كلية الدراسات العليا

Sudan University of Science & Technology

College of Graduate Studies

Development of an Automatic Segmentation Model for Quran Verses at phoneme level

تطوير نموذج تقسيم آلي لآيات القرآن على مستوي الفونيم

A thesis submitted to the College of Graduate Studies, Faculty of Computer Science and Information TechnologySudan University of Science and Technology

In Partial Fulfillment of the Requirements for the Degree of

Doctor of Philosophy in Computer Science

By

Alaa Ehab Mohamed Elsaied Ali

Supervisor

Professor: Mohsen Rashwan

JULY 2022

بسم الله الرحمن الرحيم

بِسْمِ اللهِ الرَّحْمَنِ الرَّحِيمِ

قال الله تعالي

{ اقْرَأْ وَرَبُّكَ الْأَكْرَمُ * الَّذِي عَلَّمَ بِالْقَلَمِ * عَلَّمَ الْإِنسَانَ مَا لَمْ يَعْلَمْ }

صدق الله العظيم

سورة العلق الآية (1-5)

Acknowledgement

First and foremost, praises and thanks to the God, the Almighty, for His showers of blessings throughout my research work to complete the research successfully and His support for being with me in all directions of my life.

I would like to express my deep and sincere gratitude to my research supervisor, **Professor Mohsen Rashawn** Professor and Head, Research & Development International (RDI®), Giza, Egypt, Department of Electronics and Communication Engineering, Cairo University, Giza, Egypt, Department of IT, Faculty of Computers and Information, Cairo University, Giza, Egypt for giving me the opportunity to do research and providing invaluable guidance throughout this research. His dynamism, vision, sincerity and motivation have deeply inspired me. He has taught me the methodology to carry out the research and to present the research works as clearly as possible. It was a great privilege and honor to work and study under his guidance. I am extremely grateful for what he has offered me. I would also like to thank him for his friendship, empathy, and great sense of humor.

I am extremely grateful and honored to Sudan University of Science and Technology for giving me this priceless futuristic chance to have my name scripted to hold a PhD degree from their university and the great knowledge earned through the program.

Lastly my parents for their love, prayers, caring and sacrifices for educating and preparing me for my future. Special thanks to my baby sister for tender love and continuous asking and prayers. I am very much thankful to my husband for his love, understanding, prayers and continuing support to complete this research work.

Abstract

In this research, two models were built to obtain automatic segmentation of the verses of the Holy Quran at the phoneme level, in the first model the Hidden Markov Model Tool Kit HTK was used, the second model was built using the KALDI toolkit at phoneme.

to obtain the objectives of this research two data sets have been used. The first data set consisted of a database of 2 hours with the voice of 10 Sudanese male reciters who recited the Quran carefully under the supervision of an expert in the correct and most accurate recitations of the ascription to the Messenger (Peace Be Upon Him), the recordings of 16 Surahs of the Quran. the recitation commenced orderly as in the Holy Quran, starting from Surah Al-Bayyinah to Surah An-Nas.

the second data set has been recorded or 100 reciters non-Arab reciters (Indian male) and a total speech corpus of 80 hours in the correct and most accurate recitations. The data set contains surahs (Al-Fatiha, Al-Asr, Al-Kawthar, Al-Ikhlas, Al-Falaq, and An-Nas) the dataset also contains 10 letters which have similar pronunciation. (ظ، ذ، ط، ت، ض، ع، ح، خ، ص، غ)

in this research four experiments were evaluated as follows. In the first experiment a process of training and testing the first model was conducted using the first database, the results obtained from the automatic segmentation was 62%. in the second experiment a process of training and testing the second model was conducted using the first database, the results obtained from the automatic segmentation was 62% for test set 70%, and 75% for dev set. in the third experiment a process of training and testing the second model was conducted using the second database but the 10 letters have been omitted, the results obtained from the automatic segmentation was 95% for test set, and 99% for dev set. in the fourth experiment a process of training and testing the second model was conducted using the second database but the 10 letters have been applied, the results obtained from the automatic segmentation was 99.9% for test set and dev set.

المستخلص

في هذا البحث تم بناء وتطوير نموذجين بغرض التحصل على تقطيع آلي لآيات القرآن الكريم على مستوى الفونيم، في النموذج الأول تم إستخدام HTK_Tool ، اما في النموذج الثاني تم إستخدام KALDI_Tool.

لتحقيق أهداف البحث تم إستخدام قاعدتين للبيانات، قاعدة البيانات الأولي مدتها ساعتان بصوت عشرة قراء (ذكور بجنسية سودانية) يتلون القرآن قراءة محكمة بإشراف خبير بالقراءة الصحيحة للسند إلى الرسول (صلى الله عليه وسلم)، التسجيلات لعدد ستة عشر سورة من سور القرآن وهي بالترتيب في كتاب القرآن الكريم ابتداء من سورة البينة وصولًا إلى سورة الناس، قاعدة البيانات الثانية مدتها ثمانون ساعة بصوت مائة قارئ غير ناطقين باللغة العربية (ذكور بجنسية هندية) يتلون القرآن قراءة محكمة، التسجيلات لعدد ستة سور من سور القرآن وهي بالترتيب في كتاب القرآن الكريم (الفاتحة، العصر، الكوثر، الإخلاص، الفلق و الناس) بالإضافة لتسجيل النطق لعدد عشرة أحرف عربية تتشابه في النطق وهي (ظ، ذ، ط، ت، ض، ع، ح، خ، ص، غ) .

في هذا البحث تم تقييم أربعة تجارب كالتالي، في التجربة الأولي تم إجراء عملية تدريب و اختبار النموذج الأول باستخدام قاعدة البيانات الأولي وكان دقة التقطيع الآلي 62%. في التجربة الثانية تم إجراء عملية تدريب و اختبار النموذج الثاني باستخدام قاعدة البيانات الأولي وكان مقدار دقة التقطيع الآلي لل test set 70% و لل dev set 75%. في التجربة الثالثة تم إجراء عملية تدريب و اختبار النموذج الثاني باستخدام قاعدة البيانات الثانية مع استبعاد التسجيلات الخاصة بنطق الأحرف واستخدام التسجيلات التي تحتوي علي السور فقط، وكان دقة التقطيع الآلي للtest set 95% و 99% لل dev set. في التجربة الرابعة تم إجراء عملية تدريب و اختبار النموذج الثاني باستخدام قاعدة البيانات الثانية وكان مقداردقة التقطيع الآلي 99.9 % لل test set و dev set .

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Abbreviations

HTK	-	Hidden Markov Toolkit
GMM-HMM	-	Gaussian Mixture Model-Hidden Markov Model
DNN-HMM	-	Deep Neural Network-Hidden Markov Model
DNN	-	Deep Neural Network
MFCC	-	Mel-Frequency Cepstrum Coefficients
PCA	-	Principle Component Analysis
LDA	-	Linear Discriminant Analysis
ICA	-	Independent Component Analysis
LPC	-	Linear Predictive Coding
CMS	-	Cepstral Mean Subtraction
ZCPA	-	Zero Crossing Peak Amplitude
ALSD	-	Average Localized Synchrony Detection
PMVDLR	-	Perceptual Minimum Variance Distortion Less Response
PNCC	-	Power-Normalized Cepstral Coefficients
IIF	-	Invariant Integration Features
SPARK	-	SParse Auditory Reproducing Kernel
SLM	-	Statistical Language Modeling
ART	-	Adaptive Resonance Theory
TDNN	-	Time Delay Neural Networks
ANN	-	Artificial Neural Network
LM	-	Language Model
AM	-	Acoustic Model

CNN	-	Convolutional Neural Network
RNN	-	Recurrent neural network
DAE	-	Denoising Auto Encoder
DBN	-	Deep Belief Networks
LSTM	-	Long Short-Term Memory
GPU	-	Graphic Processing Units

Chapter I

Introduction

This chapter contains and discusses background for this thesis, statement of the problem, Important of the Research Questions, Research Hypotheses, research objectives, Research Methodology, Research Scope, and finally presents the Thesis Outline.

0.1 Background

Automatic segmentation for the holy Quran is challenging because of the lexical variety and data sparseness of the Arabic language. Arabic can be considered as one of the most morphologically complex languages. reducing the entry barrier to build robust Automatic Speech segmentation system for Arabic has been a research concern over the past decade [1][2]. unlike American English, for example, which has Carnegie Mellon University dictionary, standard KALDI scripts available, the Arabic language has no freely available resource for researchers to start working on Automatic Speech segmentation systems. to build a Holy Quran Automatic Speech segmentation system, a researcher does not only need to understand the technical details, but also to have the language expertise, which is a barrier for many people. this has been the main motivation for us to release and share this with the community. Researchers who are interested in building a baseline Arabic Automatic Speech segmentation system can use it as a reference.

significant amount of research has been done in Automatic phonetic segmentation which is a technique that defines boundary locations of certain sounds in each utterance. its use is required in situations where phone boundaries or limits must be detected for very large corpora. the areas of speech and speaker recognition, speech synthesis and speech coding use segmentation of speech according to its phonetic transcription [3].

It is typically used to form sub-word units for the purpose of concatenative speech synthesis [4][5], and to determine sound boundaries in massive language corpora Also, to train neural network-based speech recognition systems, or in other applications initiated and increasingly motivated by a study of pronunciation variability based on the analysis of the results of phonetic segmentation. a comprehensive analysis of certain sound realizations can also promote the clinical diagnosis of serious diseases that affect speech production, or the analysis of the pronunciation variability in formal or informal language [6].

multiple methods were used for the purpose of phonetic segmentation of speech; the most significant method being based on is the Hidden Markov Model (HMM) (Donovan, 1996; Ljolje et al., 1996; Nefti and Boffard, 2001; Pellom, 1998; Toledano and Gmez, 2002). most modern speech recognition systems use HMM which intuitively forms the backbone of the task of speech segmentation and is well suited for it [3].

the most crucial issue in the field of speech such as ASR, speech synthesis, speech database and speech identification and speaker verification are the segmentation of continuous speech into their corresponding phonemes. the most proposed phoneme-level speech segmentation utilized is either manual segmentation or automatic segmentation techniques. manual speech segmentation requires an expert / phonetician, and its segmentation is based on listening and visual assessment of the boundaries required. However, manual phonetic segmentation is tedious, expensive, inconsistent, error-prone, and time-consuming [7].

due to what was previously said in respect of manual speech segmentation, the development and need of automatic speech segmentation became increasingly important and needed. in brief, automatic speech segmentation techniques were divided into two types, namely, supervised and unsupervised segmentation techniques. Supervised procedures require prior expert knowledge of phoneme boundaries [8] [9]. these boundaries of the phonemes were in the form of their pre-segmented ones. it also required predefined models of the phoneme set of a particular language. on the other hand, unsupervised methods do not require a predefined model and no previous expert knowledge of phoneme sets or their limits are needed. it is mostly used in automatic speech segmentation through new modelling and training data sets [7].

Hence, for a given utterance with available acoustic implementation and known content, ideally on a phonetic level, the basic solution for determining sound boundaries or limits is based on a forced alignment of trained HMM models. in training acoustic models of speech recognizers, this technique is used by default as an important step. It can be performed using various toolkits that implement HMM-based speech recognition, e.g., HTK [10], Sphinx [8] or KALDI [11]. Above all, KALDI stands as the most popular language research toolkits worldwide.

in 2006, a diversity of a new procedure for learning (DNNs) with many hidden layers of nonlinear units, introduced a paradigm escalation in machine learning and artificial intelligence. DNNs were successfully experimented for acoustic modelling and have shown outstanding performance. research showed confirmed proof that the use of DNN improves the accuracy of speech segmentation. Therefore, the development of DNN-HMM hybrid acoustic models is growing rapidly and on a large scale respectively in all knowledge areas of speech Research [12].

1.1 Motivations and problem statement

The main research problem is evaluating the correctness and accuracy of any person's reading of the verses of the holy Qur'an, because for the recitation to be correct, it is not enough only to pronounce the letters of the verse correctly, but also each letter must take its correct time lap, especially with regard to the provisions of the (muddud=extensions of the holy Quran) and Accurately calculate the time of each letter precisely.

The process of performing various types of segmentation is a very hard and time-consuming process, especially if the volume of data is as large as the Holy Quran with all the Surahs in it.

It possible to implement the segmentation process manually, and depends mainly on the focus of the person, who performs this task and his/her personal evaluation of the segmentation (expert judgement), which may differ from one person to another, and this will result in accuracy issues. therefore, the search for an alternative began to find ways to perform this task, thus avoiding the errors and inaccuracy that people may fall into and save effort and time and get better segmentation accuracy.

Manual segmentation is not time beneficial. through an experiment the time taken to perform the segmentation process at the phoneme level of Surat Al-Falaq (one audio recording), which contains 121 phonemes, was 20 minutes. to perform the same task for 100 reciter it needs 33 hours of continuous work, which is very hard and requires a great deal of patience, focus, time consuming and may end up with inaccurate segmentation. the matter becomes worse by increasing the volume of data.

One of the most prominent topics currently is how to develop a system that teaches the correct reading of the Holy Quran and to implement this, the information must be extracted from the audio database, which is often a very large one (extracting the percentage of the audio file in addition to its linguistic content). one of the most important processes of this procedure is the process of adding sentence boundaries. this matter is of great importance as it represents the correct way to read the Holy Quran in compliance with the rules of Tajwid (intonation), in addition to defining word or sentence breaks that is very necessary to avoid ambiguity in the meaning or changing it.

Number of research have been completed that address the automatic segmentation of speech into different units of speech for several languages, but there is scarce researching that addresses the automatic segmentation of the verses of the Holy Quran at the phoneme level.

In this research, more than one tool was used represented in HTK-Tool and KALDI-Tool, to obtain the automatic segmentation of the verses of the Holy Quran and the details of this are addressed in Chapter Three.

1.2 Importance of this research

This research is concerned with finding methods to perform the automatic segmentation at phoneme level for the holy Quran to obtain excellent segmentation accuracy, a process that is almost impossible to be done manually as the time length of the phoneme can be 10 milliseconds.

The importance of this research is represented in a number of points, the first of which is: creating data, in both forms text and audio for the Holy Quran, which is an important matter due to the scarcity of recordings of the verses of the Holy Quran that must be free from noise and reliable, it is a must that the recitations is done correctly and perfectly, as well as with regard to the text corpus of the verses in conformity with the language and the rules of Tajwid (intonation).

The second point is that the segmentation process is considered as the basis for the success of all systems related to speech from division and synthesis to applications for understanding speech. Accordingly, all systems that teach the Holy Quran or test the user's learning and correct it or convert the Quranic texts into audible verses primarily all depend on the segmentation process and its accuracy is reflected in the final outputs.

1.3 Research Questions

1- What speech features or their combinations would result in optimal speech segmentation?

2- What model is optimal for automatic speech segmentation?

3- What distinguishes DNN- HMM from GMM- HMM?

4- Could results be improved and better accuracy obtained? What are the requirements?

5- What are the limitations for GMM- HMM and DNN- HMM?

1.4 Research Hypotheses

H1: Using DNN-HMM achieves best results of segmentation.

H2: The increase in corpus size, will result in more accurate findings.

H3: an accurate representation of text with respect to the spoken words will have direct impact in better results at the training phase.

H4: The assurance of training phase outcomes, is directly related to the audio recording which should not contain any noise or echo. Keep in mind that is the major element of capturing good result in the training phase and can be limiting factor for the rest of the entire research.

1.5 Research Objectives

1.5.1 General Objective

The main objective of this research is , to build a model For the verses of the Holy Quran at phoneme level to evaluate the correctness and accuracy for the reciters reading the verses of the holy Qur'an, because for the reading to be correct, it is not enough only to pronounce the letters of the verse correctly, but also each letter must take its correct time, especially with regard to the provisions of the (muddud=extensions of the holy Quran) and Accurately calculate the time of each letter precisely.

1.5.2 Specific Objectives

Investigating, examining, and evaluating the impact of different Acoustic models. Contributing to filling the limitation of resources of the automatic segmentation regarding the Holy Quran, from audio or text files that are compatible with HTK and KALDI.

1.6 Research Methodology

This part provides an overview of the research methodologies used in this research.
this research adopted the GMM-HMM methodology, represented using the HTK tool and the DNN-HMM methodology, represented using the KALDI tool.

the research methodology aims at finding solutions to the research questions in a theoretical scientific form that achieves the objectives and adds something useful to the body of knowledge.

the techniques that were used aimed directly at addressing the research objectives and goals.

to achieve this, there are several stages represented in:

1. Preparing text and acoustic corpus.
2. Creating acoustic model.
3. Creating language model.
4. Training of the system.
5. Testing the system.
6. Obtaining an automatic segmentation.
7. Evaluating the model.

1.7 Research Scope

This research is concerned with finding and calculating the accuracy of automatic segmentation of verses of the Holy Quran by using different models such as HTK Tool and KALDI Tool and comparing the results obtained automatically with the results of manual segmentation that were achieved by using PRAAT program and based on the comparison the accuracy of automatic segmentation is calculated.

1.8 Thesis Outline

This thesis has the following structure: Chapter 2 provides the necessary theoretical back- ground in addition to a detailed discussion of related works and methodologies. Chapter 3 introduces the research methodology and the proposed solution for automatic phonetic segmentation for verses. Chapter 4 presents further interpretation of the results. Finally, conclusions, and future work set out in chapter 5.

Chapter II

Theoretical Background and Related Works

This chapter covers the overall concepts of the General characteristics of speech, Speech segmentation, Segmentation units, Types of automatic speech segmentation, Types of features are used for segmenting, Automatic speech segmentation methods, Phonology of the Holy Qur'an recitation, The Standard Arabic Phonology, Speech segmentation architecture, Machine learning, Supervised machine learning algorithms, Artificial neural networks, Deep learning methods, Deep neural networks and finally presents the Related work on Deep Neural Network for Acoustic Models.

2.0 Introduction

Segmentation of continuous speech into its corresponding phonemes is a very important issue in the area of speech like ASR, speech synthesis, speech database, and language identification and speaker verification. the most proposed phoneme level speech segmentations are using either manual segmentation or automatic segmentation techniques. In manual speech segmentation expert/phonetician is required and its segmentation is based on listening and visual judgment on required boundaries. However, manual phonetic segmentation is tedious, expensive, inconsistent, prone to errors and time-consuming task. there is also a disagreement between phoneticians and there are no clear, common, and coherent strategies to segment speech wave forms. considering these and other disadvantages the development of automatic speech data is becoming increasingly important [3, 4]. generally, automatic speech segmentation methods are divided into two types, namely supervised and unsupervised segmentation methods. supervised methods require a priori knowledge about phoneme boundaries [41].

these boundaries of phonemes are existed in the form of their pre-segmented. It also requires pre-defined models of phoneme set of a specific language. on the other side, unsupervised methods don't require pre-defined model and knowledge about phoneme sets and their boundaries respectively. it is most used in automatic speech segmentation through new modeling and training data sets [8]. Thus, unsupervised method yields a desirable and more flexible framework for automatic segmentation of a speech at phoneme level [41].

2.1 General characteristics of speech

A continuous speech signal consists of two main parts: one carries the speech information, and the second carries the silent or noise sections that are in between the utterances, without any verbal information. The verbal (informative) part of speech can be further divided into two main types: Voiced and Unvoiced speech. While humans speak all air paths through larynx, voiced sounds are generated when vocal cords are semi closed while when they are opened unvoiced speech is generated. In voiced speech is a relatively slowly changing periodic signal with case frequency of vibration of vocal cords called pitch. Male's contribution of pitch is commonly in the range between 50Hz to 250Hz while female's contribution of pitch lies between 120Hz and 500Hz. Unvoiced speech sounds are produced by air passed directly through vocal tract formations. Unlike voiced speech, unvoiced speech does not exhibit periodicity, and is characterized by a noise-like signal. The speech production process involves producing speech utterances which are groups of successive voiced and/or unvoiced sounds. These sounds usually are very smoothly connected. Speech utterances are separated by silence regions. There is no excitation supplied to the vocal tract during silence regions and hence there is no speech output. None-the-less, silence is considered an integral part of speech signal [45]. Acoustic signals during silence regions are mainly background noise and speech irrelevant mouth sounds.

2.2 Speech segmentation

Speech segmentation can be defined as the process of finding the limits (with specific characteristic) in natural spoken language between words, syllables or phonemes [46][47]. The main objective of Speech segmentation is to serve other speech analysis problems such as speech synthesis, data training for speech recognizers, or to fabricate and label prosodic databases. Therefore, it can be viewed as a vital sub-issue for various fields in speech analysis and research [48][49]. The traditional approach handling this issue is by manual segmentation of speech, which is generally performed by specialized phoneticians. However, this method is based on listening and visual judgment on required boundaries which makes it inconsistent and time consuming [50][51]. which is considered very convenient, another method is an automatic segmentation.

The speech can be automatically segmented into sub word units which are defined acoustically [52]. In Automatic speech Recognition ASR systems, segmentation can be performed:

i. At the system training stage, when segmentation is applied to the training set recordings.
ii. At the recognition stage [47].

2.3 Segmentation units

The Speech recognition and synthesis systems need a speech signal to be segmented into some basic units like Words, Phonemes, or syllables. Depending on the extent or size of vocabulary the decision of representative units is made. Word is the most natural unit of segmentation. It's not suitable to use words as the units for segmentation because of the absence of generalization and more memory space consumption [52].

Phonemes are the smallest segmental unit of sound employed to form meaning. The same phoneme in various words has distinctive significance. There is an over generalization of phonemes. So, the blend of phoneme and words gives rise to next level basic unit of speech called as syllables. Syllable like units are defined by rules, a syllable must have a vowel called its nucleus. [53]

The realization of a phoneme is strongly influenced by its adjacent phonemes. Phonemes are highly context dependent. Hence, the acoustic variability of basic phonetic units due to context is extremely large and is not well understood in many languages [54][52]

2.4 Types of automatic speech segmentation

Automatic speech segmentation strategies can be grouped in various perspectives; however, one very common classification is the division to blind and aided segmentation algorithms.

2.4.1 Blind segmentation

The term blind segmentation refers to methods where there is no pre-existing or external knowledge regarding linguistic properties, such as orthography or full phonetic annotation of the signal to be segmented. Blind segmentation is applied in various applications, for example speaker verification systems, speech recognition systems, language identification systems, and speech corpus segmentation and labeling [55].

Solutions to this problem comprise of algorithms which do not require any background knowledge about the phonetic content and are based predominantly on statistical signal analysis [56][47].

Due to the lack of external or top-down information, the first phase of blind segmentation relies completely on the acoustical features present in the signal. The second phase or bottom-up processing is normally built on a front-end parameterization of the speech signal, often using MFCC, LP-coefficients, or pure FFT spectrum [57].

2.4.2 Aided segmentation

Aided segmentation Algorithms use some sort of external linguistic knowledge of the speech stream to segment it into relating segments of the wanted type. An orthographic or phonetic transcription is used as a parallel contribution with the speech or training the algorithm with such data [58]. These algorithm types are computationally consuming. This group includes algorithms using recognition with Hidden Markov Models (HMMs) [59][60][61], Dynamic Time Warping (DTW) [62] or Artificial Neural Networks (ANNs) [63]. The algorithms of this group are employed only in the ASR system training stage.

One of the most common methods in ASR for utilizing phonetic annotations is with HMM- based systems [64]. HMM-based algorithms have dominated most speech recognition applications since the 1980s because of their high performance in recognition and relatively small computational complexity in the field of speech recognition [65][66][67][68].

2.5 Types of features used for segmenting

Two types of signal features are used for segmenting speech signal: time-domain features and frequency domain features.

2.5.1 Time-Domain Signal Features

Time-domain features are widely used for speech segment extractions. These features are useful when it is needed to have algorithm with simple implementation and efficient calculation.

2.5.1.1 Short-time signal energy

Short-time energy is the dominant and most natural feature that has been used. Physically, energy is a measure of how much signal there is at any one time. energy is used to discover voiced sound, which have higher energy than silence/unvoiced sound in a continuous speech. the energy of a signal is typically calculated on a short time basically by windowing the signal at a particular time, squaring the samples, and taking the average. the square root of this result is an engineering quantity, known as the Root Mean Square (RMS) value. The short-time energy function of a speech frame with length N is defined as: [69][70]

$$E_n = \frac{1}{N}\sum_{m=-\infty}^{\infty}[x(n-m)w(m)]^2 \qquad (2.1)$$

The short-term root mean square (RMS) energy of this frame is given by: [71][72][73][74]

$$E_{n(RMS)} = \sqrt{\frac{1}{N}\sum_{m=-\infty}^{\infty}[x(n-m)w(m)]^2} \qquad (2.2)$$

Where x(n) is the discrete-time audio signal and w(n) is rectangle window function [71][72] [73] [74].

$$w(n) = \begin{cases} 1 & 0 \le n \le N-1 \\ 0 & otherwise \end{cases} \qquad (2.3)$$

2.5.1.2 Short-time average zero-crossing rate

The average zero-crossing rate refers to the number of times speech samples change algebraic sign in each frame [69]. The rate at which zero crossings occur is a simple measure of the frequency content of a signal. It is a measure of number of times in a given time interval/frame that the amplitude of the speech signals passes through a value of zero [75]. Unvoiced speech components normally have much higher ZCR values than voiced ones. The short-time average zero-crossing rate is defined as:

$$Z_n = \frac{1}{2}\sum_{m=-\infty}^{\infty} |sgn[x(n-m)] - sgn[x(n-m-1)]|w(m) \qquad (2.4)$$

Were

$$sgn[x(n)] = \begin{cases} 1 & x(n) \geq 0 \\ -1 & x(n) < 0 \end{cases} \qquad (2.5)$$

And w (n) is a rectangle window of length N, given in equation (2.3) [71][73].

2.5.2 Frequency-domain signal features

The most information of speech is amassed in 250Hz-6800Hz frequency range. to extract frequency-domain features, discrete Fourier transform can be used. The Fourier representation of a signal demonstrates the spectral composition of the signal [76].

2.5.2.1 Spectral centroid

The spectral centroid is a measure used in digital signal processing to characterize a spectrum. It indicates where the center of gravity of the spectrum high values corresponding to brighter sounds [77] [78]. The spectral centroid, SCi of the i-th frame is defined as the center of gravity of its spectrum and it is given by the following equation: [76]

$$SC_i = \frac{\sum_{n=0}^{N-1} f_i(n) x_i(n)}{\sum_{n=0}^{N-1} x_i(n)} \qquad (2.6)$$

Where x(n) represents the weighted frequency value, or magnitude, of bin number n, and f(n) represents the center frequency of that bin in DFT spectrum. The DFT is given by the following equation and can be computed efficiently using a fast Fourier transform (FFT) algorithm [76].

$$X_k = \sum_{n=0}^{N-1} x_n . e^{-j2\pi k \frac{n}{N}} \quad , \qquad k = 0, \dots, N-1 \quad (2.7)$$

Here, X_k is the DFT coefficients of i-th short-term frame with length N [72][71][73].

2.5.2.2 Spectral flux

Spectral flux is a measure of how quickly the power spectrum of a signal is changing calculated by comparing the power spectrum for one frame against the power spectrum from the previous frame, also known as the Euclidean distance between the two normalized spectra.

The spectral flux can be used to determine the timbre of an audio signal, or in onset detection [79] among other things. The equation of Spectral Flux, SF_i is given by:

$$SF_i = \sum_{k=1}^{N/2} (|X_i(k)| - |X_{i-1}(k)|)^2 \qquad (2.8)$$

Here, Xi(k) is the DFT k-th coefficient of i-th short-term frame with length N, given in equation (2.7) [71][73].

2.6 Automatic speech segmentation methods

Automatic speech segmentation methods are divided into two types:

2.6.1 Supervised methods

Require a priori knowledge [82][83][84]. Most of the supervised methods are based on forced alignment techniques starting from an orthographic transcription of the speech material.

This means that the representation of the word or utterance in terms of discrete units is known from a lexicon which includes the words' pronunciations and pre-trained acoustic models of these units are needed for the forced alignment. The task of the segmentation algorithm is then to optimally locate the unit boundaries [85].

2.6.2 Unsupervised methods

Require no training data for segmenting the speech signal. Instead, they use sets of rules derived from or encoding human knowledge to segment speech. Acoustic rate of change [85]; see for early work on unsupervised automatic speech segmentation [86]; for more recent work, see below is an example of prior human knowledge that is used to solve the speech segmentation task. The task for an unsupervised segmentation algorithm then is two-fold; the number of segments in the speech signal needs to be determined this is usually determined by a parameter, and the position of the boundaries based on the acoustic signal needs to be determined [85].

There are some good reasons for using unsupervised methods. First, supervised methods require extensive training on carefully prepared speech material. The training material needs to be transcribed in terms of the units the algorithm is supposed to segment the speech signal into, usually phones. Furthermore, usually large amounts of training data are needed to train the supervised algorithms; however, large amounts of training data are not always easily obtained, and neither are transcriptions. Unsupervised methods, on the other hand, do not require training; so, obviously no training material is needed. For each new language, speech style, dialect or accent, supervised algorithms may need to be retrained, whereas unsupervised methods are based on human knowledge and understanding of the nature of speech and are therefore language and speech style independent. Furthermore, supervised methods require the units to be defined beforehand, e.g., phones, diphones, syllables, and words, to be able to train models for them, whereas unsupervised methods, in principle, do not. Thus, unsupervised methods yield a desirable and more flexible framework for the automatic segmentation of speech. Finally, unsupervised segmentation methods are generally simpler algorithms than supervised methods [87].

2.7 Phonology of the Holy Qur'an recitation

Basic requirements for a pronunciation teacher include knowledge of acoustic properties of both correct pronunciation and common pronunciation mistakes. This section discusses the different aspects of acoustic phonetic modeling of the Holy Qur'an recitation.
a phoneme is defined as "the smallest unit which can make a difference in meaning". But in the Holy Qur'an phoneme can be more accurately defined as "the smallest speech unit which

can make a difference in correctness of recitation [195].

2.8 The Standard Arabic Phonology

Studies of phonetics of the Arabic language stated that standard Arabic language has 28 consonants plus 3 short vowels and their 3 long These phonemes and their SAMBA transcription symbols are listed in table 2.1 (SAMBA) [195].

Table 2.1 Standard Arabic phonemes.

TYPE	SYMBOL	ORTHOGRAPHY
Consonants	?	أ
	b	ب
	t	ت
	T	ث
	Z	ج
	X\	ح
	x	خ
	d	د
	D	ذ
	r	ر
	z	ز
	s	س
	S	ش
	s'	ص
	d'	ض
	t'	ط
	D'	ظ

	?' (?\)	ع
	G	غ
	f	ف
	q	ق
	k	ك
	l	ل
	l'	ا
	m	م
	n	ن
	h	ه
	w	و
	j	ي
Short Vowels	i	كسرة
	a	فتحة
	u	ضمة
Long Vowels	i:	1ي
	a:	1ا
	u:	1و

Table 2.2 Shows Arabic language phoneticians' classification of these phonemes according to their acoustic characteristics [195].

Table 2.2 Acoustic characteristics of standard Arabic phonemes

PHONEME	VOICED/ UNVOICED	PLOSIVES	FRICATIVES	NASALS	TRILL
?	-	P	-	-	-
b	V	P	-	-	-
t	U	P	-	-	-
T	U	-	F	-	-
Z	V	P	-	-	-
X\	U	-	F	-	-
x	U	-	F	-	-
d	V	P	-	-	-
D	V	-	F	-	-
r	V	-	-	-	T
z	V	-	F	-	-
s	U	-	F	-	-
S	U	-	F	-	-
s'	U	-	F	-	-
d'	V	P	-	-	-
t'	V	P	-	-	-
D'	V	-	F	-	-
?'	V	-	F	-	-
G	V	-	F	-	-
f	U	-	F	-	-
q	V	P	-	-	-
k	U	P	-	-	-

l	V	-	-	-	-
m	V	-	-	N	-
n	V	-	-	N	-
h	U	-	F	-	-
w	V	-	-	-	-
j	V	-	-	-	-
i i:	V	-	-	-	-
a a:	V	-	-	-	-
u u:	V	-	-	-	-

2.9 Differences in the Holy Qur'an phonology

In correct recitation of the Holy Qur'an some phonemes deviate from their standard pronunciation to a totally new form that could differ by some properties. To represent these deviations special phonemes can be added to represent those special pronunciations. These special pronunciations are for letters 'Raa', 'Meem', 'Noon', vowels and 'Kalkala' or "agitation" [195].

2.9.1 The Letter Raa "ر"

In the Holy Qur'an phoneme Raa "r" "ر" can be emphatic or non-emphatic to distinguish these two cases we will use symbol "r" for non-emphatic Raa and " r' " for emphatic Raa [195].

2.9.2 The Letter Meem "م"

In the Holy Qur'an phoneme Meem "m" "م" has three cases:

1. non-concealed, which is similar to the corresponding phoneme in standard Arabic.
2. Repeated "Gonna" which has the same acoustic characteristics to the phoneme "m" in standard Arabic but have extra length.
3. Concealed if it was not vowelized and followed by letter Baa "b" "ب". It mainly differs than "m" in standard Arabic by duration and can also differ by acoustic features [195].

2.9.3 The Letter Noon "ن"

In the Holy Qur'an phoneme Noon "n" "ن" has five cases:

1. non-concealed which is similar to the corresponding phoneme in standard Arabic.

2. Repeated "Gonna" which has the same acoustic characteristics to the phoneme "n" in standard Arabic but have extra length.

3. Concealed if it was not vowelized and followed by any letter from the group "t T Z d D z s S s' d' t' D' f q k" "ت ث ج د ذ ز س ش ص ض ط ظ ف ق ك". It mainly differs than "n" in standard Arabic by duration and can also differ by acoustic features.

4. Consolidated in letter Waw "w" "و" when it is not vowelized and followed by letter Waw. It differs than both "n" and "w" in standard Arabic by duration and differs by acoustic features.

5. Consolidated in letter Yaa "j" "ى" when it is not vowelized and followed by letter Yaa. It differs than both "n" and "j" standard Arabic by duration and also differ by acoustic features [195].

2.9.4 Extra lengthening of Vowels and Semivowels

In correct rotation of the Holy Qur'an, vowels and semivowels may be extra lengthened than their original lengths. The lengthening is measured in a unit called "motion" which is defined as the time it takes to fold or unfold a hand finger. main possible extra lengthening is 2, 4 and 6 motions. table 2.3 shows the symbols used for each vowel and semivowel at each possible extra lengthening period [195].

2.9.5 The agitation "Kalkala" "القلقلة"

Kalkala is a vowel like sound that follows any not vowelized occurrence of the phonemes "b Z d t' q" "ب ج د ط ق". The symbol "K" will use to represent it [195].

Table 2.3 Symbols of extra-lengthened vowels and semivowel

Vowel Or Semivowel	2 motions	4 Motions	6 Motions
I	i:	i::	i:::
A	a:	a::	a:::
U	u:	u::	u:::
W	w:	w::	w:::
J	j:	j::	j:::

2.10 Speech segmentation architecture

A typical speech segmentation system is developed with major components that include acoustic front-end, acoustic model, lexicon, language model and decoder as shown in Figure 2.1 Acoustic front-end takes care of converting the speech signal into appropriate features which provides useful information for segmentation. The input audio waveform from a microphone is converted into a sequence of fixed-size acoustic vectors is a process called feature extraction. The parameters of word / phone models are estimated from the acoustic vectors of training data. The decoder operates by searching through all possible word sequences to find the sequence of words that is most likely to generate. The likelihood is defined as an acoustic model P (O|W) and P (W) is determined by a language model. The functionality of automatic speech segmentation system can be described as an extraction of a number of speech parameters from the acoustic speech signal for each word or sub-word unit. The speech parameters describe the word or sub-word by their variation over time and together they build up a pattern that characterizes the word or sub-word.

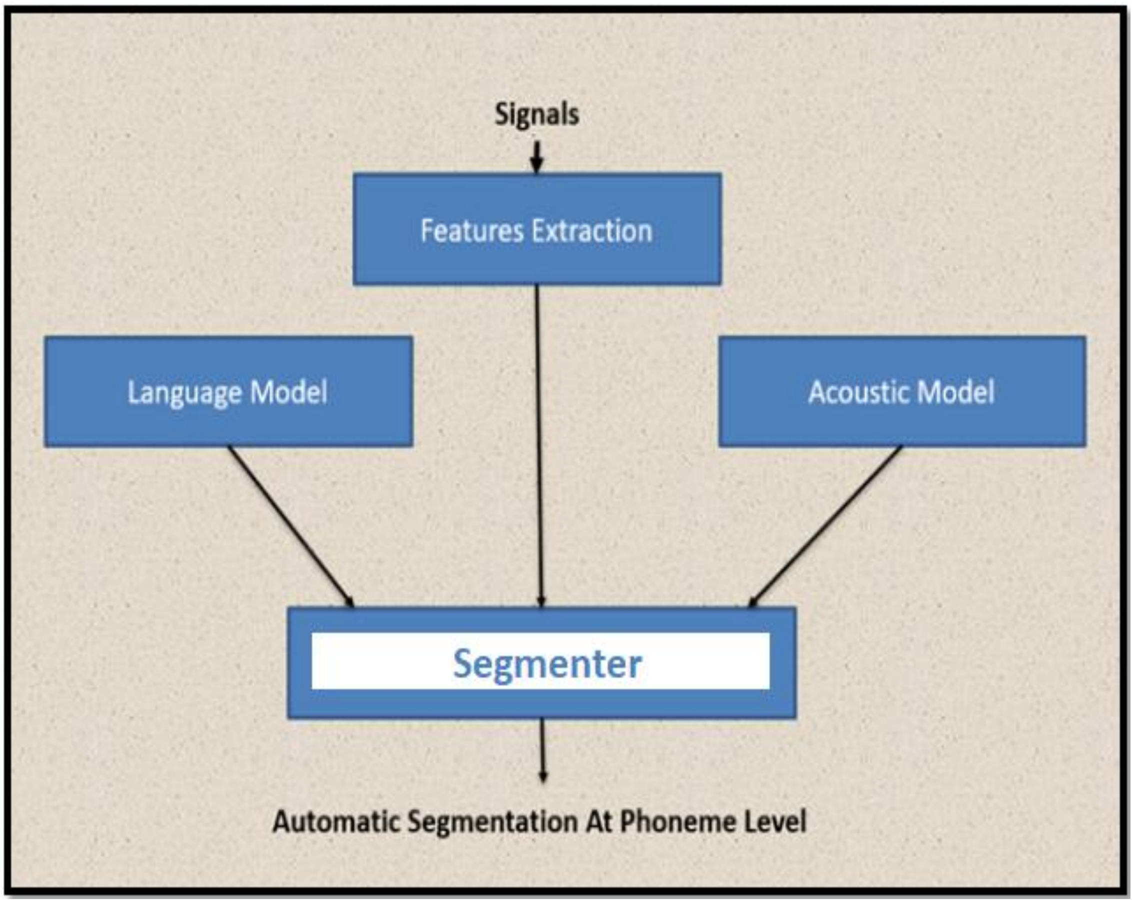


Figure 2.1 Speech segmentation architecture

2.10.1 Acoustic front-end

Acoustic front-end involves signal processing and feature extraction. In speech segmentation, the main goal of the feature extraction step is to compute a parsimonious sequence of feature vectors providing a compact representation of the given input signal [89]. The feature extraction is usually performed in three stages. The first stage is called the speech analysis or the acoustic front end. It performs spectra temporal analysis of the signal and generates raw features describing the envelope of the power spectrum of short speech intervals. The second stage compiles an extended feature vector composed of static and dynamic features. Finally, the last stage (which is not always present) transforms these extended feature vectors into more compact and robust vectors that are then supplied to the recognizer. There is no feature suitable for particular application, but the choice of features has to fulfill the following properties: they should allow an automatic system to discriminate between different through similar sounding speech sounds, they should allow for the automatic creation of acoustic models for these sounds without the need for an excessive amount of training data, and they should exhibit statistics which are largely invariant across speakers and speaking environment. To find some statistically relevant information from incoming data, it is important to have mechanisms for reducing the information of each segment in the audio signal into a relatively small number of parameters, or features. These features should describe each segment in such a characteristic way that other similar segments can be grouped together by comparing their features. There are enormous interesting and exceptional ways to describe the speech signal in terms of parameters. Some of the feature extraction methods are, (PCA), (LDA), (ICA), (LPC), Cepstral Analysis, Mel-Frequency Scale Analysis, Filter-Bank Analysis, (MFCC), Kernal Based Feature Extraction, Dynamic Feature Extraction, Wavelet based features, Spectral Subtraction and (CMS).

in noise robust speech recognition, many auditory-based feature extraction methods, such as (ZCPA), (ALSD), (PMVDR), (PNCC), (IIF), amplitude modulation spectrogram, Gammatone frequency cepstral coefficients, (SPARK), and Gabor filter bank features are effectively applied [90].

There are many feature representations in use, but the most common is the Mel frequency cepstral coefficient (MFCC) feature set [80]. The MFCC feature extraction process has many steps which are elaborated below, and the pictorial representation is given in Figure 2.2.

A. **Pre-emphasis**: This stage is used to amplify energy in the high frequencies of the input speech signal. This allows information in these regions to be more recognizable during HMM model training and recognition.

B. **Windowing**: This stage slices the input signal into discrete time segments. This is done by using a window of N milliseconds wide and at offsets of M milliseconds long. A Hamming window is commonly used to prevent edge effects associated with the sharp changes in a rectangular window.

C. **Discrete Fourier transform**: DFT is applied to the windowed speech signal, resulting in the magnitude and phase representation of the signal.

D. **Mel filter bank**: While the resulting spectrum of the DFT contains information in each frequency, human hearing is less sensitive at frequencies above 1000 Hz. This concept also has a direct effect on performance of ASR systems; therefore, the spectrum is warped using a logarithmic Mel scale. to create this effect on the DFT spectrum, a bank of filters known as triangular filters is constructed with filters distributed equally below 1000 Hz and spaced logarithmically above 1000 Hz. The output of filtering the DFT signal by each Mel filter is known as the Mel spectrum.

E. **Log**: Taking logarithm of this provides Mel spectrum co-efficient.

F. **Discrete cosines transform**: discrete cosine transforms The final step in obtaining MFCC is performing discrete cosine transform on the Mel spectrum coefficients. The output of DCT is Mel-cepstral coefficients of 13th order.

G. **Delta MFCC features**: In order to capture the changes in speech from frame to frame, the first and second derivative of the MFCC coefficients are also calculated and used [91].

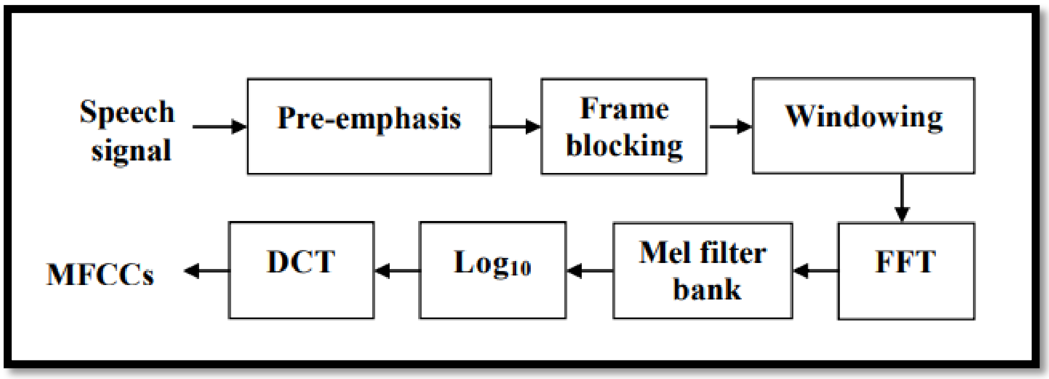


Figure 2.2 Diagram for MFCC algorithm

2.10.2 Acoustic model

Acoustic model is one of the most important knowledge sources for automatic speech segmentation system, which represents acoustic features for phonetic units to be recognized. In building an acoustic model, one fundamental and important issue is choosing of basic modeling units. When the target language of the speech is specified, there is several types of sub word unit can be used for acoustic modeling. Different acoustic modeling unit can make a dramatic difference on the performance of the speech segmentation system. Acoustic modeling of speech typically refers to the process of establishing statistical representations for the feature vector sequences computed from the speech waveform. Hidden Markov Model (HMM) is one of the most used statistical models to build acoustic models. Other acoustic models include segmental models, super segmental models (including hidden dynamic models), neural networks, maximum entropy models, and (hidden) conditional random fields, etc. An acoustic model is a file that contains statistical representations of each of the distinct sounds that makes up a word. Each of these statistical representations is assigned a label called a phoneme acoustic model is created by taking a large database of speech called a speech corpus and using special training algorithms to create statistical representations for each phoneme in a language. Each phoneme has its own HMM. The speech decoder listens for the distinct sounds spoken by a user and then looks for a matching HMM in the acoustic model. Each spoken word w is decomposed into a sequence of basic sounds called base phones. The acoustic model describes the probability of a specific observation given a base phone.

2.10.3 Language model

A language model is a collection of constraints on the sequence of words acceptable in a given language. These constraints can be represented, for example, by the rules of a generative grammar or simply by statistics on each word pair estimated on a training corpus. Although there are words that have similar sounding phone, humans generally do not find it difficult to recognize the word. This is mainly because they know the context, and also have a fairly good idea about what words or phrases can occur in the context. Providing this context to a speech segmentation system is the purpose of language model. The language model specifies what are the valid words in the language and in what sequence they can occur. Language models are usually trained that is, the n-gram probabilities are estimated by observing sequences of words in corpora of text that contain, typically, millions of word tokens and by reducing perplexity on training data [92]. Common language models are bigram and trigram models. These models contain computed probabilities of groupings of two or three particular words in a sequence, respectively. There are tools for language modeling like CMU, Statistical Language Modeling (SLM) Toolkit, Stanford Research Institute Language Modeling Toolkit.

2.10.4 Segmenter

In this stage, the task is to find the most likely word sequence W given the observation sequence O, and the acoustic-phonetic-language model. The decoding problem can be solved using dynamic programming algorithms. Rather than evaluating likelihoods of all possible model paths generating O, the focus is on finding a single path through the network yielding the best match to O. To estimate the best state sequence for the given observation sequence, the Viterbi algorithm is frequently used [93]. In the case of larger vocabulary, it would be challenging to consider all possible words during the recursive part of the Viterbi algorithm. To address this, a beam search can be used for Viterbi iteration, only the words with path probabilities above a threshold are considered when extending the paths to the next time step. This approach speeds up the searching process. The Viterbi algorithm assumes that each of the best paths at time t must be an extension of each of the best paths ending at time t − 1, which is not generally true. The path that seems to be less probable than others in the beginning may turn into being the best path for the sequence as a whole the most probable phoneme sequence does not need to correspond to the most probable word sequence). this issue is addressed by extended Viterbi and forward-backward algorithms [92].

2.11 Machine learning

Over the years, sophisticated skills were developed to recognize patterns like speech, handwriting, facial features, etc. The pursuit to computer programs that make computers learn the above skills from the past experience gave birth to machine learning. Mitchell [94] stated in the context of machine learning that, “A computer program is said to learn from experience E with respect to some class of tasks in T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E.” Few related basic terminologies used with machine learning are introduced as follows:

Example: an instance of the input.

Features: an attribute set characterizing the input, represented as a vector or linear array.

Labels: the category or class associated (e.g., positive or negative in binary classification or a real value in regression).

Training data: data used to train the ML algorithm during learning phase.

Test data: data used to test the performance of the learning algorithm during generalization phase. Depending on how the machine gains knowledge to respond correctly, learning can be categorized into four basic methods as briefed in the following sections.

Machine learning algorithms employ a variety of statistical, probabilistic and optimization methods to learn from past experience and detect useful patterns from large, unstructured and complex datasets [95]. These algorithms have a wide range of applications, including automated text categorisation [96], network intrusion detection [97], junk e-mail filtering [98], detection of credit card fraud [99], customer purchase behavior detection

[100], optimising manufacturing process [101] and disease modelling [102]. Most of these applications have been implemented using supervised variants [98][99][102] of the machine learning algorithms rather than unsupervised ones. In the supervised variant, a prediction model is developed by learning a dataset where the label is known and accordingly the outcome of unlabelled examples can be predicted [103].

2.11.1 Supervised learning

In supervised learning, the machine is trained with labelled dataset where output response or class for each input data vector is known. The assumption is that if the training data is large enough, a hypothesis that can perform well on the test data can be obtained. A simple example of supervised learning is curve-fitting problem. Given a set of input data, the machine is trained to generate the curved surface that best fits the training dataset, and during testing the machine is expected to correctly interpolate the new data over the curved surface. Feed-forward neural networks like perceptrons (adopting delta learning rule or perceptron learning rule), multilayer perceptrons (MLP, adopting back propagation), and constrained MLPs fall under this category [88].

2.11.2 Unsupervised learning

In unsupervised learning, the machine is expected to learn the patterns in the unlabelled input dataset by itself without any feedback from the environment. The problem can be stated as finding the patterns in input dataset to partition or cluster the training data into subsets in an appropriate way. Taxonomic problems, where designing efficient ways to group the data into meaningful clusters, fall under this category. Examples are Hebb and Hopfield networks (Hebbian learning), Kohonen networks/self-organizing maps, and (ART) networks/ART (competitive learning). Auto encoder is a simple network that is trained to produce what is given at the input, i.e. by setting the target output as the input. The network is trained to reproduce the input using gradient descent back propagation unsupervised learning method. Auto encoders are stacked to form a deep network that can be pre-trained using unsupervised learning to fix better initial weights and bias values [88].

2.11.3 Semi-supervised

Learning In semi-supervised learning, both labelled and unlabelled data are used for training the system. Typically, a small proportion of labelled data is used with a large amount of unlabelled data. This type of learning approach is usually adopted in problems where obtaining labelled data is very expensive [88].

2.11.4 Active learning

In active learning, the algorithm interactively queries the user to obtain the labels for the examples. This is used in scenarios where unlabelled data is abundant but labelling the data is expensive [88].

2.12 Supervised machine learning algorithms

At its most basic sense, machine learning uses programmed algorithms that learn and optimise their operations by analysing input data to make predictions within an acceptable range. With the feeding of new data, these algorithms tend to make more accurate predictions. Although there are some variations of how to group machine learning algorithms, they can be divided into four broad categories according to their purposes and the way the underlying machine is being taught.

In supervised machine learning algorithms, a labelled training dataset is used first to train the underlying algorithm. This trained algorithm is then fed on the unlabelled test dataset to categories them into similar groups. Using an abstract dataset for three diabetic patients, Figure 2.3 shows an illustration about how supervised machine learning algorithms work to categories diabetic and non-diabetic patients. Supervised learning algorithms suit well with two types of problems: classification problems; and regression problems. In classification problems, the underlying output variable is discrete. This variable is categorized into different groups or categories, such as ‘red’ or ‘black’, or it could be ‘diabetic’ and ‘non-diabetic’. The corresponding output variable is a real value in regression problems, such as the risk of developing cardiovascular disease for an individual [109]

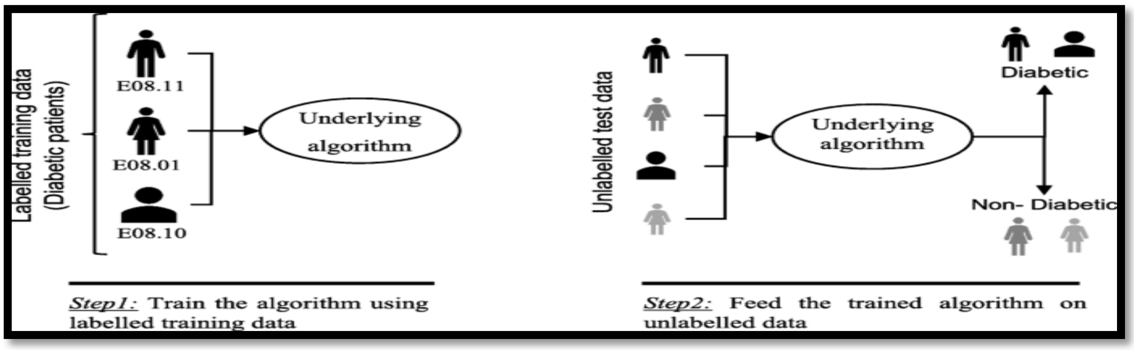

Labelled training data (Diabetic patients)
E08.11
E08.01
E08.10
Underlying algorithm
Step1: Train the algorithm using labelled training data
Unlabelled test data
Underlying algorithm
Diabetic
Non- Diabetic
Step2: Feed the trained algorithm on unlabelled data

Figure 2.3 An illustration of how supervised machine learning algorithms wor

2.12.1 Decision tree

Decision tree (DT) is one of the earliest and prominent machine learning algorithms. A decision tree models the decision logics i.e., tests and corresponds outcomes for classifying data items into a tree-like structure. The nodes of a DT tree normally have multiple levels where the first or top-most node is called the root node. All internal nodes (i.e., nodes having at least one child) represent tests on input variables or attributes. Depending on the test outcome, the classification algorithm branches towards the appropriate child node where the process of test and branching repeats until it reaches the leaf node [104]. The leaf or terminal nodes correspond to the decision outcomes. DTs have been found easy to interpret and quick to learn, and are a common component to many medical diagnostic protocols [105]. When traversing the tree for the classification of a sample, the outcomes of all tests at each node along the path will provide sufficient information to conjecture about its class. An illustration of an DT with its elements and rules is depicted in Figure 2.4.

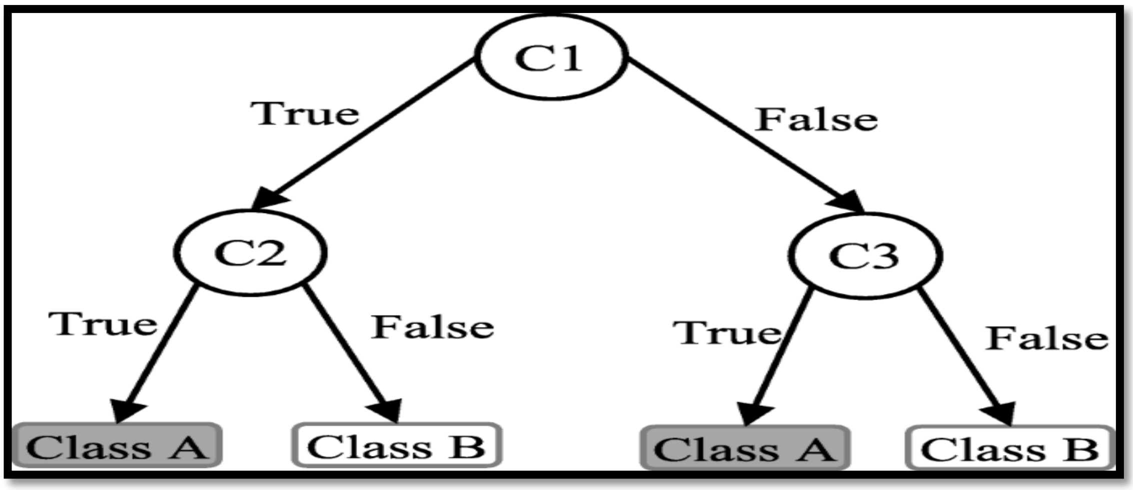


Figure 2.4 Decision tree

An illustration of a Decision tree. Each variable (C1, C2, and C3) is represented by a circle and the decision outcomes (Class A and Class B) are shown by rectangles. In order to successfully classify a sample to a class, each branch is labelled with either ‘True’ or ‘False’ based on the outcome value from the test of its ancestor node

2.12.2 Naïve Bayes

Naïve Bayes (NB) is a classification technique based on the Bayes’ theorem [106]. This theorem can describe the probability of an event based on the prior knowledge of conditions related to that event. This classifier assumes that a particular feature in a class is not directly related to any other feature although features for that class could have interdependence among themselves [107]. By considering the task of classifying a new object (white circle) to either ‘green’ class or ‘red’ class, Figure 2.5 provides an illustration about how the NB technique works. According to this figure, it is reasonable to believe that any new object is twice as likely to have ‘green’ membership rather than ‘red’ since there are twice as many ‘green’ objects (40) as ‘red’. In the Bayesian analysis, this belief is known as the prior probability. Therefore, the prior probabilities of ‘green’ and ‘red’ are 0.67 (40 ÷ 60) and 0.33 (20 ÷ 60), respectively. Now to classify the ‘white’ object, we need to draw a circle around this object which encompasses several points (to be chosen prior) irrespective of their class labels. Four points (three ‘red’ and one ‘green) were considered in this figure. Thus, the likelihood of ‘white’ given ‘green’ is 0.025 (1 ÷ 40) and the likelihood of ‘white’ given ‘red’ is 0.15 (3 ÷ 20). Although the prior probability indicates that the new ‘white’ object is more likely to have ‘green’ membership, the likelihood shows that it is more likely to be in the ‘red’ class. In the Bayesian analysis, the final classifier is produced by combining both sources of information (i.e., prior probability and likelihood value). The ‘multiplication’ function is used to combine these two types of information and the product is called the ‘posterior’ probability. Finally, the posterior probability of ‘white’ being ‘green’ is 0.017 (0.67×0.025) and the posterior probability of ‘white’ being ‘red’ is 0.049 (0.33×0.15). Thus, the new ‘white’ object should be classified as a member of the ‘red’ class according to the NB technique.

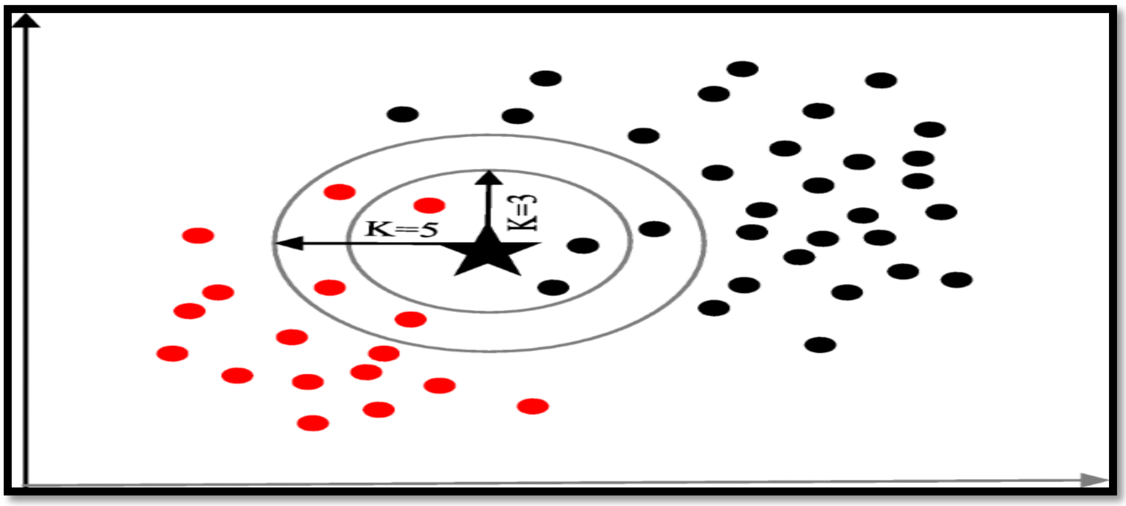


Figure 2.5 A simplified illustration of the K-nearest neighbor algorithm

When K = 3, the sample object ('star') is classified as 'black' since it gets more 'vote' from the 'black' class. However, for K = 5 the same sample object is classified as 'red' since it now gets more 'vote' from the 'red' class [109].

2.12.3 Hidden Markov Models

Hidden Markov model is derived from the Markov chain. While each state in the Markov chain represents a specific observable event, a state has a probabilistic function for observation in the hidden Markov model. Thus, the sequence of states is unobservable 'hidden' in the hidden Markov model. Modelling of a signal temporal variation problem is the main cause for using hidden states [109].

Belong to the Markov assumption, the probability of the state is depending just on the previous state. This assumption reduces the number of model parameters, thus the usage of the memory for modelling.

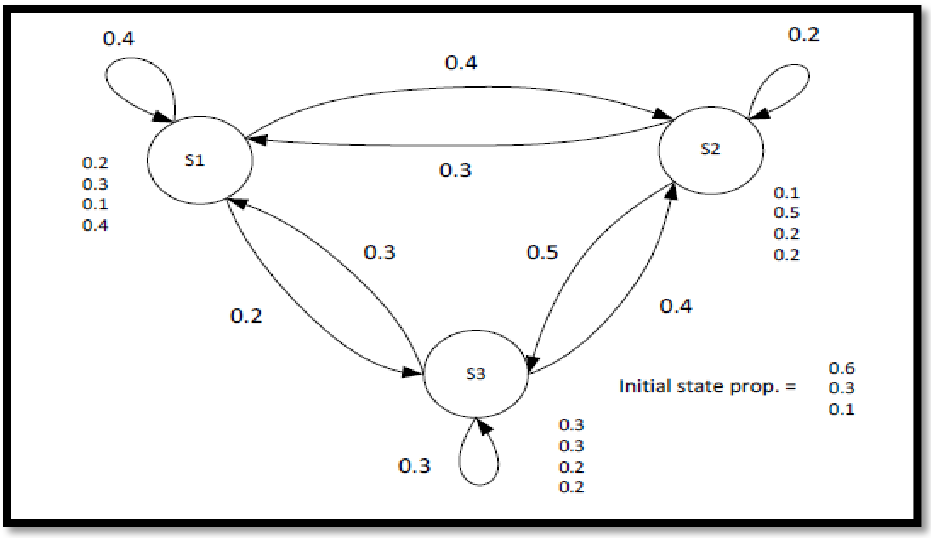


Figure 2.6 Example for a 3-states hidden Markov model

An example is shown in Figure 2.6 for more understanding of HMM. The model in the Figure has three states and four observation symbols, between each state couple there is a transition probability, and for each state there is a probability distribution for the observation symbols. For the initial state, there is a probability set represents the probability of each state to be the initial state.

We can conclude from the example that the definition of HMM needs to three probability distributions:

$\boldsymbol{A} = \{aij\}$ - A transition probability distribution, where aij is the probability of the transition from state i to state j [110].

$$aij = (st = j|st{-}1 = i) \ (2.9)$$

$\boldsymbol{B} = \{bi(k)\}$ - A matrix of output probability, where $bi(k)$ is the probability distribution of symbol ok observation in state i.

$$b(k) = P(Xt = ok|st = i) \ (2.10)$$

$\boldsymbol{\pi} = \{\pi i\}$ - Initial state distribution.

$$\pi i = (s0 = i) \ (2.11)$$

Because A, B and π are probability distributions, the summation of each distribution must equal to 1.

There are three problems must be solved to prepare the HMM for the real usage [109]. We can summarize these problems by three questions: How we can determine the probability of that the given HMM generates a given observation sequence? What is the best path of hidden states is in HMM that generates a given observation sequence? What is the proper parameters for the HMM to model a set of observation sequences?

The first question represents **the Evaluation problem**, which represented by $P(X|\Phi)$, where Φ is a given model and X is the observation sequence $X = (x1, x1, \ldots, xT\ )$ [110].

The simple solution for the evaluation problem is by generate all possible state sequences, then calculate the probability of each sequence and find the summation of these probabilities. This solution has an exponential computation (N_T) where T is the length of the observation sequence [110].

For computation simplicity, the forward algorithm is used. The algorithm benefit from the HMM assumption, where each state probability is depending just on the previous state and the observable symbol probability in the current state. Then it is possible to find the total probability with recursion on t, then it reduces the complexity to $O(N2T)$ [110].

The second problem is the Decoding problem. The decoding is helpful in many applications such as the segmentation and the searching in the continuous speech. The solution of this problem is similar to the evaluation problem solution but it instead of finding the summation of all possible sequences, just finding the sequence that score the maximum probability value. Viterbi algorithm is used for solving this problem, Viterbi is a form of the dynamic programing algorithms which partition the problem into small partial problems [110].

Learning problem is a complex problem because that the estimation of the model parameters cannot be determined analytically. Forward-backward algorithm depending on Expectation Maximization principle named Baum-Welch is the used algorithm for HMM learning [110].

2.13 Gaussian mixture models

GMMS are used for modelling continuous distribution components as parametric probability distributions (Gaussian or normal), and the entire dataset can be modelled using mixture of such distributions or Gaussians.

GMMs are powerful in forming smooth approximations over a large class of sample distributions. GMM-based HMMs or GMM/HMM system is the most used ML approach in ASR. A GMM/HMM system is represented by l D (p, A, B), where p is a vector of state prior probabilities; A D (ai,j) is the state transition matrix; B D {b1,. . .,bn} is the set of GMMS of state j. The HMM state is usually associated with a sub-segment of a phoneme in speech. A sentence is modelled by concatenating HMMs for the sequence of phones and GMM distribution is used to generate a vector in the HMM state [88].

2.13.1 GMM formulation

The spectral features extracted from speech are real valued but applying HMMs on continuous observations is not directly possible. Instead, the possible values of an observation feature vector o_t are assumed to be normally distributed. The observation likelihood function bj(ot) is represented as a Gaussian. Given a dataset, the mean and variance can be obtained from the data, but the state that corresponds to an observation is not known. Hence, a way is needed to assign each observation vector o_t to every possible state i, incorporating the probability that the HMM was in state i at time t. Let this probability be $Y_t(i)$. Each vector of observation is modelled as a multivariate Gaussian with diagonal covariance matrices, and Baum– Welch algorithm is used to estimate the probability and to compute the mean and the variance.

A mixture of Gaussians is needed to model the multidimensional function and they need to be trained. The usual procedure to train the mixture of Gaussians (GMMs) is to choose M the number of Gaussians and splitting the Gaussian into two and running the forward–backward algorithm to retrain the Gaussians. This process is repeated until M Gaussians are generated. Another approach is to do embedded training where each phone HMM embedded in an entire sentence is trained. Both word segmentation and alignment can be done as a part of the training process. Typically, CD phones are used, and decision-tree-based state tying is used to cluster the many states into various clusters [88].

2.13.2 Limitations of GMMs in GMM-HMM based acoustic models

In GMM-HMM acoustic models, GMMs are used to model the relationship between the states of the HMM and the acoustic input. These models have proven efficient in dealing with the acoustic variations related to speaker accents, pronunciation variations, and environmental noise, etc. In fact, due to these variations, modeling the state densities of HMM using a mixture of Gaussians is more accurate than using a single form of density function. In addition, the availability and the efficiency of the EM algorithm for estimating the parameters of the model have played an important role in the success of GMMs in HMM-based acoustic models. Hence, it was difficult to find a new method that can outperform GMMs.

Despite their efficiency, GMMs do suffer from several shortcomings that should be addressed. In [189], three of the major GMM problems are identified. First, GMMs assume that the data distribution is necessarily Gaussian. Second, the parameters of GMMs at each HMM state are not trained using the whole data of all states but with the subset associated with that state given the alignment. On the other hand, the number of GMM parameters needed to be estimated across all states is very big, especially in context-dependent acoustic models, and may require a large amount of training data [190]. Finally, techniques used for feature dimensionality reduction may significantly reduce the accuracy in estimating the GMM parameters due to the potential elimination of some useful information.

Another critical shortcoming of GMMs in acoustic modeling is that they may require a large number of diagonal Gaussians or full-covariance Gaussians to model highly nonlinear data; whereas other models exist and can fit such kind of data with only a few parameters [191]. In [192], it was argued that GMMs are also statistically inefficient at modeling high-dimensional data with componential structure. This inefficiency was attributed to the fact that for two significantly different sub-bands of independent patterns, when the first contains N different patterns and the second contains M different patterns, a GMM requires $N.M$ components to model such data. In fact, each data has only one single latent cause, and hence each component must fit both sub-bands. However, only $N + M$ components are necessary to explain such data for a model that uses multiple causes, in which each component is specific for a single sub-band. This shortcoming of GMM may affect the efficiency of GMM-HMM based ASR system where a large number of Gaussians at each HMM state must be estimated from a sub-set of the data derived from the alignment. Furthermore, in GMMs, every single Gaussian aims at modeling a partition from the input space. Having many Gaussians with independent means may lead to local generalization [193].

Many approaches have been proposed to overcome some of the limitations of GMMs. Since GMMs are typically trained as generative models using the EM algorithm, applying a subsequent stage of discriminative training was a first attempt to significantly improve the GMM-HMM acoustic models. The objective function of the discriminative training has a close relationship with the main goal of the ASR system. Maximum Mutual Information Estimation (MMIE) is one of the most common discriminate estimation methods. It aims at maximizing the separation between acoustic models by taking into account not only the likelihood of the training word strings given the labels, but also the probability of other possible word string hypotheses [194].

Using feed forward ANNs to replace GMMs in continuous density HMM for acoustic modeling was an alternative means to address the problem of GMMs discussed above. The introduction of an effective new procedure for learning deep neural networks has motivated researchers to apply deeper architecture for acoustic modeling [194].

On the other hand, it has been asserted that [191] the information embedded in speech can be represented with lower-dimensional data. However, GMMs are incapable of exploring highly correlated features. In consequence, they cannot handle the latent information from a large window of frames [194].

2.14 Artificial neural networks

One type of network sees the nodes as ‘artificial neurons. These are called artificial neural networks (ANNs) [111]. An artificial neuron is a computational model inspired in the natural neurons. Figure 2.7 shows Natural neurons receive signals through synapses located on the dendrites or membrane of the neuron. When the signals received are strong enough (surpass a certain threshold), the neuron is activated and emits a signal though the axon. This signal might be sent to another synapse and might activate other neurons [112].

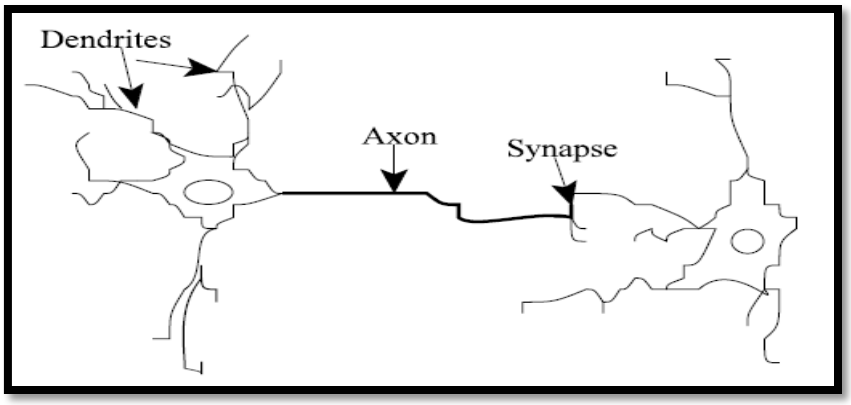


Figure 2.7 Natural neurons (artist’s conception) [112].

Tthe complexity of real neurons is highly abstracted when modelling artificial neurons. These basically consist of inputs (like synapses), which are multiplied by weights (strength of the respective signals), and then computed by a mathematical function which determines the activation of the neuron. Another function (which may be the identity) computes the output of the artificial neuron (sometimes in dependance of a certain threshold). ANNs combine artificial neurons to process information [112].

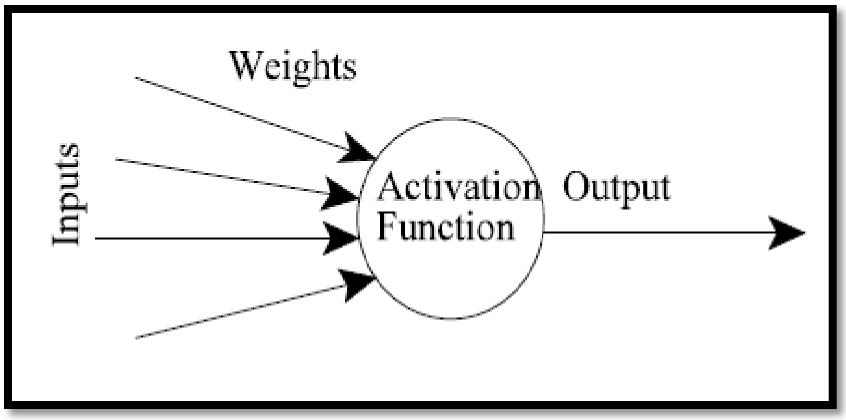


Figure 2.8 An artificial neuron [112]

The higher a weight of an artificial neuron is, the stronger the input which is multiplied by it will be. Weights can also be negative, so we can say that the signal is inhibited by the negative weight. Depending on the weights, the computation of the neuron will be different. By adjusting the weights of an artificial neuron, we can obtain the output we want for specific inputs. But when we have an ANN of hundreds or thousands of neurons, it would be quite complicated to find by hand all the necessary weights. But we can find algorithms which can adjust the weights of the ANN to obtain the desired output from the network. This process of adjusting the weights is called learning or training as shows in Figure2.8 [112][115].

The number of types of ANNs and their uses is very high. Since the first neural model [113] there have been developed hundreds of different models considered as ANNs. The differences in them might be the functions, the accepted values, the topology, the learning algorithms, etc. Also, there are many hybrid models where each neuron has more properties than the ones we are reviewing here. Because of matters of space, we will present only an ANN which learns using the backpropagation algorithm [112][115] for learning the appropriate weights, since it is one of the most common models used in ANNs, and many others are based on it [112].

Since the function of ANNs is to process information, they are used mainly in fields related with it. There are a wide variety of ANNs that are used to model real neural networks, and study behavior and control in animals and machines, but also there are ANNs which are used for engineering purposes, such as pattern recognition, forecasting, and data compression [114][115].

Applications of ANNs:

1. Control (such as cars in ALVINN).
2. Recognize/Classify (written/spoken words).
3. Predict (trends).

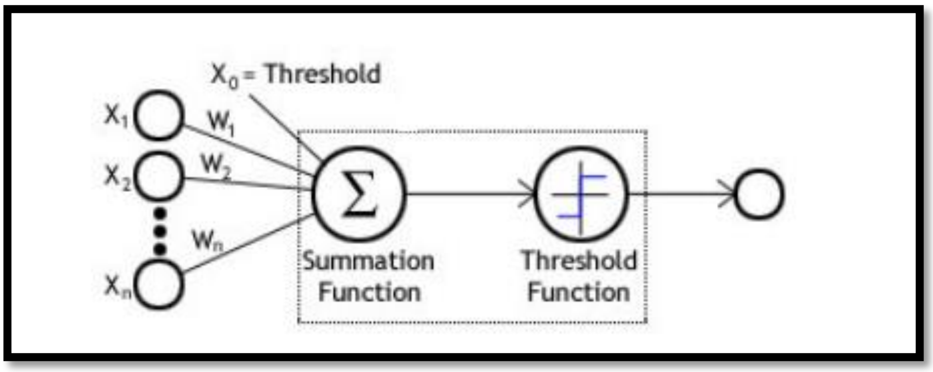


Figure 2.9 ANN unit [115]

In the Figure 2.9 shown above, there are n input units (Xi) connected with n links (Wi). It also has one output. The ANN unit is composed of two main parts: the first part sums the input and sends it to the threshold function. If the activation is greater than 0 then the unit activates and sends a “1” as the output, otherwise it sends a 0 (or –1). The X0 can be set to any value so that instead of tuning the threshold function to activate at some fixed-point Y, X0 can be set to -Y [115].

Two commonly used types of ANNs are Time-Delay Neural Networks (TDNNs) [116] and recurrent neural networks [117][118].

2.15 Deep learning methods

Deep learning algorithms have recently emerged from machine learning and soft computing techniques. Since then, several deep learning algorithms have been recently introduced to scientific communities and are applied in various application domains. Today the usage of DL has become essential due to their intelligence, efficient learning, accuracy and robustness in model building.

Convolutional neural network (CNN) Recurrent neural network (RNN), Denoising autoencoder (DAE), deep belief networks (DBNs), Long Short-Term Memory (LSTM) are the most popular deep learning methods have been widely used. In this section, the description of each method is described along with the notable applications.

2.15.1 Convolutional neural network

CNN is one of the most known architectures of DL techniques. This technique is generally employed for image processing applications. CNN contains three types of layers with different convolutional, pooling, and fully connected layers Figure 2.10 In each CNN, there are two stages for the training process, the feed-forward stage, and the back-propagation stage. The most common CNN architectures are ZFNet [119], GoogLeNet [120], VGGNet [121], AlexNet [122], ResNet [123].

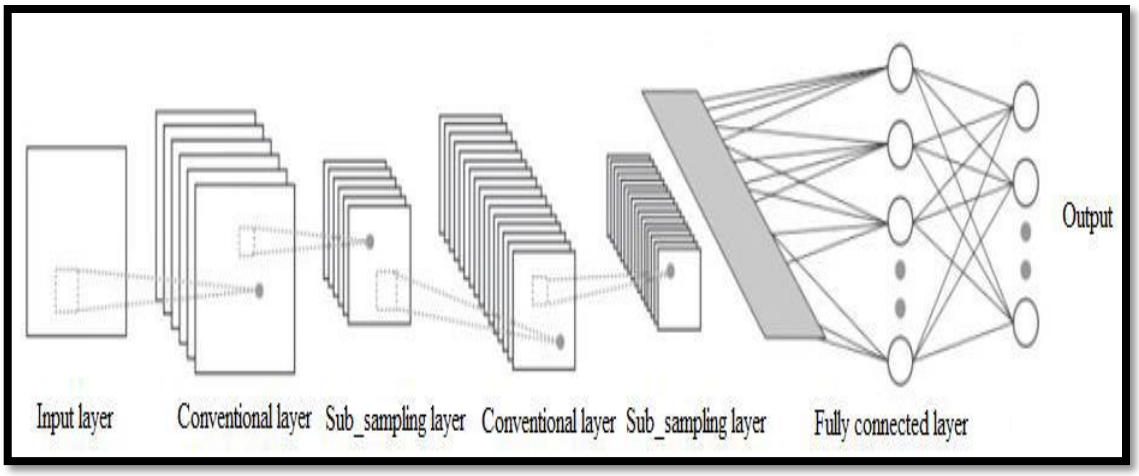


Figure 2.10 CNN architecture

Although CNN is primarily known for image processing applications, the literature includes other application domains, e.g., energy, computational mechanics, electronics systems, remote sensing, etc as shown in table 2.4.

Table 2.4 Applications of CNN

Reference	**Application**
Bhatnagar et al. 2019 [124]	Prediction of aerodynamic flow
Nevavuori et al. 2019 [125]	Crop yield prediction
Ajami et al. 2019 [126]	Advanced image processing
Lossau et al. 2019 [127]	Motion estimation and correction of medical imaging
Kong et al. 2020 [128]	Condition monitoring of wind turbines

2.15.2 Recurrent neural networks

RNN is designed to recognize sequences and patterns such as speech, handwriting, text, and such applications. RNN benefits cyclic connections in the structure which employ recurrent computations to sequentially process the in-put data [129]. RNN is basically a standard neural network that has been ex-tended across time by having edges which feed into the next time step instead of into the next layer in the same time step. Each of the previous inputs data are kept in a state vector in hidden units, and these state vectors is utilized to compute the outputs Figure.2.11 presents the architecture of RNN.

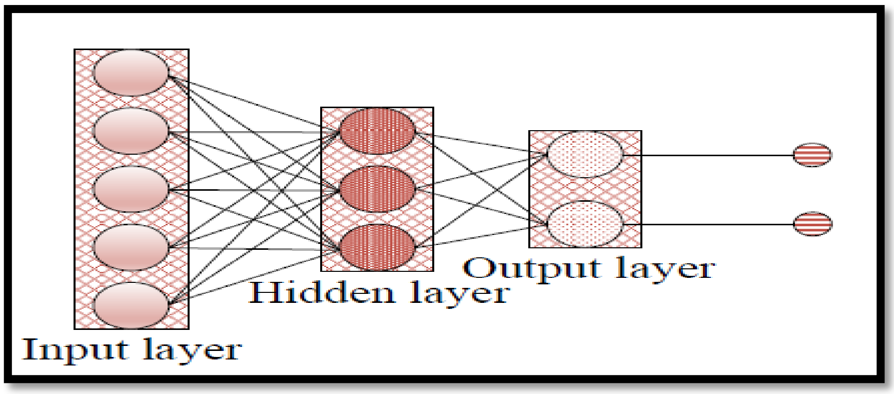


Figure 2.11 RNN architecture

RNN is relatively newer deep learning method. Therefore the application domains are still young and plenty of rooms remains for research and exploration. The energy, hydrological prediction, expert systems, navigation, and economics are the current applications reported in the literature as shown in table 2.5.

Table 2.5 Applications of RNN

Reference	Application
Zhu et al. 2019 [130]	Wind speed prediction
Pan et al. 2019 [131]	Tropical cyclone intensity prediction
Bisharad et al. 2019 [132]	Music genre recognition
Zhong et al. 2019 [133]	Ship Trajectory Restoration
Jarrah et al. 2019 [134]	Stock price trends predict

2.15.3 Denoising autoencoder

DAE has been extended from AE as asymmetrical neural network for learning features from noisy datasets. DAE consists of three main layers, including input, encoding, and decoding layers [76]. DAE is able to be aggregated for taking high-level features. Stacked Denoising Autoencoder (SDAE), as an un-supervised algorithm, is generated by the DEA method, which can be employed for nonlinear dimensionality reduction. This method is a type of feed-forward neural network and employs a deep architecture with multiple hidden layers and a pre training strategy [135][136]. Figure 2.12 presents the architecture of DEA architecture.

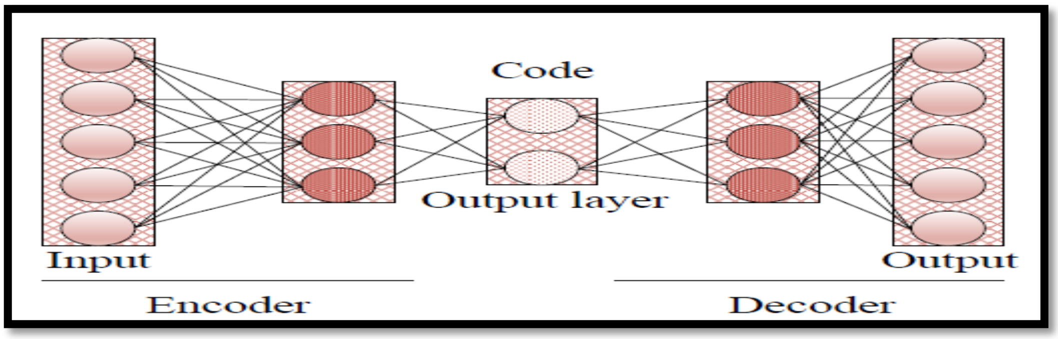


Figure 2.12 DEA architecture

DEA is slowly starting to be known among researchers as an efficient DL algorithm. DEA has already been used in various application domains with promising results. The energy forecasting, cybersecurity, banking, fraud detection, image classification, and speaker verification are among the current popular applications of DEA as shown in table 2.6.

Table 2.6 Applications of DEA

Reference	Application
Chen et al. 2019 [137]	Improving the cyberphysical systems
Liu et al. 2019 [138]	Electric load forecasting
Nicolai et al. 2018 [139]	Laser-based scan registration
Yue, et al. 2018 [140]	Collaborative Filtering
Roy et a. 2018 [141]	Noisy image classification
Tan et al. 2018 [142]	Robust Speaker Verification

2.15.4 The deep belief networks

DBNs are employed for high dimensional manifolds learning of data. This method contains multiple layers, including connections between the layers except for connections between units within each layer. DBNs can be considered as a hybrid multi-layered neural network, including directed and undirected connections. DBNs contains restricted Boltzmann machines (RBMs) which are trained in a greedy manner. Each RBM layer communicates with both the previous and subsequent layers [136][143][144]. This model is consisting of a feed-forward network and several layers of restricted Boltzmann machines or RBM as feature extractors [145]. A hidden layer and visible layer are only two layers of an RBM, Figure2.13 presents the architecture of DBN architecture [146].

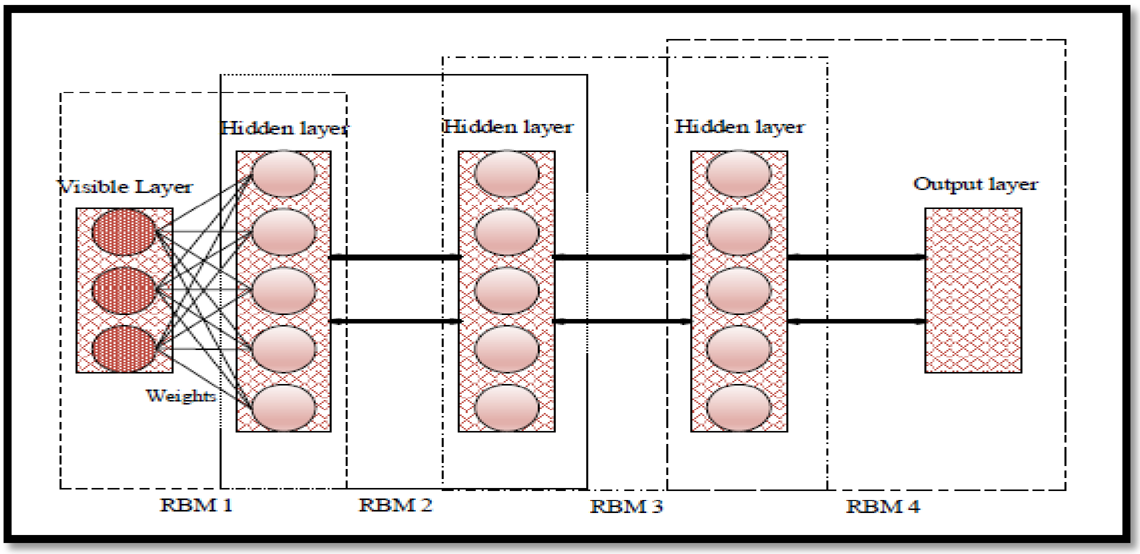


Figure 2.13 DBN architecture

DBN is one of the most reliable deep learning methods with high accuracy and computational efficiency. Thus, the application domains have been divers, including exciting application in a wide range of engineering and scientific problems. Human emotion detection, time series prediction, renewable energy prediction, economic forecasting, and cancer diagnosis have been among the public application domains as mentioned in table 2.7.

Table 2.7 Applications of DBN

Reference	**Application**
Hassan et al. 2019 [147]	Human emotion recognition
Cheng et al. 2019 [148]	Time series prediction
Yu et al. 2019 [149]	wind speed prediction
Zheng et al. 2019 [150]	Exchange rate forecasting
Ahmad et al. 2019 [151]	Automatic Liver Segmentation
Ronoud et al. 2019 [152]	Breast cancer diagnosis

2.15.5 Long short-term memory

LSTM is an RNN method which benefits feedback connections to be used as a general-purpose computer. This method can of for both sequences and pat-terns recognition and image processing applications. In general, LSTM contains three central units, including input, output, and forget gates. LSTM can control on deciding when to let the input enter the neuron and to remember what was computed in the previous time step. One of the main strengths of the LSTM method is that it decides all these based on the current input itself, Figure2.14 presents the architecture of LSTM architecture.

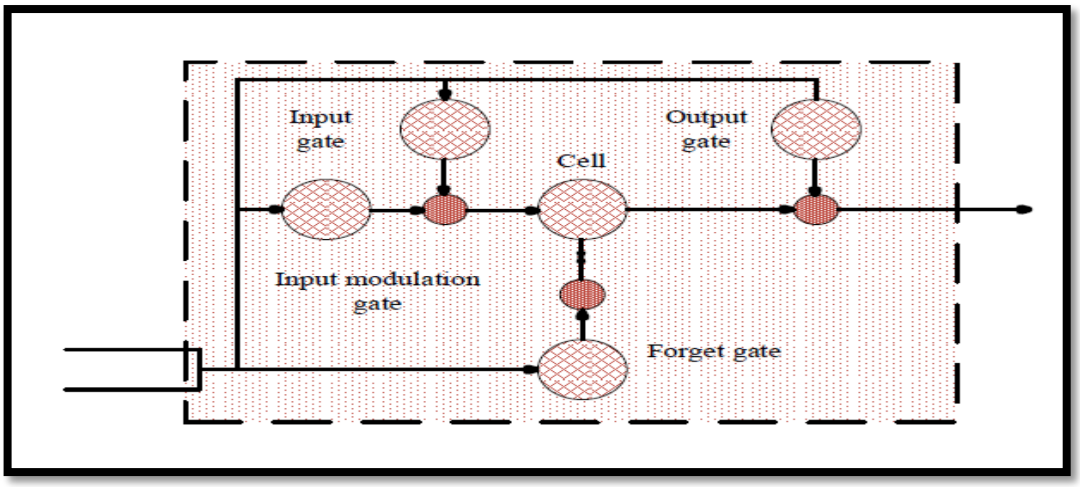


Figure 2.14 LSTM architecture

LSTM has shown great potential in environmental applications summarized in table 2.8, e.g., geo-logical modeling, hydrological prediction, air quality, and hazard modeling. Due to the generalization ability of the LSTM architecture, it can be suitable for many application domains. Energy demand and consumption, wind energy industry, and solar power modeling are the other application domains of LSTM. Further investigation is essential to explore the new deep learning methods and explore the application domains, as it has been done for machine Learning methods [153][154][155][156] as shown in table 2.8.

Table 2.8 Applications of LSTM

Reference	Application
Ghimire et al. 2019 [157]	Solar radiation forecasting
Liu 2019 [158]	Volatility forecasting
Hong et al. 2019 [159]	Fault prognosis of battery systems
Krishan 2019 [160]	Air quality prediction
Zhang et al. 2019 [161]	Structural seismic prediction
Hua et al. 2019 [162]	Time Series Prediction
Zhang et al. 2019 [163]	Wind turbine power prediction
Vardaan et al. 2019 [164]	Earthquake trend prediction

2.16 Deep neural networks

A Deep Neural Network is defined as a feed-forward Artificial Neural Network (ANN) that has multiple layers of hidden units stacked on top of each other between the input and the output layers [165][166] The main goal of DNNs is the discrimination between classes; however, they are often generatively pretrained to initialize their parameters [167].

2.16.1 Motivations for training DNNs

ANNs are connectionist models composed of highly connected computational units (neu- rons) organized in a specific manner that makes them able to perform a nonlinear trans- formation to the input data. In a feed-forward ANN, the units in one layer are connected to the units in the next layers using unidirectional connections. In fact, each hidden unit j, in layer l, is connected to all the units in the layer l 1, and has an activation function, typically logistic, that is responsible for summing the weighted input from the layer below, and for processing the result to the layer above as a single output. The output of the units i can be calculated using the following equation [166] as shown in Figure 2.15.

$$o_i = logistic(\sum_i^N w_{ij} x_i + b_j) \qquad \textbf{(2.13)}$$

where N is the number of units in the layer $l-1$, w_{ij} is the weight of the connection between unit j and unit i of the layer below, and b_j is the bias of the hidden unit j.

$$logistic(x) = 1/(1 + e^{-x})) \qquad \textbf{(2.14)}$$

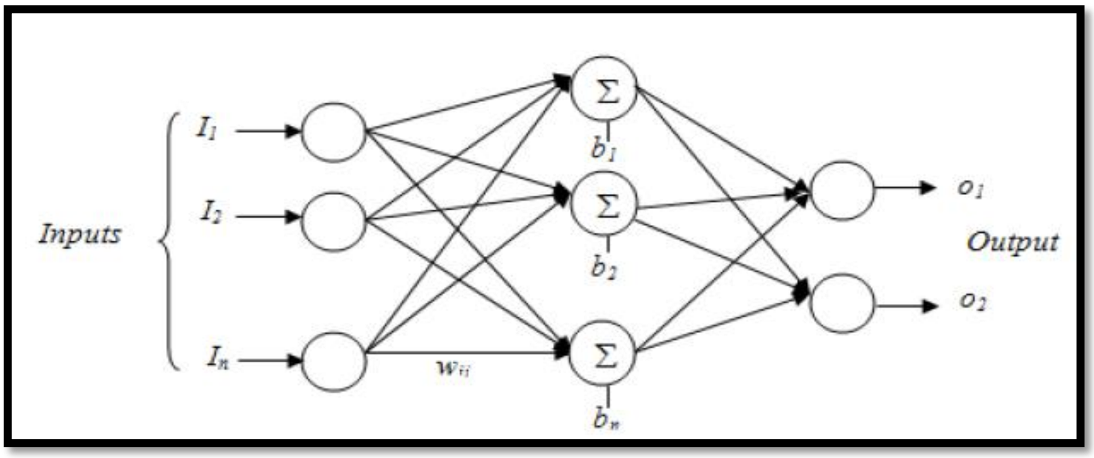


Figure 2.15 Feed-forward ANN with multiple inputs, two outputs and one hidden layer

ANNs are discriminatively trained in a supervised way using the backpropagation learning algorithm that measures the error between the network outputs and the desired outputs and updates the network parameters (the weights and biases) by applying the gradient descent optimization technique with the aim of reducing the local error [167].

The motivation behind designing ANNs is to mimic the computational power of the human brain. In fact, ANNs have shown great success in the areas of machine learning, pattern recognition and classification. Moreover, they have proven a notable strength in modeling nonlinear systems. They have been used in a wide range of applications including robotics, medical, communication systems, data mining, sales and marketing [167]. However, only shallow-structured architectures were exploited, and researchers have failed, for many years, to train networks with more than one hidden layer.

The main difficulty that arises is the" curse of dimensionality", which states that the variations in the input data grow exponentially with a linear increase in their dimensionality [167][169]. In consequence, the network may require a huge number of training data in order to model all these variations, and the learning process is inherently very complex and computationally very expensive.

This problem begs the need for different new approaches to train deep-layered neural networks. The clue might lie in understanding how the human brain processes this vast amount of information, received in a very high frequency, with such an ease. It has been argued that the brain processes the sensory input through multiple layers in a hierarchical fashion. Each layer extracts new features from its inputs and creates a higher-level representation to pass it on to the next higher level [167][169].

Thus, designing deep architectures is an attempt to emulate the representational power and the information modeling ability of the human brain. Learning a DNN to extract useful features from its input data and eliminate unnecessary variations is considered the key solution to preventing the curse of dimensionality. As a result, the number of training examples required for learning high dimensional data would drastically decrease. A DNN can achieve this features extraction and transformation using hierarchical processing of the input data akin to what the brain does. The idea is to feed the input data to the first layer which learns to extract a new representation of its input and provides its output to the layer above. The next layer then derives a new higher-level representation from its inputs. The process continues all the way to the top layer, resulting in meaningful features and a new description of the input data.

Despite the big development in the area of computer hardware and machine learning algorithms, researchers have failed to train a network with many layers of nonlinear feature detectors. The backpropagation learning algorithm has been the efficient and traditional way for discriminatively training ANNs due to its ease of implementation. However, attempts to train a DNN that learns to derive features and build internal representations from complex inputs using the back-propagation algorithm have led to dismal results. In fact, training deep networks seems to be very difficult and hard to optimize; these net- works perform worse than shallow ones in terms of generalization [168]. As networks become deeper, they tend to get stuck more often in poor local minima. This trend is attributed to the random initialization of the weights in the back-propagation algorithm. Moreover, the large number of layers drives the network to learn rare dependencies and variations in the input data, which makes it very prone to the problem of overfitting [165] [166][176] [170].

Due to these generalization problems, learning deep-layered architectures for tackling very complicated applications fell out of favor amongst both researchers and industry. Hence, many experiments were made to overcome these problems and to find a good learning algorithm able to mimic the robustness of the brain in representing information, and able to produce good classification results in terms of time and accuracy.

2.16.2 DNN-HMM for acoustic modeling

Acoustic modeling in speech segmentation refers to the process of creating statistical representations for the feature vector sequences computed from the speech waveform. As discussed in chapter one, these acoustic models should be robust and capable of modeling not only the variability intrinsic in human speech but also the noise that characterizes the background environment. In fact, the acoustic models should determine the probability P (A W) of an acoustic vector sequence A, given that the uttered word sequence is W . Due to the large number of different possible pairings of W with A, sophisticated statistical models are needed [171]. HMMs are typically the most useful acoustic models in speech segmentation.

In ASR, HMMs are typically used to model the sequential structure of speech signals [55, 58], with Gaussian densities at each state in order to model the local spectral variability in the sound wave [172]. Using the Expectation Maximization (EM) algorithm, GMMs are easy to fit to data. Indeed, they have proved effective in modeling the relationship between HMM states and the acoustic input represented]more often as a set of MFCCs. With enough components, GMMs can accurately model the probability distribution over the acoustic input feature vectors associated with each state of an HMM [166]. Nonetheless, GMMs in HMM-based acoustic models suffer from some major limitations that have motivated researchers to find other models.

2.16.3 DNN-HMM hybrid architecture for acoustic modeling

Early attempts to use feed-forward ANNs as a substitute for GMMs in HMM-based acoustic models offered several advantages over GMMs. In [173], three major ones were reported. First, in contrast to GMMs, ANNs do not require detailed assumptions about the data distribution in order to estimate the posterior probabilities over HMM states. Second, when using ANNs, it is possible to apply different types of data, including the combination of discrete and continuous features. Finally, yet importantly, ANNs make use of all the training data to model their parameters. Furthermore, ANNs have shown great capabilities at modeling highly nonlinear data.

The effectiveness of the new deep learning algorithms developed recently has strengthened the idea that neural networks are suitable for acoustic modeling. As reviewed in the previous. the algorithm for training deep neural networks consists of a two-step's procedure: first, a generative pre-training step that aims at initializing the weights of the network and extracting higher new representations of the input at each layer, followed by a discriminative fine-tuning step with the backpropagation learning algorithm. DBNs have been the most common technique for generatively pre-training deep networks where each layer extracts a new higher representation and new structures from the input. This pre training stage has the great advantage of reducing over fitting and reducing the computation cost during the discriminative fine-tuning with the back-propagation algorithm, a cost that was considered the major impediment with deep networks. It also helps the network to rapidly converge towards better local minima. In earlier literature, DNNs generatively pre trained as DBNs were often called DBNs. To eliminate any ambiguity, Hinton et al. introduced the new name DBN-DNNs in [166].

In combining pre-trained DNNs with HMMs within a single hybrid architecture for acoustic modeling, researchers intended to combine the representational power of DBN-DNNs and the sequential modeling capability of HMMs.

2.16.4 Interfacing DNNs with HMMs

To interface DNN with HMM for modeling acoustic data, the network is trained to estimate the probability distribution over the states of the HMM. Considered as static classifiers with fixed dimensionality input vectors, DNNs are unable to perform sequence segmentation with variable dimensionality of the inputs and outputs such as speech segmentation [170]. However, HMMs are the most powerful tool that can handle sequential patterns using dynamic programming. Thus, combining HMMs and DNNs fruitfully takes advantage of both static and sequential pattern recognition, which makes such hybrid architecture very useful for speech segmentation.

DNN trained to predict the posterior probability over Monophone HMM states. The training data consist of a window of n successive frames of speech coefficient. A network composed of many layers of nonlinear units was first generatively pre-trained using RBMs and contrastive divergence to extract new features of the input at each layer. Then, the network was discriminatively trained with the back-propagation algorithm to predict the label of the central frame as shown in Figure 2.16. In fact, the output layer that represents the states of the HMMs provides a probability distribution over the possible labels of the central frame. Figure 2.16 illustrates the architecture of DNN for phone segmentation.

As described in the second chapter, an HMM is defined by 3 parameters:
the initial state distribution $\pi_i = P\ (s_0 = i)$,
the transition probability $a_{ij} = P\ (s_t = j\ s_{t-1} = i)$, which is the probability of taking a transition from state i to state j at time t,
the emission probability $b_j(x_t) = P\ (x_t|s_t = j)$ defined as the probability that the state j generates the acoustic observation x_t at time t. In GMM-HMM, this emission probability is estimated using the GMMs.
In DNN-HMM, DNN is used instead of GMM to estimate the emission probability at each state. In phoneme segmentation, the DNN produces the posterior probability, P (st|xt), of the mono-phone HMM states given the acoustic observation xt. However,
p(xt|st) = p(st|xt). p(xt)/p(st).

The probability p(st) can be estimated from an initial state-level alignment on the coustic training data using a Viterbi decoder. The probability p(xt) is assumed to be” independent of the word sequence and can be ignored during decoding" [175]. Hence, to get the emission probability P (xt|st), the posterior probability p(st|xt) should be divided by the prior p(st). However, it is asserted that this division may not provide improvement in segmentation accuracy under some conditions [175].

The DNN-HMM hybrid architecture can also be used for acoustic modeling in continuous speech segmentation task, as presented in [175][176] In contrast to the context-independent (CI) DNN-HMM hybrid architecture described above for phoneme segmentation, the DNN can be trained to predict probability distributions over tri-phone HMMs, forming thus a context-dependent (CD) DNN-HMM hybrid architecture for large-vocabulary continuous speech segmentation task. The idea is to use senones as modeling units rather than mono- phones. The method involves two main steps [175]. First, the Viterbi algorithm is used to generate the senone-level alignment on a tied tri-phone GMM-HMM baseline. Then, the DNN-HMM is trained to predict the senones in each frame or sequence of frames. It has been proven that the better the baseline system used during forced alignment, the better the final results of the CD-DNN-HMM system [175] [166].

DNN-HMM hybrid architecture has shown successful results and good segmentation per- formance in both isolated and continuous speech segmentation tasks. In fact, DBN-DNNs have proven capability to outperform GMMs in acoustic modeling.

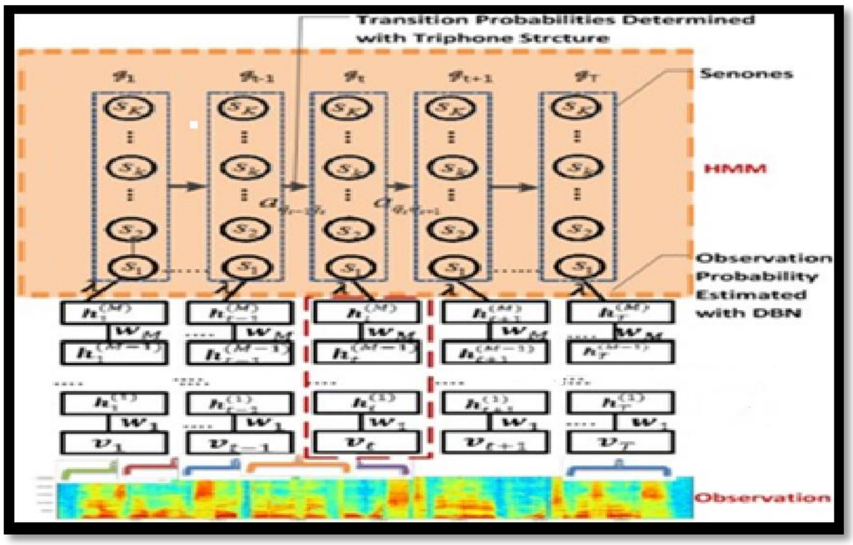

Transition Probabilities Determined with Triphone Strcture
Senones
HMM
Observation Probability Estimated with DBN
Observation

Figure 2.16 Interfacing DNN and HMM for continuous speech [177]

2.16.5 Time delay neural network

In a time delay neural network, the temporal context is modeled by using a hierarchical architecture. Each layer in a TDNN operates at a different temporal resolution. The outputs of the activation from previous hidden layer are spliced as the input of the current layer. Therefore, the current layer operates at a much wider context, compared with the previous layer. As we go to higher layers of the network, increasingly wide context is seen by the network.

Like Convolutional Neural Networks (CNNs [200]), the transforms in the same layer of a TDNN are tied across time to reduce the number of parameters and make the transformation invariant to time shift of the input [200]. TDNNs are seen as a precursor to the CNNs. proposed a method to subsample the TDNN network. The splicing configuration {-1,1} means that we splice the input at current time step minus 1 and the current time step plus 1 (i.e. the current frame is dropped). Sub-sampling reduces the dimension of the input and thus the model size.

The overall input contexts of TDNNs are limited, for example, asymmetric context windows of up to 16 frames in past and 9 frames in the future are investigated in [200]. The success of TDNNs indicates that the most valuable information for the recognition of the current frame lies in a relatively narrow context. This is true even when recurrent models are used. Truncated Back Propagation Through Time.

2.16.6 What distinguishes DNN- HMM from GMM- HMM

DNNs have proven efficient in acoustic modeling and shown results competitive with GMMs. However, many differences exist between GMMs and DNNs, which have made DNNs more powerful for speech segmentation. DNNs are more efficient at modeling highly non-linear data [166]. with many hidden units, DNN can easily model multiple simultaneous events within one frame or window. However, doing so is very difficult for GMM because each data point is assumed to be generated by a single Gaussian in the mixture. In addition, DNN can efficiently be trained using multiple frames of acoustic features; whereas GMM, which requires uncorrelated data, is unable to exploit multiple frames [166].

Despite the advantage of the EM algorithm for training GMMs, which can be easily parallelized on a cluster machine [166], DNNs present many advantages that have made them the most prominent technique for acoustic modeling in speech segmentation. In [178], the success of DNNs over GMMs in acoustic modeling was attributed to three main facts. First, neural networks are very flexible at modeling the data without any preliminary assumptions about its distribution. In addition, DNNs are exponentially more compact than GMMs due to the" distributed representation" of the input, which can be explained by the big number of units that together can represent a single input vector. Second, the deep architecture of DNNs that are composed of many nonlinear layers plays an important role in improving the flexibility and the modeling capacity of the network. In fact, extracting higher representation and a highly non-linear statistical structure at each layer has a great impact on improving the segmentation accuracy of the system. Finally, the generative pre-training phase using an unsupervised layer-wise training algorithm represents the most important advantage of DBN-DNNs. On one hand, this phase aims at extracting regularities in the input data so as to adjust the weights for a good generative model; on the other hand, it initializes the weights of the network for a good discrimination between the labels during the supervised learning stage. Indeed, the pre-training phase plays an important role in reducing over fitting, improving the generalization, and reducing the computation during the discriminative fine-tuning with back-propagation learning algorithm.

Due to these advantages, generatively pre-trained DNNs are competitive to GMMs for acoustic modeling.

2.17

2.18 Related work on Deep Neural Network for Acoustic Models

In the past few years, research has been done in the field of speech knowledge research. Deep learning has arisen as a recent area of machine learning and use in speech systems. Deep Neural Networks having GPU cards with high compute capability are used for modeling for the speech systems.

The HTK toolkit is provides DNN for speech systems‘ KALDI toolkit is having good documentation of DNNs script that is useful in the field of speech knowledge research. So, the KALDI toolkit is preferable over other toolkits for the implementation of speech models based on DNN. The research work on the KALDI ASR toolkit using DNN was studied which shows the experimental results of sequence discriminative training based on n-gram model [202][203][204] They recommend the training of a DNN based acoustic model of continuous speech recognition system for large vocabulary. The researchers evaluated the recognition of the phone on the Italian corpus with the help of the KALDI toolkit [205]. Even then the obtained results using DNNs were not superior to the results of the baseline method. Cosi explains that it's because of insuffi cient size of corpus that failed to tune the DNN architecture for language corpus. Researchers performed a case study on the Analysis of ASR for the diff erent Indian Languages [206]. In this paper, the significance of Punjabi language has been analyzed so that relevant research work can be implemented for Punjabi language. The HTK and SPHINX Speech recognition toolkits were used by them. Feature extraction techniques i.e., MFCC and Extended MFCC have also been discussed. M. Dua and Aggarwal used the concept of HMM for speech command recognition (speaker-dependent) using the Punjabi Language and turned a machine to an intelligent one that responded to the instructions in Punjabi Language [207].

In many recent papers, it was shown that DNN-HMM models outperform traditional GMM-HMM models. In [208], context-depended model based on a deep belief network for large-vocabulary speech recognition is presented. Deep belief networks have un-directed connections between the 2 top layers and directed connections to all other layers from the layer above. In that research, a hybrid DNN-HMM architecture was used; it was shown that DNN-HMM model can outperform GMM-HMMs, and the authors have achieved a relative sentence error reduction of 5.8 %. In [209], context-depended DNNs-HMMs (CD-DNN-HMMs) are described. CDDNN-HMMs combine ANN-based HMMs with tied-state triphones and deep-belief network pre-training. Efficiency of the models was evaluated on the phone call transcription task. The application of CD-DNN-HMMs has reduced the word error rate (WER) from 27.4 % to 18.5 %. An application of the tandem approach to acoustic modeling is presented in [210]. The input of the network was a window of successive feature vectors. Training of the network was performed according to the standard procedure that is used for a hybrid DNN-HMM system. Then extracted features were fed to the GMM-HMM system. The training was performed according to the standard expectation-maximization procedure. The authors have obtained a relative WER reduction of 31 % over baseline MFCC and PLP acoustic features with the context-independent models. In [211], the possibility of obtaining the features directly from DNN without a conversion of output probabilities to features suitable for GMM-HMM system was researched. Experiments with the use of a 5-layer perceptron in a bottle-neck layer were conducted. After training the DNN, the outputs of the bottle-neck layer were used as features for GMM-HMM system for speech recognition system. There was obtained the reduction of WER comparing to the system with probabilistic features, as well as the reduction of model size because only a part of the network was used. Research of DNN for acoustic modeling for large vocabulary continuous speech recognition (LVCSR) was also presented in [212]. In this paper, the authors have conducted an empirical investigation on what aspects of the DNN-based AM design are most important for performance of a speech recognition system. It was shown that increasing model size and depth is effective only up to a certain point. In addition, a comparison of standard DNNs, convolution NNs and deep locally untied NNs was made. It was found out that deep locally untied NNs perform slightly better. In [213], the KALDI toolkit was used for DNN-based children speech recognition for Italian. Karel's and Dan's DNN training was explored. Speech recognition results obtained using the Karel's implementation were slightly better than the Dan's DNN, but both implementations significantly outperformed non-DNN configuration. The KALDI toolkit was used for Serbian speech recognition in [214]. The DNN models were trained using the Karel's implementation on a single CUDA GPU. Depending on the test set a relative WER reduction of 15–22 % comparing to the GMM-HMM system was obtained. In [215], KALDI was used in conjunction with PDNN (Python deep learning toolkit) developed under Theano environment (http://deeplearning.net/software/theano/). The authors used KALDI for training GMMs. DNN was trained with the help of PDNN, and then obtained DNN models were loaded into KALDI for speech recognition. Four receipts were described in [215]: DNN Hybrid, Deep Bottleneck Feature (BNF) Tandem, BNF+DNN Hybrid, convolution NN Hybrid.

Chapter III

Research Methodology

In this chapter discuss the overall design of the proposed models, Gaussian mixture models, GMM formulation, Limitations of GMMs in GMM-HMM based acoustic models, DNN-HMM hybrid architecture for acoustic modeling, Interfacing DNNs with HMMs, Time delay neural network, Automatic segmentation model implementation using HTK, the standard KALDI receipt for DNN-based acoustic modeling steps, Implementation of automatic segmentation model at phoneme level using KALDI toolkit.

3.0 Introduction

Investigations of combining artificial neural networks (ANNs) and hidden Markov models (HMMs) for acoustic modeling were started between the end of the 1980s and the beginning of the 1990s [179]. At present the usage of ANNs in automatic speech recognition (ASR) becomes very popular because of increasing performance of computers. For acoustic modeling, ANNs are often combined with HMMs using hybrid and tandem methods [179]. In the hybrid method, ANNs are used for estimating the posterior probabilities of an HMM state. In the tandem method, outputs of ANNs are used as an additional stream of input features for HMM-GMM (Gaussian Mixture Models) system.

In this research, present a study on deep neural network (DNN) based acoustic models (AMs) for Quran verses phoneme segmentation. For training and testing the model and used the open-source KALDI toolkit [180]. The KALDI software is written in C++ and based on the Open FST library and uses BLAS and LAPACK libraries for linear algebra. There are two implementations of DNNs in KALDI. The first one is Karel's implementation [181]. It supports Restricted Boltzmann Machines (RBM) pre-training, stochastic gradient training using graphics processing units (GPU), and discriminative training. The second implementation is Dan's implementation [182]. It does not support Restricted Boltzmann Machine pre-training; instead, a method like the greedy layer-wise supervised training [183] or the "layer-wise backpropagation" [184] is used. For the given research, we have chosen the Karel's implementation.

3.1 Overview of the theoretical model

3.1.1 HMMS for phonetic segmentation

The objective here is to build on the extensive knowledge and infrastructure available in the speech field to discover alternative phoneme pronunciations for words. The sampling rate is 16 ksps, and analysis window is 25.6 msec (about 410 samples), with consecutive frames overlap by 10 msec. Each window is pre-emphasized and is multiplied by a Hamming window [185]. The basic feature vector uses the Mel Frequency Cepstrum Coefficients MFCC. The Mel-frequency scale is linear frequency spacing below 1000 Hz and a logarithmic spacing above 1000 Hz. The MFCCs are obtained by taking the Discrete Cosine Transform (DCT) of the log power spectrum from Mel spaced filter banks. The system uses a 12-coefficients basic feature vector. The basic feature vector is usually normalized by subtracting the mean over the sentence utterance. x (0) represents the log Mel spectrum energy and is used to derive other feature parameters. The basic feature vector is highly localized. To account for the temporal properties, 3 other derived vectors are constructed from the basic MFCC coefficients: a 40-ms and 80-ms differenced MFCCs (24 parameters), a 12-coefficient second order differenced MFCCs, and 3–dimensional vector representing the normalized power (log energy), differenced power, and second-order differenced power The HMM shown in Figure 3.1 to represent the speech phonemes. The model, known as Bakis model, has a fixed topology consisting of 3 emitting sates and one output state. Output probability distributions in HMM states are modeled with mixtures of 8 diagonal covariance Gaussians. First, context independent HMMs of one Gaussian were trained. Then these HMMs were successively extended to one more Gaussian and re-estimated to up to 8 Gaussians. Baum-

Welch re-estimation was used during the whole process. The phonetic labels used were generated automatically from the orthographic transcription using a set of rules. Alternative pronunciations were not considered, but we do not expect this to constitute a major problem because the training material was recorded and manually verified to avoid important dialectal and pronunciation variations. It is a common practice to use context independent HMMs for speech segmentation [186][187]. Context-dependent HMMs can better model the spectral movements in phonetic transitions. However, the segmentations they produce tend to be less precise than the ones produced by context independent HMMs. A theoretical explanation for this behavior was presented in [188], where it was argued that the cause is the loss of alignment, during the training process, between the context dependent HMMs and the phones. Context-dependent HMMs are always trained with realizations of phones in the same context. For that reason, the HMMs do not have any information to discriminate between the phone and its context. As a result, the HMM (particularly the lateral states) can end up modeling part of other phones or not all the phones. Context-independent HMMs, on the other hand, are trained with realizations of phones in different contexts. For that reason, they should be able to discriminate between the phone to model (invariable in all the training examples) and its context (which varies).

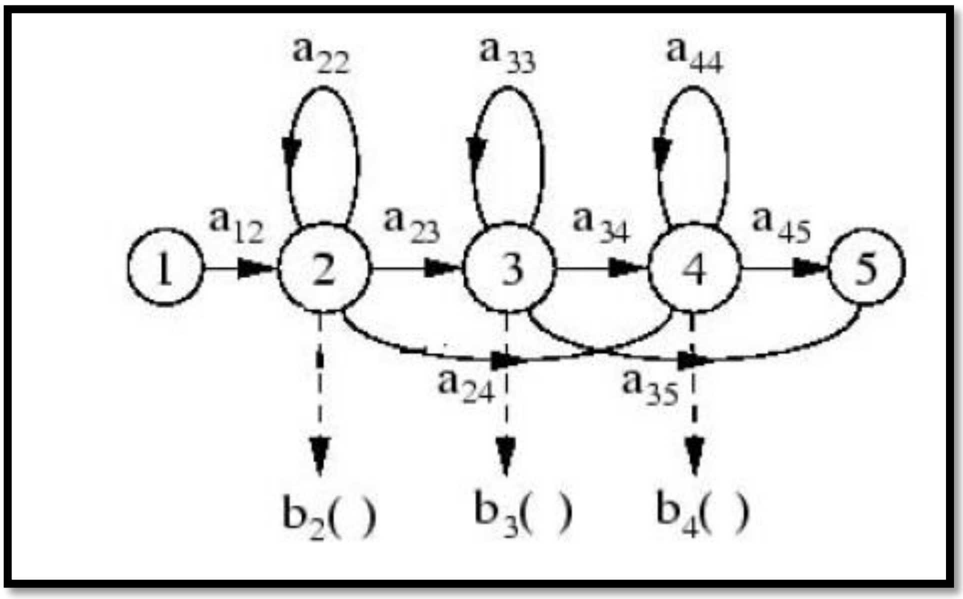


Figure 3.1 The 5-states HMM phoneme model

Once a phonetic transcription has been selected, automatic segmentation can proceed in the following way. Sentence models are generated by simply concatenating all relevant phoneme models. Next, the speech data are assigned by respectively Viterbi to the acoustic model of the complete phoneme sequence.

The Viterbi algorithm returns the single best path through the model given the observed speech signal *x i T i* , = 1,2,... , where *T* is the number of frames in the utterance.

$$s_i = arg\{ \underset{s_i \subset S}{max} \prod_{i=1}^{T} f(x_i \mid s_i) p(s_i \mid s_{i-1}) \qquad (3.1)$$

With si a sequence of HMM states (one state for each time frame) which is consistent with the sequence model S,T being the number of time frames. Thus, the Viterbi algorithm results in the segmentation which reaches maximum likelihood for the given feature vectors.

3.1.2 Gaussian mixture models

GMMS are used for modelling continuous distribution components as parametric probability distributions (Gaussian or normal), and the entire dataset can be modelled using mixture of such distributions or Gaussians.

GMMs are powerful in forming smooth approximations over a large class of sample distributions. GMM-based HMMs or GMM/HMM system is the most used ML approach in ASR. A GMM/HMM system is represented by l D (p, A, B), where p is a vector of state prior probabilities; A D (ai,j) is the state transition matrix; B D {b1,. . .,bn} is the set of GMMS of state j. The HMM state is usually associated with a sub-segment of a phoneme in speech. A sentence is modelled by concatenating HMMs for the sequence of phones and GMM distribution is used to generate a vector in the HMM state [88].

3.1.3 GMM formulation

The spectral features extracted from speech are real valued but applying HMMs on continuous observations is not directly possible. Instead, the possible values of an observation feature vector o_t are assumed to be normally distributed. The observation likelihood function bj(ot) is represented as a Gaussian. Given a dataset, the mean and variance can be obtained from the data, but the state that corresponds to an observation is not known. Hence, a way is needed to assign each observation vector o_t to every possible state i, incorporating the probability that the HMM was in state i at time t. Let this probability be $Y_t(i)$. Each vector of observation is modelled as a multivariate Gaussian with diagonal covariance matrices, and Baum– Welch algorithm is used to estimate the probability and to compute the mean and the variance.

A mixture of Gaussians is needed to model the multidimensional function and they need to be trained. The usual procedure to train the mixture of Gaussians (GMMs) is to choose M the number of Gaussians and splitting the Gaussian into two and running the forward–backward algorithm to retrain the Gaussians. This process is repeated until M Gaussians are generated. Another approach is to do embedded training where each phone HMM embedded in an entire sentence is trained. Both word segmentation and alignment can be done as a part of the training process. Typically, CD phones are used, and decision-tree-based state tying is used to cluster the many states into various clusters [88].

3.1.4 Limitations of GMMs in GMM-HMM based acoustic models

In GMM-HMM acoustic models, GMMs are used to model the relationship between the states of the HMM and the acoustic input. These models have proven efficient in dealing with the acoustic variations related to speaker accents, pronunciation variations, and environmental noise, etc. In fact, due to these variations, modeling the state densities of HMM using a mixture of Gaussians is more accurate than using a single form of density function. In addition, the availability and the efficiency of the EM algorithm for estimating the parameters of the model have played an important role in the success of GMMs in HMM-based acoustic models. Hence, it was difficult to find a new method that can outperform GMMs.

Despite their efficiency, GMMs do suffer from several shortcomings that should be addressed. In [189], three of the major GMM problems are identified. First, GMMs assume that the data distribution is necessarily Gaussian. Second, the parameters of GMMs at each HMM state are not trained using the whole data of all states but with the subset associated with that state given the alignment. On the other hand, the number of GMM parameters needed to be estimated across all states is very big, especially in context-dependent acoustic models, and may require a large amount of training data [190]. Finally, techniques used for feature dimensionality reduction may significantly reduce the accuracy in estimating the GMM parameters due to the potential elimination of some useful information.

Another critical shortcoming of GMMs in acoustic modeling is that they may require a large number of diagonal Gaussians or full-covariance Gaussians to model highly nonlinear data;

whereas other models exist and can fit such kind of data with only a few parameters [191]. In [192], it was argued that GMMs are also statistically inefficient at modeling high-dimensional data with componential structure. This inefficiency was attributed to the fact that for two significantly different sub-bands of independent patterns, when the first contains N different patterns and the second contains M different patterns, a GMM requires $N.M$ components to model such data. In fact, each data has only one single latent cause, and hence each component must fit both sub-bands. However, only $N + M$ components are necessary to explain such data for a model that uses multiple causes, in which each component is specific for a single sub-band. This shortcoming of GMM may affect the efficiency of GMM-HMM based ASR system where a large number of Gaussians at each HMM state must be estimated from a sub-set of the data derived from the alignment. Furthermore, in GMMs, every single Gaussian aims at modeling a partition from the input space. Having many Gaussians with independent means may lead to local generalization [193].

Many approaches have been proposed to overcome some of the limitations of GMMs. Since GMMs are typically trained as generative models using the EM algorithm, applying a subsequent stage of discriminative training was a first attempt to significantly improve the GMM-HMM acoustic models. The objective function of the discriminative training has a close relationship with the main goal of the ASR system. Maximum Mutual Information Estimation (MMIE) is one of the most common discriminate estimation methods. It aims at maximizing the separation between acoustic models by considering not only the likelihood of the training word strings given the labels, but also the probability of other possible word string hypotheses [194].

Using feed forward ANNs to replace GMMs in continuous density HMM for acoustic modeling was an alternative means to address the problem of GMMs discussed above. The introduction of an effective new procedure for learning deep neural networks has motivated researchers to apply deeper architecture for acoustic modeling.

On the other hand, it has been asserted that [191] the information embedded in speech can be represented with lower-dimensional data. However, GMMs are incapable of exploring highly correlated features. In consequence, they cannot handle the latent information from a large window of frames.

3.1.5 DNN-HMM hybrid architecture for acoustic modeling

Early attempts to use feed-forward ANNs as a substitute for GMMs in HMM-based acoustic models offered several advantages over GMMs. In [173], three major ones were reported. First, in contrast to GMMs, ANNs do not require detailed assumptions about the data distribution in order to estimate the posterior probabilities over HMM states. Second, when using ANNs, it is possible to apply different types of data, including the combination of discrete and continuous features. Finally, yet importantly, ANNs make use of all the training data to model their parameters. Furthermore, ANNs have shown great capabilities at modeling highly nonlinear data.

The effectiveness of the new deep learning algorithms developed recently has strengthened the idea that neural networks are suitable for acoustic modeling. As reviewed in the previous, the algorithm for training deep neural networks consists of a two-step's procedure: first, a generative pre-training step that aims at initializing the weights of the network and extracting higher new representations of the input at each layer, followed by a discriminative fine-tuning step with the backpropagation learning algorithm. DBNs have been the most common technique for generatively pre-training deep networks where each layer extracts a new higher representation and new structures from the input. This pre training stage has the great advantage of reducing over fitting and reducing the computation cost during the discriminative fine-tuning with the back-propagation algorithm, a cost that was considered the major impediment with deep networks. It also helps the network to rapidly converge towards better local minima. In earlier literature, DNNs generatively pre trained as DBNs were often called DBNs. To eliminate any ambiguity, Hinton et al. introduced the new name DBN-DNNs in [166].

In combining pre-trained DNNs with HMMs within a single hybrid architecture for acoustic modeling, researchers intended to combine the representational power of DBN-DNNs and the sequential modeling capability of HMMs.

3.1.6 Interfacing DNNs with HMMs

To interface DNN with HMM for modeling acoustic data, the network is trained to estimate the probability distribution over the states of the HMM. Considered as static classifiers with fixed dimensionality input vectors, DNNs are unable to perform sequence

segmentation with variable dimensionality of the inputs and outputs such as speech segmentation [170]. However, HMMs are the most powerful tool that can handle sequential patterns using dynamic programming. Thus, combining HMMs and DNNs fruitfully takes advantage of both static and sequential pattern recognition, which makes such hybrid architecture very useful for speech segmentation.

DNN trained to predict the posterior probability over Monophone HMM states. The training data consist of a window of n successive frames of speech coefficient. A network composed of many layers of nonlinear units was first generatively pre-trained using RBMs and contrastive divergence to extract new features of the input at each layer. Then, the network was discriminatively trained with the back-propagation algorithm to predict the label of the central frame. In fact, the output layer that represents the states of the HMMs provides a probability distribution over the possible labels of the central frame. Figure 3.2 illustrates the architecture of DNN for phone segmentation.

As described in the second chapter, an HMM is defined by 3 parameters:
the initial state distribution $\pi_i = P\ (s_0 = i)$, the transition probability $a_{ij} = P\ (s_t = j\ s_{t-1} = i)$, which is the probability of taking a transition from state *i* to state *j* at time *t*, the emission probability $b_j(x_t) = P\ (x_t|s_t = j)$ defined as the probability that the state *j* generates the acoustic observation x_t at time *t*. In GMM-HMM, this emission probability is estimated using the GMMs.

In DNN-HMM, DNN is used instead of GMM to estimate the emission probability at each state. In phoneme segmentation, the DNN produces the posterior probability, P (st|xt), of the mono-phone HMM states given the acoustic observation xt. However,
p(xt|st) = p(st|xt). p(xt)/p(st).

The probability p(st) can be estimated from an initial state-level alignment on the coustic training data using a Viterbi decoder. The probability p(xt) is assumed to be" independent of the word sequence and can be ignored during decoding" [175]. Hence, to get the emission probability P (xt|st), the posterior probability p(st|xt) should be divided by the prior p(st). However, it is asserted that this division may not provide improvement in segmentation accuracy under some conditions [175].

The DNN-HMM hybrid architecture can also be used for acoustic modeling in continuous speech segmentation task, as presented in [175][176] In contrast to the context-independent (CI) DNN-HMM hybrid architecture described above for phoneme segmentation, the DNN can be trained to predict probability distributions over tri-phone HMMs, forming thus a context-dependent (CD) DNN-HMM hybrid architecture for large-vocabulary continuous speech segmentation task. The idea is to use senones as modeling units rather than mono- phones. The method involves two main steps [175]. First, the Viterbi algorithm is used to generate the senone-level alignment on a tied tri-phone GMM-HMM baseline. Then, the DNN-HMM is trained to predict the senones in each frame or sequence of frames. It has been proven that the better the baseline system used during forced alignment, the better the results of the CD-DNN-HMM system [175] [166].

DNN-HMM hybrid architecture has shown successful results and good segmentation performance in both isolated and continuous speech segmentation tasks. In fact, DBN-DNNs have proven capability to outperform GMMs in acoustic modeling. The following subsection

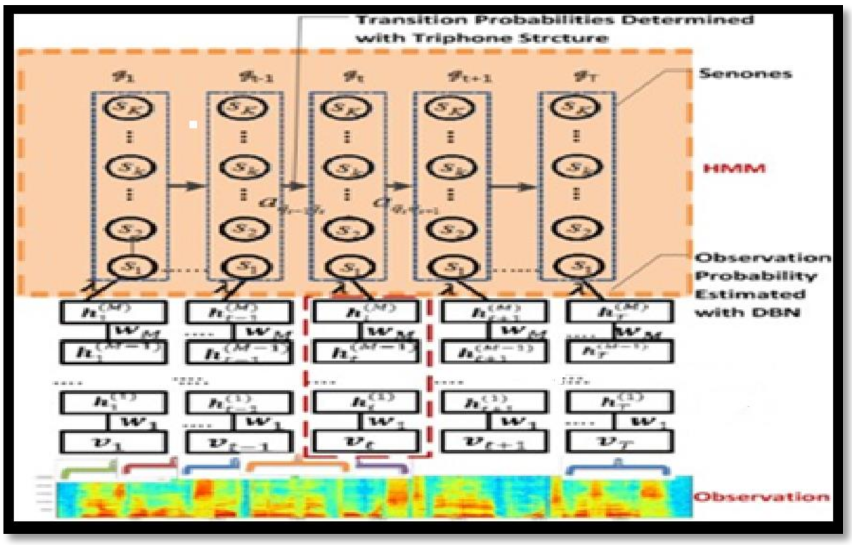

Transition Probabilities Determined with Triphone Strcture
Senones
HMM
Observation Probability Estimated with DBN
Observation

Figure 3.2 Interfacing DNN and HMM for continuous speech [177]

explores the main differences between GMMs and DNNs, and the main strengths of both deep-layered networks and the new learning algorithm for acoustic modeling.

3.1.7 Time delay neural network

In a time delay neural network, the temporal context is modeled by using a hierarchical architecture. Each layer in a TDNN operates at a different temporal resolution. The outputs of the activation from previous hidden layer are spliced as the input of the current layer. Therefore, the current layer operates at a much wider context, compared with the previous layer. As we go to higher layers of the network, increasingly wide context is seen by the network.

Convolutional Neural Networks (CNNs [200]), the transforms in the same layer of a TDNN are tied across time in order to reduce the number of parameters and make the transformation invariant to time shift of the input [200]. TDNNs are seen as a precursor to the CNNs. proposed a method to subsample the TDNN network. The splicing configuration {-1,1} means that we splice the input at current time step minus 1 and the current time step plus 1 (i.e. the current frame is dropped). Sub-sampling reduces the dimension of the input and thus the model size.

The overall input contexts of TDNNs are limited, for example, asymmetric context windows of up to 16 frames in past and 9 frames in the future are investigated in [200]. The success of TDNNs indicates that the most valuable information for the recognition of the current frame lies in a relatively narrow context. This is true even when recurrent models are used. Truncated Back Propagation Through Time.

3.2 Automatic verses Segmentation framework

In the Automatic verses Segmentation framework, all the steps mentioned below is followed

3.2.1. Data preparation

Data preparation is an important step of solving machine learning problems. An accurate, properly prepared text and speech corpus is very important for speech research areas. It is the
first stage of automatic speech segmentation model. It has high contribution to the performance of automatic speech segmenter.

the steps for getting data ready for speech segmentation model includes data collection, manual segmentation, lexicon preparation and pronunciation dictionary preparation.

3.2.2. Feature extraction

The final stage of data preparation is to parameterize the raw speech of the waveforms into sequences of feature vectors. This means that HTK is not as efficient in processing wav files as it is with its internal format. Therefore, should convert wav files to another format called MFCC format. Feature extraction is the process of transforming the speech waveform into a set of feature vectors. Mel Frequency Cepstral Coefficients is used to parameterize the speech signals into feature vectors with MFCC coefficients.

3.2.3. HMM model building

An acoustic model is a file that contains a statistical representation of each distinct sound that make-up each word used in grammar. Acoustic models are statistical models which capture the correspondence between a short sequence of acoustic vectors and an elementary unit of speech. The elementary units of speech that are used in our research is the phoneme.

HMM is used to model the acoustic component, Acoustic modeling process takes pronunciation dictionary, training text corpus, feature vectors of the training speech corpus as main inputs.

3.2.4 Automatic Verses Segmentation Framework

The proposed models are composed of three stages to get Automatic verses Segmentation include audio segmentation, features extraction and phoneme boundary detection.

the input audio is segmented into non-overlapping frames. These smaller frames are used in feature extraction for classification of speech/non-speech segments.

Figure 3.3 summarized the general steps which need to get the Automatic verses Segmentation by using different models represented in htk tool and kaldi tppl.

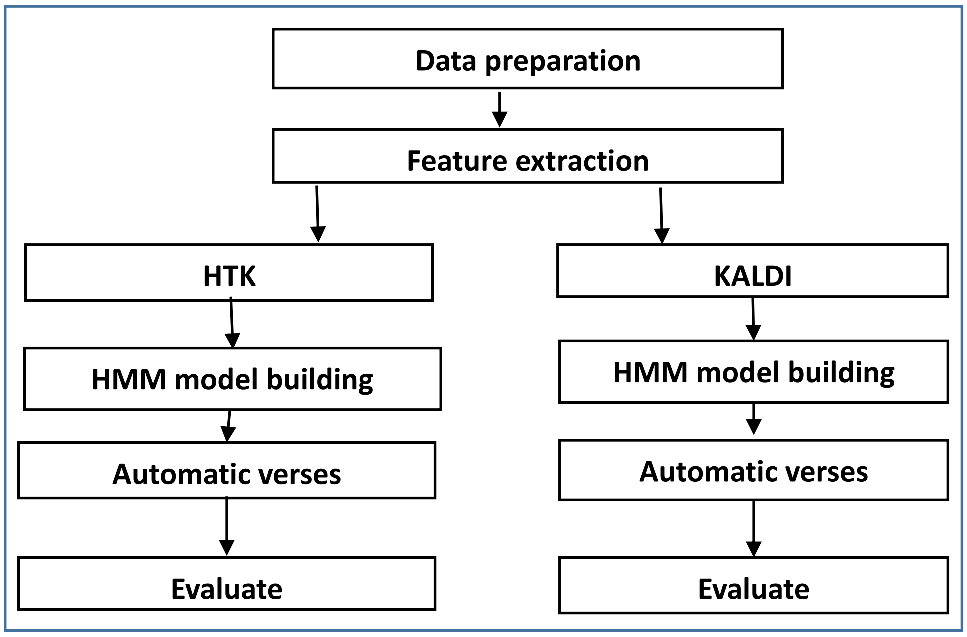
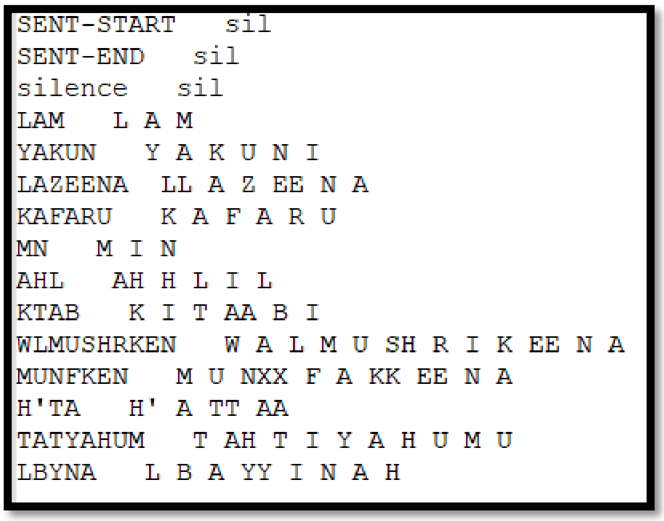


Figure 3.3 Automatic verses Segmentation framework

3.3 Automatic segmentation model implementation using HTK

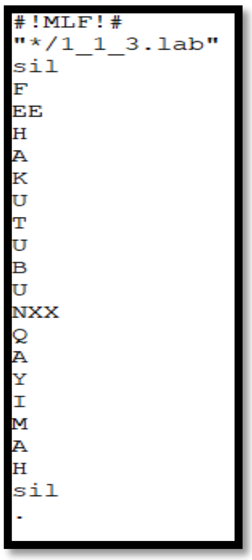
3.3.1 Dictionary preparation

The audio file and its word-level transcription are input. A wordlist and a dictionary are created as shown in Figure 3.4, also a list of phones was created for which HMMs will be estimated. The model created initially will lack small pause. Orthographic transcriptions are converted into the HTK label format- mlf (master label file). The same thing is done for the phones in transcripts as shown in Figure 3.5.



```
SENT-START    sil
SENT-END    sil
silence    sil
LAM    L A M
YAKUN    Y A K U N I
LAZEENA   LL A Z EE N A
KAFARU    K A F A R U
MN    M I N
AHL    AH H L I L
KTAB    K I T AA B I
WLMUSHRKEN    W A L M U SH R I K EE N A
MUNFKEN    M U NXX F A KK EE N A
H'TA    H' A TT AA
TATYAHUM    T AH T I Y A H U M U
LBYNA    L B A YY I N A H
```

Figure 3.4 Section of dictionary file



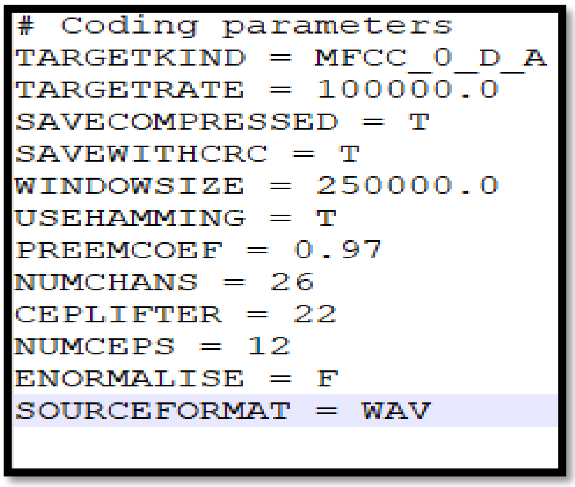
```
#!MLF!#
"*/1_1_3.lab"
sil
F
EE
H
A
K
U
T
U
B
U
NXX
Q
A
Y
I
M
A
H
sil
.
```

Figure 3.5 Section of Phones file

3.3.2 Creating MFCCs

Mel Frequency Cepstral Coefficients (MFCCs) is the standard in this research. To create the cepstral, which is the raw data used to form HMMs, we use the HTK tool HCopy which takes a single configuration file as input. This configuration file contains information such as sampling rate, pre-emphasis coefficient, window size etc.

The final stage of data preparation is to parameterize the raw speech waveforms into sequences of feature vectors. Here configuration parameters for feature extraction are given from configuration script config as shown in Figure 3.6.



```
# Coding parameters
TARGETKIND = MFCC_0_D_A
TARGETRATE = 100000.0
SAVECOMPRESSED = T
SAVEWITHCRC = T
WINDOWSIZE = 250000.0
USEHAMMING = T
PREEMCOEF = 0.97
NUMCHANS = 26
CEPLIFTER = 22
NUMCEPS = 12
ENORMALISE = F
SOURCEFORMAT = WAV
```

Figure 3.6 Config file [218]

In config file TARGETKIND = MFCC_0_D_A,

For each signal frame, the following coefficients are extracted:

The 12 first MFCC coefficients [c1,…, c12] (since NUMCEPS = 12).

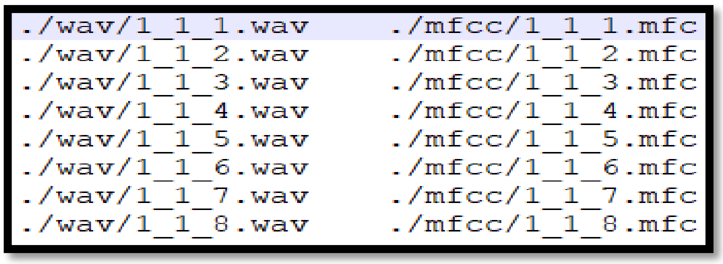
The "null" MFCC coefficient c0, which is proportional to the total energy in the frame (suffix "_0" in TARGETKIND).

13 "Delta coefficients", estimating the first order derivative of [c0, c1,…, c12] (suffix "_D" in TARGETKIND).

13 "Acceleration coefficients", estimating the second order derivative of [c0, c1,…, c12] (suffix "_A" in TARGETKIND).

Altogether, a 39-coefficient vector is extracted from each signal frame.

hcopy.scp is a script which contains source/target paths for the wav*s files and its correspondent MFCCS for the training data set as shown in Figure 3.7.



```
./wav/1_1_1.wav    ./mfcc/1_1_1.mfc
./wav/1_1_2.wav    ./mfcc/1_1_2.mfc
./wav/1_1_3.wav    ./mfcc/1_1_3.mfc
./wav/1_1_4.wav    ./mfcc/1_1_4.mfc
./wav/1_1_5.wav    ./mfcc/1_1_5.mfc
./wav/1_1_6.wav    ./mfcc/1_1_6.mfc
./wav/1_1_7.wav    ./mfcc/1_1_7.mfc
./wav/1_1_8.wav    ./mfcc/1_1_8.mfc
```

Figure 3.7 section of the hcopy file

3.3.3 HMM Initialization and Training

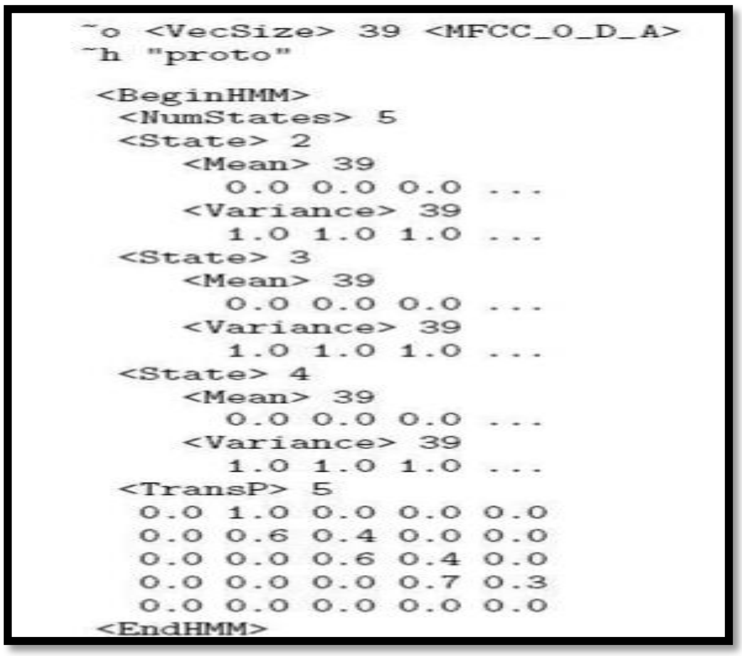
3.3.3.1 HMM Definition

The first step is to choose a priori a topology for each HMM:

1. Number of states
2. Form of the observation functions (associated with each state)
3. Disposition of transitions between states

Such a definition is not straightforward. There is no fixed rule for it. In HTK, an HMM is described in a text description file as shown in Figure 3.8.

After continuous researching and reviews from past papers [3]. I have concluded that the best state-of-art for this model would be 7 states, and in every state a single mixture.



```
~o <VecSize> 39 <MFCC_O_D_A>
~h "proto"

 <BeginHMM>
  <NumStates> 5
  <State> 2
     <Mean> 39
       0.0 0.0 0.0 ...
     <Variance> 39
       1.0 1.0 1.0 ...
  <State> 3
     <Mean> 39
       0.0 0.0 0.0 ...
     <Variance> 39
       1.0 1.0 1.0 ...
  <State> 4
     <Mean> 39
       0.0 0.0 0.0 ...
     <Variance> 39
       1.0 1.0 1.0 ...
  <TransP> 5
   0.0 1.0 0.0 0.0 0.0
   0.0 0.6 0.4 0.0 0.0
   0.0 0.0 0.6 0.4 0.0
   0.0 0.0 0.0 0.7 0.3
   0.0 0.0 0.0 0.0 0.0
 <EndHMM>
```

Figure 3.8 The description file for the HMM [218]

<NumStates> 5 gives the total number of states in the HMM, including the 2 non-emitting states 1 and 5.

<State> 2 introduces the description of the observation function of state 2. Here we have chosen to use single-gaussian observation functions, with diagonal matrices. Such a function is entirely described by a mean vector and a variance vector (the diagonal elements of the autocorrelation matrix). States 1 and 6 are not described since they have no observation function.

<Mean> 39 0.0 0.0 (...) 0.0 (x 39) gives the mean vector (in a 39-dimension observation space) of the current observation function. Every element is arbitrary initialized to 0: the file only gives the “prototype” of the HMM (its global topology). These coefficients will be trained later.

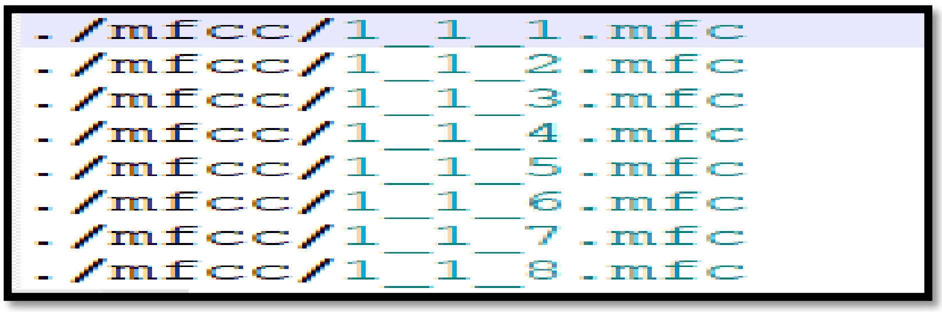
<Variance> 39 1.0 1.0 (...) 1.0 (x 39) gives the variance vector of the current observation function. Every element is arbitrary initialized to 1.

<TransP> 5 Gives the 5x5 transition matrix of the HMM.

Where a_{ij} is the probability of transition from state i to state j. Null values indicate that the corresponding transitions are not allowed. The other values are arbitrary initialized (but each line of the matrix must sum to 1). They will be later modified, during the training process.

3.3.3.2 HMM flat-start Initialization

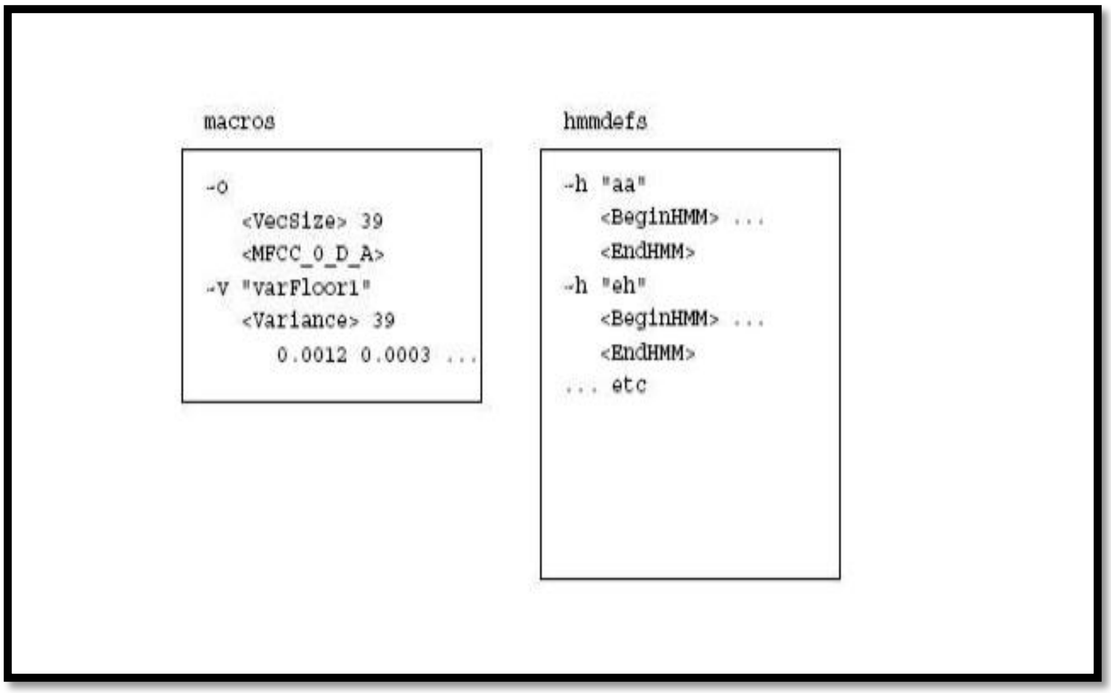
The HTK tool HCompV will scan a set of data files, compute the global mean and variance and set all of the Gaussians in a given HMM to have the same mean and variance. Hence, assuming that a list of all the training files is stored in train.scp as shown in Figure 3.9, Here train.scp is a script containing list of MFCC files, which are participating in Training. hmm0 is output directory name. proto is basic HMM definition file، then get output macro file vFloors and hmmdefs file.



```
./mfcc/1_1_1.mfc
./mfcc/1_1_2.mfc
./mfcc/1_1_3.mfc
./mfcc/1_1_4.mfc
./mfcc/1_1_5.mfc
./mfcc/1_1_6.mfc
./mfcc/1_1_7.mfc
./mfcc/1_1_8.mfc
```

Figure 3.9 section of the train.scp file

HCompV has several options specified for it. The -f option causes a variance floor macro (called vFloors) to be generated which is equal to 0.01 times the global variance. This is a vector of values which will be used to set a floor on the variances estimated in the subsequent steps. The –m option asks for means to be computed as well as variances. Given this new prototype model stored in the directory hmm0, a Master Macro File (MMF) called hmmdefs containing a copy for each of the required Monophone (in this model we have 72 monophones) HMMs is constructed by manually copying the prototype and relabeling it for each required monophone (including “sil”). The format of an MMF is similar to that of an MLF and it serves a similar purpose in that it avoids having a large number of individuals HMM definition files as shown in Figure 3.10.



```
macros
~o
   <VecSize> 39
   <MFCC_0_D_A>
~v "varFloor1"
   <Variance> 39
      0.0012 0.0003 ...
```

```
hmmdefs
~h "aa"
   <BeginHMM> ...
   <EndHMM>
~h "eh"
   <BeginHMM> ...
   <EndHMM>
... etc
```

Figure 3.10 section from Macros and hmmdefs files [218]

3.3.4 Re-estimation HTK

allows re-estimating the flat start monophones using the HTK tool HERest. For this, HMM definitions are created in which each unique phoneme is defined by the prototype. Re-estimation is done thrice to improve the model.

re-estimated using the embedded re-estimation tool HERest invoked as illustrated in figure 3.11.

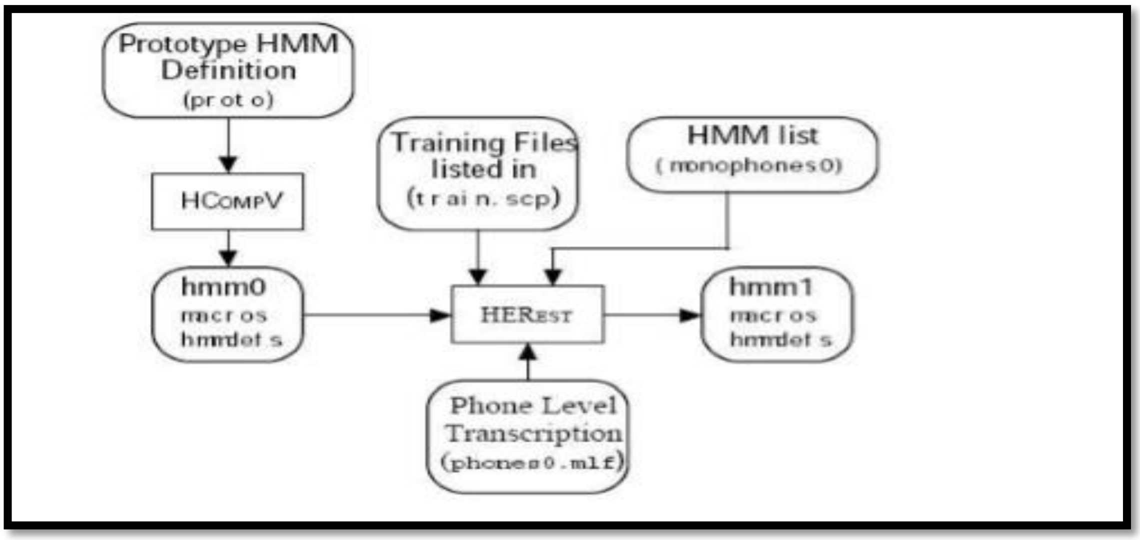

Prototype HMM Definition (proto)
HCompV
hmm0 macros hmmdefs
Training Files listed in (train.scp)
HMM list (monophones0)
HERest
hmm1 macros hmmdefs
Phone Level Transcription (phones0.mlf)

Figure 3.11 Re-estimation tool HERest [218]

3.3.5 Training and segmenting

Now that several versions of the model to be used have been created and trained, it's time to fix a few assumptions that have been made on the way. The first is the two types of "silence" in the corpus- sil, goes at the beginning and end of sentences, and sp, which lacks an HMM.

The two should be similar, but not entirely the same, HMM and phones.

To make the models more robust following steps are

3.3.5.1 Fixing the silence models

The middle state from the model for sil is copied to build a model for small pause sp. A script-based editor for HMMs, HHed is used.

3.3.5.2 Training Re-estimation

is performed twice more with the new model for sp which has been introduced while fixing the silence model.

3.3.5.3 Re-aligning data

The point of this re-alignment is to check for alternate pronunciations of words in the dictionary. The generated dictionary may contain multiple pronunciations; at this step, HTK decides which pronunciation is more applicable. Here Viterbi algorithm is implemented using the HVite, HTK tool, as shown in Figure 3.12.

As noted earlier, the dictionary contains multiple pronunciations for some words, particularly function words. The phone models created so far can be used to realign the training data and create new transcriptions. This can be done with a single invocation of the HTK recognition tool HVite.

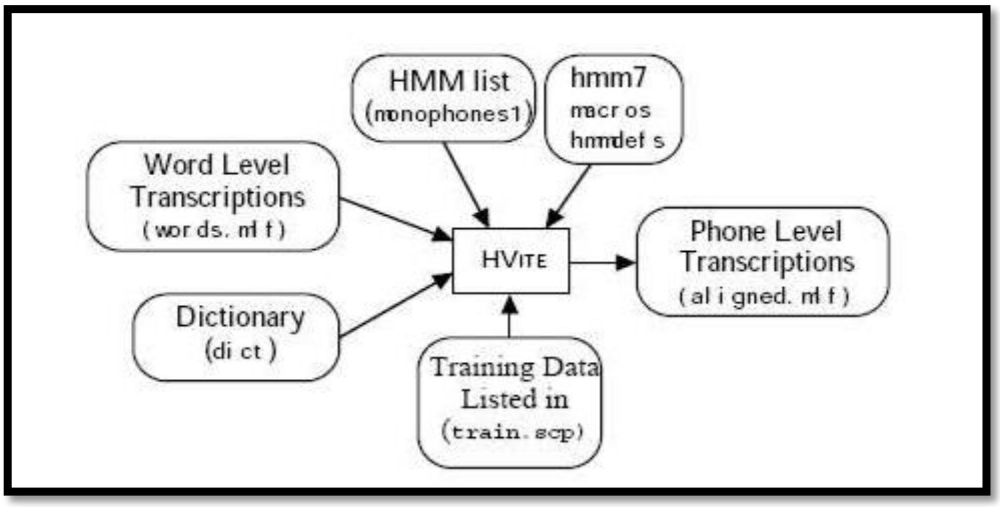


Figure 3. 12 HVite tool [218]

3.3.5.4 More training

After the most likely pronunciation has been chosen for each item in the dictionary in the previous step, two more rounds of training are performed using HERest.

3.3.5.5 The creation of Triphones from Monophones

The monophones are subsequently copied into a set of tied triphone models (we used the script driven editor" HHEd" from the HTK toolkit for tying), and triphone models are computed using embedded re-estimation.

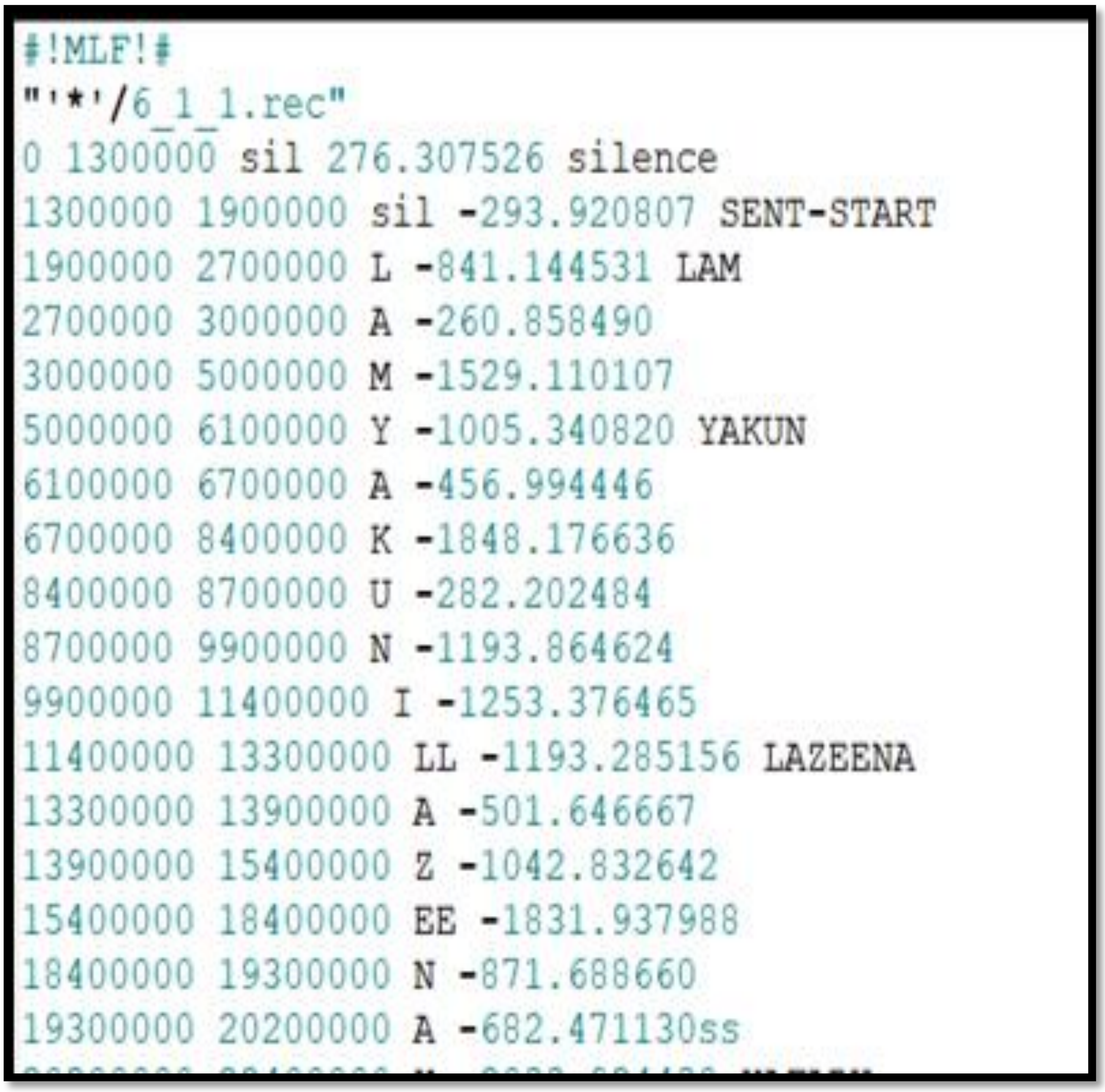
3.3.5.6 Initiating the Adapted System

Although the training techniques described previously can produce high performance, these systems can be improved upon by customizing the HMMs to the characteristics of a particular speaker. HTK provides the tools HEREST and HVITE to perform adaptation using a small amount of enrollment or adaptation data. The two tools differ in that HEREST performs offline supervised adaptation while HVITE recognizes the adaptation data and uses the generated transcriptions to perform the adaptation. Generally, more robust adaptation is performed in a supervised mode, as provided by HEREST, but given an initial well trained model set, HVITE can still achieve noticeable improvements in performance.

3.3.5.7 Segmenting

We have a sufficient model to obtain time-aligned phoneme transcriptions. The model works by adjusting alignments to maximize the degree to which words cluster, so HTK should have computed the most likely location of every phoneme using Viterbi algorithm (within the linear order of a sentence), using the model we've built so far. At this point, there is another possibility for refining the model before outputting the segmentations. HVite is used once more to output the final segments.

Aligned is a master label file, which contains phoneme-level Transcriptions with time stamps. Units for time stamps as shown in Figure 3.13.



```
#!MLF!#
"'*'/6_1_1.rec"
0 1300000 sil 276.307526 silence
1300000 1900000 sil -293.920807 SENT-START
1900000 2700000 L -841.144531 LAM
2700000 3000000 A -260.858490
3000000 5000000 M -1529.110107
5000000 6100000 Y -1005.340820 YAKUN
6100000 6700000 A -456.994446
6700000 8400000 K -1848.176636
8400000 8700000 U -282.202484
8700000 9900000 N -1193.864624
9900000 11400000 I -1253.376465
11400000 13300000 LL -1193.285156 LAZEENA
13300000 13900000 A -501.646667
13900000 15400000 Z -1042.832642
15400000 18400000 EE -1831.937988
18400000 19300000 N -871.688660
19300000 20200000 A -682.471130ss
```

Figure 3.13 section from the Aligned.mlf file

All the steps to build GMM-HMM base automatic segmentation model at phoneme level summarized in Figure 3.14.

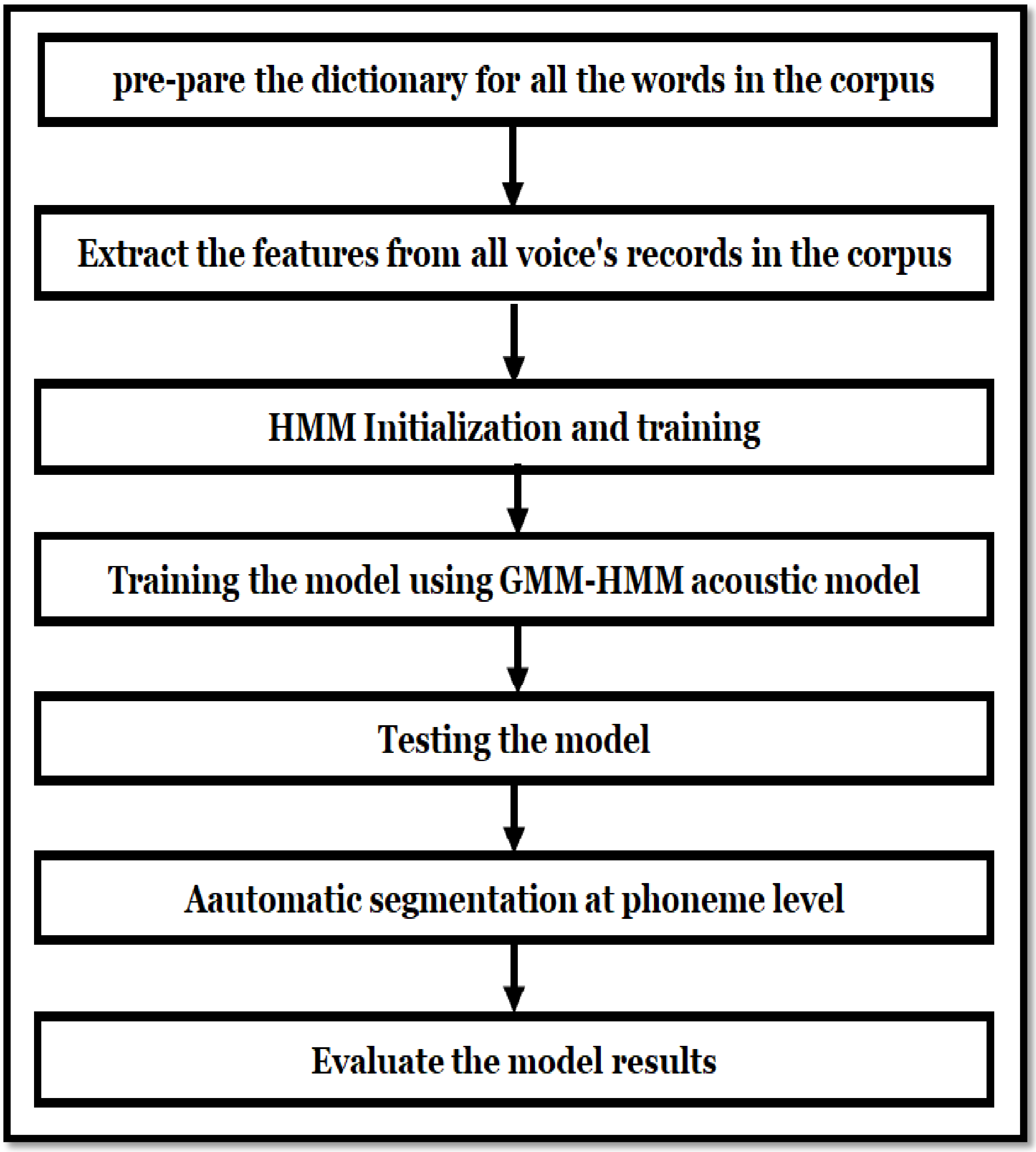

pre-pare the dictionary for all the words in the corpus
Extract the features from all voice's records in the corpus
HMM Initialization and training
Training the model using GMM-HMM acoustic model
Testing the model
Aautomatic segmentation at phoneme level
Evaluate the model results

Figure 3.14 Architecture of the GMM-HMM base automatic segmentation at phoneme level

3.4 The standard KALDI receipt for DNN-based acoustic modeling steps

KALDI is an open-source toolkit for speech recognition written in C++, the goal of KALDI is to have modern and flexible code that is easy to understand, modify and extend, KALDI is available on Source Forge (see http://KALDI.sf.net/). The tools compile on the commonly used Unix-like systems and on Microsoft Windows. The standard KALDI receipt for DNN-based on acoustic modeling steps

3.4.1 Data preparation

In the data preparation step, we will create directories in data which will store any training and test sets, features and eventually a language model.

Create files for:

3.4.1.1 Text

The text file is essentially the utterance-by-utterance transcript of the corpus. This is a text file with the following format:

utt_id WORD1 WORD2 WORD3 WORD4 …

utt_id = utterance ID

Example text file:

110236_20091006_82330_F_0001 I'M WORRIED ABOUT THAT

110236_20091006_82330_F_0002 AT LEAST NOW WE HAVE THE BENEFIT

110236_20091006_82330_F_0003 DID YOU EVER GO ON STRIKE

…

120958_20100126_97016_M_0285 SOMETIMES LESS IS BETTER

120958_20100126_97016_M_0286 YOU MUST LOVE TO COOK

Once you've created text, the lexicon will also need to be reduced to only the words present in the corpus. This will ensure that there are no extraneous phones that we are "training".

3.4.1.2 Wav.scp

av.scp contains the location for each of the audio files. If your audio files are already in wav format, use the following template:

file_id path/file, Example wav.scp file:

110236_20091006_82330_F path/110236_20091006_82330_F.wav

111138_20091215_82636_F path/111138_20091215_82636_F.wav

111138_20091217_82636_F path/111138_20091217_82636_F.wav

…

120947_20100125_59427_F path/120947_20100125_59427_F.wav

120953_20100125_79293_F path/120953_20100125_79293_F.wav

120958_20100126_97016_M path/120958_20100126_97016_M.wav

If your audio files are in a different format (sphere, mp3, flac, speex), you will have to convert them to wav format.

3.4.1.3 Utt2spk

utt2spk contains the mapping of each utterance to its corresponding speaker. As a side note, engineers will often conflate the term speaker with recording session, such that each recording session is a different "speaker". Therefore, the concept of "speaker" does not have to be related to a person – it can be a room, accent, gender, or anything that could influence the recording. When speaker normalization is performed then, the normalization may actually be removing effects due to the recording quality or particular accent type. This definition of "speaker" then is left up to the modeler.

utt2spk is a text file with the following format:

utt_id spkr

utt_id = utterance ID

spkr = speaker ID

Example utt2spk file:

```
110236_20091006_82330_F_0001 110236
110236_20091006_82330_F_0002 110236
110236_20091006_82330_F_0003 110236
110236_20091006_82330_F_0004 110236
…
120958_20100126_97016_M_0284 120958
120958_20100126_97016_M_0285 120958
120958_20100126_97016_M_0286 120958
```

3.4.1.4 Spk2utt

spk2utt is a file that contains the speaker to utterance mapping. This information is already contained in utt2spk, but in the wrong format. The following line of code will automatically create the spk2utt file and simultaneously verify that all data files are present and in the correct format:

```
utils/fix_data_dir.sh data/train
```

While spk2utt has already been created, you can verify that it has the following format:

spkr utt_id1 utt_id2 utt_id3

3.4.1.5 lexicon.txt

You will need a pronunciation lexicon of the language you are working on. A good English lexicon is the CMU dictionary, which you can find here. The lexicon should list each word on its own line, capitalized, followed by its phonemic pronunciation

WORD W ER D

LEXICON L EH K S IH K AH N

The pronunciation alphabet must be based on the same phonemes you wish to use for your acoustic models. You must also include lexical entries for each “silence” or “out of vocabulary” phone model you wish to train.

3.4.2 Features extraction

Mel Frequency Cepstral Coefficients (MFCC) are the most used features, but Perceptual Linear Prediction (PLP) features and other features are also an option. These features serve as the basis for the acoustic models.

We’ll now generate the features and the corresponding feats.scp script file, that will map utterance ids to positions in an archive, e.g. feats.ark. For GMM-HMM systems we typically use MFCC or PLP features, and then apply cepstral mean and variance normalisation.

feature extraction and waveform-reading code aims to create standard MFCC features, setting reasonable defaults but leaving available the options that people are most likely to want to tweak (for example, the number of mel bins, minimum and maximum frequency cutoffs, and so on). This code only reads from .wav files containing pcm data. These files commonly have the suffix .wav or. pcm (although sometimes the .pcm suffix is applied to sphere files; in this case the file would have to be converted). If the source data is not a wave file then it is up to the user to find a command-line tool to convert.

The command-line tools compute-mfcc-feats and compute the features; as with other KALDI tools, running them without arguments will give a list of options. The example scripts demonstrate the usage of these tools.

3.4.2.1 Computing MFCC features

Here we describe how MFCC features are computed by the command-line tool compute-mfcc-feats. This program requires two command-line arguments: an rspecifier to read the .wav data (indexed by utterance) and a w specifier to write the features (indexed by utterance); see The Table concept and Specifying Table formats: wspecifiers and rspecifiers for more explanation of these terms. In typical usage, we will write the data to one big "archive" file but also write out an "scp" file for easy random access; see Writing an archive and a script file simultaneously for explanation. The program does not add delta features (for that, see add-deltas). It accepts an option –channel to select the channel (e.g. –channel=0, –channel=1), which is useful when reading stereo data.

The computation of MFCC features is done by an object of type Mfcc, which has a function Compute() to compute the features from the waveform.

The overall MFCC computation is as follows:

1. Work out the number of frames in the file (typically 25 ms frames shifted by 10ms each time).
2. For each frame:
3. Extract the data, do optional dithering, preemphasis and dc offset removal, and multiply it by a windowing function (various options are supported here, e.g. Hamming)
4. Work out the energy at this point (if using log-energy not C0).
5. Do FFT and compute the power spectrum
6. Compute the energy in each mel bin; these are e.g. 23 triangular overlapping bins whose centers are equally spaced in the mel-frequency domain.
7. Compute the log of the energies and take the cosine transform, keeping as many coefficients as specified (e.g. 13)
8. Optionally do cepstral liftering; this is just a scaling of the coefficients, which ensures they have a reasonable range.

The lower and upper cutoff of the frequency range covered by the triangular mel bins are controlled by the options –low-freq and –high-freq, which are usually set close to zero and the Nyquist frequency respectively, e.g. –low-freq=20 and –high-freq=7800 for 16kHz sampled speech.

The features differ from HTK features in several ways, but almost all of these relate to having different defaults. With the option –HTK-compat=true, and setting parameters correctly, it is possible to get very close to HTK features. One possibly important option that we do not support is energy max-normalization. This is because we prefer normalization methods that can be applied in a stateless way and would like to keep the feature computation such that it could in principle be done frame by frame and still give the same results. The program compute-mfcc-feats does, however, have an option –subtract-mean to subtract the mean of the features. This is done per utterance; there are different ways to do it per speaker (e.g. search for "cmvn", meaning cepstral mean and variance normalization, in the scripts).

3.4.3 Training a Monophone model

A Monophone model is an acoustic model that does not include any contextual information about the preceding or following phone. It is used as a building block for the triphone models, which do make use of contextual information.

*Note: from this point forward, we will be assuming a Gaussian Mixture Model/Hidden Markov Model (GMM/HMM) framework. This is in contrast to a deep neural network (DNN) system.

The parameters of the acoustic model are estimated in acoustic training steps; however, the process can be better optimized by cycling through training and alignment phases. This is also known as Viterbi training (related, but more computationally expensive procedures include the Forward-Backward algorithm and Expectation Maximization). By aligning the audio to the reference transcript with the most current acoustic model, additional training algorithms can then use this output to improve or refine the parameters of the model. Therefore, each training step will be followed by an alignment step where the audio and text can be realigned.

3.4.4 Training a triphone model

While Monophone models simply represent the acoustic parameters of a single phoneme, we know that phonemes will vary considerably depending on their particular context. The triphone models represent a phoneme variant in the context of two other (left and right) phonemes.

At this point, we'll also need to deal with the fact that not all triphone units are present (or will ever be present) in the dataset. There are (# of phonemes)3 possible triphone models, but only a subset of those will occur in the data. Furthermore, the unit must also occur multiple times in the data to gather sufficient statistics for the data. A phonetic decision tree groups these triphones into a smaller amount of acoustically distinct units, thereby reducing the number of parameters and making the problem computationally feasible.

3.4.5 Training a triphone model with delta and delta-delta features

computes delta and double-delta features, or dynamic coefficients, to supplement the MFCC features. Delta and delta-delta features are numerical estimates of the first and second order derivatives of the signal (features). As such, the computation is usually performed on a larger window of feature vectors. While a window of two feature vectors would probably work, it would be a very crude approximation (similar to how a delta-difference is a very crude approximation of the derivative). Delta features are computed on the window of the original features; the delta-delta are then computed on the window of the delta-features.

3.4.6 Training a triphone model with Linear Discriminative Analysis (LDA) and Maximum Likelihood Linear Transform (MLLT)

stands for Linear Discriminant Analysis – Maximum Likelihood Linear Transform. The Linear Discriminant Analysis takes the feature vectors and builds HMM states, but with a reduced feature space for all data. The Maximum Likelihood Linear Transform takes the reduced feature space from the LDA and derives a unique transformation for each speaker. MLLT is therefore a step towards speaker normalization, as it minimizes differences among speakers.

3.4.7 Speaker adapted training (SAT)

stands for Speaker Adaptive Training. SAT also performs speaker and noise normalization by adapting to each specific speaker with a particular data transform. This results in more homogenous or standardized data, allowing the model to use its parameters on estimating variance due to the phoneme, as opposed to the speaker or recording environment.

3.4.8 Training on feature space maximum likelihood linear regression (fMLLR) adapted features

stands for Feature Space Maximum Likelihood Linear Regression. After SAT training, the acoustic model is no longer trained on the original features, but on speaker-normalized features. For alignment, we essentially must remove the speaker identity from the features by estimating the speaker identity (with the inverse of the fMLLR matrix), then removing it from the model (by multiplying the inverse matrix with the feature vector). These quasi-speaker-independent acoustic models can then be used in the alignment process.

3.4.9 Training the final DNN-HMM model

The DNN-HMM model is trained using fMLLR-adapted features; the decision tree and alignments are obtained from the SAT-fMLLR GMM system.

CTM is a master label file, which contains phoneme-level Transcriptions with time stamps. Units for time stamps, All the steps to build DNN-HMM base automatic segmentation model at phoneme level summarized in Figure 3.15.

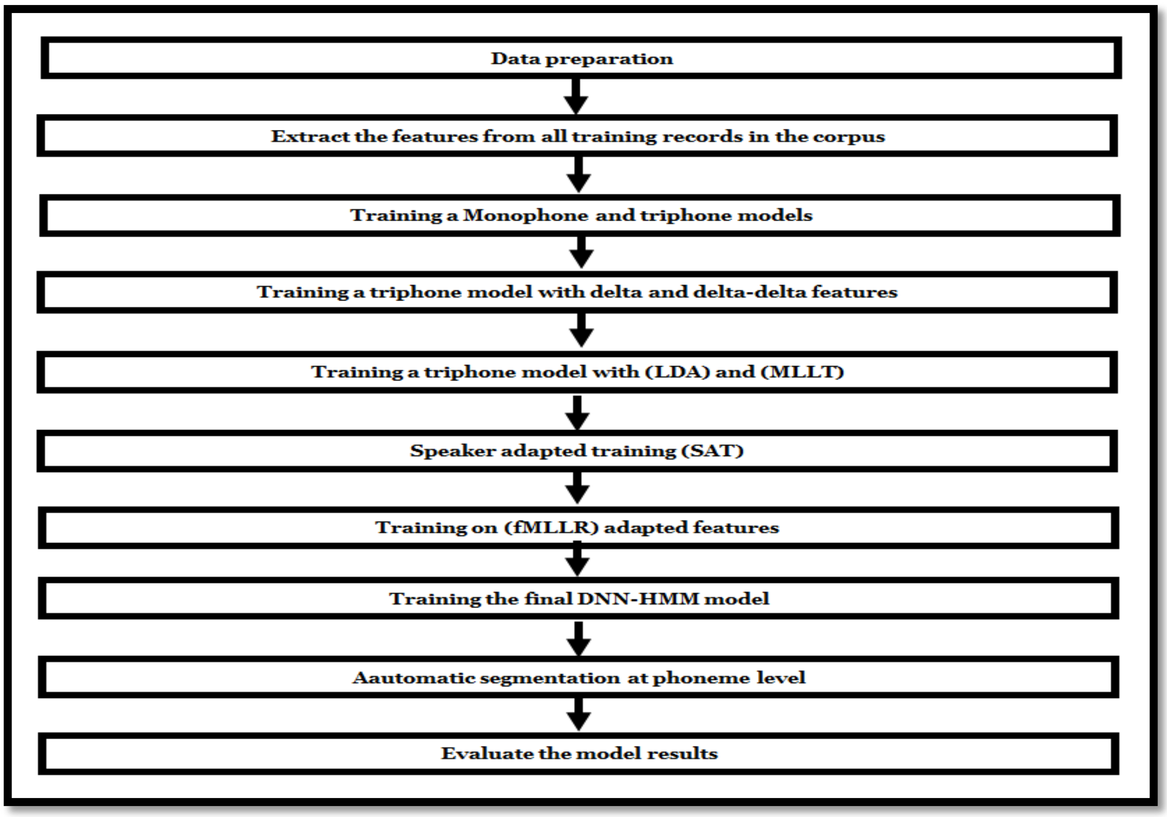

Data preparation
Extract the features from all training records in the corpus
Training a Monophone and triphone models
Training a triphone model with delta and delta-delta features
Training a triphone model with (LDA) and (MLLT)
Speaker adapted training (SAT)
Training on (fMLLR) adapted features
Training the final DNN-HMM model
Aautomatic segmentation at phoneme level
Evaluate the model results

Figure 3.15 Architecture of the DNN-HMM base automatic segmentation at phoneme level

3.5 Implementation of automatic segmentation model at phoneme level using KALDI toolkit

The model started with the initialization phase. According to KALDI toolkit, Speech features were calculated and used in KALDI recipes. As primary Cepstral features (used in AMs mono and tri1), we utilised 13 MFCCs including the zeros Cepstral coefficient, which were calculated for the short-term frame with the length of 25 ms and with the step of 10 ms above the signal have been shifted. The Cepstral mean of normalization was designated to this 13-element vector of short-term static features, and delta (dynamic) and delta-delta (acceleration) features accomplished them to the final length of 39. Linear Discriminative Analysis LDA features (used in AM tri2) was evaluated from the context obtained by splicing 5 short term feature vectors on both sides, continuing by LDA and Maximum Likelihood Linear Transform MLLT, which recognises decorrelation and the reduction of the dimension to the length of 40.

After the generation of the feature, all the words in the transcription must be implemented by the creation of a record-specific dictionary. Given that the orthography is to be phonetical, so the words in the lexicon are made up of their graphemes. KALDI uses the lexicon, acoustic model, and transcripts to create dataset-specific finite state transducers as a final preparation for the alignment. The baseline 3-gram model was created using SRI Language Modelling Toolkit (SRILM).

I have used the 3 as the value of N, due to the past researching and it was found that it reduces the perplexity which is the performance measurement metrics for the N-gram language model evaluation [216] [217].

In pursuance of AM tri3, a linear regression with maximum probability feature-space Maximum Likelihood Linear Regression (fMLLR) followed in the feature space for every speaker (also called Speaker Adaptive Training SAT). Ultimately, these 40-dimensional fMLLR features with mean and variance normalization in a mutual context were applied as input in this AM, we selected TDNN based on acoustic modeling with 6 hidden layers and 1024 units in each hidden layer.

The phonetic segmentation depends on the quality of the inputs, obviously, but it also depends heavily on the accuracy of the phonetic content entered. The phonetic content of utterances can be acquired by a grapheme-to-phoneme conversion or from a pronunciation lexicon, which can also cover pronunciation variability by including more pronunciation variants. This approach must be used when setting sound boundaries for spontaneous and informal speaking, a higher diversity of language dialects and other conditions with relatively high pronunciation variability. It can be held manually (for some very special condition) or automatically (to add certain sound substitutions or reductions to the regular pronunciation based on pre-specified conditions.

using a GPU speeds up training by about a factor of (mor than 10) faster than just using the CPU in our setup. Without using a GPU, it would take about three months to train the best model.

Chapter VI

Results Interpretation

In this chapter, described the experiments of phoneme level automatic speech segmentation based on the design developed in Chapter three and then the experiments results are presented for evaluation. Theis experiments covers two models of speech segmentation.

Firstly, automatic segmentation is implemented with HTK toolkit Secondly, automatic segmentation is implemented with KALDI toolkit.

4.0 Experiment No (1)

The model was built with the same mentioned steps in section3. 3

The data split chosen for all models in this research was 70% for training and 30% for testing because all research in this field chooses the same split, for example. [219] [220]

4.0.1 Dataset

Quran verses data set has been recorded with a duration of (2 hours) with having its corresponding text corpus, its contains 1.060 recordings of the verses of the Holy Quran with the voice of 10 reciters who recite the Quran carefully under the supervision of an expert of the correct reading of the ascription to the Messenger (Peace Be Upon Him), the recordings of 16 Surahs of the Quran, which are in order in the Holy Quran, starting from Surah Al-Bayyinah to Surah An-Nas, then using the MFCC to perform the Features Extraction Process, then building the Language Model and the Acoustic Model, then performing training process for the model at the level of Monophone and Triphone, Dataset Details summarized in table 4.1.

Table 4.1: Dataset Details in experiment no 1

	Num.Speakers	Num.Waves
Training Set	7	742
Test Set	3	318

4.0.2 Evaluation criteria

There are several methods by which the performance of the automatic segmentation is evaluated, which is explained in detail in [158]. the most suitable evaluation method for my research was accuracy accordingly to past papers [18] [222].

The direct method of evaluating the segmentation is to measure the accuracy, i.e., to find the number of correct boundaries. The accuracy is usually given as a percentage and is calculated as:

$$\text{Accuracy} = \frac{\text{Correctboundaries}}{\text{Totalboundaries}} \times 100 \qquad \text{Based on [18],}$$

For implementing Boundary Comparison For the given sets of reference (ref) and segmented (seg) phone boundaries, we proceed as follows. First, make a search space of 10 ms (10 ms to the left and 10 ms to the right) around each reference boundary. If the search spaces of two ref boundaries overlap, then shrink the search spaces by truncating at the middle of the overlapping area. This shrinking of the search space is done to prevent a single seg boundary from appearing in the search space of two neighboring ref boundaries. For comparison, a single boundary from the ref set is taken along with its search space. Any seg boundary that lies within the search space of this ref boundary is considered a match otherwise a miss. This is repeated for all ref boundaries. The result is obtained by taking the mean of the results from all the utterances.

The accuracy was calculated in all the experiments conducted in this research in the same way as previously explained.

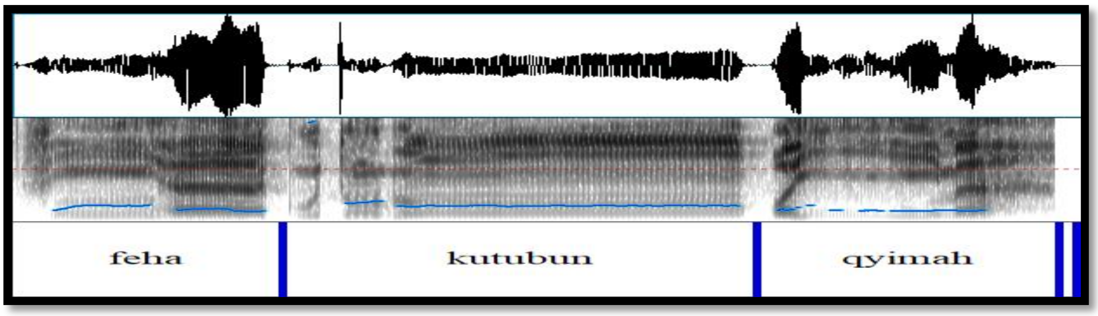


Figure 4.1 Manual Segmentation at word level by PRAAT tool

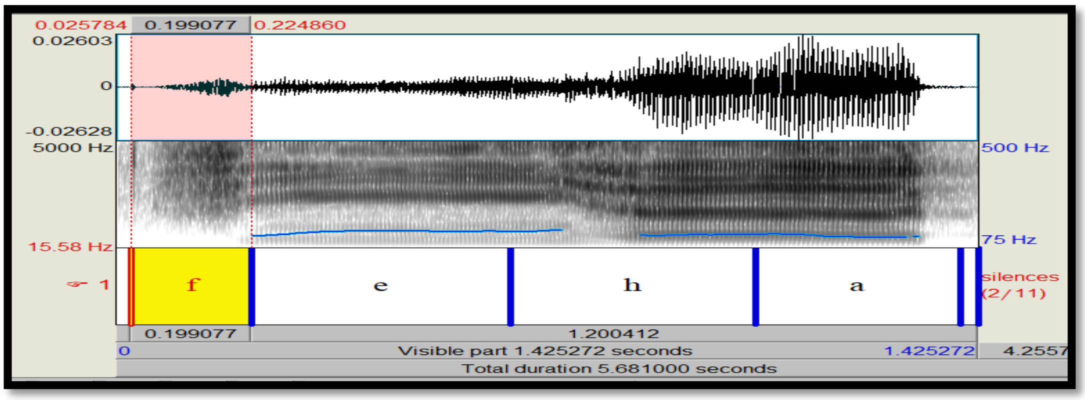


Figure 4.2 Manual Segmentation at phonetic level using PRAAT tool

Figure 4.1 shows the representation of a word by the PRAAT tool for one of the verses from Surat Al-Bayinah at the word level. Figure 4.2 shows the representation of a word at the phonemic level. As shown in the red color, PRAAT tool allows you to determine the beginning and end of each phoneme by time, in a fraction of a second, and depending on that, it is determining the begin boundary and end boundary for every phoneme (manual segmentation).

4.0.3 Results

The goal in speech segmentation is not to achieve a perfect phonetic segmentation. Automatic phonetic segmentations are generally evaluated by comparison with segmentations produced manually, which is the most accurate segmentation method known so far, but by no means error-free.

To measure the performance of automatic speech segmentation, phoneme-mapping concept is used. The manually segmented phonemes boundaries are compared with each phoneme found during automatic speech segmentation. The formulas used to calculate the accuracy of the Automatic phonetic segmentations mention in the Evaluation Criteria section,
in our study the accuracy was evaluated using two different tolerance values 5ms, 10ms as can be seen in table 4.2.

Table no. 4.2 shows the results of comparison between the segmented results obtained using HTK with the manually segmented results using PRAAT tool.

Table 4.2 The obtained results for automatic phonetic segmentation accuracy in experiment no 1

	% Performance for ≤ 5ms	**% Performance for ≥ 10 ms**
Test Set	60%	70 %
Adaptation Set	62%	70 %

4.1 Experiment No (2)

The model was built with the same steps mentioned in the section with the same parameters mentioned in sections 3.4, 3.5.

4.1.1 Dataset

Quran verses data set has been recorded with a duration of (2 hours) with having its corresponding text corpus, containing 1.060 recordings of the verses of the Holy Quran with the voice of 10 reciters who recite the Quran carefully under the supervision of an expert of the correct reading of the ascription to the Messenger (Peace Be Upon Him), the recordings of 16 Surahs of the Quran, which are in order in the Holy Quran, starting from Surah Al-Bayyinah to Surah An-Nas, Dataset Details summarized in table 4.3.

Table 4.3 Dataset Details in experiment no 2

	Num.Speakers	Num.Waves
Training Set	**7**	**742**
Test Set	**2**	**212**
Development Set	**1**	**106**

4.1.2 Results

Table 4.4 The obtained results for automatic phoneme segmentation accuracy in experiment no 2

	% Performance for ≤ 5ms	% Performance for ≤ 10ms
Test Set	70 %	72%
Dev Set	75 %	78%

4.2 Experiment No (3)

The model was built with the same steps mentioned in the section with the same parameters mentioned in sections 3.4, 3.5.

4.2.1 Dataset

Quran verses data set has been recorded with having its corresponding text corpus and number of waves 100 for 100 reciters and a total speech corpus of (5.3 hours). All Quran verses text corpus are recorded in mono channel, *.wav format and 16 kHz sample frequency. The dataset was split into a training set, a test set, and development set to simulate the real data collection and training procedure and to avoid having overlap between training, test, and development sets. The data set contains surah (al-Fatiha, al-Asr, al-Kawthar, al-Ikhlas, al-Falaq and an-Nas) and assuming that the reciters were non-Arab speakers The training phase was implemented, by selecting 70 waves for 70 reciters. The testing of the model was done using 20 waves for 20 reciters. Lastly, the development dataset was selected from 10 reciters with a total of 10 waves, all respectively without overlap or repetition, Dataset Details summarized in table 4.5.

Table 4.5 Dataset Details in experiment no 3

	Speakers	Hours
Training Set	70	5.3
Test Set	20	1.3
Development Set	10	0.6

4.2.2 Results

Table 4.6 The obtained results for automatic phonetic segmentation accuracy in experiment no 3

	% Performance for ≤ 5ms	% Performance for ≤ 10ms
Test Set	95 %	97%
Dev Set	99 %	99.9%

4.3 Experiment No (4)

The model was built with the same steps mentioned in the section with the same parameters mentioned in section 3.5

4.3.1 Dataset

Quran verses data set has been recorded with having its corresponding text corpus and number of waves 1100 for 100 reciters and a total speech corpus of (80 hours). All Quran verses text corpus are recorded in mono channel, *.wav format and 16 kHz sample frequency. The dataset was split into a training set, a test set, and development set to simulate the real data collection and training procedure and to avoid having overlap between training, test, and development sets. The data set contains verses from (al-Fatiha, al-Asr, al-Kawthar, al-Ikhlas, al-Falaq and an-Nas) and assuming that the reciters were non-Arab speakers. and it contains also records for 10 letters which have similarities in pronunciation to increase the system learning capabilities. The training phase was implemented, by selecting 770 waves for 70 reciters. The testing of the model was done using 220 waves for 20 reciters. Lastly, the development dataset was selected from 10 reciters with a total of 110 waves, all respectively without overlap or repetition, Dataset Details summarized in table 4.7.

Table 4.7 Dataset D.etails in experiment no 4

	Speakers	**Hours**
Training Set	70	56
Test Set	20	16
Development Set	10	8

4.3.2 Results

To measure the performance of automatic speech segmentation, phoneme-mapping concept is used. The manually segmented phoneme boundaries are compared with each phoneme sequences found during automatic speech segmentation. The formulas used to calculate the accuracy of the Automatic phonetic segmentations mention in the Evaluation Criteria section.

Table 4.8 The obtained results for automatic phoneme segmentation accuracy in experiment no 4

	% Performance for ≤ 5ms
Test Set	99.9 %
Dev Set	99.9 %

on automatic segmentation results. The best result was obtained when DNN with 9 hidden layers and 1024 units in each hidden layer, in dev set

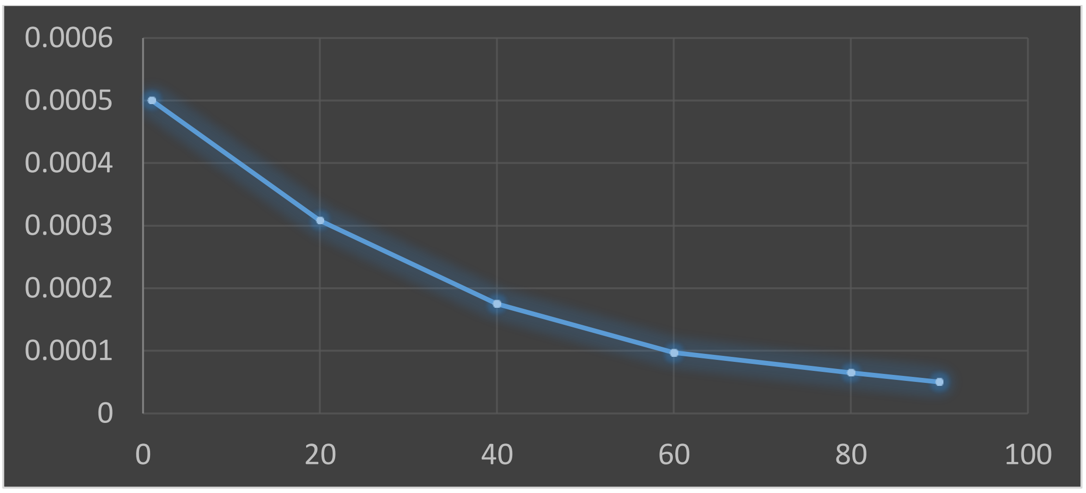


Figure 4.3 decreasing error rate while increasing number of iterations

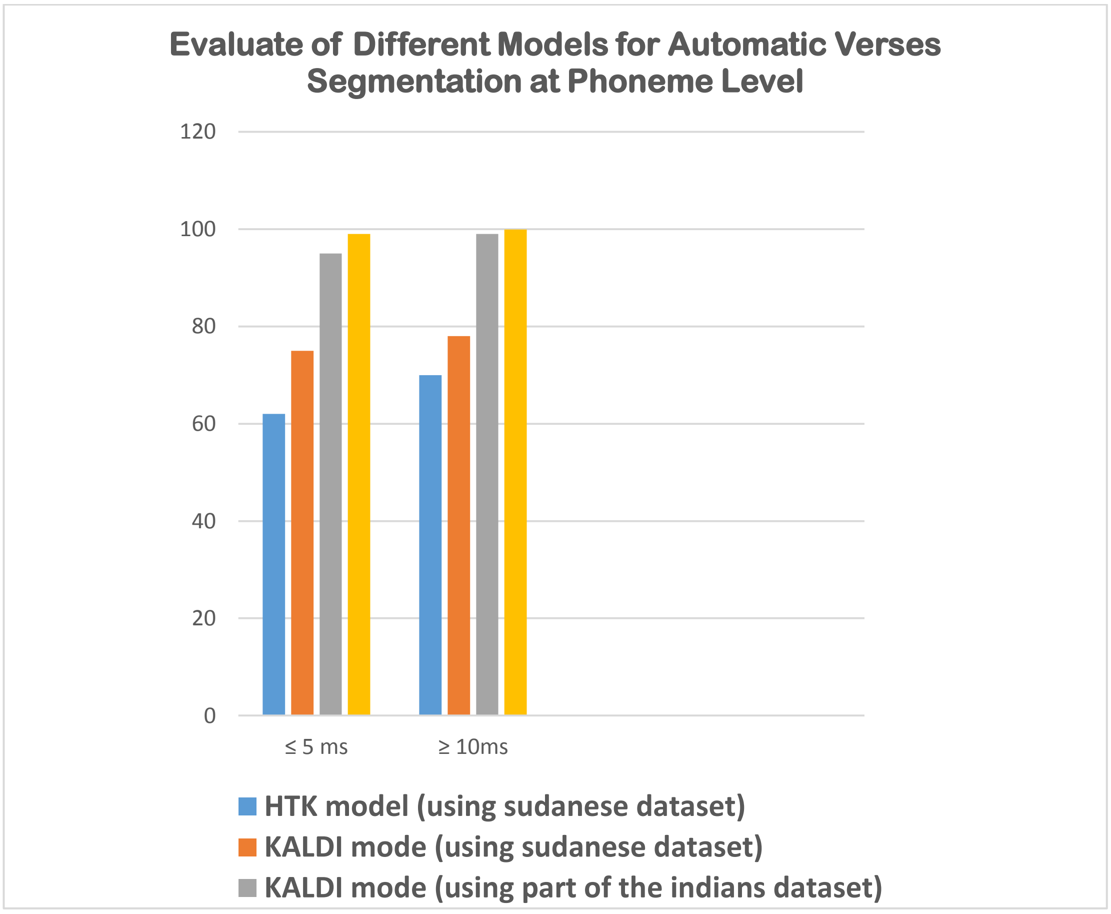


Figure 4.4 Evaluate of Different Models for Automatic Verses Segmentation at Phoneme Level

4.4 Result discussion

This research has demonstrated a clearer insight of using DNN-HMM at phonetic automatic segmentation by using the verses of the noble holy Qur'an with different parameters. comparing the efficiency of the different tools and involving different sizes of data set to draw a clear understanding to which methodology may result in the highest accuracy. The methodological choices were constrained, we built more than one model using different tools and using two different datasets. through 4 different experiments, interpreting with the previous results noted in this study that comparing the details of the first and second experiment, that the DNN has contributed to improve the results of automatic segmentation. when looking at comparing the details of the second and third experiment, we note that the large size of the database on which the model is trained contributes significantly to improve the accuracy of the results obtained. finally, when comparing the results obtained from the third and fourth experiment, the influence and role of the language model appears significantly in improving the results obtained to reach an accuracy of 99.9%.

4.5 Limitations

The main limitation we faced is the creation of the dataset. For training the model we must record at least 5 hours for one hundred reciters to improve the accuracy. This number of recording hours takes a long time for the recording and the pre-processing, and in our work, we must cooperate with one hundred expert reciters to record the selected Holy Quran verses. We expect that the creation of dataset with perfect properties take about two years at least. In the addition of the need for a lot of efforts and times, the recording in proper environment cost a lot of money. It costs about US $100 for each recording hour, according to these limitations we create one dataset for only 10 speakers. (I would like at this point to thank RDI that has given me the second data set ready which I believe has given them great trouble and effort including time to do the recording)

Chapter VII

Conclusion and Further Work

5.0 Conclusion

In this thesis we have studied GMM-HMM and DNN-HMM based on AM for holy Quran verses, using the HTK toolkit and Kaldi toolkit. We have experimented with DNNs with different numbers of hidden layers, the automatic phonetic segmentation experiments showed that the results obtained was 99.9% compared with manual segmentation. the main goal of this thesis was the development of Deep Neural Network-Hidden Markov Model (DNN-HMM) hybrid acoustic models for phoneme Automatic segmentation, with an emphasis on providing a systematic implementation procedure. This thesis also aimed to empirically confirm the capability of DNNs to outperform Gaussian Mixture Models (GMMs), and to investigate the performance of DNNs in acoustic modeling.

In this thesis, a thorough overview of the fundamentals of speech segmentation was presented, including the description of the different components required to build Automatic Speech segmentation models. Theories about GMM and DNNs, their architectures and learning algorithms were reviewed. The problems of GMMs in HMM-based acoustic models were discussed.

Training a neural network with many hidden layers and a large output layer on a large amount of data was computationally very expensive. To accelerate the training, parallel computing using Graphic Processing Units (GPUs) was exploited.

The investigations proved that by increasing the number of layers the segmentation accuracy does necessarily increase Finally, it was proved that DNNs can provide better acoustic estimations than GMMs, which often underestimate.

5.1 Future Work

For further research, we will investigate in the holy Quran by using some other DNN’s configurations and types with larger data set.

Further investigations are required to identify other major factors behind the performance of DNNs in acoustic modeling. Our next steps will focus on incorporating more advanced speech-modeling and feature-extraction techniques available to build the automatic segmentation model.

References

[1] F. Diehl, M. J. F. Gales, M. Tomalin, and P. C. Woodland,m “ Morphological decomposition in Arabic ASR systems,” Comput. Speech Lang., vol. 26, no. 4, pp. 229–243, Aug. 2012.

[2] B. Kingsbury, H. Soltau, G. Saon, S. Chu, H.-K. Kuo, L. Mangu, S. Ravuri, N. Morgan, and A. Janin, “The IBM 2009 GALE Arabic speech transcription system,” in 2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 4672–4675, 2011.

[3] Dines, John, Sridha Sridharan, and Miles Moody. "Automatic speech segmentation with hmm." In Proceedings of the 9th Australian Conference on Speech Science and Technology, pp. 544-549, 2002.

[4] Matouˇsek, J., Tihelka, D., Psutka, J.: Experiments with automatic segmentation for Czech speech synthesis. In: Matouˇsek, V., Mautner, P. (eds.) TSD 2003. LNCS (LNAI), vol. 2807, pp. 287–294. Springer, Heidelberg, 2003.

[5] Rendel, A., Sorin, A., Hoory, R., Breen, A.: Toward automatic phonetic segmentation for TTS. In: Acoustics, Speech and Signal Processing (ICASSP), Kyoto, Japan, pp. 4533–4536, 2012.

[6] Mizera, P., Pollak, P., Kolman, A., Ernestus, M.: Impact of irregular pronunciation on phonetic segmentation of Nijmegen corpus of casual Czech. In: Sojka, P., Hor´ak, A., Kopeˇcek, I., Pala, K. (eds.) TSD 2014. LNCS (LNAI), vol. 8655, pp. 499–506. Springer, Cham, 2014.

[7] Y. C. a. Q. Wang, "A Speaker Based Unsupervised Speech Segmentation Algorithm Used in Conversational Speech," Springer-Verlag Berlin Heidelberg 2007, pp. 396–402, 2007.

[8] F. D. Brugnara F., and Omologo M., "Automatic segmentation and labeling of speech based on hidden Markov models," Speech Commun. , vol. 12, pp. 357-370, 1993.

[9] B. L. Pellom, and Hansen, J. H. L., "Automatic segmentation of speech recorded in unknown noisy channel characteristics," Speech Commun.25, pp. 97-116, 1998.

[10] Young, S., et al.: The HTK Book, Version 3.4.1. Cambridge, 2009.

[11] Povey, D., et al.: The Kaldi speech recognition toolkit. In: Proceedings of the IEEE 2011 Workshop on Automatic Speech Recognition and Understanding, ASRU 2011, 2011.

[12] Automatic Phonemes Segmentation for Quran Verses Using Kaldi Toolkit 2022

[13] Scharenborg O, Wan V, Ernestus M. Unsupervised speech segmentation: An analysis of the hypothesized phone boundaries, 2010.

[14] I. Mporas, T. Ganchev, and N. Fakotakis, “A hybrid architecture for automatic segmentation of speech waveforms,” in 2008 IEEE International Conference on Acoustics, Speech and Signal Processing. IEEE, pp. 4457–4460, 2008.

[15] B. Zio´łko, S. Manandhar, R. C. Wilson, and M. Zio´łko, “Wavelet method of speech segmentation,” in Signal Processing Conference, 2006 14th European. IEEE, pp. 1–5, 2006

[16] S. M. Siniscalchi, P. Schwarz, and C.-H. Lee, “High-accuracy phone recognition by combining high-performance lattice gen- eration and knowledge based rescoring,” in IEEE International Conference on Acoustics, Speech and Signal Processing-ICASSP’07, vol. 4. IEEE, 2007.

[17] Anh, Tuan Tran, Mong Nguyen Huu, and Khanh Nguyen Trong. "A Method for Automatic Vietnamese Speech Segmentation.", 2019

[18] Demuynck, Kris, and Tom Laureys. "A comparison of different approaches to automatic speech segmentation." International Conference on Text, Speech and Dialogue. Springer, Berlin, Heidelberg, 2002.

[19] Coelho, Luis Pinto, and Daniela Braga. "Automatic phonetic segmentation and labelling of spatanous speech." IV Jornadas en Tecnologia del Habla IVJTH’06, 2006.

[20] Rahman, Md Mijanur, and Md Al-Amin Bhuiyan. "Continuous bangla speech segmentation using short-term speech features extraction approaches." Int J Adv Comput Sci Appl (IJACSA) 3.11, 2012.

[21] Makowski, Ryszard, and Robert Hossa. "Automatic speech signal segmentation based on the innovation adaptive filter." International Journal of Applied Mathematics and Computer Science 24.2, 2014.

[22] Matoušek, Jindřich, and Michal Klíma. "Automatic phonetic segmentation using the kaldi toolkit." International Conference on Text, Speech, and Dialogue. Springer, Cham, 2017.

[23] Mizera, Petr, and Petr Pollak. "Automatic phonetic segmentation and pronunciation detection with various approaches of acoustic modeling." International Conference on Speech and Computer. Springer, Cham, 2018.

[24] Bigi, Brigitte, and Christine Meunier. "Automatic segmentation of spontaneous speech." Revista de Estudos da Linguagem 26.4, 2018.

[25] Anwar, Muhammad Jamil, et al. "Automatic Arabic speech segmentation system." International Journal of Information Technology 12.6, 2006.

[26] Kamarauskas, Juozas. "Automatic Segmetation of Phonemes using Artificial Neural Networks." Elektronika ir Elektrotechnika 72.8, 2006.

[27] Rahman, Md Mijanur, Fatema Khatun, and Md Al-Amin Bhuiyan. "Blocking black area method for speech segmentation." Editorial Preface 4.2, 2015.

[28] Sinclair, Mark, et al. "A semi-markov model for speech segmentation with an utterance-break prior." Fifteenth Annual Conference of the International Speech Communication Association, 2014.

[29] Kaur, Er Amanpreet, and Er Tarandeep Singh. "Segmentation of continuous punjabi speech signal into syllables." Proceedings of the World Congress on Engineering and Computer Science. Vol. 1, 2010.

[30] Stolcke, Andreas, et al. "Highly accurate phonetic segmentation using boundary correction models and system fusion." 2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE, 2014.

[31] Juneja, Amit, and Carol Espy-Wilson. "Speech segmentation using probabilistic phonetic feature hierarchy and support vector machines." Proceedings of the International Joint Conference on Neural Networks, Vol. 1. IEEE, 2003.

[32] Hosom, John-Paul. "Automatic phoneme alignment based on acoustic-phonetic modeling." INTERSPEECH, 2002.

[33] Saksamudre, Suman K., and R. R. Deshmukh. "Isolated Word Recognition System for Hindi Language." International Journal of Computer Science andEngineering 3.7, 2015.

[34] Bansal, Poonam, et al. "Speech synthesis-automatic segmentation." International Journal of Computer Applications 98.4, 2014.

[35] Zhang, Jin Xi, et al. "The Phoneme Automatic Segmentation Algorithms Study of Tibetan Lhasa Words Continuous Speech Stream." Advanced Materials Research. Vol. 765. Trans Tech Publications Ltd, 2013.

[36] Nagarajan, T., Hema A. Murthy, and Rajesh M. Hegde. "Segmentation of speech into syllable-like units." Eighth European Conference on Speech Communication and Technology, 2003.

[37] Park, Jun, Young-jik Lee, and Jae-woo Yang. "Spontaneous Speech Translation System Development." Proceedings of the Acoustical Society of Korea Conference. The Acoustical Society of Korea, 1998.

[38Park, Jun, Young-jik Lee, and Jae-woo Yang. "Spontaneous Speech Translation System Development." Proceedings of the Acoustical Society of Korea Conference. The Acoustical Society of Korea, 1998.

[39] Prasad, V. Kamakshi, T. Nagarajan, and Hema A. Murthy. "Automatic segmentation of continuous speech using minimum phase group delay functions." speech communication 42.3-4, 2004.

[40] Brognaux, Sandrine, and Thomas Drugman. "HMM-based speech segmentation: Improvements of fully automatic approaches." IEEE/ACM Transactions on Audio, Speech, and Language Processing 24.1, 2015.

[41] Emiru, Eshete Derb, and Walelign Tewabe Sewunetie. "Automatic Speech Segmentation for Amharic Phonemes Using Hidden Markov Model Toolkit (HTK)", 2016.

[42] Räsänen, Okko, Gabriel Doyle, and Michael C. Frank. "Unsupervised word discovery from speech using automatic segmentation into syllable-like units." Sixteenth Annual Conference of the International Speech Communication Association, 2015.

[43] Baby, Arun, et al. "Significance of spectral cues in automatic speech segmentation for Indian language speech synthesizers." Speech Communication 123, 2020.

[44] Karim, Riksa Meidy. "Optimizing Parameters of Automatic Speech Segmentation into Syllable Units." International Journal of Intelligent Systems and Applications 10.5, 2019.

[45] M. Nilsson and M. Ejnarsson, "Speech recognition using hidden markov model", 2002.

[46] M. A. Al-Manie, M. I. Alkanhal, M. M. Al-Ghamdi, N. Mas- torakis, A. Croitoru, V. Balas, E. Son, and V. Mladenov, "Auto- matic speech segmentation using the arabic phonetic database," in WSEAS International Conference. Proceedings. Mathematics and Computers in Science and Engineering, no. 10. World Scientific and Engineering Academy and Society, 2009.

[47] R. Makowski and R. Hossa, "Automatic speech signal segmentation based on the innovation adaptive filter," International Journal of Applied Mathematics and Computer Science, vol. 24, no. 2, pp. 259–270, 2014.

[48] A. Cherif, L. Bouafif, and T. Dabbabi, "Pitch detection and for- mant analysis of arabic speech processing," Applied Acoustics, vol. 62, no. 10, pp. 1129–1140, 2001.

[49] M. Sharma and R. Mammone, "Subword-based text-dependent speaker verification system with user-selectable passwords," in Acoustics, Speech, and Signal Processing, 1996. ICASSP-96. Conference Proceedings., IEEE International Conference on, vol. 1., pp. 93–96, 1996.

[50] J. P. van Hemert, "Automatic segmentation of speech," IEEE Transactions on Signal Processing, vol. 39, no. 4, pp. 1008–1012, 1991.

[51] Y.-S. Lee, K. Papineni, S. Roukos, O. Emam, and H. Hassan, "Language model based arabic word segmentation," in Proceed- ings of the 41st Annual Meeting on Association for Computa- tional Linguistics-Volume 1. Association for Computational Linguistics, pp. 399–406, 2003.

[52] E. A. Kaur and E. T. Singh, “Segmentation of continuous punjabi speech signal into syllables,” in Proceedings of the World Congress on Engineering and Computer Science, vol. 1. Citeseer, pp. 20–22, 2010.

[53] T. Nagarajan, H. A. Murthy, and R. M. Hegde, “Segmentation of speech into syllable-like units,” Energy, vol. 1, no. 1.5, p. 2, 2003.

[54] R. Thangarajan and A. Natarajan, “Syllable based continuous speech recognition for tamil,” South Asian language review, vol. 18, no. 1, pp. 72–85, 2008.

[55] M. Sharma and R. Mammone, “” blind” speech segmentation: Automatic segmentation of speech without linguistic knowl- edge”, 1996.

[56] O. Scharenborg, V. Wan, and M. Ernestus, “Unsupervised speech segmentation: An analysis of the hypothesized phone boundaries,” The Journal of the Acoustical Society of America, vol. 127, no. 2, pp. 1084–1095, 2010.

[57] A. SaiJayram, V. Ramasubramanian, and T. Sreenivas, “Robust parameters for automatic segmentation of speech,” in Acoustics, Speech, and Signal Processing (ICASSP), 2002 IEEE Interna- tional Conference on, vol. 1., pp. I–513, 2002.

[58] F. Schiel, “Automatic phonetic transcription of non-prompted speech”, 1999.

[59] Jettmarová, Zuzana. "Review of “Investigating translation: Selected papers from the 4th International Congress on Translation” by Allison Beeby, Doris Ensinger and Marisa Presas (eds.)." Target. International Journal of Translation Studies” 15.1: 153-158, 2003.

[60] D. T. Toledano, L. A. H. Go´ mez, and L. V. Grande, “Automatic phonetic segmentation,” IEEE transactions on speech and audio processing, vol. 11, no. 6, pp. 617–625, 2003.

[61] I. Mporas, T. Ganchev, and N. Fakotakis, “A hybrid architecture for automatic segmentation of speech waveforms,” in 2008 IEEE International Conference on Acoustics, Speech and Signal Processing. IEEE, pp. 4457–4460, 2008.

[62] J. A. Go´ mez and M. Calvo, “Improvements on automatic speech segmentation at the phonetic level,” in Iberoamerican Congress on Pattern Recognition. Springer, pp. 557–564, 2011.

[63] S. M. Siniscalchi, P. Schwarz, and C.-H. Lee, “High-accuracy phone recognition by combining high-performance lattice gen- eration and knowledge based rescoring,” in 2007 IEEE Interna- tional Conference on Acoustics, Speech and Signal Processing-ICASSP’07, vol. 4. IEEE, pp. IV–869, 2007.

[64] K. Knill and S. Young, “Hidden markov models in speech and language processing,” in Corpus-based methods in language and speech processing. Springer, pp. 27–68, 1997.

[65] B.-H. Juang and L. R. Rabiner, “Automatic speech recognition– a brief history of the technology development,” Georgia Insti- tute of Technology. Atlanta Rutgers University and the Univer- sity of California. Santa Barbara, vol. 1, p. 67, 2005.

[66] N. Chowdhury, M. A. Sattar, and A. K. Bishwas, “Separating words from continuous bangla speech,” Global Journal of Computer Science and Technology, vol. 4, pp. 172–175, 2009.

[67] C.-T. Hsieh, “Segmentation of continuous speech into phonemic units,” IEICS, pp. 420–424, 1991.

[68] M. M. Rahman, M. F. Khan, and M. A. Moni, “Speech recognition front-end for segmenting and clustering continuous bangla speech,” Daffodil International University Journal of Science and Technology, vol. 5, no. 1, pp. 67–72, 2010.

[69] T. Zhang and C.-C. Kuo, “Hierarchical classification of audio data for archiving and retrieving,” in Acoustics, Speech, and Sig- nal Processing, 1999. Proceedings., 1999 IEEE International Conference on, vol. 6. IEEE, pp. 3001–3004, 1999.

[70] G. Hemakumar and P. Punitha, “Automatic segmentation of kannada speech signal into syllables and sub-words: noised and noiseless signals,” International Journal of Scientific & Engineering Research, vol. 5, no. 1, pp. 1707–1711, 2014.

[71] M. M. Rahman and M. Bhuiyan, “Continuous bangla speech segmentation using short-term speech features extraction ap- proaches,” International Journal of Advanced Computer Sci- ences and Applications, vol. 3, no. 11, p. 485, 2012.

[72] I. Khaing and L. KZin, “Automatic speech segmentation for myanmar language”, 2014.

[73] M. Kalamani, S. Valarmathy, and S. Anitha, “Hybrid speech segmentation algorithm for continuous speech recognition”, 2015.

[74] J.-J. Chen, , and L.-S. Lee, “Automatic segmentation techniques for mandarin speech recognition,” in International Computer Symposium, Tamkang University, Taipei (1988.12), 1988.

[75] 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.

[76] R. Niederjohn and J. Grotelueschen, “The enhancement of speech intelligibility in high noise levels by high-pass filtering followed by rapid amplitude compression,” IEEE Transactions on Acoustics, Speech, and Signal Processing, vol. 24, no. 4, pp.277–282, 1976.

[77] T. Giannakopoulos, “Study and application of acoustic infor- mation for the detection of harmful content, and fusion with visual information,” Department of Informatics and Telecom- munications, vol. PhD. University of Athens, Greece, 2009.

[78] S. Ratsameewichai, N. Theera-Umpon, J. Vilasdechanon, S. Ua- trongjit, and K. Likit-Anurucks, “Thai phoneme segmentation using dual-band energy contour,” ITC-CSCC: 2002 Proceed- ings, pp. 111–113, 2002.

[79] J. P. Bello, L. Daudet, S. Abdallah, C. Duxbury, M. Davies, and M. B. Sandler, “A tutorial on onset detection in music signals,” IEEE Transactions on speech and audio processing, vol. 13, no. 5, pp. 1035–1047, 2005.

[80] Sturim, Douglas E., et al. "Speaker verification using text-constrained Gaussian mixture models." 2002 IEEE International Conference on Acoustics, Speech, and Signal Processing. Vol. 1. IEEE, 2002.

[81] Huang, Chengquan, et al. "A strategy for estimating tree canopy density using Landsat 7 ETM+ and high resolution images over large areas", 2001.

[82] Brugnara, Fabio, Daniele Falavigna, and Maurizio Omologo. "Automatic segmentation and labeling of speech based on Hidden Markov Models." Speech Communication 12.4, 1993.

[83] Kim, Yeon-Jun, and Alistair Conkie. "Automatic segmentation combining an HMM-based approach and spectral boundary correction." INTERSPEECH, 2002.

[84] Hansen, J. H., & Pellom, B. L. An effective quality evaluation protocol for speech enhancement algorithms. In ICSLP (Vol. 7, pp. 2819-2822), 1998.

[85] Sharma, Manish, and Richard Mammone. "" Blind" speech segmentation: automatic segmentation of speech without linguistic knowledge." Proceeding of Fourth International Conference on Spoken Language Processing. ICSLP'96. Vol. 2. IEEE, 1996.

[86] Bridle, J., and N. Sedgwick. "A method for segmenting acoustic patterns, with applications to automatic speech recognition." ICASSP'77. IEEE International Conference on Acoustics, Speech, and Signal Processing. Vol. 2. IEEE, 1977.

[87] Hou, J., Rabiner, L., & Dusan, S. Automatic speech attribute transcription (asat)-the front end processor. In 2006 IEEE International Conference on Acoustics Speech and Signal Processing Proceedings (Vol. 1, pp. I-I). IEEE, 2006.

[88] Padmanabhan, Jayashree, and Melvin Jose Johnson Premkumar. "Machine learning in automatic speech recognition: A survey." IETE Technical Review 32.4, 2015.

[89] M. A. Anusuya and S. Katti, ―Front end analysis of speech recognition: a review‖, Int. J. Speech Technology, vol. 14, no. 2, pp. 99–145, 2011.

[90] J. Li, L. Deng, Y. Gong and R.H.-Umbach, ―An Overview of Noise-Robust Automatic Speech Recognition‖, IEEE/ACM Transactions on Audio, Speech, and Language Processing, vol. 22, no. 4, pp. 745 – 777, 2014.

[91] S. B. Davis, and P. Mermelstein, "Comparison of Parametric Representations for Monosyllabic Word Recognition in Continuously Spoken Sentences", IEEE Transactions on Acoustics, Speech, and Signal Processing, vol. 28, no. 4, pp. 357–366, 1980.

[92] D. Jurafsky and J. H. Martin, ―Speech and Language Processing - An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition‖, Prentice Hall, 2009.

[93] R. Lawrence and B.-H. Juang ―Fundamentals of Speech Recognition‖, Prentice-Hall, Inc., (Engelwood, NJ), 1993.

[94] Davis DA, Chawla NV, Christakis NA, Barabási A-L. Time to CARE: a collaborative engine for practical disease prediction. Data Min Knowl Disc. ;20(3):388–415, 2010.

[95] T. M. Mitchell, "Machine learning WCB": McGraw-Hill Boston, MA, 1997.

[96] Sebastiani F. Machine learning in automated text categorization. ACM Comput Surveys (CSUR), 34(1):1–47, 2002.

[97] Sinclair C, Pierce L, Matzner S. An application of machine learning to network intrusion detection. In: Computer Security Applications Conference, 1999. (ACSAC'99) Proceedings. 15th Annual, p. 371–7. IEEE, 1999.

[98] Sahami M, Dumais S, Heckerman D, Horvitz E. A Bayesian approach to filtering junk e-mail. In: Learning for Text Categorization: Papers from the 1998 workshop, vol. 62, p. 98–105. Madison, Wisconsin, 1998.

[99] Aleskerov E, Freisleben B, Rao B. Cardwatch: A neural network based database mining system for credit card fraud detection. In: Computational Intelligence for Financial Engineering (CIFEr), Proceedings of the IEEE/IAFE, p. 220–6. IEEE, 1997.

[100] Kim E, Kim W, Lee Y. Combination of multiple classifiers for the customer's purchase behavior prediction. Decis Support Syst, 34(2):167–75, 2003.

[101] Mahadevan S, Theocharous G. "Optimizing Production Manufacturing Using Reinforcement Learning," in FLAIRS Conference, p. 372–7, 1998.

[102] Yao D, Yang J, Zhan X. A novel method for disease prediction: hybrid of random forest and multivariate adaptive regression splines. J Comput, 8(1):170–7, 2013.

[103] R. S. Michalski, J. G. Carbonell, and T. M. Mitchell, Machine learning: an artificial intelligence approach. Springer Science & Business Media, 2013.

[104] Quinlan JR. Induction of decision trees. Mach Learn, 1(1):81–106, 1986.

[105] Cruz JA, Wishart DS. Applications of machine learning in cancer prediction and prognosis. Cancer Informat, 2:59–77, 2006.

[106] Lindley DV. Fiducial distributions and Bayes' theorem. J Royal Stat Soc. Series B (Methodological), 1958.

[107] I. Rish, "An empirical study of the naive Bayes classifier," in IJCAI 2001 workshop on empirical methods in artificial intelligence, vol. 3, 22, pp. 41–46: IBM New York, 2001.

[108] Uddin, Shahadat, et al. "Comparing different supervised machine learning algorithms for disease prediction." BMC medical informatics and decision making 19.1, 2019.

[109] Rabiner, Lawrence, and Biinghwang Juang. "An introduction to hidden Markov models." ieee assp magazine 3.1, 1986.

[110] Alshaikhkhalil, Iyas AI. Automatic Spoken Qur'anic Phonemes The Islamic University of Gaza, 2018.

[111] Aggarwal, Charu C., et al. "Active learning: A survey." Data Classification: Algorithms and Applications. CRC Press, 2014.

[112] Cabreira, Ariel G., Martín Tripode, and Adrián Madirolas. "Artificial neural networks for fish-species identification." ICES Journal of Marine Science 66.6, 2009.

[113] McCulloch, Warren S., and Walter Pitts. "A logical calculus of the ideas immanent in nervous activity." The bulletin of mathematical biophysics 5.4, 1943.

[114] Rumelhart, David E., Geoffrey E. Hinton, and James L. McClelland. "A general framework for parallel distributed processing." Parallel distributed processing: Explorations in the microstructure of cognition 1.45-76, 1986.

[115] Gershenson, Carlos. "Artificial neural networks for beginners." arXiv preprint cs/0308031, 2003.

[116] Waibel, T. Hanazawa, G. Hinton, K. Shikano, and K. Lang,"Phoneme recognition using time-delay neural networks," IEEE Trans. Acoust. Speech Signal Process., Vol. 37, pp.328_39, Mar, 1989.

[117] Waibel, T. Hanazawa, G. Hinton, K. Shikano, and K. Lang,"Phoneme recognition using time-delay neural networks," IEEE Trans. Acoust. Speech Signal Process., Vol. 37, pp.328_39, Mar, 1989.

[118] M. L. Jordan, "Serial order: A parallel distributed processingapproach," UCSD, Tech. Rep. 8604, La Jolla, California, 1986.

[119] Zeiler, M.D. and R. Fergus. Visualizing and understanding convolutional networks. in European conference on computer vision, 2014.

[120] Szegedy, C., et al. Going deeper with convolutions. in Proceedings of the IEEE conference on computer vision and pattern recognition, 2015.

[121]Simonyan, K. and A. Zisserman, Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556, 2014.

[122] Krizhevsky, A., I. Sutskever, and G.E. Hinton. Imagenet classification with deep convolutional neural networks. in Advances in neural information processing systems, 2012.

[123] He, K., et al. Deep residual learning for image recognition. in Proceedings of the IEEE conference on computer vision and pattern recognition, 2016.

[124] Bhatnagar, S., et al., Prediction of aerodynamic flow fields using convolutional neural networks. Computational Mechanics, 64(2): p. 525-545, 2019.

[125] Nevavuori, P., N. Narra, and T. Lipping, Crop yield prediction with deep convolutional neural networks. Computers and Electronics in Agriculture, 2019.

[126] Ajami, A., et al., Identifying a slums' degree of deprivation from VHR images using convolutional neural networks. Remote Sensing, 2019.

[127] Lossau, T., et al., Motion estimation and correction in cardiac CT angiography images using convolutional neural networks. Computerized Medical Imaging and Graphics, 2019.

[128] Kong, Z., et al., Condition monitoring of wind turbines based on spatio-temporal fusion of SCADA data by convolutional neural networks and gated recurrent units. Renewable Energy, 146: p. 760-768, 2020.

[129] Min, S., B. Lee, and S. Yoon, Deep learning in bioinformatics. Briefings in bioinformatics, 18(5): p. 851-869, 2017.

[130] Zhu, S., et al., Gaussian mixture model coupled recurrent neural networks for wind speed interval forecast. Energy Conversion and Management, 2019.

[131] Pan, B., X. Xu, and Z. Shi, Tropical cyclone intensity prediction based on recurrent neural networks. Electronics Letters, 55(7): p. 413-415, 2019.

[132] Bisharad, D. and R.H. Laskar, Music genre recognition using convolutional recurrent neural network architecture. Expert Systems, 2019.

[133] Zhong, C., et al., Inland Ship Trajectory Restoration by Recurrent Neural Network. Journal of Navigation, 2019.

[134] Jarrah, M. and N. Salim, A recurrent neural network and a discrete wavelet transform to predict the Saudi stock price trends. International Journal of Advanced Computer Science and Applications, 10(4): p. 155-162, 2019.

[135] Yin, Z. and J. Zhang, Cross-session classification of mental workload levels using EEG and an adaptive deep learning model. Biomedical Signal Processing and Control, 33: p. 30-47, 2017.

[136] Sun, W., B. Zheng, and W. Qian, Automatic feature learning using multichannel ROI based on deep structured algorithms for computerized lung cancer diagnosis. Computers in biology and medicine, 89: p. 530-539, 2017.

[137] Chen, Y., et al., Indoor location method of interference source based on deep learning of spectrum fingerprint features in Smart Cyber-Physical systems. Eurasip Journal on Wireless Communications and Networking, 2019. 2019.

[138] Liu, P., P. Zheng, and Z. Chen, Deep learning with stacked denoising auto-encoder for short-term electric load forecasting. Energies, 2019.

[139] Nicolai, A. and G.A. Hollinger, Denoising Autoencoders for Laser-Based Scan Registration. IEEE Robotics and Automation Letters, 3(4): p. 4391-4398, 2018.

[140] Yue, L., et al., Multiple Auxiliary Information Based Deep Model for Collaborative Filtering. Journal of Computer Science and Technology, 33(4): p. 668-681, 2018.

[141] Roy, S.S., M. Ahmed, and M.A.H. Akhand, Noisy image classification using hybrid deep learning methods. Journal of Information and Communication Technology, 17(2): p. 233-269, 2018.

[142] Tan, Z., et al., Denoised Senone I-Vectors for Robust Speaker Verification. IEEE/ACM Transactions on Audio Speech and Language Processing, 26(4): p. 820-830, 2018.

[143] Zhang, Q., et al., Deep learning based classification of breast tumors with shear-wave elastography. Ultrasonics, 72: p. 150-157, 2016.

[144] Wulsin, D., et al., Modeling electroencephalography waveforms with semi-supervised deep belief nets: fast classification and anomaly measurement. Journal of neural engineering, 8(3): p. 036015, 2011.

[145] Patterson, J. and A. Gibson, Deep Learning: A Practitioner's Approach, " O'Reilly Media, Inc.", 2017.

[146] Vieira, S., W.H. Pinaya, and A. Mechelli, Using deep learning to investigate the neuroimaging correlates of psychiatric and neurological disorders: Methods and applications. Neuroscience & Biobehavioral Reviews, 74: p. 58-75, 2017.

[147] Hassan, M.M., et al., Human emotion recognition using deep belief network architecture. Information Fusion, 51: p. 10-18, 2019.

[148] Cheng, Y., et al., Deep belief network for meteorological time series prediction in the internet of things. IEEE Internet of Things Journal, 6(3): p. 4369-4376, 2019.

[149] Yu, Y., et al., Forecasting a short-term wind speed using a deep belief network combined with a local predictor. IEEJ Transactions on Electrical and Electronic Engineering, 14(2): p. 238-244, 2019.

[150] Zheng, J., X. Fu, and G. Zhang, Research on exchange rate forecasting based on deep belief network. Neural Computing and Applications, 31: p. 573-582, 2019.

[151] Ahmad, M., et al., Deep Belief Network Modeling for Automatic Liver Segmentation. IEEE Access, 7: p. 20585-20595, 2019.

[152] Ronoud, S. and S. Asadi, An evolutionary deep belief network extreme learning-based for breast cancer diagnosis. Soft Computing, 2019.

[153] Mesri Gundoshmian, T., Ardabili, S., Mosavi, A., Varkonyi-Koczy, A., Prediction of combine harvester performance using hybrid machine learning modeling and re-sponse surface methodology, Preprints 2019.

[154] Ardabili, S., Mosavi, A., Varkonyi-Koczy, A., Systematic review of deep learning and machine learning models in biofuels research, Preprints 2019.

[155] Ardabili, S., Mosavi, A., Varkonyi-Koczy, A., Advances in machine learning model-ing reviewing hybrid and ensemble methods, Preprints 2019.

[156] Ardabili, S., Mosavi, A., Varkonyi-Koczy, A., Building Energy information: demand and consumption prediction with Machine Learning models for sustainable and smart cities, Preprints 2019.

[157] Ghimire, S., et al., Deep solar radiation forecasting with convolutional neural network and long short-term memory network algorithms. Applied Energy, 2019.

[158] Liu, Y., Novel volatility forecasting using deep learning–Long Short Term Memory Recurrent Neural Networks. Expert Systems with Applications, 132: p. 99-109, 2019.

[159] Hong, J., Z. Wang, and Y. Yao, Fault prognosis of battery system based on accurate voltage abnormity prognosis using long short-term memory neural networks. Applied Energy, 2019.

[160] Krishan, M., et al., Air quality modelling using long short-term memory (LSTM) over NCT-Delhi, India. Air Quality, Atmosphere and Health, 12(8): p. 899-908, 2019.

[161] Zhang, R., et al., Deep long short-term memory networks for nonlinear structural seismic response prediction. Computers and Structures, 220: p. 55-68, 2019.

[162] Hua, Y., et al., Deep Learning with Long Short-Term Memory for Time Series Prediction. IEEE Communications Magazine, 57(6): p. 114-119, 2019.

[163] Zhang, J., et al., Short-term forecasting and uncertainty analysis of wind turbine power based on long short-term memory network and Gaussian mixture model. Applied Energy: p. 229-244, 2019.

[164] Vardaan, K., et al., Earthquake trend prediction using long short-term memory RNN. International Journal of Electrical and Computer Engineering, 9(2): p. 1304-1312, 2019.

[165] L. Deng and D. Yu. Deep Learning for Signal and Information Processing. Microsoft, 2013.

[166] G. Hinton, L. Deng, D. Yu, A. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. Sainath, G. Dahl, and B. Kingsbury. Deep Neural Networks for Acoustic Modeling in Speech Recognition. IEEE Signal Processing Magazine, 29(6):82–97, November 2012.

[167] F.O. Karray and C. De Silva. Soft Computing and Intelligent Systems Design: Theory, Tools and Applications. Addison Wesley, 2004.

[168] Y. Bengio. Learning deep architectures for AI. Foundations and Trends in Machine Learning, 2(1):1–127, 2009.

[169] A. Itamar, R. Derek, and T. P. Karnowski. Deep Machine Learning - A New Frontier in Artificial Intelligence Research. IEEE Computational Intelligence Mag., 5(4):13–18, 2010.

[170] L. Deng. Three Classes of Deep Learning Architectures and Their Applications: A Tutorial Survey. APSIPA Transactions on Signal and Information Processing, 2012.

[171] F. Jelinek. Statistical Methods for Speech Recognition. MIT Press, Cambridge, MA, USA, 1997.

[172] A. Mohamed, G. Dahl, and G. Hinton. Deep Belief Networks for Phone Recognition. In NIPS Workshop on Deep Learning for Speech Recognition and Related Applications, 2009.

[173] A. Mohamed, G. Dahl, and G. Hinton. Acoustic Modeling Using Deep Belief Networks. Audio, Speech, and Language Processing, IEEE Transactions on, 20(1):14 – 22, jan. 2012.

[174] L. Deng, G. Hinton, and B. Kingsbury. New types of deep neural network learning for speech recognition and related applications: An overview. In in Proc. Int. Conf. Acoust., Speech, Signal Process, 2013.

[175] Wang, Linlin, et al. "Improved DNN-based segmentation for multi-genre broadcast audio." 2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE, 2016.

[176] ten Bosch, Louis, and Thomas Niesler. "Unconstrained speech segmentation using deep neural networks." International conference on pattern recognition applications and methods. Vol. 2. SCITEPRESS, 2015.

[177] D. Yu, L. Deng, and G. Dahl. Roles of Pre-Training and Fine-Tuning in Context-Dependent DBN-HMMs for Real-World Speech Recognition. In NIPS workshop on Deep Learning and Unsupervised Feature Learning, December 2010.

[178] A. Mohamed, G. Hinton, and G. Penn. Understanding How Deep Belief Networks Perform Acoustic Modelling. In ICASSP, 2012.

[179] Yu, D., Deng, L.: Automatic Speech Recognition - A Deep Learning Approach. Springer,London , 2015.

[180] Povey, D. et al.: The Kaldi speech recognition toolkit. In: IEEE Workshop on Automatic Speech Recognition and Understanding ASRU, 2011.

[181] Veselý, K. et al.: Sequence-discriminative training of deep neural networks. In: INTERSPEECH 2013, pp. 2345–2349, 2013.

[182] Povey, D., Zhang, X., Khudanpur, S.: Parallel training of DNNs with natural gradient and parameter averaging. preprint arXiv:1410.7455 http://arxiv.org/pdf/1410.7455v8.pdf, 2014.

[183] Bengio, Y., Lamblin, P., Popovici, D., Larochelle, H.: Greedy layer-wise training of deep networks. Adv. Neural Inf. Process. Syst. (NIPS) 19, 153–160, 2007.

[184] Seide, F., Li, G., Yu, D.: Conversational speech transcription using context-dependent deep neural networks. In: INTERSPEECH-2011, pp. 437–440, 2011.

[184] X.Huang, A. Acero, and H. Hon, Spoken Language Processing, Prentice Hall PTR, 2001.

[185] S. Cox, R. Brady, and P. Jackson, “ Techniques for accurate automatic annotation of speech waveforms”, inProceedings of the International Conference on Spoken Language Processing, Vol V., Sydney, NSW, pp. 1947-1950, 1998.

[186] Malfrère, Fabrice, Olivier Deroo, and Thierry Dutoit. "Phonetic alignment: speech synthesis based vs. hybrid HMM/ANN." ICSLP, HMM/ANN”, in Proceedings of the International Conference on Spoken Language Processing, Vol IV., Sydney, NSW, pp. 1571-1574, 1998.

[187] Ljolje, Andrej, Julia Hirschberg, and Jan PH van Santen. "Automatic speech segmentation for concatenative inventory selection." In Progress in speech synthesis, pp. 305-311. Springer, New York, NY, 1997.

[188] inventory selection”, in Progress in Speech Synthesis, J.P.H. Van Santen, Ed: Springer, pp. 305-311, 1997.

[189] T. N. Sainath, B. Kingsbury, B. Ramabhadran, P. Fousek, P. Novak, and A. Mohamed. Making Deep Belief Networks effective for large vocabulary continuous speech recognition. In ASRU, pages 30–35, 2011.

[190] L. Lu. Subspace Gaussian Mixture Models For Automatic Speech Recognition. PhD thesis, Institute of Language, Cognition and Computation, School of Informatics, Unversity of Edinburgh, 2013.

[191] G. Hinton, L. Deng, D. Yu, A. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. Sainath, G. Dahl, and B. Kingsbury. Deep Neural Networks for Acoustic Modeling in Speech Recognition. IEEE Signal Processing Magazine, 29(6):82–97, November 2012.

[192] A. Mohamed, G. Dahl, and G. Hinton. Acoustic Modeling Using Deep Belief Networks. Audio, Speech, and Language Processing, IEEE Transactions on, 20(1):14 – 22, jan. 2012.

[193] G. Dahl, D. Yu, L. Deng, and A. Acero. Context-Dependent Pre-trained Deep Neu- ral Networks for Large Vocabulary speech recognition. IEEE Transactions on Au- dio, Speech, and Language Processing (receiving 2013 IEEE SPS Best Paper Award), 20(1):30–42, January 2012.

[194] P.C. Woodland and D. Povey. Large scale discriminative training of hidden Markov models for speech recognition. Computer Speech & Language, 16(1):25–47, 2002.

[195] Metwalli, S. E. H. M Computer Aided Pronunciation Learning System Using Statistical Based Automatic Speech Recognition Techniques. Faculty of Engineering at Cairo University in Partial Fulfillment of the Requirements for the Degree of DOCTOR OF PHILOSOPHY in ELECTRONICS AND COMMUNICATION ENGINEERING FACULTY OF ENGINEERING, CAIRO UNIVERSITY GIZA, 2005.

[196] L. Deng. Three Classes of Deep Learning Architectures and Their Applications: A Tutorial Survey. APSIPA Transactions on Signal and Information Processing, 2012.

[197] A. Mohamed, G. Dahl, and G. Hinton. Deep Belief Networks for Phone Recognition. In NIPS Workshop on Deep Learning for Speech Recognition and Related Applications, 2009.

[198] G. Dahl, D. Yu, L. Deng, and A. Acero. Large Vocabulary Continuous Speech Recognition With context-dependent DBN-HMMS. In Proc. ICASSP, Prague. IEEE, May 2011.

[199] D. Yu, L. Deng, and G. Dahl. Roles of Pre-Training and Fine-Tuning in Context-Dependent DBN-HMMs for Real-World Speech Recognition. In NIPS workshop on Deep Learning and Unsupervised Feature Learning, December 2010.

[200] V. Peddinti, D. Povey, and S. Khudanpur "A time delay neural network architecture for efficient modeling of long temporal contexts Interspeech", pp 3214–3218, 2015.

[201] A. Mohamed, G. Hinton, and G. Penn. Understanding How Deep Belief Networks Perform Acoustic Modelling. In ICASSP, 2012.

[202] Weninger, Felix, et al. "Speech enhancement with LSTM recurrent neural networks and its application to noise-robust ASR." International conference on latent variable analysis and signal separation. Springer, Cham, 2015.

[203] Snyder, David, et al. "Deep neural network-based speaker embeddings for end-to-end speaker verification." 2016 IEEE Spoken Language Technology Workshop (SLT). IEEE, 2016.

[204] Vu, Ngoc Thang, Jochen Weiner, and Tanja Schultz. "Investigating the learning effect of multilingual bottle-neck features for ASR." Fifteenth Annual Conference of the International Speech Communication Association. 2014.

[205] Cosi, Piero, and Sede Secondaria di Padova–Italy. "Phone recognition experiments on ArtiPhon with KALDI." of the Final Workshop 7 December 2016, Naples. 2016.

[206] Ghai, Wiqas, and Navdeep Singh. "Continuous speech recognition for Punjabi language." International Journal of Computer Applications 72.14, 2013.

[207] Dua, Mohit, Rajesh Kumar Aggarwal, and Mantosh Biswas. "GFCC based discriminatively trained noise robust continuous ASR system for Hindi language." Journal of Ambient Intelligence and Humanized Computing, 2019.

[208] Dahl, G., Yu, D., Deng, L., Acero, A.: Context-dependent pre-trained deep neural networks for large vocabulary speech recognition. IEEE Trans. Audio Speech Lang. Process. 20(1), 30–42, 2012.

[209] Seide, F., Li, G., Yu, D.: Conversational speech transcription using context-dependent deep neural networks. In: INTERSPEECH-2011, pp. 437–440, 2011.

[210] Ellis, D.P.W., Singh, R., Sivadas, S.: Tandem acoustic modeling in large-vocabulary recognition. In: International Conference on Acoustics, Speech and Signal Processing ICASSP 2001, pp. 517–520, 2001.

[211] Grezl, F., Karafiat, M., Kontar, S., Cernocky, J.: Probabilistic and bottle-neck features for LVCSR of meetings. In: ICASSP 2007, pp. 757–760, 2007.

[212] Maas, A.L. et al.: Building DNN Acoustic Models for Large Vocabulary Speech Recognition. preprint arXiv:1406.7806, http://arxiv.org/pdf/1406.7806.pdf, 2015.

[213] Cosi, P.: A KALDI-DNN-based ASR system for Italian. In: IEEE International Joint Conference on Neural Networks IJCNN, pp. 1–5, 2015.

[214] Popović, B., Ostrogonac, S., Pakoci, E., Jakovljević, N., Delić, V.: Deep neural network based continuous speech recognition for Serbian using the Kaldi toolkit. In: Ronzhin, A., Potapova, R., Fakotakis, N. (eds.) SPECOM, LNCS, vol. 9319, pp. 186–192. Springer, Heidelberg, 2015.

[215] Miao, Y.: Kaldi+PDNN: building DNN-based ASR systems with Kaldi and PDNN. arXiv preprint arXiv:1401.6984, https://arxiv.org/abs/1401.6984, 2014.

[216] Mandery, Christian. Distributed N-Gram Language Models: Application of Large Models to Automatic Speech Recognition. Diss. Doctoral Dissertation, Informatics Institute, 2011.

[217] Keselj, Vlado. "Book Review: Speech and Language Processing by Daniel Jurafsky and James H. Martin." Computational Linguistics 35.3 (2009).

[218] Young, Steve, et al. "The HTK book." Cambridge university engineering department 3.175, 2002.

[219] Krishnaiah, R. R., and L. N. Kanal. "Dimensionality and Sample Size Considerations in Pattern Recognition Practice. Handbook of Statistics." , 1982.

[220] Raudys, S. J., and Anil K. Jain. "Small sample size effects in statistical pattern recognition: recommendations for practitioners and open problems." [1990] Proceedings. 10th International Conference on Pattern Recognition. Vol. 1. IEEE, 1990.

[221] Gałka, Jakub, and Bartosz Ziółko. "Study of performance evaluation methods for non-uniform speech segmentation." International Journal Of Circuits, Systems And Signal Processing, NAUN, 2008.

[222] Khan, Arif, and Ingmar Steiner. "Qualitative Evaluation and Error Analysis of Phonetic Segmentation." Studientexte zur Sprachk ommunikation: Elektronische Sprachsignalverarbeitung , 2017.