule-based	F-measure value
0	2179.3	8.1	7.9	5.8	2.7	50.0
1	1740.3	9.0	6.8	8.9	8.6	66.7
2	2164.7	4.4	2.2	5.2	2.2	99.4

Case 3: Optimizing the linear regression fusion method. In this case, the parameters are: frame size, minimum number of frames/segment unit for each individual segmentation, and the linear regression coefficient parameters corresponding to the individual segmentation methods. We started by considering all the individual segmentation methods (Energy, ZCR and Entropy), with the F-measure results, after applying optimization, displayed in Table 3.7. We can see that with the optimal values, we get an accuracy of 89%. We then performed an experiment in which we only considered two individual segmentation methods, namely Energy and Entropy. In this case, we were able to achieve an almost perfect segmentation of 98.7% as shown in Table 3.7. The experiments show that the pair (Energy and Entropy) is compatible with linear regression combination and leads to excellent segmentation. Note that when all three individual segmentations are fused and because of diversity in the three approaches, the combination did not achieve perfect accuracy [47].

**Table 3.7: F-measure values for linear regression fusion technique with different error range**

	Frame width	Min Syllable length in Energy	Min Syllable length in ZC	Min Syllable length in Entropy	A1	A2	A3	F-measure value
Energy, ZCR and Entropy	1957.1	3.9	3.9	3.9	1.0094	0.8665	1.0266	88.9
Energy and Entropy	207.11	3.6	-	3.6	0.9685	-	0.9981	98.7

To summarize our results, we display in Figure 3.9, the performance in terms of precision, recall, and F-measure for all of the scenarios discussed in our experiments. Few notes can be outlined from the figure. We see that among the individual segmentation techniques, Entropy provides the best results. As we outlined before, to the best of our knowledge, this is the first time that entropy is being used for Arabic speech segmentation. While the linear regression approach provides consistent results, the accuracy is not outstanding compared to the individual segmentation techniques. More importantly, we see that the proposed segmentation technique provides the best results consistently and with all performance measures.



**Figure 3.9 PRC, RCL and F-measure of our experimental results**

### 3.4 Conclusion

In this chapter, we introduced a new hybrid speech segmentation algorithm using a GA optimization scheme over multiple features. The algorithm uses numerous independent individual segmentation methods, to produce multiple predictions of boundary positions. These predictions are used as input to the proposed fusion method. To optimize the overall system, we consider two main challenges: frame size, and minimum number of frames / segment unit. To handle these challenges, we use a GA to optimize these two parameters together with the combination coefficients, to obtain the final optimal predicted boundaries. The experimental results show significant improvements in segmentation accuracy of the proposed method, when compared to the best performing baseline segmentation technique. The results show an improvement accuracy of 16% over the best performing single feature, and 10% improvement over the



linear regression based approach. Overall, an almost perfect accuracy is obtained with respect to manual segmentation. These results were obtained over the KACST database, but superior results are also expected over other similar databases. The findings for this work are now being used for further classification and quality of recitation/reading tasks, as related to Quran recitation and Arabic language.

After detecting all the segment units based on our proposed algorithm, in the next chapter, we will discuss how these segment units can be identified using machine learning techniques.

# **CHAPTER 4**

## **ENHANCEMENT SPEECH SEGMENTATION USING ENSEMBLE-BASE CLASSIFIERS AND A HIERARCHICAL TREE STRUCTURE**

### **4.1 Introduction**

Machine learning is a field within the greater area artificial intelligence which focuses on learning algorithms and decision making based on some acquired data. With respect to how machine learning approaches work, three main directions are taken [138]: a) Supervised learning where the algorithms learn how to make decisions based on available input-output pairs, b) Unsupervised learning where the algorithms try to find hidden structures within the input data, and c) Reinforcement learning where the algorithms actively interact with the environment and learn based on the concept of rewards and punishments.

When considering the types of applications, machine learning algorithms can also be categorized under several types [138]:

1. Classification, where the algorithm learns to assign class labels (from a certain set) to inputs, therefore deciding which input is a part of which class; equivalent to a supervised approach to classification.

2. Regression, where the output is discrete, and the regression algorithm produces possibly continuous outputs such as in the case of data fitting.
3. Clustering, where the algorithm divides data into clusters, but in an unsupervised fashion as classes are apriori unknown.
4. Dimensionality reduction where the input data is mapped into a lower dimensional space.

In this chapter, we will present our approach to studying the problem of speech segmentation and recognition using machine learning technique. Segment unit recognition is defined as the process of assigning a given input signal to one of the several prescribed segment unit classes. The training of an individual classifier that performs recognition for all the classes in one time, is unpractical especially when the number of classes is large and the similarity among these different classes is high [139]. In this chapter, we developed a hierarchical approach to overcome the limitations of individual classifiers. Under hierarchical approaches, the large number of classes is grouped into fewer subgroups with a separate network being trained for each subgroup [140]. The proposed hierarchical recognition system forms a tree like structure, in which many paths can be traversed from the root node down to the terminal nodes (leaves). It is based on the principle of "Divide and Conquer", where a large problem is recursively divided into smaller and easier problems, whose solutions can be combined to yield a solution to the overall complex problem [141]. In contrast to conventional non-hierarchical (flat) baseline recognition algorithms, where each data sample is tested against all possible classes, in a hierarchical tree structure, a sample is tested against only

certain subsets of classes, thus eliminating unnecessary computations. Hierarchical Tree Classification Algorithms (HTCs) have the flexibility of choosing different feature subsets and using different decision rules at the different stages of recognition, in addition to the ability of trading-off between recognition accuracy and time-space efficiency. However, in large HTC systems, the effect of classification errors can propagate from one level to another (accumulation), which can be considered a significant drawback of HTC systems pointing out to the fact that one cannot simultaneously optimize both accuracy and efficiency of a system. Moreover, difficulties might be encountered in designing an optimal HTC structure. The performance of an HTC system is strongly dependent upon how well the tree is designed [142].

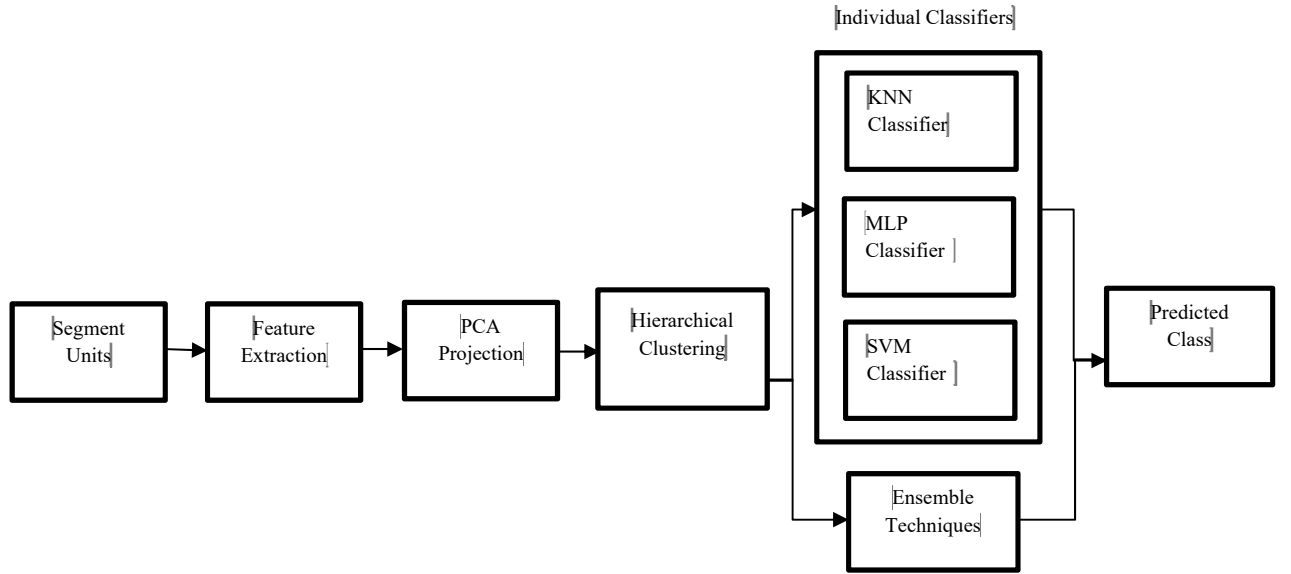
The main purpose of most real-world pattern recognition systems, is to discriminate between test instances belonging to different classes. The diverse classification algorithms have their own advantages and disadvantages depending upon the application of interest. Here, we propose to fuse evidence from several classifiers to enhance recognition accuracy. Researchers discussed several reasons for using ensemble systems including Statistical Reasons, Large Volumes of Data, Too Little Data, Divide and Conquer, and Data Fusion [143].

This chapter is organized as follows: The system architecture is first described in Section 4.1. Feature extraction and feature vector dimension reduction using PCA/LDA for the input data are described in sections 4.2 and 4.3. The hierarchical classification is discussed in section 4.4. Then, the individual and the ensemble based technique are described in sections 4.5 and 4.6. The experimental results are discussed in section 4.7. A conclusion is finally drawn in section 4.8.

## 4.2 The system Architecture

In this chapter, our main focus is on developing a Quran speech classification system using individual and ensemble-based classifier techniques. For our experiments, we used KACST database which was described in chapter 3. The HTR approach is discussed in section 4.4.

The proposed system architecture is organized around the following steps (Figure 4.1): Segment units as input, Feature extraction, PCA technique, Hierarchical classification, Three individual classifications: MLP [144], KNN [145] and SVM [146], then Ensemble based classification fusion to determine the predicted class. The input data is the segment units which were extracted based on our proposed algorithm described in chapter 3. These segments were extracted using an optimization technique to extract the boundaries. The work focuses on Quran recitation but is also applicable to other types of speech data. The total number of segment units obtained was around 4300 segment units. In the following sections, we will briefly discuss the different stages of the proposed algorithm.



**Figure 4.1: Flow Chart for the Proposed Arabic Speech Recognition System**

### 4.3 Feature Extraction

Before classification, we need to extract robust features from the speech data. Energy is the most important feature for identifying segment units, as vowels have more energy than consonants. Here, energy can distinguish between the main structure of the syllables. Pitch is commonly estimated using the autocorrelation method. Formants are the other important features. The popular linear predictive coding (LPC) method is commonly used to extract these formants. The Mel Frequency Cepstral Coefficients (MFCCs) are also very important in speech processing as they represent the short-term power spectrum in human-like Mel scale frequencies [147]. All of the above features have been used for representing segment units. In what follows, we will briefly describe these features and their importance with respect to identifying the segment unit type. It is worth noting that

this final list of features was obtained after numerous experiments with a larger pool of potential features.

### 4.3.1 The Energy Feature

The energy is one of the most important features used in speech processing. It is an excellent characteristic for discriminating between consonant and vowel syllables, as it has a high value for vowel syllables, and a low value for consonant syllables. The energy of a speech signal is calculated using Equation 4.1 [148]:

$$E_i = \sum_{n=1}^N x_i(n)^2 \quad (4.1)$$

where  $E_i$  represents the total energy for utterance  $i$  and  $x_i(n)$  represents sample  $n$  in segment unit  $i$ ,  $N$  is the total number of speech samples in the same segment unit. The resulting segment unit energy is usually normalized over the number of samples  $N$  to obtain an average energy per sample.

### 4.3.2 Pitch Feature

Pitch is commonly used to describe the perceived rise and fall in voice tone and is represented in the form of the fundamental frequency. It represents the vibration frequency of the vocal folds while speaking [149].

Numerous techniques can be used to estimate the pitch from a given speech signal. Here,

the autocorrelation method is used. The method uses a short-term analysis technique to maintain the characteristics for each frame. For each frame, we start by estimating the autocorrelation feature using Equation 4.2.

$$R(k) = \sum_{m=1}^N x(m)x(m+k) \quad (4.2)$$

where  $N$  is the frame length,  $\{x(m)\}$  is the signal frame,  $k$  is the shift or lag parameter, and  $R(k)$  represents the estimated autocorrelation function.

To address the variations in the pitch frequency values across the frames, we decided to extract 2 features related to it. These are: Variance and Max values across segment units.

### 4.3.3 Formant Frequencies

Formant frequencies are defined as resonance frequencies of the vocal tract and characterize timbre in vowels [150]. The formants are also very useful features for speech recognition and have been used substantially in diverse speech processing applications.

The peaks of the frequency response from the linear prediction filter are defined as the formant frequencies. These are estimated by computing the roots of the linear prediction coding (LPC) polynomial [151]. Linear prediction coding, as its name indicates, is used in representing a current sample as a linear combination of past samples [149].

$$\tilde{x}[n] = \sum_{k=1}^p a_k x[n-k] \quad (4.3)$$



where  $\tilde{x}(n)$  is the predicted sample at time  $n$ ,  $p$  is the number of past samples, and  $\{a_k\}$  are the LP coefficients. The prediction error is written as:

$$e[n] = x[n] - \tilde{x}[n] \quad (4.4)$$

It is easy to notice that the formant frequencies are estimated from the LP model by finding the poles of the all the pole AR or LP filter [151].

The first formant is an important feature for identifying the type of segment unit. To address the variations of the first formant across frames, we opted to extract 3 features related to it. These features are: Mean, Variance and Maximum of the first formant.

#### 4.3.4 The Mel-Frequency Cepstrum Features

The Mel-Frequency Cepstrum Coefficients (MFCCs) are used to represent the short time power spectrum of a sound. The MFCCs imitate the reaction of the human ear to sounds using Mel scales instead of linearly spaced frequency bands [149]. The expression for converting linear frequencies into the Mel scale is given by:

$$M(f) = 1125 \ln\left(1 + \frac{f}{700}\right) \quad (4.5)$$

where  $f$  is the frequency given in Hz.

The process of extracting the MFCCs from a speech signal is shown in Figure 4.2. First, the Fourier Transform of the windowed frame is computed. The results are squared to

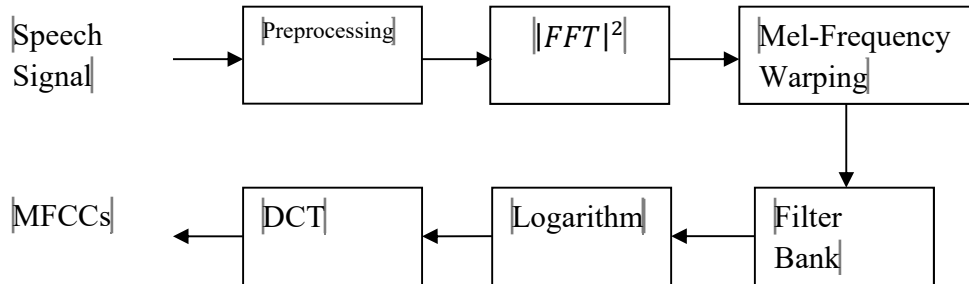
obtain the frame power spectrum. The Discrete Fourier Transform (DFT) is calculated using Equation 4.6.

$$S_i(k) = \sum_{n=1}^N S_i(n) h(n) e^{-\frac{j2\pi kn}{N}} \quad (4.6)$$

where  $i$  represent the speech signal,  $\{h(n)\}$  is an  $N$ -sample window, and  $k$  is the length of the DFT. The power spectrum is given by Equation 4.7.

$$P_i(k) = \frac{1}{N} |S_i(k)|^2 \quad (4.7)$$

Following the above, the Mel filter bank is applied to the power spectrum. The log of the resulting sequence is computed followed by the Discrete Cosine Transform (DCT) to obtain the final MFCC coefficients as shown in Figure 4.2.



**Figure 4.2: MFCC parameter estimation**

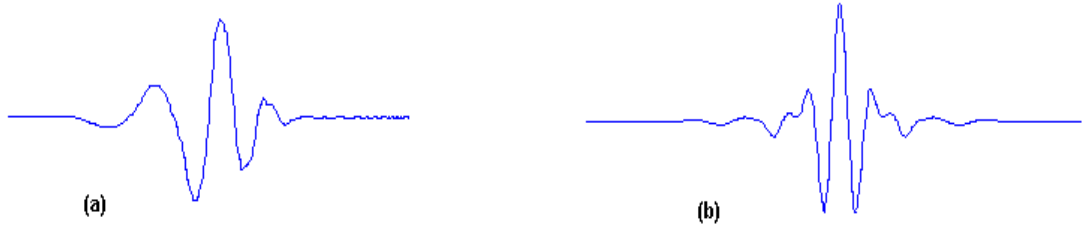
In this work, we estimated the mean and the variance of the first ten MFCCs. These coefficients cover frequencies below 1000 Hz which represent most of the energy in the signal. However, the first MFCC coefficient mainly measures the overall the

loudness. While in most speech processing applications, the first coefficient is ignored, here, we did consider it as loudness is affected by type of the segment unit.

### 4.3.5 The Discrete Wavelet Transform

In recent times, we have witnessed a lot of research directed towards the use of spectral-based features for diverse applications in speech processing. Among the different approaches, the Discrete Wavelet Transform (DWT), was shown to provide a good time and frequency resolutions, and can be used to extract robust features from speech. It can also be used to suppress noise from speech signals, and can provide to good representation for both stationary and non-stationary signals. The wavelet transform decomposes signals into successive levels from low to high frequencies, in what is known as a multiresolution decomposition. Such decomposition allows a detailed description of signals at both low and high frequencies. This is important for speech recognition, as low frequencies describe some important acoustic phenomena, which are essential to determine the segment unit type. Based on our own observations, and state-of-the-art, as well as our aim to keep computational complexity low, we decided to use the wavelet transform to extract a feature vector consisting of signal energies across different levels [152]. Instead of using the traditional dyadic decomposition; we have used here the wavelet packet decomposition with 7 levels as shown in Figure 4.4, where  $\Omega_{0,0}$  (the root node of the tree) represents the original signal space. The original signal is first projected over 2 subspaces, namely,  $\Omega_{1,0}$ , for the first half frequencies, and subspace  $\Omega_{1,1}$ , for the second half frequencies, and so on. This subspace decomposition continuous with each of

the components from the previous stage and so on. Here, we have used a simple Daubechies Mother Wavelet [153]. For example, the Daubechies wavelets as shown in Figure 4.3, are orthogonal and have compact support, but they do not have a closed analytic form and the lowest-order families do not have continuous derivatives everywhere. On the other hand, wavelets like the modulated Gaussian function or harmonic waveform are particularly useful for harmonic analysis due to its smoothness. This is the case of the Morlet and the Meyer (Figure 4.3 b) wavelets which are able to reflect amplitude information [153].

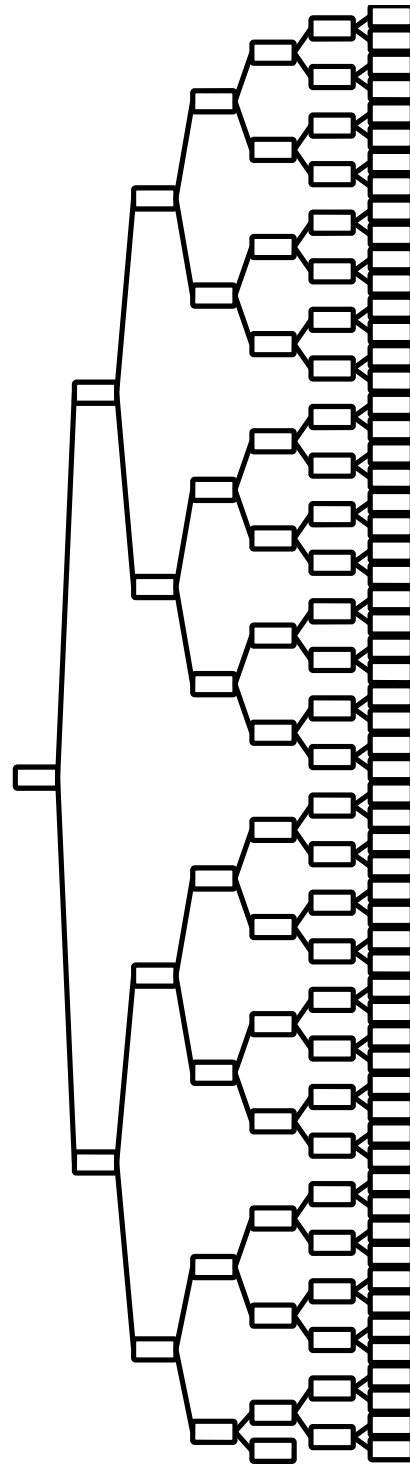


**Figure 4.3: (a) Daubechies-5 and (b) Meyer wavelets.**

The "optimal" choice of the wavelet basis has been shown to depend on the application of interest. For our case, the traditional Daubechies wavelets was a good option to visualize the time-varying frequency components of speech. In our work, the resulting feature vector consists of 255 energy values covering all frequency bands.

In addition to the traditional Daubechies mother wavelet (dB2), we also tested the performance of other types of mother wavelets. Our experiments showed very little difference in performance across different wavelets. A simple reason is that the DWT feature vectors obtained (using any of the mother wavelets) is further transformed into

uncorrelated components using PCA-LDA (see next section). Furthermore, it is worth noting that the daubechies family with its non-symmetry and overlapping properties has been shown to be among the most appropriate mother wavelets in speech applications [154], [155].



**Figure 4.4:** The wavelet packet frequency bands decomposition with seven levels

#### 4.4 Dimension Reduction using PCA

The initial results based on the features discussed above, were satisfactory but not excellent. In this work, we decided to further enhance this feature extraction stage by transforming the above features through projection over an orthogonal basis. For this purpose, we used a simple 2-stage transformation based on Principal Component Analysis (PCA) and Linear Discriminant Analysis (LDA).

First, the high dimension feature vector is transformed into a reduced-dimension feature vector consisting of uncorrelated components using PCA. PCA is implemented using eigenvalue decomposition the estimated covariance (or correlation) matrix. This is usually done after mean centering the data matrix for each attribute. For an original zero-mean dataset  $X$ , the estimated covariance matrix is given by:

$$S_X = \frac{1}{n}XX^T \quad (4.8)$$

Where  $X$  is an  $m \times n$  matrix,  $m$  is the number of features, i.e. the dimension, and  $n$  is the number of observations.  $S_X$  is a square symmetric  $m \times m$  matrix. The diagonal terms of  $S_X$  are estimates of the variances of different variables, and the off-diagonal terms represent the covariance between the variables. Now, let's start with the following PCA transformation:

$$Y = PX \quad (4.9)$$

Y is a representation of X over the new basis matrix  $P$ .  $P$  is a matrix that transforms X into the new coordinate system with sorted variances from large to small. The estimated covariance matrix of Y is given by:

$$\begin{aligned}
S_Y &= \frac{1}{n} Y Y^T \\
&= \frac{1}{n} (P X) (P X)^T \\
&= \frac{1}{n} P X X^T P^T \\
&= P \left( \frac{1}{n} X X^T \right) P^T \\
&= P S_X P^T
\end{aligned} \tag{4.10}$$

The estimated covariance matrix  $S_X$  can be decomposed using eigenvalue decomposition as:

$$S_X = U D U^{-1} \tag{4.11}$$

Where D is a diagonal matrix with entries being the eigenvalues ranked in descending order, and U is the matrix of the corresponding eigenvectors (as columns).

Since  $S_X$  is symmetric, the eigenvalues are real and  $U^{-1} = U^T$ , hence we can write:

$$S_X = U D U^T \tag{4.12}$$

Returning back to  $S_Y$ , assume we choose  $P = U^T$ , then:



$$\begin{aligned}
S_Y &= PS_X P^T \\
&= U^T S_X U \\
&= U^T (U D U^T) U \\
&= (P P^T) D (P P^T) \\
&= D
\end{aligned} \tag{4.13}$$

We can see that when the transformation matrix is chosen as  $P = U^T$ , the resulting transformed features (elements of Y) become uncorrelated since the results covariance matrix is diagonal. Since the eigenvalues and corresponding eigenvectors are ranked by order of importance, we can limit our transformation to only  $d$  elements hence P becomes an  $d \times n$  matrix.

In our work, we only kept the top two components after the transformation from the linear mapping, which we found to be enough for distinguishing between consonants and vowels in speech signals.

While simple and powerful, PCA can't account for class information as it assumes that every observation in the training set is a class by itself. To solve this problem, we followed PCA by a simple LDA (Linear Discriminant Analysis) projection. The LDA transformation is obtained by solving a generalized e-value e-vector decomposition problem. Listed below are the general steps for performing LDA performance:

1. Compute the  $d$ -dimensional mean vectors (for the PCA projection) for the different classes from the dataset, where  $\mathbf{m}_i$ 's are the mean vectors for different classes ( $i=1, 2, \dots, K$ ). Here  $d=2$ .

2. Compute the scatter matrices: LDA computes a transformation that the vector maximizes the between-class scatter while minimizing the within-class scatter. The projection matrix is obtained by maximizing the ratio  $S_b / S_w$ , where  $S_w$  is the within-class scatter matrix and  $S_b$  is the between-class scatter matrix.
  - a) Compute the Within-class scatter matrix  $S_w$

$$S_w = \sum_{i=1}^K S_i \quad (4.14)$$

Where  $S_i$  is the scatter matrix for every class and is computed by:

$$S_i = \sum_{x_j \in D_i} (x_j - m_i)(x_j - m_i)^T \quad (4.15)$$

Where  $m_i$  is the mean vector for class  $i$  and is computed by:

$$m_i = \frac{1}{n_i} \sum_{x_j \in D_i} x_j \quad (4.16)$$

- b) Compute the Between-Class scatter matrix  $S_b$

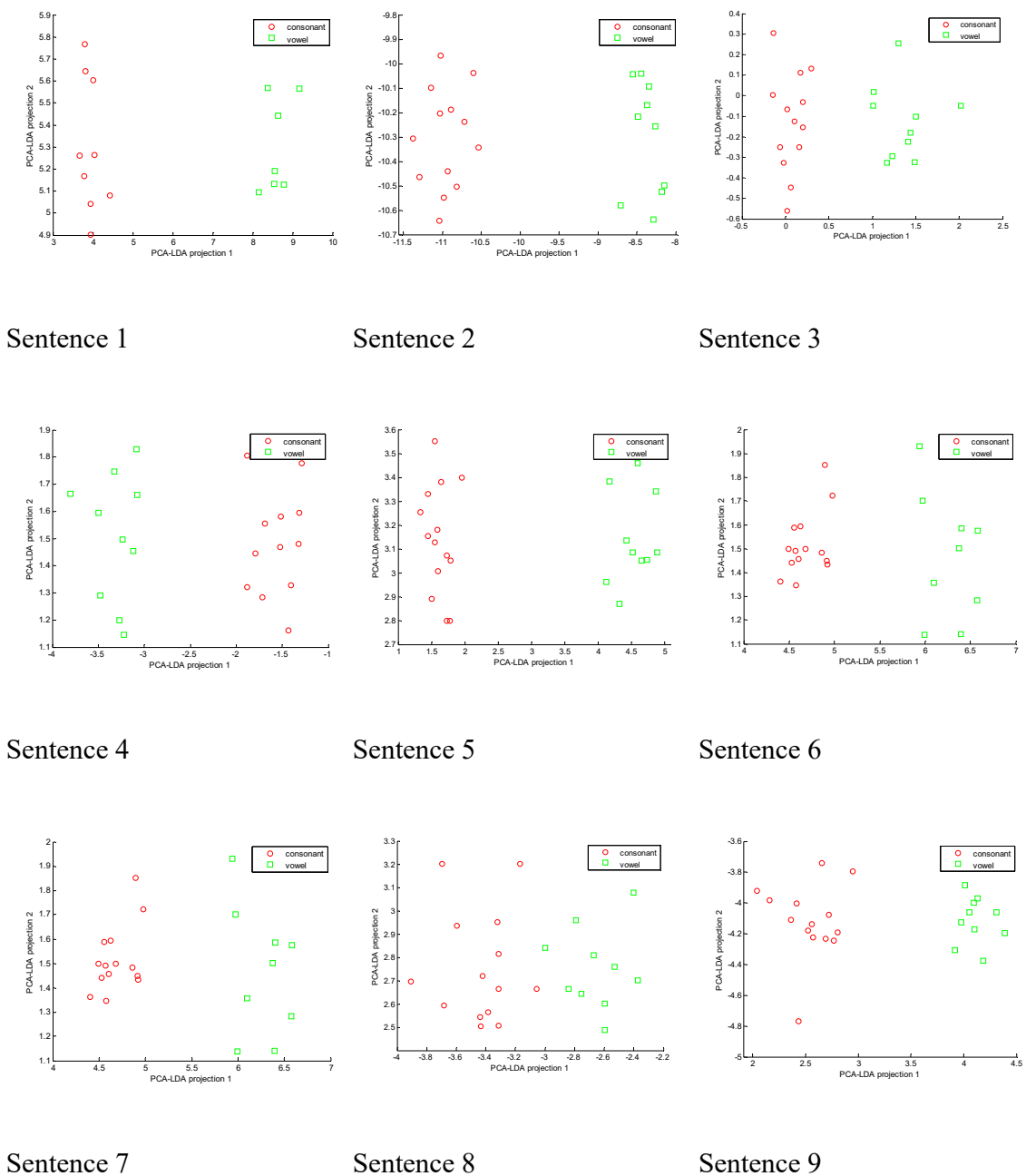
$$S_b = \sum_{i=1}^K n_i (m_i - m)(m_i - m)^T \quad (4.17)$$

Where  $m$  is the overall mean, and  $m_i$  and  $n_i$  are the sample mean and size of the respective classes in the training dataset.

3. Compute the generalized eigenvectors ( $e_1, e_2, \dots, e_d$ ) and corresponding eigenvalues ( $\lambda_1, \lambda_2, \dots, \lambda_d$ ) of  $S_w^{-1} S_b$ . The eigenvectors are basically the direction of a distortion in the linear transformation, and the eigenvalues are the scaling factors for the eigenvectors that describe the magnitude of the distortion.
4. Use the eigenvectors as rows of the transformed matrix  $W$  ( $d \times m$ ).

5. Use this  $d \times m$  matrix to transform the samples onto the new subspace:  $Y = W \times X$  (where  $X$  is a  $m \times n$ -dimensional matrix, and  $Y$  are the transformed  $d \times n$ -dimensional samples in the new subspace) [4].

From the above PCA-LDA projection, we obtain a 2-dimensional feature vector of uncorrelated and class dependent components. For our experiments, the two stage PCA-LDA projection resulted in optimal 2-dimensional feature vectors able to achieve excellent consonant-vowel recognition accuracy. The scatter plot for the consonant and vowels from the data after applying the PCA-LDA transformation is shown in Figure 4.5.



**Figure 4.5** Scatter plot of first and second of PCA-LDA projections.

In this new space, we see that consonant projections are well separated from vowel projections, with a clear boundary between these. In Figure 4.6, we used the top feature from the projection to plot the histograms for consonants and vowels, and their

corresponding simulated probability distribution functions (pdfs) as Gaussian distributions. We see that the two classes are well separated hence the resulting features can be used to achieve excellent accuracy in speech classification between consonants and vowels.

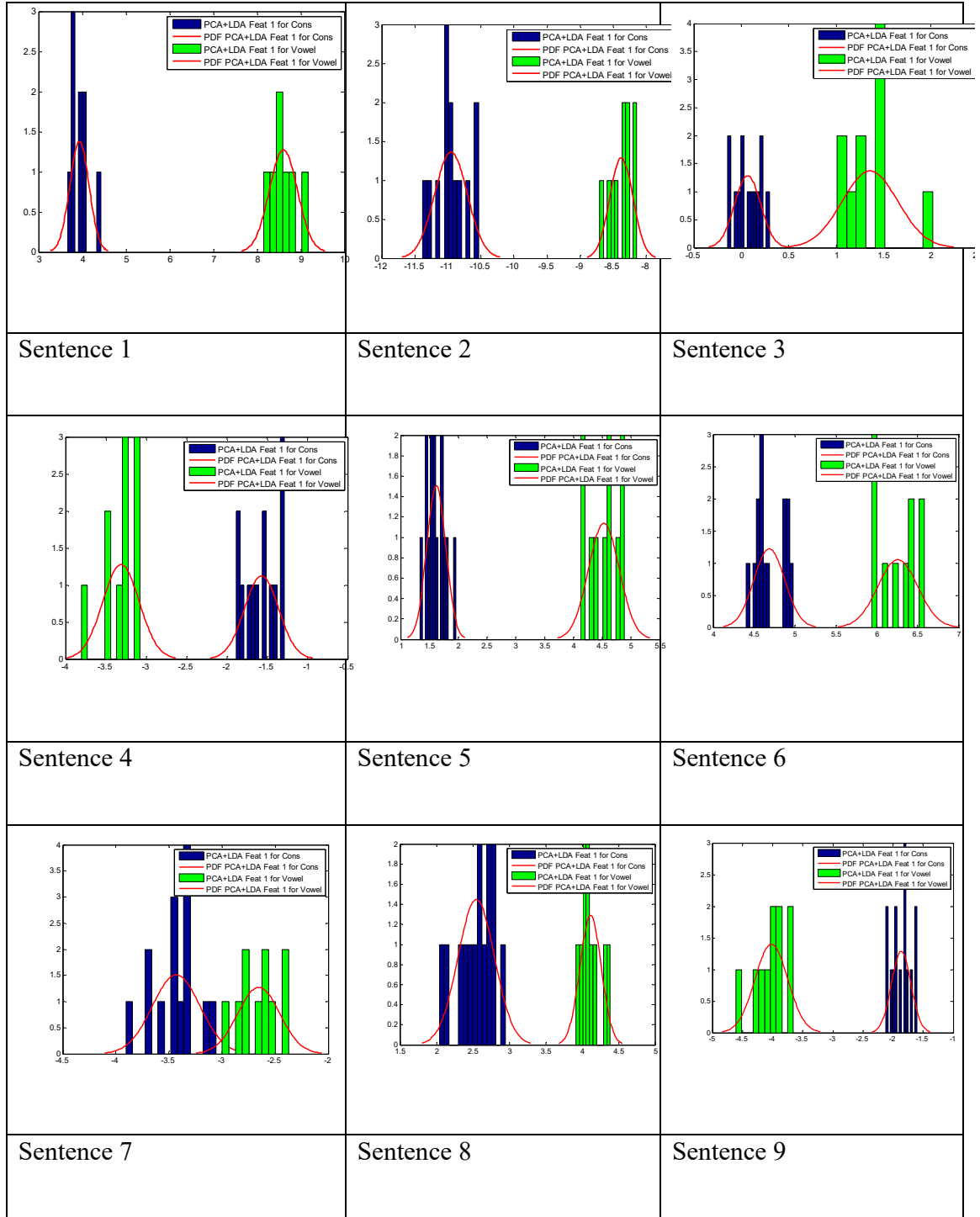


Figure 4.6 Histograms and simulated pdfs for the top feature from PCA-LDA projections

## 4.5 Hierarchical tree classification

In designing Hierarchical Tree Classification algorithms (HTC), it is important to search for the appropriate tree structure and the feature subsets to be used at each node. The simplest approach is to divide the problem into sub-problems that have no common elements, also called a “hard split” [156]. Our proposed HTC system is shown in Figure 4.7. The proposed HTC here, has partly been based on the prior knowledge of segment unit classes based on Tajweed rules [157]. This was motivated by the observation that these classes are easy to be distinguished acoustically. At the root of the tree, CVs and CVCs classes are to be discriminated by recognizing the last three frames for each segment units. Then, both are further split into subsets of classes. CVs are split into four leaves (terminal nodes) that represent: bold and nasal, bold and not nasal, nasal, not bold and nasal, and not bold not nasal. Then each terminal, also splits into two leaves that represent short vowel (V), and long vowel (VV). CVCs are also divided into bold and nasal (e.g. **مِنْ** قال), bold and not nasal (e.g. **قَالَ**), not bold and nasal (e.g. **أَنْتُمْ**), and not bold not nasal (e.g. **سِئَالٍ**). Then, each terminal, split into four leaves that represent short vowel (V) (e.g. **قَالَ**), long vowel (VV) (e.g. **قَالَ**), double long vowel (V4) (e.g. **سِئَالٍ**), and triple long vowel (V6) (e.g. **سِيعْلُمُون**). We notice in Figure 4.7 that V4 and V6 classes are not included in CVs branch. The class V4 is satisfied, when the vowel is followed by letter (ء). Where V6 is satisfied when the vowel is followed by a consonant when the reader stops reading the ayah. However, branching can be continued until each branch in the tree is assigned only one segment unit.

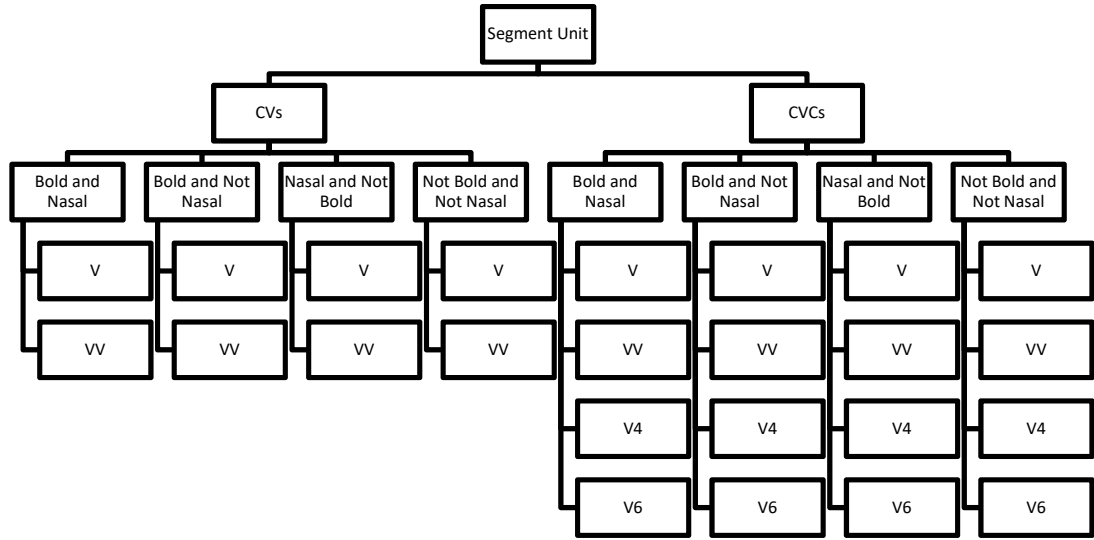


Figure 4.7: The proposed segment unit classification tree

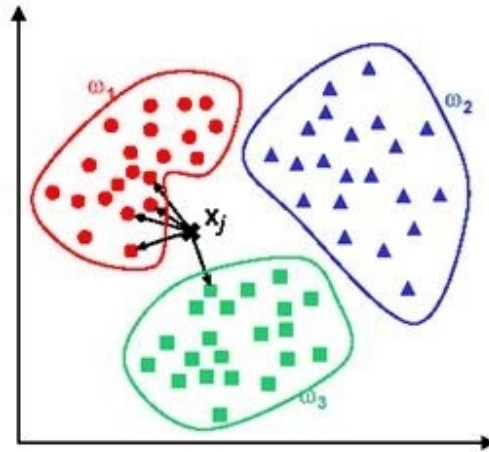
## 4.6 Individual Classifiers

As the focus of the work is on the classifier combination techniques, we selected here 3 basic classifiers commonly used in the literature (KNN, MLP and SVM). These are briefly described below:

### 4.6.1 K-Nearest Neighbor (KNN)

The K-Nearest-Neighbor classifier is one of the oldest classical classification techniques. When a new feature vector needs to be classified, its k-nearest-neighbor vectors are found together with their class labels. The assigned class of the new vector is decided by considering the majority of its neighbors as shown in Figure 4.8.





**Figure 4.8:** An example for assigning a new vector to one of the classes using the KNN classifier

The KNN refers back to the raw training data in the classification for each new sample. Therefore, one can say that the entire training set is the classifier. The basic idea is that the similar tuples most likely belong to the same class. Based on some pre-selected distance metric, we find the  $k$  most similar or nearest training samples of the sample to be classified and assign the plurality class of those  $k$  samples to the new sample. The value for  $k$  is pre-selected. Using relatively larger  $k$  may include some samples not so similar samples and, on the other hand, using very smaller  $k$  may exclude some potential candidate samples. In both cases, the classification accuracy decreases. The optimal value of  $k$  depends on the size and nature of the data. The typical value for  $k$  is 3, 5 or 7 [73].

The main steps for the classification process are:

1. Determine a suitable distance metric.
2. Find the  $k$  nearest neighbors using the selected distance metric.
3. Find the plurality class of the  $k$ -nearest neighbors (voting on the class labels of the NNs).

4. Assign the sample to the class with highest number of votes.

When a new sample arrives, KNN finds the  $k$  neighbors nearest to the new sample from the training space based on some suitable similarity or closeness metric. A common similarity function is based on the Euclidian distance between two data tuples. For two tuples,  $X = (x_1, x_2, x_3, \dots, x_n)$  and  $Y = (y_1, y_2, y_3, \dots, y_n)$  (excluding the class labels), the Euclidian similarity function is:

$$d_2(X, Y) = \sqrt{\sum_{i=1}^n (x_i - y_i)^2} \quad (4.18)$$

A generalization of the Euclidean function is the Minkowski similarity function given by:

$$d_2(X, Y) = \sqrt[q]{\sum_{i=1}^n w_i |x_i - y_i|^q} \quad (4.19)$$

The Euclidean distance is obtained by setting  $q$  to 2, and each weight,  $w_i$ , to 1. The Manhattan distance is defined as:

$$d_1(X, Y) = \sum_{i=1}^n |x_i - y_i| \quad (4.20)$$

Which is obtained by setting  $q$  to 1 and using equal weights. Setting  $q$  to  $\infty$ , results in the max function:

$$d_{\infty}(X, Y) = \max_{i=1, \dots, n} |x_i - y_i| \quad (4.21)$$

After finding the  $k$  nearest tuples based on the selected distance metric, the plurality class label of those  $k$  tuples can be assigned to the new sample as its class. If there is more than one class label in the plurality, one of them can be chosen arbitrarily [71].

The KNN does not build a residual classifier, but instead, searches again for the  $k$ -nearest neighbor set for each new sample. This approach is simple and can be very accurate. The KNN is a good choice when simplicity and accuracy are the predominant issues. The KNN can be superior when a residual, trained and tested classifier has a short lifespan, such as in the case with data streams, where new data arrives rapidly and the training set is ever changing [71].

The major drawback of this classifier is that all the training data should be stored beforehand for classifying new patterns. In addition, the classification performance is significantly affected by the choice of  $k$ . If  $k$  is too small, the classification performance is directly affected by the outliers in the training set. So, for large datasets, KNN classification is not the preferred method, but commonly used in benchmarking studies and cross-validation.

#### 4.6.2 The Multi-Layer Perceptron (MLP) Network

The study of neural networks (NN) attracted much attention over the last two decades. During this time, various types of neural network structures were suggested. The basic NN structure is a single layer feed forward network. This basic processing unit is called perceptron. It is trained using a simple delta rule. It can form simple decision boundaries for a given classification problem. The perceptron properties and relation with the simple statistical classifiers were explored in [60].

Later, the multi-layer perceptron network (MLP) was introduced. Learning mechanisms are similar to traditional NN but more complex than the simple perceptron network. It was proven that a three layer MLP network can produce arbitrary complex decision regions over a given feature space. The details of the analysis are available in textbooks and references such as [59].

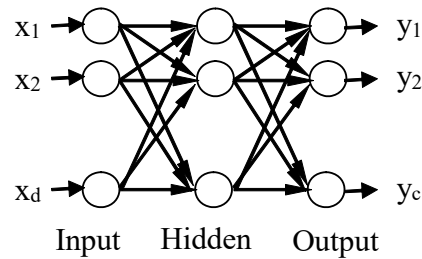
Before exploring the training algorithms for MLPs, we discuss the topology and characteristics of the MLPs. An MLP is a directed graph of nodes that have simple and similar behavior. Traditionally, a node in such a graph is called neuron. Each neuron is a processing unit which has some inputs, some outputs and a transfer function (frequently called as activation function). Simple linear neuron adds up the input signals and transfers the result to its output. However, the output function has other forms for stabilizing the network and leading to a smooth learning phase. The most commonly used output functions are threshold (Equation 4.22) and sigmoid (Equation 4.23) functions.

$$g(x) = \begin{cases} 0 & x < 0 \\ 1 & x \geq 0 \end{cases} \quad (4.22)$$

$$g(x) = \frac{1}{1 + e^{-x}} \quad (4.23)$$

An MLP can be constructed by arranging neurons as layers and connecting the outputs of each layer as input to the next layer. In the simple MLP, each consecutive layer is fully connected, but other nonsymmetrical connections are also possible.

We mentioned that an MLP with three layers could learn arbitrary mappings between input and output. Actually, this is mostly related to the number of adjustable parameters of the network. For MLP, the adjustable parameters are the number of hidden layer neurons. This choice implicitly raises or lowers the number of network weights that are adjusted in the training stage. The basic topology for a three-layer MLP is given in Figure 4.9.



**Figure 4.9: Typical standard of Multilayer perceptron network.**

Assume that there are  $d$  inputs,  $c$  outputs (or classes), and  $M$  hidden units. Then,  $w_{ji}^k$  denotes the weight of the link connecting  $i^{\text{th}}$  unit of layer  $(k-1)$  to the  $j^{\text{th}}$  unit of layer  $k$ . The unit indices start from one and the zero weight is reserved for bias values. So, the output of the  $j^{\text{th}}$  unit in the first hidden layer for a given input vector  $x$ , and activation function  $g(\cdot)$  will be as follows:

$$o_j^1 = g\left(\sum_{i=0}^d w_{ji}^1 x_i\right) \quad (4.24)$$

There are two training algorithms for MLPs. The first one is called the gradient descent (GD) training algorithm. The second one is called Conjugate Gradient Descent (CGD) algorithm that is, in fact, a well-known optimization algorithm. However, several other optimization algorithms can be used in training multi-layer perceptron networks, such as Newton's method, the Quasi-Newton method involving other techniques for speeding-up the process [74].

### 4.6.3 The Support Vector Machines (SVM) Algorithm

Support Vector Machines (SVMs) are classifiers drew particular interest in recent years. Although, the theoretical basis for SVMs were known decades ago, Their full use was not explored until the 90's [76]. The review provided here is mostly inspired from the tutorial in [77]. SVMs are supervised classification and regression engines, although there are some recent suggestions about their semi supervised variants [76]. Much of the

work on SVM classifiers relates to the binary classification problem, with multiclass classifier constructed by combining several binary classifiers.

Given a model (classifier), *capacity* can be defined informally as the extent of data that the model could successfully learn. Capacity is related to the number of free parameters of the given model. In pattern recognition, one of the major challenges is what we call the *bias-variance tradeoff*. In other words, a classifier is successful if it could generalize the instance of data it has seen during the training stage, rather than memorizing it. Therefore, the number of free parameters for a model plays the most important role in the overall performance. For example, in the case of neural networks, the free parameters are usually fixed after several experiments on the given dataset.

The SVM can be seen as a classification machine that addresses diverse classification problem more than other classifiers. The basic idea behind SVMs stems from the concept of *risk-minimization*. Let  $T$  be the set of input-output pairs, and  $l$  the number of elements in the set  $T$ :

$$\begin{aligned} T &= \{(\mathbf{x}_i, y_i) \mid \mathbf{x}_i \in \mathfrak{X}, y_i \in \{-1, 1\}\} \quad \text{where } \mathfrak{X} = \mathbb{R}^d, \\ \lambda &= |T|, \text{ the number of elements in set } T, \\ T &\text{ is called input - output pairs.} \end{aligned} \tag{4.25}$$

The classification (or regression) machine,  $f(\cdot)$ , becomes:

$$\begin{aligned} f : x_i &\alpha y_i \quad \forall i \in \{1 \wedge \lambda\} \\ y_i &= f(x, \alpha) \quad \text{where } \alpha \text{ be all adjustable parameters of } f \end{aligned} \tag{4.26}$$

SVMs can be used to classify data under two possible scenarios: The linearly separable projection and the nonlinear projection. In our work, we used the linearly separable scenario.

### Linearly Separable Datasets

SVM is traditionally used to project non-linearly separable data into higher-dimensional space using kernel techniques. In higher dimensions, the data points become linearly separable. SVM based classifiers, are learn as universal classifiers that are statistically robust learning methods based on structural risk minimization. SVM based classifiers seek to find optimal separating hyper-planes, in order to maximize the margin between classes of data in the projection space.

Assume that we have  $m$  training samples, and each sample consists of an  $(x_i, y_i)$  pair where  $x_i$  is  $N \times 1$  containing attributes of the  $i^{\text{th}}$  sample, and  $y_i \in \{+1, -1\}$  is the class label for the sample. The objective of the SVM is to find the optimal separating hyper-plane  $\vec{w} \cdot \vec{x} + b = 0$ , between the two classes of data. Where  $\vec{w}$  is the normal vector to the hyperplane. The parameter  $\frac{b}{\|\vec{w}\|}$  determines the offset of the hyperplane from the original along the normal vector  $\vec{w}$  as shown in Figure 4.10.

If the training data is linearly separable, we can select two parallel hyperplanes that separate the two classes of data, so that the distance between them is as large as possible. The region bounded by these two hyperplanes is called the "margin", and the maximum-margin hyperplane is the hyperplane that lies halfway between them. With proper dataset rescaling, these hyperplanes can be described by the following equations:



$\vec{w} \cdot \vec{x} + b = 1$  (anything on or above this boundary is of one class, with label 1)

And

$\vec{w} \cdot \vec{x} + b = 0$  (anything on or above this boundary is of one class, with label -1)

The distance between these two hyperplanes is  $\frac{b}{\|\vec{w}\|}$ , so to maximize the distance between the planes we want to minimize  $\|\vec{w}\|$ . The distance is computed using the distance from a point to a plane equation. We also have to prevent data points from falling into the margin, we add the following constraint: for each  $i$  either

$$\vec{w} \cdot \vec{x} + b \geq 1, \text{ if } y_i = 1$$

Or

$$\vec{w} \cdot \vec{x} + b \leq -1, \text{ if } y_i = -1$$

These constraints state that each data point must lie on the correct side of the margin. This can be rewritten as:

$$\arg \min_w \frac{1}{2} \|\vec{w}\|^2 \tag{4.28}$$

subject to

$$y_i(w \cdot x_i + b) \geq 1, \text{ for all } 1 \leq i \leq m$$

Where  $m$  is number of samples. The corresponding classifier is  $sgn(\vec{w} \cdot \vec{x} + b)$ .

The constraints aim to put the samples with positive labels at one side of the margin  $w \cdot x + b \geq 1$ , and the ones with negative labels at the other side  $w \cdot x + b \leq -1$ .

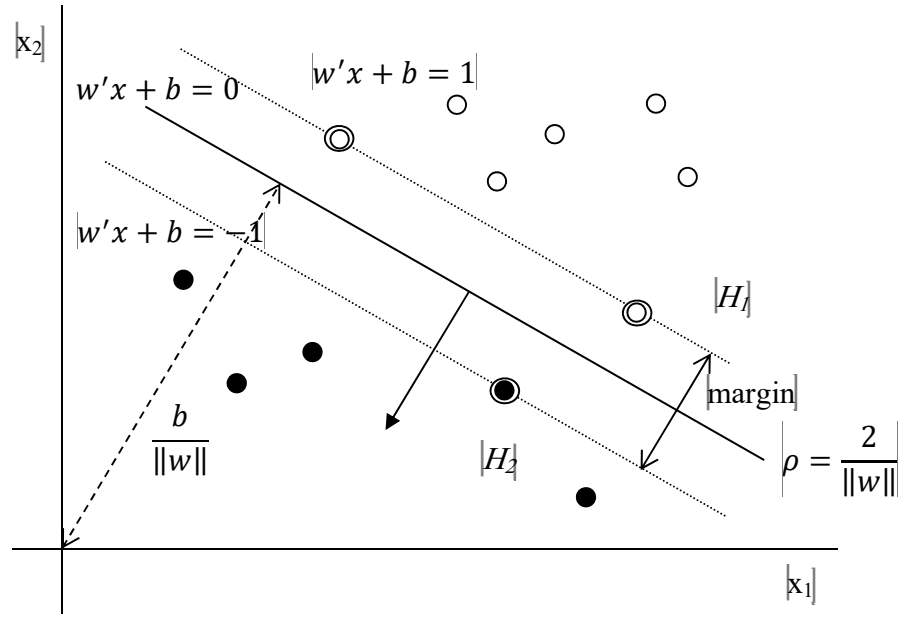


Figure 4.10: SVM widen the margin between the two classes of data that are needed to be classified.

The classification approached outlined above have their own advantages and disadvantages. In our work, we will use ensemble classifiers as a combination tool between the individual classifiers to enhance the results. The details of ensemble classifiers will be explain in the next section. .

## 4.7 Ensemble Classifiers

Classifier combinations in pattern recognition attracted much attention in the last decade due to several reasons. First, combining classifiers give rise to extra dependability and proven to decrease the variance of the different classifiers. This is shown to be true even if the combination performs equal or slightly less than the best classifier in the combination [81].

Another reason is different classifiers split the feature space differently and each classifier probably performs better on different regions of feature space. In addition, some initialization dependent classifiers, such as neural networks, converge to different sub optimum solutions [81]. By combining the classifiers, the learning attempts of different networks are utilized efficiently.

Classically, a combiner combines the outputs of the given classifiers to make the final decision. Therefore, the combination scheme depends on the selected architecture of the overall system. Parallel, cascade, or hierarchical combinations of classifiers are possible.

In parallel combination architecture, each classifier is fed with the given feature and the outputs are combined. In the cascaded architecture, classifiers are applied in sequence while the possible number of classes is decreased. The hierarchical approach is similar to the decision tree classifiers. Instead of the decision nodes acting on features, classifiers nodes are placed to form a classifier tree. These three basic architectures can also be combined to form more complex classification systems.

Simple analysis shows that the discrimination ability of the combination increases as the individual classifiers pose low correlation. Therefore, one should utilize the techniques for guaranteeing this independence to certain level. The simplest way to ensure the classifier independence is to use different training sets for each of the classifier in the combination. If sample is abundant, independent classifiers are easily produced. In general, this is not the case for the expert systems where manual labeling should be done by the expert.

An important issue in combining classifiers is that the outputs of the classifiers should be compatible in order for the combination to make sense. Furthermore, the behavior of the individual classifiers should be carefully studied. The combination techniques are divided into three categories as mentioned in [87]. In confidence level, the classifier makes the membership probably estimation for each class. In rank level, a classifier assigns ranks to the classes. In abstract level, the classifier only outputs a class label. Thus, the information level decreases from confidence level to abstract level.

For example, MLP networks with sigmoid output units belong to the confidence level combination, since the output nodes of MLP are known to represent the approximation of class probabilities for the given input pattern. KNN belongs to the abstract level since it only outputs a single label.

The common combination rules by the individual classifier output information level are presented in Figure 4.11. In this Figure, there are three main levels; Abstract level, confidence level and rank level. The abstract classifiers can be combined with majority voting method. The confidence level classifier can be combined by any combination

method since they convey the richest information. The outputs can simply be converted to ranks by sorting the class probabilities in descending order and ranking, or, the maximum output can just be picked to perform majority voting in abstract level. Combination of the rank based classifiers is the least studied one among the three types. Recently, such a rank based method is suggested in [88], to formulate a discrete optimization problem as a maximization in search for the best total probability of correct decision.

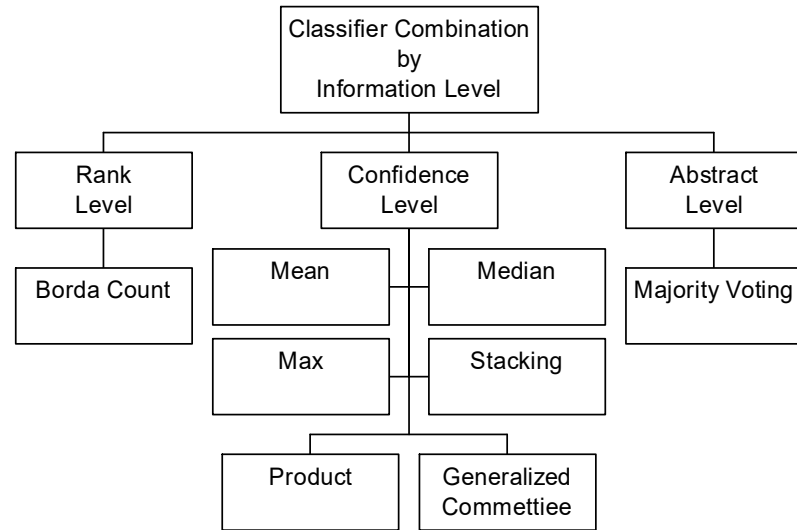


Figure 4.11: Classifier combination methods used by information level.

In the following discussion, we assume that only the class labels are available from the classifier outputs. Let us define the decision of the  $t^{\text{th}}$  classifier as  $d_{t,j} \in \{0, 1\}$ ,  $t = 1, \dots, T$  and  $j = 1, \dots, C$ , where  $T$  is the number of classifiers and  $C$  is the number of classes. If  $t^{\text{th}}$  classifier chooses class  $\omega_j$ , then  $d_{t,j} = 1$ , and 0, otherwise.

The ensemble decision for the plurality voting can be described as follows: choose class  $\omega_J$ , if:

$$\sum_{t=1}^T d_{t,J} = \max_{j=1} \sum_{t=1}^T d_{t,j} \quad (4.30)$$

There are three assumptions for the majority voting: 1. Number of classifiers should be odd number; 2. The classifier outputs are independent; 3. The probability of each classifier choosing the correct class is  $p$ . Hence,  $P_{ens}$ , the probability of ensemble success is:

$$P_{ens} = \sum_{k=(\frac{T}{2})+1}^T \binom{T}{k} p^k (1-p)^{T-k} \quad (4.31)$$

Assume that the outputs of the classifiers are given in confidence level. Then each classifier gives a probability estimate for each of the possible classes. The Mean or Sum Rule adds up the given output vectors to form a decision. If the outputs of the sum rule it to be used in another classifier combination stage, then it should be normalized. If the decision should be made, the class with maximum value will be selected. In addition, by imposing a certain threshold of the maximum output value, rejection measure can be employed for the problem if required.

In the Product Rule, the outputs are multiplied element by element to form the combined output vector and the result is normalized. This rule is very sensitive to the most pessimistic classifiers: a low support (close to 0) for a class from any of the classifiers can effectively remove any chance of that class being selected. However, if individual posterior probabilities are estimated correctly at the classifier outputs, then this rule provides the best estimate of the overall posterior probability of the class selected by the ensemble.

$$\mu_j(x) = \frac{1}{T} \prod_{t=1}^T d_{t,j}(x) \quad (4.32)$$

For the Median Rule of combination, each output probability estimate is grouped and sorted among them. The median of each group is taken to form the output vector. If there is even number of classifiers, the mean value of the values, belonging to the nearest integer indices is computed. The Borda Count method is originally suggested for the election systems. Since classifier combination can be thought as some form of election, it can be applied for combining classifier outputs. In this method, each classifier output vector is arranged in ascending order and a rank score equal to the index of the class is given to the corresponding class. Then, class scores are added up to find the output vector from which the class having maximum value is taken as winner. The output vector can be normalized to be fed into another stage of classifiers.

Another type of combination technique is to combine the output of the classifiers by linear weighting. The weights should be chosen such that the output error of the

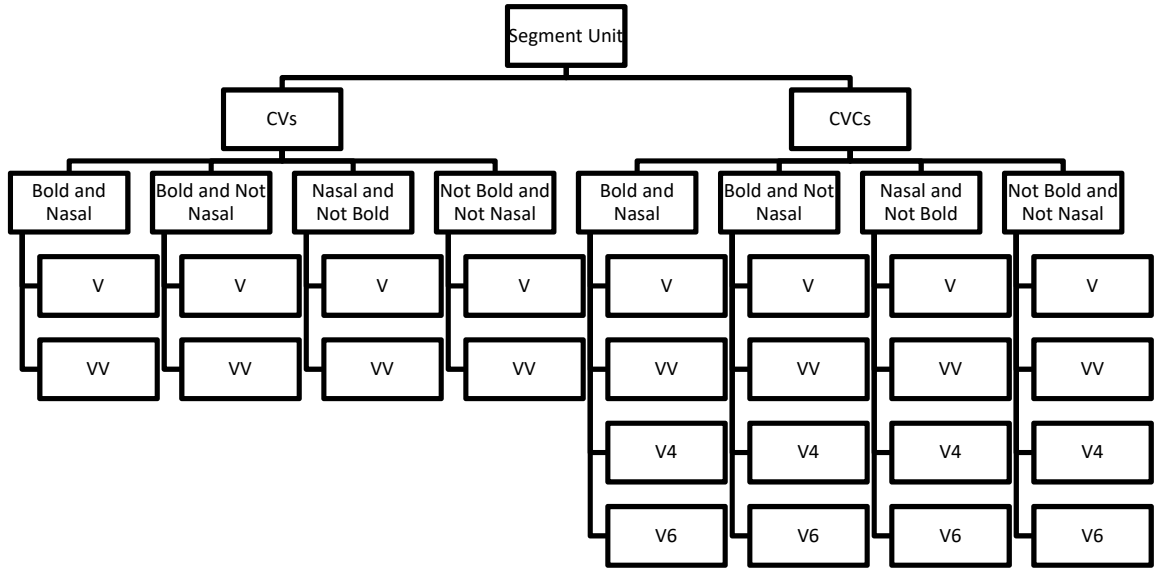
combination with respect to some known dataset is minimized. The application of such a combination is certainly expected to perform better than the mean rule since it is a generalized version of the mean rule.

In our work, we have selected combination algorithms to cover different type of combinations. In particular, we selected the following combinations: Algebraic combiners (Minimum/ Maximum/ Median/ Product Rule), and Majority voting. The main advantage of using these ensemble classifiers is decreasing the variance of the different classifiers. In the next section, we will use the HTC and the ensemble classifiers in our work.

## **4.8 Experimental Results**

In this section, we discuss our results following the experimental settings described above using the segment units that were extracted as explained in chapter 3. The total number of segment units is around 4300. These segment units contain CVs and CVCs, and the total number of classes is 24 classes as shown in Figure 4.12.





**Figure 4.12: The proposed segment unit classification tree**

In our work, we performed segment unit classification experiments for each stage separately using the following the following 281 features [161]:

- The energy (1 feature).
- Variance and max the pitch (2 features).
- Mean, variance, and max the first formant (3 features)
- Mean and variance the first ten coefficients of MFCC (20 features).
- Mean the seven-layer energy coefficients of Wavelet (255 features).

For our first set of experiments, we started by a binary classification between the main CVs and CVCs categories. We segmented the speech units into frames of 1200 samples. The consecutive frames were then overlapped with 300 samples. Then, each frame was windowed using a Hamming Window. We used the LDA-PCA transformation to project

the features into 50 uncorrelated features for each frame. These 50 uncorrelated features had a cumulative explained variance of more than 90% of the full variance of the data [162]. Hence, the input data is a 4300 x 50 matrix, and this feature matrix is passed to our HTC, as described in section 4.4. This tree represents the classification of the segment units, according to the prior knowledge about these segments based on the Tajweed rules.

In our experimental setup, we used 80% of the data for training, and the remaining 20% for testing. We used the individual and the ensemble-based classifiers as classification tools to distinguish between the classes. For individual classification, we used 3 types: MLP, KNN and SVM. Whereas for ensemble-based classifiers, we used Majority, Maximum rule, Summation rule, Minimum rule, Average rule, and Product rule.

First, we analyzed the results for the first stage (CVs and CVCs). The main difference between these is in the last part of the segment unit as either a consonant or a vowel. So in this stage, the last three frames are selected to identify these frames as consonant or vowel, where the minimum number of frames for the consonant or the vowel was chosen to be three. The results were then repeated over 10 runs randomly shuffled. Across all the runs, we have around 8600 (860x10) test segment units covering all the class types. The average accuracies for this stage are shown in Table 4.1. We notice in this table that the average accuracy is around 99.3%.

Table 4.1 The confusion matrix for the first stage in our HTR system (CV vs. CVC)

	CVs	CVCs	Accuracy
CVs	5478	21	99.62%
CVCs	32	3069	98.97%
Average Accuracy			99.3%

Second, we passed the results from the first stage to the second stage in our proposed HTC algorithm. In this stage, we have four main classes as follows: bold and nasals, bold and not nasal, not bold and nasal, and not bold and not nasal. The results are shown in Tables 4.2 and 4.3.

Table 4.2 Confusion matrix for second stage recognition in CVs

Predicted \ True Class	not bold and not nasal	not bold and nasal	not bold and not nasal	not bold and nasal	Recognition Accuracy
not bold and not nasal	4131	46	73	91	95.18 %
not bold and nasal	91	146	18	27	52.09 %
bold and not nasal	255	27	356	36	52.34 %
bold and nasals	55	18	27	100	51.58 %
Average Accuracy					86.09 %

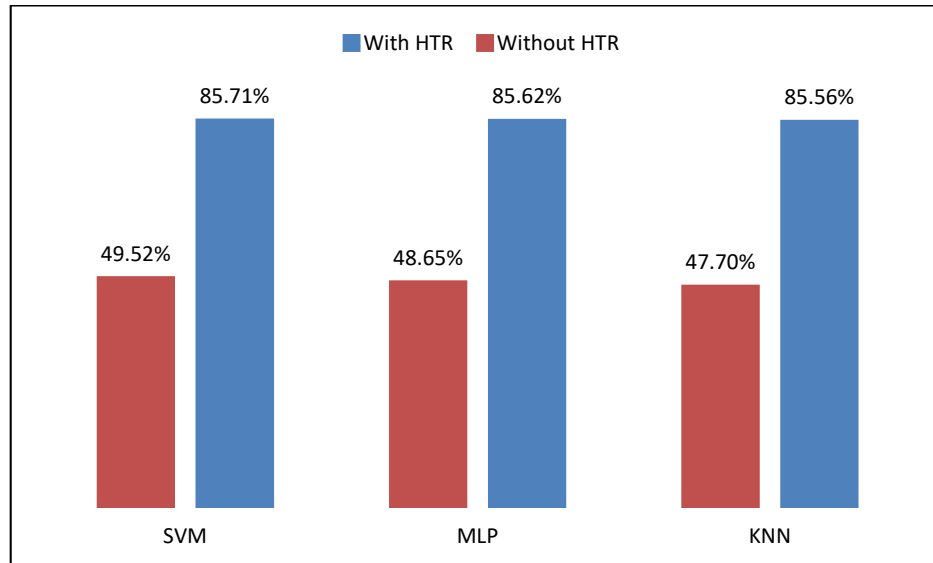
**Table 4.3 Confusion matrix for second stage recognition in CVCs**

<div> <div>Predicted</div> <div>True Class</div> </div>	not bold and not nasal	not bold and not nasal	not bold and not nasal	not bold and not nasal	Recognition Accuracy
not bold and not nasal	<b>2597</b>	26	9	17	97.75 %
not bold and nasal	26	<b>60</b>	9	17	53.33 %
bold and not nasal	68	34	<b>111</b>	9	48.96 %
bold and nasals	34	17	26	<b>43</b>	36.73 %
<b>Average Accuracy</b>					<b>90.03 %</b>

The results show clearly the importance of identifying the consonants and vowels before identifying the type of CVs or CVCs classes. The proposed algorithm achieved 87% in overall recognition under CVs and 90% in overall recognition under CVCs compared to the case of segment unit recognition without considering the HTC (49%) as was shown in Table 4.2. The proposed framework using HTC showed an improvement of about 34% in recognition accuracy. While the segment unit recognition accuracy was barely 49% when all classes were mixed, such accuracy reached around 90% (89.13%) with the inclusion of HTC framework.

In the third stage, we used the segment unit length as a feature to distinguish between V and VV for CVs classes, and V, VV, V4, and V6 for CVCs classes. The

overall average accuracies of the individual classification methods with and without HTC are presented in Figure 4.12. We observe, from Figure 4.12, the accuracy using HTC is much better compared to the results without HTC.



**Figure 4.12: The Overall Average Accuracy of Individual Classifiers with and without HTR**

To improve the performance of the individual classifiers, we implemented 6 types of ensemble-base classifiers: Majority voting rule, Maximum rule, Summation rule, Minimum rule, Average rule, and Product rule[163]. The average accuracies using these methods are presented in Tables 4.4.

**Table 4.4 Average accuracy of Ensemble Classifiers with and without HTR**

	Ensemble Techniques						Average Improvement
	Majority	Maximum	Summation	Minimum	Average	Product	
Average Accuracy without HTR	47.75	47.69	48.01	48.11	48.02	48.03	<b>47.82</b>
Average Accuracy with HTR	85.69	85.36	<b>85.74</b>	<b>85.74</b>	<b>85.74</b>	85.67	<b>85.50</b>

Based on the above results, the average accuracies of the individual classifiers are very comparable. But in almost cases, the best performance is obtained with the SVM algorithm, which is shown to be effective more in managing high dimensional data. The results show that, we can achieve a slight improvement by using ensemble-based classifiers with a negligible additional computational cost. However, we must take into consideration the fact that the performance of each of the methods is depending upon the base algorithms used, hence both methods and combination approaches must be considered in conjunction.

To summarize our results, we conducted a statistical analysis of the proposed approach considering both CVs and CVCs stages. In particular, we used our results from the confusion matrices to derive the accuracy, precision, recall, and finally F-measure. The F-measure has been used as an excellent index for measuring test accuracy. Instead

of considering only accuracy, the F-measure considers both precision and recall. It reaches its best value at 1 and worst at 0.

The values for accuracy, precision, recall and F-measure are displayed in Tables 4.5 and 4.6. For the case of CVs, we reach a maximum F-measure of 0.97 for the two classes: the not bold and nasal class and the bold and nasal class. As expected, the nasals case is usually easier for identifying because it has longer periodic time than the other types of classes. Overall, we see that the proposed HTC methodology achieves a very good recognition accuracy comparing without using HTC methodology.

**Table 4.5 Accuracy, precision, recall and F-measure using the second stage under CVs**

	<b>Accuracy</b>	<b>Precision</b>	<b>Recall</b>	<b>Fmeasure</b>
<b>not bold and not nasal</b>	95.18	78.36	65.42	0.71
<b>not bold and nasal</b>	52.09	97.42	98.23	0.97
<b>bold and not nasal</b>	52.34	93.55	97.56	0.95
<b>bold and nasals</b>	51.58	98.13	97.06	0.97

**Table 4.6 Accuracy, precision, recall and F-measure using the second stage under CVCs**

	<b>Accuracy</b>	<b>Precision</b>	<b>Recall</b>	<b>Fmeasure</b>
<b>not bold and not nasal</b>	97.75	84.28	70.53	0.77
<b>not bold and nasal</b>	53.33	98.31	97.29	0.98
<b>bold and not nasal</b>	48.96	96.03	98.50	0.97
<b>bold and nasals</b>	36.73	97.55	98.56	0.98

## 4.9 Conclusion

In this chapter, we have explored the benefits of using HTC and fusion approaches of evidence from multiple classifiers to achieve improved recognition accuracy for Quran speech. At the beginning, we proposed a HTC based on Tajweed rules. The first level in HTC classifies the segment units into two classes: CVs and CVCs. In the second level, we have 4 classes: not bold not nasal, not bold and nasal, bold and not nasal and finally bold and nasal. The total number of nodes in HTC is 24 which considered as super classes. After using a HTC, the accuracy performance enhanced around 34% over when all the segment units are mixed. In the second stage, we used ensemble-based classifiers to enhance the classification accuracy of Quran recitation. We started with three individual classifiers: KNN, MLP, and SVM. A simple SVM was able to provide good performance in terms of recognition accuracy. In ensemble-based classifiers, we used 6 methods: Algebraic combiners (Minimum/ Maximum/ Median/ Product Rule), and



Majority voting. The experiments showed that combining classifiers results improved slightly the accuracy than individual classifiers. More importantly, the experiments showed that the difference in performance between diverse ensemble-based classifiers is not substantial. The work on HTC and ensemble-based classifiers can be extended to other applications such as emotion recognition, visual recognition, language recognition...etc.

# **CHAPTER 5**

## **SPEECH SEGMENT UNIT CLASSIFICATION USING DEEP NEURAL NETWORKS**

### **5.1 Introduction**

In chapter 4, we used a hierarchical tree recognition system as a supervised approach for identifying speech units. The main aim for using HTR was to enhance the recognition performance, especially when we have a large number of classes. In our case, we have considered 700 classes.

In this chapter, we discuss another method for recognizing speech units without the need a tree structure for when dealing with large number of classes. This method is based the recently developed networks called Deep Learning Neural Networks (DNNs). In our work, we will use an unsupervised learning, where the algorithm divides data into clusters, then uses the deep learning neural network in the classification phase.

Deep Neural Networks (DNNs) are believed to achieve high performance on complicated real applications such as vision and speech by better representing their complex functions. Deep learning is a category of machine learning techniques, where hierarchical architectures are used to process natural signals (e.g. speech signals) using several non-linear information stages. Eventually, such signals often contain features that are inherently their characteristics.

The acoustic modelling of speech considered is seen as a key component for most state-of-the-art speech recognition systems. Researchers, in this area, have recently used Deep Neural Networks (DNNs) with high degree of success for acoustic modelling of the English language [94], [96], [98], [111]. This success prompted us to investigate the efficiency of such networks for representing acoustic models for Arabic speech. The aim of this chapter, is therefore, to explore deep learning techniques for segment unit classification and acoustic modelling of Quran recitation. In order to achieve this aim, we have explored different network architectures based on the Deep Neural network introduced in [102].

For a given a dataset, finding the optimum number of classes is a very important issue. One method that can be used for finding different classes is clustering. Clustering is used to find structure in unlabeled data. It is the most common form of unsupervised learning. Clustering methods can discover groups of objects where the average distances between the members of each cluster are smaller than to the members in other clusters.

This chapter is organized as follows: Deep NNs models for Quran acoustic modelling are presented in Section 5.1. In section 5.2, the DNN concepts with their related technical details and the software methodologies are discussed. This is followed by experimental results using our segment units to train Deep Neural Networks (DNNs) with the Back-propagation supervised learning algorithm. A conclusion is finally drawn in Section 5.3.

## 5.2 The system Architecture

In this chapter, our focus is on developing a Quran speech classification system using deep learning neural networks as a classification technique. For our experiments, the data input is our segment units were extracted as discussed in chapter 3 using KACST database.

The proposed system is organized around the following steps (Figure 5.1): Segment units, Feature extraction, Clustering and selecting the optimum number of classes, Deep learning neural network for classification. The first and the second stage were explained before in chapter 4. We will start our discussion with the third stage.

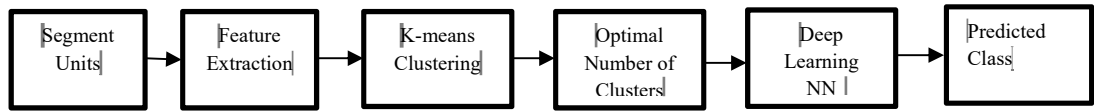


Figure. 5.1: Flow Chart for the Proposed Quran Speech Recognition System

## 5.3 K-means Clustering

K-means is a very popular approach used for clustering. Here, our main is to determine the main spectral groupings present in Quran recitation. The K-means algorithm takes as input parameter,  $k$ , then partitions a set of  $n$  objects into  $k$  clusters. Cluster similarity is measured with regards to the mean value of the objects in a cluster, which can be viewed as the cluster's centroid or center of gravity. Since the K-means

approach is iterative, it is computationally intensive and hence applied only to pattern subareas rather than to full scenes and can be seen as an unsupervised approach. The algorithm is described mathematically as follows:

Given a set of  $N$ -observations  $(X_1, X_2, \dots, X_n)$ , where each observation is a  $d$ -dimensional real vector, the K-Means clustering algorithm aims at partitioning the  $N$ -observations into  $K$  sets  $S$ , where  $(K \leq N)$  and  $S = \{S_1, S_2, \dots, S_k\}$  so as to minimize Within-Cluster Sum of Squares (WCSS), formally written as [164]:

$$\min_K \sum_{i=1}^K \sum_{X_j \in S_i} \|X_j - \mu_i\|^2 \quad (5.1)$$

Where  $\mu_i$  is the centroid of cluster  $i$ . The K-Means clustering algorithm is commonly used in different applications for unsupervised learning to split data into segments or clusters. Such segments or clusters can further be used in classification. The basic steps of K-Means clustering are described below [164]:

1. Starts by a certain number value of  $K$ .
2. Initialize the  $\mu_{is}$  to be the means of the clusters.
3. Assign each example in the data set to the closest group (represented by  $\mu_i$ ).
4. Recalculate  $\mu_i$ , based on the observations that are currently assigned to it.
5. Repeat steps 2-3 until convergence.

## 5.4 Determining the optimal number of clusters

A fundamental problem in cluster analysis is to determine the optimum number of clusters, which is usually taken as known apriori in most clustering algorithms. A clustering technique would most possibly recover the underlying cluster structure given a good estimate of the true number of clusters. The correct choice of the number of clusters is often ambiguous, with interpretations depending upon the shape and scale of the distribution of points in the data and the desired clustering resolution of the user. In addition, increasing number of clusters without penalty will always reduce the amount of the overall error in the resulting clustering, to the extreme case of zero error if each data point is considered its own cluster. Automatically then, the optimal choice number of clusters, should strike a balance between maximum compression of the data using a single cluster, and maximum accuracy by assigning each data point to its own cluster [166].

A number of strategies have been proposed for estimating the optimal number of clusters. Here, we present the Silhouette and Elbow techniques commonly used in determining the optimal number of clusters.

### 5.4.1 The Silhouette Index

For a given cluster,  $C_j$  ( $j = 1, \dots, K$ ), this method assigns to each sample  $X_i$  in  $C_j$  a quality measure,  $s(i)$  ( $i = 1, \dots, m$ ), known as *the silhouette width*. The Silhouette width is a confidence indicator on the membership of the  $i^{\text{th}}$  sample  $X_i$  in cluster  $C_j$ . The

Silhouette width for the  $i^{\text{th}}$  sample  $X_i$  in cluster  $C_j$  is defined as [165] [166].

$$s(i) = \frac{b(i) - a(i)}{\max\{a(i), b(i)\}} \quad , i=1,2,\dots,m \quad (5.2)$$

where  $a(i)$  is the average distance between the  $i^{\text{th}}$  sample  $X_i$  and all of the samples included in  $C_j$ , and  $b(i)$  is the minimum average distance between the  $i^{\text{th}}$  sample  $X_i$ , and all of the samples in the data. From this formula it shows that  $s(i)$  has a value between -1 and 1.

Thus, for a given cluster,  $C_j$ , it is possible to calculate a cluster silhouette,  $S_j$ , which characterizes the heterogeneity and isolation properties of such a cluster. It is calculated as the sum of all silhouette widths in  $C_j$ . The formula of  $S_j$  is determined as:

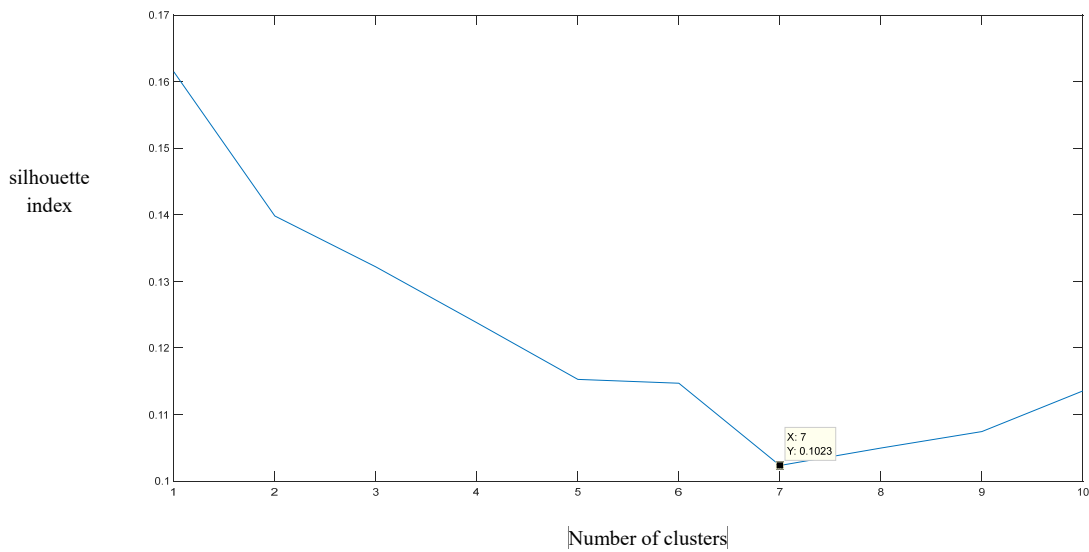
$$S_j = \sum_{X_i \in S_j} s(i) \quad (5.3)$$

Moreover, for any partition, a Global Silhouette ( $GS_U$ ) value or silhouette index, can be used as an effective validity index for a partition  $U$ .

$$GS_u = \frac{1}{c} \sum_{j=1}^c S_j \quad (5.4)$$

where  $S_j$  is silhouette index value for cluster  $C_j$ . In this case, the minimum silhouette

index value is taken as the value corresponding to the optimal partition. In our work, we used the silhouette method to find the optimum number of clusters. The results are shown in Figure 5.2, which clearly show that the optimum number is around 700 clusters for our dataset.



**Figure 5.2: Results for optimal number of clustering using the silhouette method**

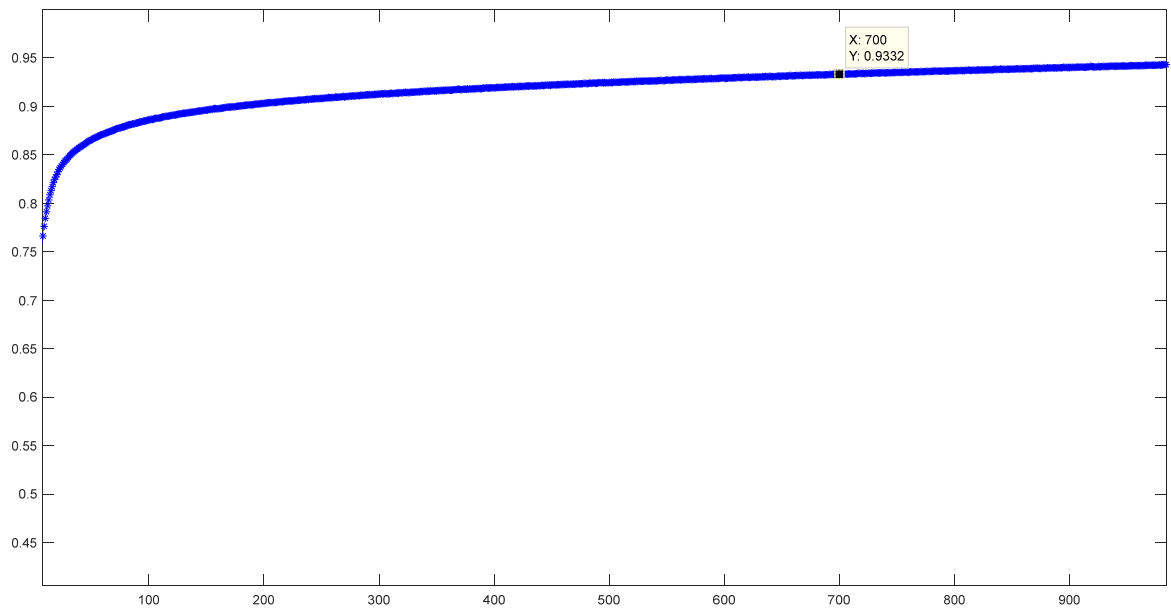
### 5.4.2 The Elbow Method

The elbow method looks at the percentage of variance explained as a function of the number of clusters. One should choose a number of clusters so that adding another cluster doesn't give a much better partition of the data. More precisely, if one plots the percentage of variance explained by the clusters against the varying number of clusters, the first clusters will add much information (explain a lot of variance), but at some point, the marginal gain drops, giving an angle in the graph. The number of clusters is chosen based on the "elbow criterion". The percentage of variance explained is the ratio of the



between-group variance to the total variance. In our work, we have around 4300 segment units as the input data, and we use the elbow criteria to find the optimum number of clusters. The percentage of variance explained for our data is shown in Figure 5.3. The optimum number of clusters based on the silhouette index method was around 700 clusters. This number results in a percentage of variance explained of around 93.3%.

Based on both (silhouette index and Elbow), in our experiments, we will consider the optimum number of clusters is 700 clusters according to our dataset.



**Figure 5.3: The percentage of variance explained for our data**

## **5.5 Deep learning for speech analysis**

Until recently, most machine learning techniques have exploited shallow-structured architectures which contain a single layer of nonlinear feature transformation. Examples

of these architectures are Support Vector Machines (SVMs), Kernel regression, Logistic regression, and Multi-Layer Perceptron (MLP) neural networks with a single hidden layer. The main properties of shallow learning models is that they have a simple architecture and are only effective in solving simple problems. However, these shallow architectures have limited representational and modeling powers which can lead to difficulties when dealing with perception related problems such as human speech.

In recent times, Deep Neural Networks (DNNs) have been shown to achieve high performance on complicated real applications such as vision and speech by using excellent representations of complex functions or tasks. Deep learning is a category of machine learning techniques, where hierarchical architectures are used to process natural signals (e.g. speech signals) using several non-linear information stages. Feed-forward neural networks (FNNs) (e.g., Multi-Layer Perceptron (MLPs)) with many hidden layers are considered a good example of such deep models. Usually MLPs use the Backpropagation (BP) algorithm for learning the network weights. However, such learning algorithms do not work well for learning when several hidden layers are used [90][91]. DNNs are feed-forward Neural Networks that have, on the other hand, many layers of non-linear hidden units between their inputs and their outputs.

There are two key properties of deep learning techniques. A DNN is trained as a generative model, and then an additional top layer is used to perform the discriminative tasks. An unsupervised pre-training phase which train the multilayer generative neural network in one layer at a time, making these effective in extracting structures that represent input features in large unlabeled training data. Most of the work that uses deep learning, can be categorized based on how the architectures are intended to be used:

- 1) Generative deep architecture: this architecture aims at distinguishing the high-order correlation properties of the visible or observed data for pattern synthesis or analysis purposes. In this architecture, DNNs are similar to other dimensionality reduction methods such as Principle Component Analysis (PCA). Generally, the use of generative models plays a significant role in feature coding and recognition applications [97]–[99].
- 2) Discriminative deep architecture: this type of architecture is, often used to characterize the posterior distributions of classes, conditioned upon the observed data, to provide discriminative power for pattern classification.

For large vocabulary speech recognition, deep neural networks must be trained with a large number of parameters. Therefore, overfitting is a potentially serious problem. The term overfitting refers to the gap between training error and test error; i.e. the neural net has learned the training examples very well, but has ‘lost’ its ability to generalize the learning to a new situation. There are several techniques used to prevent overfitting like cross-validation, early stopping, regularization, pruning...etc [167]. These techniques can be used to indicate when more training leads to worse generalizations. More recently, dropout has been discussed as regularization technique for addressing overfitting [100]. This technique randomly drops units from the neural net during training to prevent units from co-adapting too much. The authors in [100], showed that dropout can improve the

performance of the deep neural networks on supervised learning tasks in speech recognition and benchmark datasets. The authors in [101], also used a stacking training method that trains multiple simultaneous predictors to simulate the overfitting problem in early layers of the network.

Deep learning has been successfully exploited for feature learning and pattern classification in diverse applications [90], [102], [103]. A recent work in learning algorithms for deep neural networks has shown excellent performance results in classification tasks [104], [105], in regression [106], in dimensionality reduction [102], [106], in modeling motion [107], in object segmentation [108], in information retrieval [106], in robotics [108], in natural language processing [97], [109], and most notably in the area of Large Vocabulary Automatic Speech Recognition (LVASR) [96], [97].

In our work, we need a DNN classifier task as a process in which, given a certain classes, to pick from a set of features those which better fit each of these classes. When that set consists of too many features the task might not be so easy and it could be better to have a smaller set. Applying this concept to DNN, we want to find a low dimensional representation of a high dimensional data and that is what autoencoders can do. This process of dimensionality reduction is performed by such autoencoder which consists of a multilayer neural network with a small central layer and whose aim is to reconstruct its high-dimensional input vectors. More details about autoencoder will be in the next section.

### 5.5.1 Autoencoder

Autoencoders is an artificial neural network for reconstructing the input signal. An autoencoder is a three layer multi-layer perceptron (MLP) consisting of an input layer, a code layer and a reconstruction layer, as illustrated in Figure 5.4. The goal of an autoencoder is to learn a good code representation that preserves as much information as possible about an input to allow the input to then be reconstructed from the code. An autoencoder is trained on a dataset by minimizing a function of the reconstruction error such as the mean squared error.

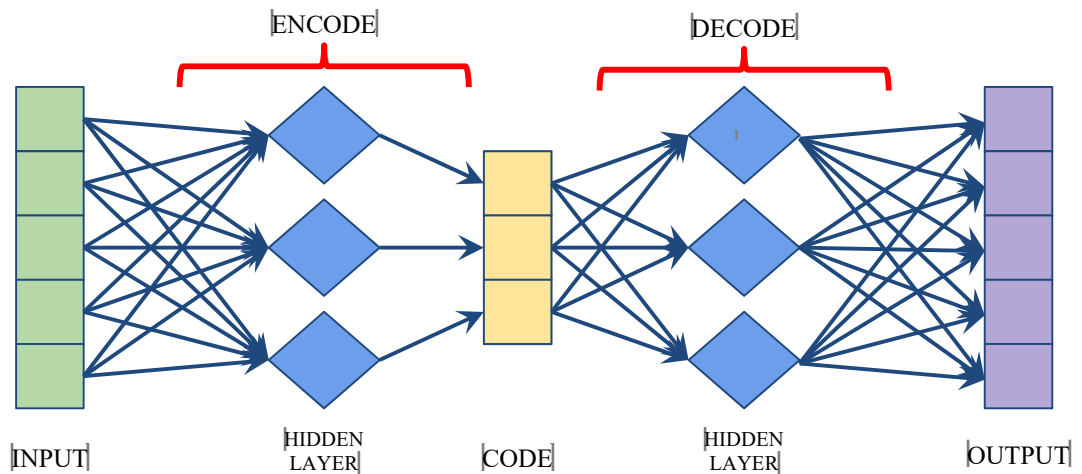


Figure 5.4: Autoencoder structure

If there is one linear hidden layer and the mean squared error criterion is used to train the network, then the  $k$  hidden units learn to project the input in the span of the first  $k$  principal components of the data. If the hidden layer is non-linear, the autoencoder behaves differently from Principal Component Analysis (PCA), with the ability to capture multi-modal aspects of the input distribution. The hope is that the code  $z$  is a

distributed representation that captures the main factors of variation in the data: because  $z$  is viewed as a lossy representation of  $x$ , it cannot be a good representation (with small loss) for all  $x$ . So learning drives it to be one that is a good representation in particular for training examples, and hopefully for others as well (and that is the sense in which an autoencoder generalizes), but not for arbitrary inputs. It can typically be used for dimensionality reduction by learning a compressed ( $m < n$ ) representation of the data.

An autoencoder is special kind of neural network with two components: an encoder and a decoder. The encoder takes the input  $x$ , and maps it to a hidden representation  $y$ , which is given by

$$y = \sigma(Wx + b_h) \quad (5.5)$$

The latent or hidden representation  $y$  is mapped back to the original input, using a decoder which is given as:

$$y = \sigma(Wx + b_h) \quad (5.6)$$

The latent or hidden representation  $y$  is mapped back to the original input, using a decoder which is given as:

$$z = \sigma(W'\hat{y} + b_v) \quad (5.7)$$

where  $y$  is the encoded value,  $\hat{y}$  is a possibly corrupted version of the encoded value,  $b_h$ ,  $b_v$  are the bias values of encoder and decoder respectively,  $W$  is the weight matrix of the encoder, while  $W'$  represents its transpose. This network tries to minimise the reconstruction error given as:

$$L_c(x, z) = \|x - z\|^2 \quad (5.8)$$

$$L_b = -\sum_{k=1}^d [x_k \log z_k + (1 - x_k) \log(1 - z_k)] \quad (5.9)$$

The first equation 5.8 is for continuous input, while the second equation 5.9 is used for classes and binary vectors. It is basically the cross-entropy error already defined above. Although we have expressed the equations with  $\sigma$  function, the activation function could be any other prominent activation functions. If the hidden layer of the auto-encoder has a lower dimensionality than the input, the model will perform non-linear dimensionality reduction. If it is of equal or greater dimensionality, special care must be put to avoid that the model learns a trivial mapping (identity function). For this reason, the input or the hidden representations may be corrupted during training.

### 5.5.1.1 Manifold training with autoencoder

A reason as to why autoencoders do so well is that they exploit the idea that data is generally concentrated around a manifold, or several subsets of manifolds. The general principle behind all autoencoders is a trade-off between two ideas: first, to learn a representation  $y$  of a training example  $x$  such that  $x$  can be approximately recovered from

y through a decoder.  $x$  should be drawn from the training data, because it means that the autoencoder need not successfully reconstruct the inputs that are not probable under the data generating distribution. The other complementary idea is to satisfy the generalization or regularization penalty, the presence of this term will encourage solutions which are less sensitive to small perturbations in the data

Variational autoencoder is trained in a supervised fashion like a neural network using the backpropagation algorithm. The whole network is shown in Figure 5.5.

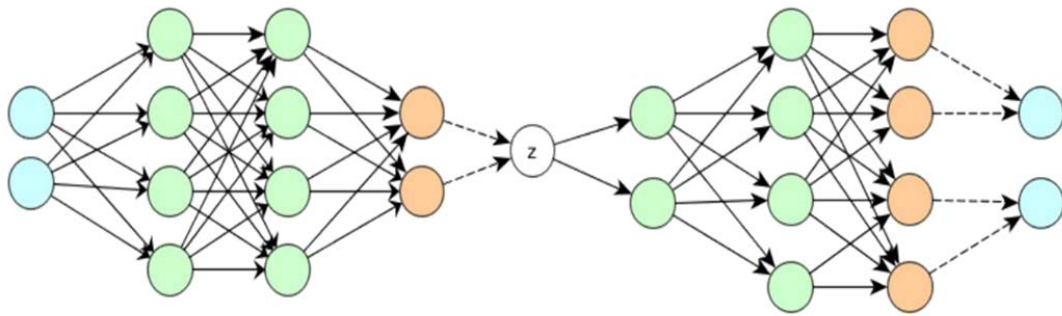


Figure 5.5: Variational autoencoder [122]

The idea of the network is to maximize the probability that the sample shown to the network will be generated as output. This approach is well known in statistics and is called MLE (maximum likelihood estimation). By using MLE, error function of the network is found. Error is also referred to as cost, and accordingly, error function is the cost function.



First, log-likelihood of the random variable which represents the data sample  $\mathbf{x}$  (from the training set) being the output of the network is specified in Equation 2.42 (logarithm is applied because it makes the calculation easier). The goal of training the neural network is to maximize that likelihood.

$$L = \log(p(\mathbf{x})) \quad (5.10)$$

Distribution which approximates  $p_{\phi}(\mathbf{z}|\mathbf{x})$  is denoted as  $q(\mathbf{z}|\mathbf{x})$ . Parameters of distribution  $q$  are now considered independent from the parameters of distribution  $p_{\phi}(\mathbf{x}|\mathbf{z})$ . Equation 5.10, can be multiplied by the integral over the entire space of  $q(\mathbf{z}|\mathbf{x})$ , therefore:

$$= \int_{\mathbf{z}} q(\mathbf{z}|\mathbf{x}) \log(p(\mathbf{x})) \quad (5.11)$$

Then, a series of simple transformations are done:

$$= \int_{\mathbf{z}} q(\mathbf{z}|\mathbf{x}) \log\left(\frac{p(\mathbf{z}, \mathbf{x})}{p(\mathbf{z}|\mathbf{x})}\right) \quad (5.12)$$

$$= \int_{\mathbf{z}} q(\mathbf{z}|\mathbf{x}) \log\left(\frac{p(\mathbf{z}, \mathbf{x})}{p(\mathbf{z}|\mathbf{x})}\right) \quad (5.13)$$

$$= \int_z q(z/x) \log\left(\frac{p(z,x)}{p(z/x)}\right) \quad (5.14)$$

$$= \int_z q(z/x) \log\left(\frac{p(z,x)q(z/x)}{p(z/x)q(z/x)}\right) \quad (5.15)$$

$$= \int_z q(z/x) \log\left(\frac{p(z,x)}{q(z/x)}\right) + \int_z q(z/x) \log\left(\frac{q(z/x)}{p(z/x)}\right) \quad (5.16)$$

$$= L^V + D_{KL}(q(z/x)||p(z/x)) \quad (5.17)$$

The first term  $L^V$  is called lower variation bound of the likelihood. The second term  $D_{KL}$  is the Kullback-Leibler divergence which measures the similarity of behavior between two distributions. Here it measures how well does  $q(z|x)$  approximate  $p(z|x)$ . To maximize the likelihood, it is necessary to maximize the lower variation bound. Therefore, lower variation bound is further analyzed.

$$L^V = \int_z q(z/x) \log\left(\frac{p(z,x)}{q(z/x)}\right) \quad (5.18)$$

$$L^V = \int_z q(z/x) \log\left(\frac{p(x/z)p(z)}{q(z/x)}\right) \quad (5.19)$$

$$L^V = \int_z q(z/x) \log\left(\frac{p(z)}{q(z/x)}\right) + \int_z q(z/x) \log(p(x/z)) \quad (5.20)$$

$$L^V = -D_{KL}(q(z/x)|p(z)) + E_{q(z/x)}(\log(p(x/z))) \quad (5.21)$$

The left term in Equation 5.21 is the Kullback-Leibler divergence which measures the similarity between  $q(z|x)$  and  $p(z)$ . Distribution  $p(z)$  can be freely chosen but is usually a normal distribution with zero mean and unit variance. This context this term acts as a regularization term. Second term is the reconstruction quality of the Autoencoder. It measures how well the approximation of  $p(x|z)$  produces data sample from the given latent state. Maximization of both of those terms is the goal of training the network.

First term  $DKL(q(z|x)|p(z))$  can be calculated using Equation 5.21, since  $q(z|x)$  is a normal distribution with parameters  $\mu_z, \sigma_z$  (vectors) produced by the encoder [122]. Variable  $J$  is the size of the latent space, thus the dimension of mentioned vectors.

$$-D_{KL}(q(z/x)|p(z)) = 0.5 \sum_{j=1}^J (1 + \log(\sigma_{zj})^2 - \mu_{zj}^2 - \sigma_{zj}^2) \quad (5.22)$$

In order to calculate the second term sampling of latent variable is needed. After sampling the latent variable many times, the average of log probability over all the samples should be calculated to estimate the expectation. But usually, only a single sample is sufficient for training to work well. Therefore, only the log probability remains to be calculated using the sample taken from the latent space. The calculation can be done in the following way by using the expression for the probability distribution of a normal distribution  $p(\mathbf{x}|\mathbf{z})$  [122]:

$$\log(p(\mathbf{x}/\mathbf{z})) = -\sum_{j=1}^D(0.5 \log(\sigma_{zj})^2 + \frac{(x_j - \mu_{xj}^2)^2}{2\sigma_{xj}^2}) \quad (5.23)$$

Parameters  $\boldsymbol{\mu}_x$ ,  $\boldsymbol{\sigma}_x$  (vectors) are produced by the decoder and  $D$  is the dimension of the data.

Both expressions on the right hand side in Equations 5.22 and 5.23 need to be maximized in order maximize the likelihood of the data sample. Thus, the cost function is the sum of those two expressions multiplied by minus one:

$$-0.5 \sum_{j=1}^J(1 + \log(\sigma_{zj})^2 - \mu_{zj}^2 - \sigma_{zj}^2) + \sum_{j=1}^D(0.5 \log(\sigma_{xj})^2 + \frac{(x_j - \mu_{xj}^2)^2}{2\sigma_{xj}^2}) \quad (5.24)$$

Equation 5.18 is a cost function for a single data sample. Usually, for a single step of training, cost function is calculated for a batch of instances from the available training set. Total cost is then calculated as the average cost among the batch.

In our work, we used softmax method as activation function for classification the output. The general concept of softmax activation function is in the following subsection.

### 5.5.2 Softmax activation function

The softmax function is used as part of a machine learning network, as a basis function to classify the outputs. The main idea of the softmax is computing its derivative using the multivariate chain rule. Where cross-entropy is commonly used to take a look at a loss function for training a network.

Cross-entropy has an interesting probabilistic and information-theoretic interpretation. For two discrete probability distributions  $p$  and  $q$ , the cross-entropy function is defined as:

$$xent(p, q) = -\sum_k p(k)\log(q(k)) \quad (5.25)$$

Where  $k$  is over all the possible values of the random variable in the distributions. In our case, there are  $T$  output classes, so  $k$  would go from 1 to  $T$ .

If we start from the softmax output  $P$ , where  $P$  is one probability distribution. The other probability distribution is the "correct" classification output, usually denoted by  $Y$ . This is a one-hot encoded vector of size  $T$ , where all elements except one are 0.0, and one element is 1.0. which 1 means the correct class for the data being classified. The cross-entropy loss formula can be written as:

$$xent(Y, P) = -\sum_{k=1}^T Y(k) \log(P(k)) \quad (5.26)$$

Where  $k$  is over all the output classes.  $P(k)$  is the probability of the class as predicted by the model.  $Y(k)$  is the "true" probability of the class as provided by the data. The sole index will be  $y$  if  $Y(k) = 1.0$ . Since for all  $k \neq y$  we have  $Y(k) = 0$ , the cross-entropy formula can be simplified to:

$$xent(Y, P) = -\log(P(y)) \quad (5.27)$$

Moreover, since in our case  $P$  is a vector, we can express  $P(y)$  as the  $y$ -th element of  $P$  as:

$$xent(P) = -\log(P_y) \quad (5.28)$$

The Jacobian of  $xent$  is a  $1 \times T$  matrix (a row vector), since the output is a scalar and we have  $T$  inputs (the vector  $P$  has  $T$  elements):

$$Dxent = [D_1xent \ D_2xent \ \dots \ D_Txent] \quad (5.29)$$

Now recall that  $P$  can be expressed as a function of input weights:

$$P(W) = S(g(W)) \quad (5.30)$$

So we have another function composition:

$$xent(W) = (xent \circ P)(W) \quad (5.31)$$

$$= xent(P(W)) \quad (5.32)$$

We can, once again, use the multivariate chain rule to find the gradient of  $xent$  w.r.t.  $W$ :

$$Dxent(W) = D(xent \circ P)(W)$$

$$= Dxent(P(W)).DP(W) \quad (5.33)$$

The dimension of the Jacobian matrix  $DP(W)$  is  $T \times NT$ , and The dimension of the Jacobian matrix  $DxentP(W)$  is  $1 \times T$ . So the resulting Jacobian  $Dxent(W)$  is  $1 \times NT$ , where the whole network has one output (the cross-entropy loss - a scalar value) and  $NT$  inputs (the weights).

Here again, there is a straightforward way to find a simple formula for  $Dxent(W)$ , since many elements in the matrix multiplication will be zero. Note that  $xent(P)$  depends only on the  $y$ -th element of  $P$ . Therefore, only  $D_y xent$  is non-zero in the Jacobian:

$$Dxent = [0 \ 0 \ D_y xent \ \dots \ 0] \quad (5.34)$$

And  $D_y xent = -\frac{1}{P_y}$ . Then, we multiply  $Dxent(P)$  by each column of  $D(P(W))$  to get each element in the resulting row-vector. The row vector represents the linearized weight matrix  $W$ . We will index into it with  $i$  and  $j$  for simplicity ( $D_{ij}$  points to element number  $iN + j$  in the row vector):

$$D_{ij}xent(W) = \sum_{k=1}^T D_k xent(P).D_{ij}P_k(W) \quad (5.35)$$



Since only the  $y$ -th element in  $D_k xent(P)$  is non-zero, we get the following, also substituting the derivative of the softmax layer as the following:

$$D_{ij}xent(W) = D_yxent(P).D_{ij}P_k(W) \quad (5.36)$$

$$= -\frac{1}{P_y}.S_y(\delta_{yi} - S_i)x_j \quad (5.37)$$

Let's,  $P_y = S_y$ , so we get:

$$D_{ij}xent(W) = -\frac{1}{S_y}.S_y(\delta_{yi} - S_i)x_j \quad (5.38)$$

$$= -(\delta_{yi} - S_i)x_j \quad (5.39)$$

$$= (S_i - \delta_{yi})x_j \quad (5.40)$$

The formula for  $D_{ij}xent(W)$  could end up being a fairly involved sum (or sum of sums). The technique of multiplying Jacobian matrices is ignorant to all this. For multivariate chain rule, we compute the individual Jacobians which is usually easier

because non-composed functions.

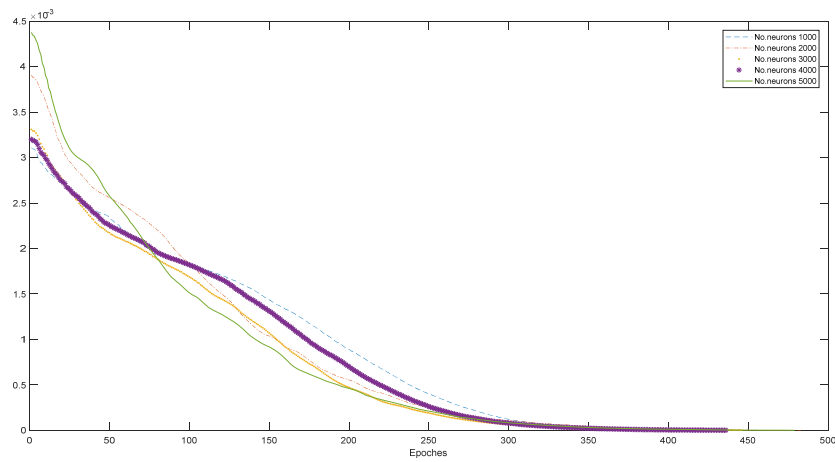
## 5.6 Experimental Results

As explained earlier, our database consists of 4300 observations (or speech segment units). Each of these observations is represented with 281 dimensional feature vectors. Using the silhouette method as a criterion, we clustered the data into 700 clusters using the k-means algorithm. The resulting cluster indices are used as class labels. In other words, we assume here that we have 700 classes. Given the large number of classes, we opted to use the Autoencoder DNN for classification.

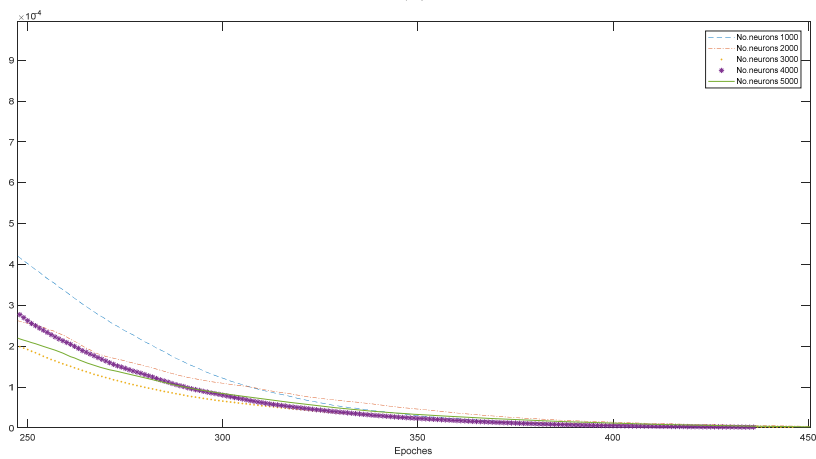
For performance analysis, we used 80% of the data for training and the remaining 20% for testing. The stochastic gradient decent algorithm was used for the pre-training phase with a mini-batch size of 100 training cases using a learning rate of 0.1, a weight cost of 0.0002, and a momentum of 0.9. In the fine-tuning stage, we used 4000 epochs for the first hidden layer, and 2000 epochs for the second hidden layer. We ran the experiments using the Matlab code as used in [168] to implement the developed systems.

As with common applications of neural networks, there are no specific rules on how to select the number of hidden layers and the number of nodes at each layer. Thus, as suggested in [169], to optimize the number of nodes in the hidden layers, the number of hidden layers should be changed once a time. For simplicity, the network used was optimized by fixing the number of hidden layers and changing the number of neurons. Figure. 5.6 shows the effect of choosing the number of neurons on the recognition

performance while fixing the number of hidden layers. We notice in Figure 5.5 that increasing the number of neurons to 4000 in the first hidden layer gives a slightly improved performance. However, 5000 nodes in the first hidden layer was not better than 4000 nodes, meaning that 4000 nodes are enough to achieve good recognition performance.



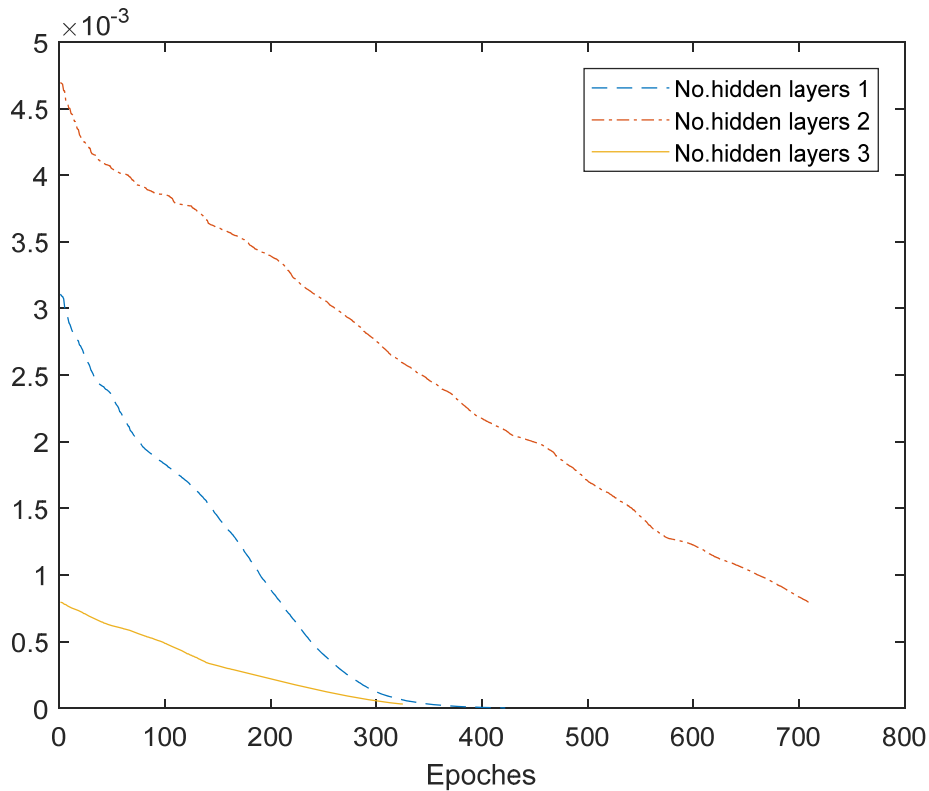
(a)



(b)

**Figure 5.6: Convergence of NN training (a), Zoomed region (b)**

As a second optimization stage, we also studied the effects of the number of hidden layers on the overall performance. Here, the number of neurons in the first stage was 4000 neurons, and 2000 neurons for the second hidden layer and 1000 neurons for the third hidden layer[168]. The number of neurons in each hidden layer were classes based on the number of observations and the number of outputs. We performant three experiments: One hidden layer with a number of neurons of 4000, two hidden layers with a number of neurons of 4000, and 1000 respectively, and three hidden layers with the number of neurons of 4000, 2000, and 1000 respectively. The results of these experiments are shown in Figure 5.7.



**Figure 5.7: Convergence of the NN with different number of hidden layers**

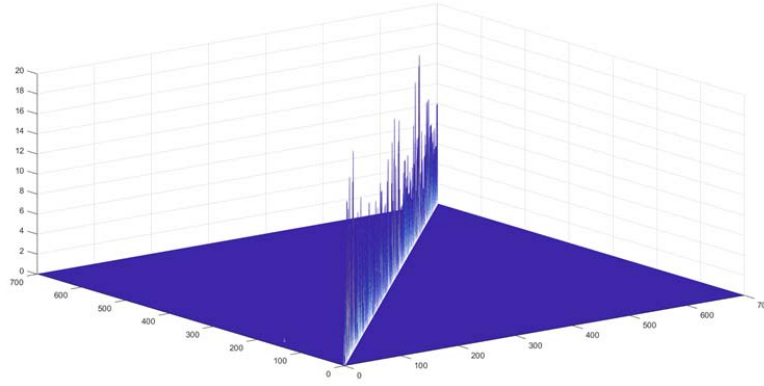
We notice in Figure 5.7 that, when we increase the number of hidden layers, the convergence is faster. These results were obtained confirming previous work discussed in [98]. Therefore, the structure of the hidden nodes for our experiments was chosen based on these optimized experiments to be three hidden layers with 4000, 2000, and 1000 neurons respectively.

The results using the above architectures are summarized in Table 5.1, showing the performance of the seven different models that were considered to evaluate the optimum number of neurons and number of hidden layers for our application. It is clear from Table 5.1 that the three hidden layers NN, provides the best classification performance across all structures. Increasing the number of neurons in the hidden layer up to 4000 hidden nodes improved the classification performance of the network. The significance tests also proved that the three hidden layers-4000-2000-1000 structure gives the best DNN architecture.

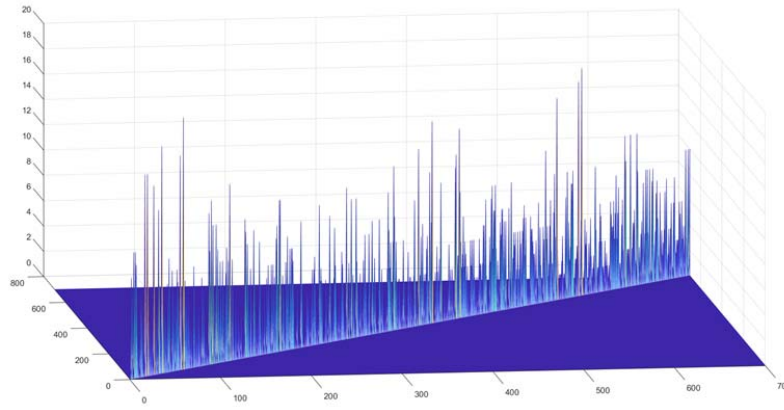
**Table 5.1 - Performance with different setups of the DNN**

Number of hidden layers	Number of neurons	The performance when the number of iteration is 400
1	1000	98.7 %
1	2000	99.1 %
1	3000	99.5 %
1	4000	99.7 %
1	5000	99.6 %
2	4000-2000	99.75 %
<b>3</b>	<b>4000-2000-1000</b>	<b>99.9 %</b>

In the classification stage, it is desirable to present the confusion matrix but this is difficult when the number of classes is very large. Here, we display the confusion matrix as a mesh plot. The mesh function in Matlab represents 3D data as a wireframe with color determined by the Z axis. The confusion matrix is displayed in Figure 5.8. It is clear that confusion matrix is very close to a diagonal one as the accuracy is close to 100%.



(a)



(b)

**Figure 5.8: a) Mesh plot of the confusion matrix b) Zoom the mesh plot**

As the KACST corpus had been labelled to test our work, we could not evaluate the developed system compared to other systems reported in the literature. However, the results were obtained from Chapter 4, can be compared directly with the DNN model results. Table 5.2 shows the comparison between the ensemble-based classifier and DNN classifiers.

**Table 5.2 - A comparison results between MLP and DNN**

	<b>Ensemble-based Classifier with HTR</b>	<b>DNN</b>
<b>Accuracy</b>	85.58%	100%

As can be seen for Table 5.2, the performance of the DNN is about 15% better than the ensemble-based classifier with HTR. This advantage would be more significant when training more complex models with larger number of classes [170]. However, when the feature set is less complex, the results from the DNN are the same as the individual classifiers, and no significant improvement can be noticed [171], [172]. On the other hand, when the feature set becomes more complex, DNNs achieve the best results with deeper layers and larger hidden units. This implies that DNN is better suited for more complex features where the inclusion of additional parameters and layers can better capture the nature of the data [172].

## **5.7 Conclusion**

The objective of this chapter was to explore the potential advantages of developing an Autoencoder DNN-based system for Quran speech recognition. In order to achieve this aim, we used the silhouette method as a technique to find the optimum number of clusters and used the k-means algorithm to assign each observation to one of the resulting clusters. Several Autoencoder DNN-based Quran speech recognition



systems were developed and evaluated using our dataset. It was shown that, the performance of the Autoencoder DNN with three hidden layers can be almost perfect. These results showed that, when the feature set becomes more complex, the DNNs achieve excellent results with deeper layers and larger hidden units. Also, the results showed that the DNNs are better suited for more complex features where the inclusion of additional parameters and layers can better capture the nature of the data and differences between classes.

## CHAPTER 6

### CONCLUSION AND FUTURE WORK

#### 6.1 Conclusion

In this dissertation, we introduced new models for speech segmentation and recognition in relation to Quran recitation. The road map of our work consists of two main branches: the first branch is finding the optimal set of segment units based on our proposed segmentation technique, and the second branch is to use our segmentation results from the first branch to find the optimal number of classes and identify such classes from test data. During the course of the work, we achieved the following results:

- We developed a Quran database that is manually timed and syllabically labelled. This dataset was constructed based on the KACST database for 10 professional reciters.
- We used our proposed automatic segmentation techniques for finding the segment units. In our initial setup, we used three individual segmentation engines (Energy, ZCR, and Entropy).

- We then proposed a new hybrid speech segmentation algorithm using a GA optimization scheme over multiple features. The algorithm uses numerous independent individual segmentation methods, to produce multiple predictions of boundary positions. To optimize the overall system, we considered two main parameters: frame size, and minimum number of frames / segment unit. The results show an improvement accuracy of 16% over the best performing single feature algorithms, and 10% improvement over the traditional linear regression based approach.
- We then explored the concept of Hierarchical Tree Recognition (HTR) and that of evidence fusion from multiple classifiers to achieve improved recognition accuracy for Quran speech recitation. Our work showed that, when we used HTR, the accuracy performance improved more than 30%. The experiments also showed that combining classifiers results in improved recognition accuracy.
- Finally, we explored the advantages of using an Autoencoder DNN-based system for Quran speech recognition. The driving focus behind this was the large number of classes considered in our approach. We used the silhouette method as a technique to find the optimum number of clusters and k-means algorithm to assign each observation to one of the clusters. Several variants of Autoencoder DNN-based classifier for speech recognition were developed and evaluated across our dataset. We show that, an almost perfect recognition can be achieved using the different variants of the Autoencoder DNN.

## 6.2 Future work

There are several research directions that can be followed to further improve the proposed systems for segmentation and recognition of Quran recitation. Some of these are outlined below:

- **Database Development and Labeling.** A fundamental key point in this research is using a suitable database for the proposed tasks. Our dataset is just for the last part of the Quran. This database needs to be extended to include all chapters from the Holy Quran and cover more expert recitations. Moreover, we need to incorporate Quran experts/reciters from other regions and language backgrounds.
- **Feature extraction selection.** More efforts need to be put in finding robust features for Quran speech systems. Other feature types could be investigated for speech recognition task to improve the speech recognition performance. In addition, we can use deep learning neural network features such as bottle-neck features and Tandem features can be considered with DNNs.
- **Hybrid systems.** One of the suggestions in the future is to develop hybrid schemes combining segmentation techniques with and without linguistic reference. We believe that there is scope in the future work to add linguistic knowledge into the segmentation models, such as language modelling scores and even syntactic bracketing information. This would require running segmentation as an iterative

procedure, on the output of an ASR model, before feeding it back in as the input to an ASR system

- **Several languages and recitations (Qiraat):** Beyond the general characterization of the segmentation approaches, on recent direction can be to analyze techniques need to be modified and adapted to a particular language or reciter, and to a particular task (our work is recitation of Quran in Hafs). So, future work should attempt to adopt the developed techniques to other languages and recitations.
- **Audio Visual processing:** In future work, the algorithms can be developed as a preprocessing step in automatic visual speech recognition, video conferencing, human-computer interaction systems (for identifying human activities involving speech), and speaker tracking. Also such algorithms can be extended to video content analysis, audio retrieval, and indexing.
- **Practical tests with noise:** In our system, the database was recorded in a special room without noise. We expect that in practical setups noise can be affect, and substantially segmentation accuracy. In the future, we intend to adopt our techniques for adverse speech conditions where different types of noises may be present.
- **Children learning:** Our system also can be adopted to help children with some types of speaking disabilities. The system can be used to check where the mistake

in pronunciation are, and can even help such children in correcting these mistakes in pronunciation.

- **Optimization algorithms.** In our work, we used an empirical method for finding the number of neurons and also number of hidden layers. Various optimization approaches can be used to find the optimal number of hidden layers and the optimal number of neurons for each layer.

## References

- [1] X. He and L. Deng, "Discriminative learning for speech recognition: Theory and practice," vol. 4. 2008.
- [2] O. Räsänen, "Speech Segmentation and Clustering Methods for a New Speech Recognition Architecture," *Acousticshutfi*, 2007.
- [3] J. Van Hemert, "Automatic segmentation of speech," *Signal Process. IEEE Trans.*, vol. 39, no. 4, pp. 1008–1012, 1991.
- [4] M. Liu and T. S. Huang, "A Bayesian Predictive Method for Automatic Speech Segmentation," *Pattern Recognit.*, pp. 18–21, 2006.
- [5] F. B. Sofya and M. Al-obadi, "Syllabic Segmentation Algorithm of Arabic Word," *Iraq J. Stat. Sci.*, no. 14, pp. 21–31, 2008.
- [6] M. Abdulatif Ibrahim, "The Shadow Area: A Contrastive Review of the Syllabic Template and Syllabification in English and Fawi Arabic," *Int. J. Linguist.*, vol. 5, no. 4, p. 9, 2013.
- [7] J. Tebelskis, R. Lippmann, and M. I. T. L. Labs, "Speech Recognition using Neural Networks," *Contract*, vol. 9, no. May, pp. 290–296, 1995.
- [8] G. Yukl, "An evaluation of conceptual weaknesses in transformational and charismatic leadership theories," *Leadersh. Q.*, vol. 10, no. 2, pp. 285–305, 1999.
- [9] A. Raake, "Speech Quality of VoIP: Assessment and Prediction," *Speech Qual. VoIP Assess. Predict.*, pp. 1–309, 2006.
- [10] S. Buk, J. Macutek, and A. Rovenchak, "Some properties of the Ukrainian writing system," *CoRR*, vol. abs/0802.4, 2008.
- [11] M. A. For and Q. Rule, "MFCC-VQ Approach For QalqalahTajweed Rule Checking . pp 275 - 293," vol. 27, no. 4, pp. 275–293, 2014.
- [12] N. Hmad and T. Allen, "Biologically inspired Continuous Arabic Speech Recognition," *Res. Dev. Intell. Syst. XX,IX*, pp. 245–258, 2012.
- [13] W. J. Chen, R. Padmanabha, and C. Glover, "Isolation, sequencing and disruption of the CKAI gene encoding the alpha subunit of yeast casein kinase II," *Mol Cell Bio*, vol. 8, no. 11, pp. 4981–4990, 1988.
- [14] K. Waheed, K. Weaver, and F. M. Salam, "A robust algorithm for detecting speech

- segments using an entropic contrast,” *2002 45th Midwest Symp. Circuits Syst. 2002 MWSCAS2002*, vol. 3, p. III-328-III-331, 2002.
- [15] P. Cox, Stephen; Brady, Richard ;Jackson, “TECHNIQUES FOR ACCURATE AUTOMATIC ANNOTATION OF SPEECH WAVEFORMS,” 1998, pp. 5–8.
  - [16] P. Cossi, J.-P. Hosom, and F. Tesser, “High performance Italian continuous ‘digit’ recognition,” *Proc. ICSLP-2000, Int. Conf. Spok. Lang. Process.*, pp. 242–245, 2000.
  - [17] D. T. Toledano, “Neural network boundary refining for automatic speech segmentation,” *ICASSP ’00 Proc. Acoust. Speech, Signal Process. 2000. IEEE Int. Conf.*, pp. 3438–3441, 2000.
  - [18] J. Yuan, N. Ryant, M. Liberman, A. Stolcke, V. Mitra, and W. Wang, “Automatic phonetic segmentation using boundary models,” *Proc. Annu. Conf. Int. Speech Commun. Assoc. INTERSPEECH*, pp. 2306–2310, 2013.
  - [19] P. Mermelstein, “Automatic segmentation of speech into syllable units,” *Acoust. Soc. Am.*, 1975.
  - [20] M. G. Beghi, “Modeling and Measurement Methods for Acoustic Waves and for Acoustic Microdevices,” *InTech*, 2013, pp. 413–472.
  - [21] L. Rabiner and B.-H. Juang, “Fundamentals of Speech Recognition,” *Prentice Hall*, vol. 103. p. 507, 1993.
  - [22] L. R. Rabiner and M. R. Sambur, “Some preliminary experiments in the recognition of connected digits,” *IEEE Trans. Acoust.*, vol. ASSP-24, no. 2, pp. 170–182, 1976.
  - [23] T. Jehan, *Musical Signal Parameter Estimation*. CNMAT 1750, Arch Street BERKELEY, CA 94709 USA, 1997.
  - [24] J. C. Ingram, *Neurolinguistics: An introduction to spoken language processing and its disorders*. Cambridge Textbooks in Linguistics. New York: Cambridge University Press, 2007.
  - [25] C. Science, S. By, H. Kaur, S. By, and C. Applications, “Acoustic Features Based Automatic Segmentation of Syllables,” no. 601103006, 2013.
  - [26] Y. A. Alotaibi and A. Hussain, “Comparative Analysis of Arabic Vowels using Formants and an Automatic Speech Recognition System,” *Pattern Recognit.*, vol. 3, no. 2, pp. 11–22, 2010.



- [27] K. Kvale, "segmentation and labelling of speech," the norwegian institute of technology, 1993.
- [28] I. Mporas, T. Ganchev, and N. Fakotakis, "A hybrid architecture for automatic segmentation of speech waveforms," *2008 IEEE Int. Conf. Acoust. Speech Signal Process.*, no. MAY 2008, pp. 4457–4460, 2008.
- [29] B. N. L. Li and J. N. K. Liu, "A comparative study of speech segmentation and feature extraction on the recognition of different dialects," *IEEE SMC'99 Conf. Proceedings. 1999 IEEE Int. Conf. Syst. Man, Cybern. (Cat. No.99CH37028)*, no. 852, pp. 538–542, 1999.
- [30] S. Greenberg, "Strategies for Automatic Multi-Tier Annotation of Spoken Language Corpora," *SALTMIL*, pp. 45–48, 2003.
- [31] I. Mporas, T. Ganchev, and N. Fakotakis, "Speech segmentation using regression fusion of boundary predictions," *Comput. Speech Lang.*, vol. 24, no. 2, pp. 273–288, 2010.
- [32] P. W. Jusczyk, "How infants begin to extract words from speech," *Trends Cogn. Sci.*, vol. 3, no. 9, pp. 323–328, 1999.
- [33] V. K. Kooijman, *Continuous-Speech Segmentation at the Beginning of Language Acquisition: Electrophysiological Evidence*. 2007.
- [34] M. R. Brent, "Speech segmentation and word discovery: A computational perspective," *Trends Cogn. Sci.*, vol. 3, no. 8, pp. 294–301, 1999.
- [35] V. K. Prasad, T. Nagarajan, and H. a Murthy, "Automatic segmentation of continuous speech using minimum phase group delay functions," *Speech Commun.*, vol. 42, pp. 429–446, 2004.
- [36] T. V. Sarkar, Anindya ;Sreenivas, "AUTOMATIC SPEECH SEGMENTATION USING AVERAGE LEVEL CROSSING RATE INFORMATION Department of Electrical Communication Engineering , Indian Institute of Science , Bangalore - 560 012 , India .," *Electr. Commun.*, pp. 397–400, 2005.
- [37] A. Esposito and G. Aversano, "Text Independent Methods for Speech Segmentation," *Nonlinear Speech Model.*, pp. 261–290, 2005.
- [38] S. Jarifi, D. Pastor, and O. Rosec, "Brandt ' S Glr Method & Refined Hmm Segmentation for Tts Synthesis Application," *Eur. Signal Process. Conf.*, no. September, pp. 3–6, 2005.
- [39] a Sethy and S. S. Narayanan, "Refined speech segmentation for concatenative

- speech synthesis,” *Seventh Int. Conf. Spok. Lang. Process.*, vol. 2002, pp. 149–152, 2002.
- [40] K. Demuynck and T. Laureys, “A comparison of different approaches to automatic speech segmentation,” *Text, Speech and Dialogue*, pp. 385–406, 2006.
  - [41] Y. Runqiang, Z. Yiqing, and Z. Yisheng, “Automatic speech segmentation combining an HMM-based approach and recurrence trend analysis,” *Proc. {IEEE} Conf. Acoust. Speech Signal Process. -- Proc. 1*, no. 1, pp. I797–I800, 2006.
  - [42] I. Shafran and R. Rose, “Robust speech detection and segmentation for real-time ASR applications,” *2003 IEEE Int. Conf. Acoust. Speech, Signal Process. 2003. Proceedings. (ICASSP '03).*, vol. 1, pp. 432–435, 2003.
  - [43] M. Basseville and I. V Nikiforov, *Detection of Abrupt Changes : Mich `ele Basseville*, vol. 2, no. 4. Prentice-Hall, Inc., 1993.
  - [44] T. Nagarajan, H. a. Murthy, and R. M. Hegde, “Segmentation of speech into syllable-like units,” *8th Eur. Conf. Speech Commun. Technol. (EUROSPEECH '03)*, pp. 2893–2896, 2003.
  - [45] F. Pwint, Moe;Sattar, “A SEGMENTATIONMETHOD FOR NOISY SPEECH USING GENETIC ALGORITHM,” *Acoust. Speech, Signal Process. 2005. Proceedings. (ICASSP '05). IEEE Int. Conf. (Volume5 )*, pp. 521–524, 2005.
  - [46] S. A. Samad *et al.*, “Automatic Segmentation and Labeling for Malay Speech Recognition,” *Proc. 6th WSEAS Int. Conf. Signal Process. Comput. Geom. Artif. Vision, Elounda, Greece, August 21-23, 2006*, vol. 2006, pp. 217–221, 2006.
  - [47] R. Andre-Obrecht, “A new statistical approach for the automatic segmentation of continuous speech signals,” *Acoust. Speech Signal Process. IEEE Trans.*, vol. 36, no. 1, pp. 29–40, 1988.
  - [48] B. Sudhakar and R. B. Raj, “Automatic Speech Segmentation to Improve Speech Synthesis Performance,” *2013 Int. Conf. Circuits, Power Comput. Technol. [ICCPCT-2013]*, pp. 835–839, 2013.
  - [49] H. Jeong, C.G.;Jeong, “Neural Network Architecture for Speech Segmentation Using Mean Field Annealing,” *Neural Networks, 1994. IEEE World Congr. Comput. Intell. 1994 IEEE Int. Conf. (Volume7 )*, pp. 4442–4447, 1994.
  - [50] M. F. Tolba, T. Nazmy, A. A. Abdelhamid, and M. E. Gadallah, “A NOVEL METHOD FOR ARABIC CONSONANT / VOWEL SEGMENTATION USING WAVELET TRANSFORM,” *IJICIS, Vol. 5, No. 1*, vol. 5, no. 1, pp. 353–364, 2005.

- [51] M. J. Anwar, M. M. Awais, S. Masud, and S. Shamail, "Automatic Arabic Speech Segmentation System," *Int. J. Inf. Technol.*, pp. 102–111, 2006.
- [52] M. M. Awais, S. Masud, and S. Shamail, "CONTINUOUS ARABIC SPEECH SEGMENTATION USING FFT SPECTROGRAM," *IEE Conf.*, 2006.
- [53] H. R. Iqbal, M. M. Awais, S. Masud, and S. Shamail, "On vowels segmentation and identification using formant transitions in continuous recitation of quranic arabic," *Stud. Comput. Intell.*, vol. 134, pp. 155–162, 2008.
- [54] A. M. Abdullah and E. J. Harfash, "The Segmentation of Arabic word signal based on Eigen values and Eigen vectors principles," *J. Basrah Res.*, 2010.
- [55] N. Lachachi, "Unsupervised Phoneme Segmentation Based on Main Energy Change for Arabic Speech," *J. Telecommun. Inf. Technol.*, 2017.
- [56] H. Frigui and R. Krishnapuram, "A robust competitive clustering algorithm with applications in computer vision," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 21, no. 5, pp. 450–465, 1999.
- [57] S. K. Bhatia and J. S. Deogun, "Conceptual Clustering in Information Retrieval," *IEEE Trans. Syst. MAN, Cybern. B Cybern.*, vol. 28, no. 3, pp. 427–435, 1998.
- [58] C. M. Bishop, *Pattern Recognition and Machine Learning*, vol. 53, no. 9. 2013.
- [59] A. K. Jain, R. P. W. Duin, and J. Mao, "Statistical pattern recognition: a review," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 1, pp. 4–37, 2000.
- [60] Š. Raudys, "Evolution and generalization of a single neurone: I. Single-layer perceptron as seven statistical classifiers," *Neural Networks*, vol. 11, no. 2, pp. 283–296, 1998.
- [61] W.-Y. Loh, "Classification and regression trees," *Wiley Interdiscip. Rev. Data Min. Knowl. Discov.*, vol. 1, no. 1, pp. 14–23, 2011.
- [62] "PAMI 1991-13-4-6 Optimal partitioning for classification and regression trees.pdf." .
- [63] S. B. G. J. Delp, "An iterative growing and pruning algorithm for classification tree design," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 13, 1991.
- [64] M. J. Anwar, M. M. Awais, S. Masud, and S. Shamail, "Automatic Arabic Speech Segmentation System," *Int. J. Inf. Technol. Vol. 12 No.6 2006*, vol. 12, no. March 2014, pp. 102–111, 2006.
- [65] T. K. Ho, "The random subspace method for constructing decision forests," *IEEE*

- Trans. Pattern Anal. Mach. Intell.*, vol. 20, no. 8, pp. 832–844, 1998.
- [66] J. Platt and S. Haykin, “Information-Maximization Approach to Blind Separation and Blind Deconvolution,” *Technology*, vol. 1159, no. 6, pp. 1129–1159, 1995.
  - [67] S. Bhattacharyya *et al.*, “Feature selection for automatic burst detection in neonatal electroencephalogram,” *IEEE J. Emerg. Sel. Top. Circuits Syst.*, vol. 1, no. 4, pp. 469–479, 2011.
  - [68] A. P. Dempster, N. M. Laird, and D. B. Rubin, “Maximum likelihood from incomplete data via the EM algorithm,” *J. R. Stat. Soc. Ser. B Methodol.*, vol. 39, no. 1, pp. 1–38, 1977.
  - [69] M. I. Jordan and R. A. Jacobs, “Hierarchical Mixtures of Experts and the EM Algorithm,” *Neural Comput.*, vol. 6, no. 2, pp. 181–214, 1994.
  - [70] H. Avnion and T. Diep, “Arbitrarily tight upper and lower bounds on the bayesian probability of error,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 18, no. 1, pp. 89–91, 1996.
  - [71] A. Antos, L. Devroye, and L. Györfi, “Lower bounds for Bayes error estimation,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 21, no. 7, pp. 643–645, 1999.
  - [72] R. P. W. Duin, “A Note on Comparing Classifiers,” *Pattern Recognit. Lett.*, vol. 17, pp. 529–536, 1996.
  - [73] M. Khan, Q. Ding, and W. Perrizo, “k -nearest Neighbor Classification on Spatial Data Streams Using P-trees,” *Adv. Knowl. Discov. Data Min.*, pp. 517–528, 2002.
  - [74] R. Fletcher, *Practical Methods of Optimization; (2Nd Ed.)*. New York, NY, USA: Wiley-Interscience, 1987.
  - [75] C. M. Bishop, *Neural Networks for Pattern Recognition*. New York, NY, USA: Oxford University Press, Inc., 1995.
  - [76] V. N. Vapnik, *The Nature of Statistical Learning Theory*. New York, NY, USA: Springer-Verlag New York, Inc., 1995.
  - [77] C. J. C. Burges, “A Tutorial on Support Vector Machines for Pattern Recognition,” *Data Min. Knowl. Discov.*, vol. 2, no. 2, pp. 121–167, 1998.
  - [78] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification (2Nd Edition)*. Wiley-Interscience, 2000.
  - [79] E. Osuna, R. Freund, and F. Girosi, “An improved training algorithm for support

- vector machines,” *Neural Networks Signal Process. [1997] VII. Proc. 1997 IEEE Work.*, pp. 276–285, 1997.
- [80] J. C. Platt, “Fast training of support vector machines using sequential minimal optimization,” *Adv. kernel methods*, pp. 185–208, 1999.
  - [81] J. Kittler, M. Hater, and R. P. W. Duin, “Combining classifiers,” *Proc. - Int. Conf. Pattern Recognit.*, vol. 2, no. 3, pp. 897–901, 1996.
  - [82] A. Jain and D. Zongker, “Feature Selection: Evaluation, Application, and Small Sample Performance,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 19, no. 2, pp. 153–158, 1997.
  - [83] D. H. Wolpert, “Stacked Generalization,” vol. 87545, no. 505.
  - [84] L. Breiman, “Bagging predictors,” *Mach. Learn.*, vol. 24, no. 2, pp. 123–140, 1996.
  - [85] Y. Freund and R. E. Schapire, “Experiments with a New Boosting Algorithm,” *Proc. Int. Conf. Mach. Learn.*, pp. 148–156, 1996.
  - [86] L. K. Hansen and P. Salamon, “Neural Network Ensembles,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 12, no. 10, pp. 993–1001, 1990.
  - [87] L. Xu, A. Krzyżak, and C. Y. Suen, “Methods of Combining Multiple Classifiers and Their Applications to Handwriting Recognition,” *IEEE Trans. Syst. Man Cybern.*, vol. 22, no. 3, pp. 418–435, 1992.
  - [88] A. Saranli and M. Demirekler, “A unified view of rank-based decision combination,” vol. 15. pp. 479–482 vol.2, 2000.
  - [89] T. B. Laboratories, “Covariance Matrix Estimation and Classification with Limited Covariance Matrix Estimation and Classification with Limited Training Data,” vol. 18, no. 7, pp. 763–767, 1996.
  - [90] Y. Bengio, “Learning Deep Architectures for AI,” *Found. Trends® Mach. Learn.*, vol. 2, no. 1, pp. 1–127, 2009.
  - [91] X. Glorot and Y. Bengio, “Understanding the difficulty of training deep feedforward neural networks,” *Proc. 13th Int. Conf. Artif. Intell. Stat.*, vol. 9, pp. 249–256, 2010.
  - [92] P. Smolensky, “Information processing in dynamical systems: Foundations of harmony theory,” *Parallel Distributed Processing Explorations in the Microstructure of Cognition*, vol. 1, no. 1. pp. 194–281, 1986.

- [93] G. E. Dahl, M. Ranzato, A. Mohamed, and G. Hinton, "Phone Recognition with the Mean-covariance Restricted Boltzmann Machine," in *Proceedings of the 23rd International Conference on Neural Information Processing Systems - Volume 1*, 2010, pp. 469–477.
- [94] A.-R. Mohamed, G. Dahl, and G. Hinton, "Deep Belief Networks for Phone Recognition," *Scholarpedia*, vol. 4, no. 5, pp. 1–9, 2009.
- [95] A. Mohamed, D. Yu, and L. Deng, "Investigation of full-sequence training of deep belief networks for speech recognition," *Interspeech*, no. September, pp. 2846–2849, 2010.
- [96] a. Mohamed, G. E. Dahl, and G. Hinton, "Acoustic Modeling Using Deep Belief Networks," *IEEE Trans. Audio. Speech. Lang. Processing*, vol. 20, no. 1, pp. 14–22, 2012.
- [97] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov, "Improving neural networks by preventing co-adaptation of feature detectors," pp. 1–18, 2012.
- [98] G. E. Dahl, D. Yu, L. Deng, and A. Acero, "Context-dependent pre-trained deep neural networks for large-vocabulary speech recognition," *IEEE Trans. Audio, Speech Lang. Process.*, vol. 20, no. 1, pp. 30–42, 2012.
- [99] L. Deng, M. Seltzer, D. Yu, A. Acero, A.-R. Mohamed, and G. Hinton, "Binary Coding of Speech Spectrograms Using a Deep Auto-encoder," *Interspeech*, no. September, pp. 1692–1695, 2010.
- [100] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A Simple Way to Prevent Neural Networks from Overfitting," *J. Mach. Learn. Res.*, vol. 15, pp. 1929–1958, 2014.
- [101] A. Grubb and J. Bagnell, "Stacked Training for Overfitting Avoidance in Deep Networks," *Int. Conf. Mach. Learn.*, p. 1, 2013.
- [102] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A Fast Learning Algorithm for Deep Belief Nets," *Neural Comput.*, vol. 18, no. 7, pp. 1527–1554, 2006.
- [103] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 8, pp. 1798–1828, 2013.
- [104] Y. Bengio, P. Lamblin, D. Popovici, and H. Larochelle, "Greedy Layer-Wise Training of Deep Networks," *Adv. Neural Inf. Process. Syst.*, vol. 19, no. 1, p. 153, 2007.

- [105] A. Ahmed, K. Yu, W. Xu, Y. Gong, and E. Xing, "Training hierarchical feed-forward visual recognition models using transfer learning from Pseudo-tasks," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 5304 LNCS, no. PART 3, pp. 69–82, 2008.
- [106] R. Salakhutdinov and G. Hinton, "Using Deep Belief Nets to Learn Covariance Kernels for Gaussian Processes," *Adv. Neural Inf. Process. Syst. 20*, vol. 20, pp. 1–8, 2008.
- [107] G. W. Taylor and G. E. Hinton, "Factored conditional restricted Boltzmann Machines for modeling motion style," *Proc. 26th Annu. Int. Conf. Mach. Learn. - ICML '09*, pp. 1–8, 2009.
- [108] J. Yang, B. Price, S. Cohen, Z. Lin, and M. H. Yang, "PatchCut: Data-driven object segmentation via local shape transfer," *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, vol. 07–12–June, pp. 1770–1778, 2015.
- [109] R. Collobert and J. Weston, "A unified architecture for natural language processing," *Proc. 25th Int. Conf. Mach. Learn. - ICML '08*, pp. 160–167, 2008.
- [110] F. Seide, G. Li, and D. Yu, "Conversational speech transcription using Context-Dependent Deep Neural Networks," *Proc. Annu. Conf. Int. Speech Commun. Assoc. INTERSPEECH*, no. August, pp. 437–440, 2011.
- [111] G. Hinton *et al.*, "Deep Neural Networks for Acoustic Modeling in Speech Recognition," *IEEE Signal Process. Mag.*, no. November, pp. 82–97, 2012.
- [112] C. K. I. Williams, "Learning With Kernels: Support Vector Machines, Regularization, Optimization, and Beyond," *Journal of the American Statistical Association*, vol. 98, no. 462, pp. 489–489, 2003.
- [113] J. Martens, "Generating Text with Recurrent Neural Networks," *Neural Networks*, vol. 131, no. 1, pp. 1017–1024, 2011.
- [114] K. Doya, "Bifurcations in the learning of recurrent neural networks 3," *Learn.*, vol. 3, no. 0, p. 17, 1992.
- [115] T. P. Schmidt, M. A. Wiering, A. C. van Rossum, R. A. J. van Elburg, T. C. Andringa, and B. Valkenier, "Robust Real-Time Vowel Classification with an Echo State Network," *Work. Cogn. neural Model. Autom. Process. speech text*, pp. 1–8, 2010.
- [116] A. Graves, "Supervised Sequence Labelling with Recurrent Neural Networks," vol. 385, 2012.

- [117] A. Graves and N. Jaitly, "Towards End-To-End Speech Recognition with Recurrent Neural Networks," in *Proceedings of the 31st International Conference on Machine Learning*, 2014, vol. 32, no. 2, pp. 1764–1772.
- [118] A. Graves, A. Mohamed, and G. Hinton, "Speech Recognition With Deep Recurrent Neural Networks," *Icassp*, no. 3, pp. 6645–6649, 2013.
- [119] H. Jaeger, *Long Short-Term Memory in Echo State Networks: Details of a Simulation Study*. IRC-Library, Information Resource Center der Jacobs University Bremen, 2012.
- [120] T. Mikolov, M. Karafiat, L. Burget, J. Cernocky, and S. Khudanpur, "Recurrent Neural Network based Language Model," *Interspeech*, no. September, pp. 1045–1048, 2010.
- [121] B. Schrauwen and L. Using, "A hierarchy of recurrent networks for speech recognition," 2017.
- [122] D. P. Kingma and M. Welling, "Auto-Encoding Variational Bayes," no. ML, pp. 1–14, 2013.
- [123] M. K. Sharma, "Speech Recognition : A Review," in *Special Conference Issue: National Conference on Cloud Computing & Big Data*, 2015.
- [124] R. K. Aggarwal and M. Dave, "Implementing a Speech Recognition System Interface for Indian Languages," *Proc. IJCNLP-08 Work. NLP Less Privil. Lang.*, no. January, pp. 105–112, 2008.
- [125] I. Many, "An easy way to pronounce the consonants in Arabic," *English*, pp. 1–10, 2009.
- [126] and M. A. Yahya Ould Mohamed Elhadj, Mansour Alghamdi, "Phoneme-Based Recognizer to Assist Reading the Holy Quran," *Adv. Intell. Syst. Comput.*, vol. 235, pp. 141–152, 2014.
- [127] T. Zhang and C. J. Kuo, "Hierarchical classification of audio data for archiving and retrieving," *Acoustics, Speech, and Signal Processing, IEEE*, vol. 6, pp. 3001–3004, 1999.
- [128] L. R. Rabiner and M. R. Sambur, "An Algorithm for Determining the Endpoints of Isolated Utterances," *Bell System Technical Journal*, vol. 54, no. 2, pp. 297–315, 1975.
- [129] M. M. Rahman and M. Al-Amin Bhuiyan, "Continuous Bangla Speech Segmentation using Short-term Speech Features Extraction Approaches," *IJACSA*



*Int. J. Adv. Comput. Sci. Appl.*, vol. 3, no. 11, pp. 131–138, 2012.

- [130] G. Bitesize, “Advantages and disadvantages of mean, median and mode,” 2017. [Online]. Available: <http://www.bbc.co.uk/schools/gcsebitesize/maths/statistics/measuresofaveragev6.shtml>. [Accessed: 18-Oct-2017].
- [131] M. Kelt, “Advantages and Disadvantages of Advantages and Disadvantages of,” no. October. p. 2017, 2017.
- [132] “Notes on the Advantages and Disadvantages of Median.” [Online]. Available: <http://www.preservearticles.com/201104125330/advantages-and-disadvantages-of-median.html>. [Accessed: 18-Oct-2017].
- [133] S. Manikandan, “Measures of central tendency : The mean,” vol. 2, no. 2. pp. 140–142, 2011.
- [134] “The advantages of mean, median and mode.” [Online]. Available: <https://www.quora.com/What-are-the-advantages-of-mean-median-and-mode>. [Accessed: 18-Oct-2017].
- [135] D. H. Klatt, “Letter: Interaction between two factors that influence vowel duration,” *J. Acoust. Soc. Am.*, vol. 54, pp. 1102–1104, 1973.
- [136] C. GAMMAS, D., JUDITH, N. & NORA, “ArabicPod101.” [Online]. Available: <https://itunes.apple.com/us/podcast/learn-arabic-arabicpod101-com/id280208841?mt=2>.
- [137] F. Sattar, F. Rudzicz, and M. Pwint, “An evolutionary approach for segmentation of noisy speech signals for efficient voice activity detection,” *Artif. Intell. Res.*, vol. 5, no. 1, 2015.
- [138] S. J. Russell and P. Norvig, *Artificial Intelligence. A Modern Approach*. 2010.
- [139] P. A. A. Ali and I. T. Hwaidy, “HIERARCHICAL ARABIC PHONEME RECOGNITION USING MFCC ANALYSIS,” *Iraq J. Electr. Electron. Eng.*, vol. 3, no. 1, 2007.
- [140] F. Triefenbach, A. Jalalvand, K. Demuynck, and J. P. Martens, “Acoustic modeling with hierarchical reservoirs,” *IEEE Trans. Audio, Speech Lang. Process.*, vol. 21, no. 11, pp. 2439–2450, 2013.
- [141] T. Zhang and C. J. Kuo, “Hierarchical classification,” *Acoust. Speech, Signal Process. IEEE*, vol. 6, pp. 3001–3004, 1999.

- [142] F. Triefenbach, A. Jalalvand, B. Schrauwen, and J.-P. Martens, "Phoneme Recognition with Large Hierarchical Reservoirs," *Adv. Neural Inf. Process. Syst.* 23, vol. 23, pp. 1–9, 2010.
- [143] R. Polikar, "Ensemble based systems in decision making," *Circuits Syst. Mag. IEEE*, vol. 6, no. 3, pp. 21–45, 2006.
- [144] E. M. Essa, A. S. Tolba, and S. Elmougy, "A comparison of combined classifier architectures for arabic speech recognition," *2008 Int. Conf. Comput. Eng. Syst. ICCES 2008*, pp. 149–153, 2008.
- [145] N. N. Radio, "Neural Networks used for speech recognition," in *NINETEENTH NATIONAL RADIO SCIENCE CONFERENCE, ALEXANDRIA*, 2002, vol. 2, no. 4, pp. 19–21.
- [146] J. Hai and E. M. Joo, "Improved linear predictive coding method for speech recognition," *Information, Commun. Signal Process. 2003 Fourth Pacific Rim Conf. Multimedia. Proc. 2003 Jt. Conf. Fourth Int. Conf.*, vol. 3, no. December, pp. 1614–1618 vol.3, 2003.
- [147] F. O. F. Engineering, "Parametric Speech Emotion Recognition Using Neural Network," 2014.
- [148] A. Lilia and R. Herrera, "Un Método para la Identificación Automática del Lenguaje Hablado Basado en Características Suprasegmentales Ana Lilia Reyes Herrera Doctor en Ciencias en el área de Ciencias Computacionales," 2007.
- [149] D. G. M. John G. Proakis, *Digital Signal Processing*, Third. New Jersey, USA: Pearson Education, 1996.
- [150] F. Snell, Roy; Milinazzo, "Formant Location From LPC Analysis Data," *IEEE Tansaction speech audio Process.*, vol. 1, 1993.
- [151] M. W. Bhatti, Y. Wang, and L. Guan, "A Neural Network Approach for Human Emotion Recognition in Speech," *ISCAS*, pp. 0–3, 2006.
- [152] S. M. Al-qaraawi and S. S. Mahmood, "Wavelet Transform Based Features Vector Extraction in Isolated Words Speech Recognition System," *Int. Symp. Commun. Syst. Networks Digit. Sign*, pp. 847–850, 2014.
- [153] A. L. Reyes-herrera, L. Villaseñor-pineda, M. Montes-y-gómez, and L. E. Erro, "Automatic Language Identification using Wavelets," *INTERSPEECH*, 2006.
- [154] M. Elrgaby, A. Amoura, and A. Ganoun, "Spoken Arabic Digits Recognition Using Discrete Wavelet," *Int. Conf. Comput. Simul.*, pp. 274–278, 2014.

- [155] S. Sunny, D. P. S, and K. P. Jacob, "Performance Analysis of Different Wavelet Families in Recognizing Speech," *Int. mjournal Eng. trends Technol.*, vol. 4, pp. 512–517, 2013.
- [156] S. R. Safavian and D. Landgrebe, "A Survey of Decision Tree Classifier Methodology," *IEEE Trans. Syst. Man Cybern.*, vol. 21, no. 3, pp. 660–674, 1991.
- [157] A. Alfaries, M. Albahlal, M. Almazruea, and A. Almazruea, "Rules from Quran," 2015.
- [158] GÜÇLÜ ONGUN, "AN AUTOMATED DIFFERENTIAL BLOOD COUNT SYSTEM," NATURAL AND APPLIED SCIENCES THE MIDDLE EAST TECHNICAL UNIVERSITY, 2001.
- [159] L. I. Kuncheva, J. C. Bezdek, and R. P. W. Duin, "Decision templates for multiple classifier fusion: an experimental comparison," *Pattern Recognit.*, vol. 34, no. 2, pp. 299–314, 2001.
- [160] Y. Lu, "Knowledge integration in a multiple classifier system," *Appl. Intell.*, vol. 6, no. 2, pp. 75–86, 1996.
- [161] M. Deriche and A. H. Abo absa, "A Two-Stage Hierarchical Bilingual Emotion Recognition System Using a Hidden Markov Model and Neural Networks," *Arab. J. Sci. Eng.*, vol. 42, no. 12, pp. 5231–5249, 2017.
- [162] L. Hatcher, "Principal Component Analysis," *A Step-By-Step Approach to Using SAS Syst. Factor Anal. Struct. Equ. Model.*, pp. 1–56, 1994.
- [163] M. A. Bagheri, G. A. Montazer, and E. Kabir, "A subspace approach to error correcting output codes," *Pattern Recognit. Lett.*, vol. 34, no. 2, pp. 176–184, 2013.
- [164] J. Macqueen, "Some methods for classification and analysis of multivariate observations," *Proc. Fifth Berkeley Symp. Math. Stat. Probab.*, vol. 1, no. 233, pp. 281–297, 1967.
- [165] R. M. K. D. Suresh, "An Approach to Reveal the Impact of Anomalies in an Analysis of Clustering Gene Expression Data," *RSM Int. J. Eng. Technol. Manag.*, 2009.
- [166] N. Bolshakova and F. Azuaje, "Cluster validation techniques for genome expression data," *Signal Processing*, vol. 83, no. 4, pp. 825–833, 2003.
- [167] N. Morgan, "Deep and wide: Multiple layers in automatic speech recognition," *IEEE Trans. Audio, Speech Lang. Process.*, vol. 20, no. 1, pp. 7–13, 2012.

- [168] M. Kielhorn and G. Hinton, "Home Page of Geoffrey Hinton," 2017. [Online]. Available: <http://www.cs.toronto.edu/~hinton/MatlabForSciencePaper.html>. [Accessed: 05-Dec-2017].
- [169] A. Mohamed, "Deep Neural Network acoustic models for ASR," no. February, 2014.
- [170] S. Ben Driss, M. Soua, R. Kachouri, and M. Akil, "A comparison study between MLP and Convolutional Neural Network models for character recognition," 2017.
- [171] D. Yu, J. Li, and L. Deng, "Calibration of Confidence Measures in Speech Recognition," *IEEE Trans. Audio, Speech Lang. Process.*, vol. 19, no. 8, pp. 2461–2473, 2011.
- [172] P. Huang, K. Kumar, C. Liu, Y. Gong, L. Deng, and M. Corporation, "Predicting Speech Recognition Confidence Using Deep Learning With Word Identity And Score Features" Department of Electrical and Computer Engineering , University of Illinois at Urbana-Champaign , USA," *Word J. Int. Linguist. Assoc.*, no. 1, pp. 7413–7417, 2013.

## Vitae

Name	:Ahmed Hamdi Abo absa
Nationality	:Palestinian
Date of Birth	:4/1/1983
Email	:a.absah@gmail.com
Address	:Palestine, Gaza strip, University of Palestine
Academic Background	:Electrical Engineering, Signal Processing
Second Master certificate	:Electrical Engineering, communication systems, 2010
First Master certificate	:Computer Science, University of Northern Virginia, 2007
Bachelor certificate	:Electrical Engineering, control and communications, 2005