A Comparison of the Network Speech Recognition and Distributed Speech Recognition Systems and their effect on Speech Enabling Mobile Devices

Prepared by: Dale Isaacs

Supervised by: Dr Daniel J. Mashao

Speech Technology and Research Group
Department of Electrical Engineering
University of Cape Town
February 2010

This dissertation is submitted to the University of Cape Town in fulfillment of the academic requirements for the Degree of Master of Science in Engineering

Declaration

The work in this thesis is based on research carried out in the Speech Technology and Research Group at the University of Cape Town, South Africa. No part of this thesis has been submitted elsewhere for any other degree or qualification and it is all my own work unless referenced to the contrary in the text.

Signature of Author : .....

Dale Isaacs

February 2010

Acknowledgments

First and foremost, I want to thank God for keeping me safe thus far and giving me strength to focus on my work throughout this period of time. Thanks must go to Dr D.J. Mashao who as my supervisor has guided and shared with me his knowledge and experience, thank you. Thanks to all the members of the STAR research group at UCT, who were always on hand to assist me. To all those who willingly volunteered to help me by proof reading, many thanks!! To all my friends who encouraged and believed in me, this was truly appreciated.

To my family, especially my mother and my father who stood by me throughout my studies, always showing much love and support, thank you. Without you I would not be here and without your guidance and love, would not have become the person who I am today. To Astrid, thanks for all the encouragement, love and support, it has been a long and hard road. To my brother Chad, for your comic relief...thanks. To everyone else whom I have not mentioned, but who has contributed to my success, thank you all.

Abstract

Over the past 10 years there has been an exponential increase in the number of mobile subscribers worldwide. Market research has shown that the number of mobile subscribers rose to 4.3 billion towards end of Q1 in 2009. The unprecedented development of the telecommunication industry over the last decade has brought about the need for ubiquitous access to a host of different information resources and services. Today, speech remains the best medium of communication between people and it is conceivable that speech enabling mobile devices will allow users who only have mobile devices, to access all the information which is now available over the world wide web.

In this context, the term *"speech enabling"* mobile devices refers to enhancing the usability of existing applications on mobile devices to include speech recognition functionalities e.g. When sending an SMS, instead of using the keypad as an input mode, use voice (speech) to enter the message. This type of input modality for mobile devices will also provide an interface for those who only have speech at their disposal, such as the visually impaired and various other physically challenged people who mostly rely on the use of their ears, hands and speech to perform daily tasks.

Today automatic speech recognition (ASR) systems and text-to-speech (TTS) systems are quite well established. These systems, using the latest technologies, are operating at accuracies in excess of 90%. It must be taken into consideration that the main reason for their high recognition accuracy is that they operate on high powered desktop machines and servers. When implementing ASR on mobile devices, there are many challenges which need to be overcome. These include the physical size of the device, processing power capability, memory capacity and battery power,

all of which would be required to perform complex calculations.

This thesis is primarily aimed at investigating techniques for implementing automatic speech recognition on mobile devices and developing a baseline system for future development in the speech research field. The two techniques which are compared in this thesis are Network Speech Recognition (NSR) and Distributed Speech Recognition (DSR). The well-known speech database, TIMIT is used when comparing these two techniques. The test framework which was developed in this thesis used the European Telecommunications Standards Institute (ETSI) front-end for the mobile client and a Sphinx-4 decoder for the back-end server. The results showed that for a NSR implementation using a sub-set of the TIMIT database a recognition accuracy of 56.27% was obtained together with a word-error-rate (WER) of 16.19%. The DSR implementation proved more successful with a recognition accuracy of 75.63% and a WER of only 9.79%. This meant that there was an overall improvement of 34.40% in the recognition accuracy and a 39.53% improvement in the WER using the DSR baseline system. The framework which was developed in this study provides a good platform for future research in the speech technology area.

Contents

List of Figures

List of Tables

List of Acronyms and Abbreviations

ASR - Automatic Speech Recognition

NSR - Network Speech Recognition

DSR - Distributed Speech Recognition

PC - Personal Computer

TTS - Text-to-Speech

SMS - Short Message Service

MMS - Multimedia Message Service

PDA - Personal Digital Assistant

ETSI - European Telecommunications Standards Institute

AFE - Advanced Front-End

DTW - Dynamic Time Warping

HMM - Hidden Markov Model

IVR - Interactive Voice Response

DP - Dynamic Programming

ANN - Artificial Neural Networks

CMS - Cepstral Mean Subtraction

MFCC - Mel-Frequency Cepstral Co-effiecients

OS - Operating System

GSM - Global System for Mobile communications

ITU-T - The Telecommunication Standardization Sector

Kbps - Kilo bits per second

FFT - Fast Fourier Transform

VQ - Vector Quantization

CRC - Cyclical Redundancy Checking

GMM - Gaussian Mixture Models

HTK - Hidden Markov Model Toolkit

AVCSR - Audio-Visual Continuous Speech Recognition

ISIP - Institute for Signal and Information Processing

JSGF - Java Speech API Grammar Format

FST - Finite State Transducer

ASCII - American Standard Code for Information Interchange

RAM - Random Access Memory

CPU - Central Processing Unit

HDD - Hard Disk Drive

WER - Word Error-Rate

SER - Sentence Error-Rate

ALU - Arithmetic Logic Unit

IC - Integrated Circuit

DSP - Digital Signal Processor

SD - Secure Digital

NIST - National Institute of Standards and Technology

Chapter 1

Introduction

Over the past 10 years there has been an exponential increase in the number of mobile subscribers worldwide. Market research has shown that the number of mobile subscribers rose to 4.3 billion towards the end of Q1 in 2009 [1]. At the same time in South Africa, the number of mobile device owners were said to be 51.9 million [2]. A well-known mobile market research group, Portio Research, predicts that the worldwide wireless market will expand to more than 5.8 billion by 2013 [3]. This, together with the unprecedented development of the telecommunication industry over the last decade, has brought about the need for ubiquitous access to a host of different information resources and services.

Today, speech remains the best medium of communication between people and it is conceivable that speech enabling mobile devices will allow users who only have mobile devices, to access all the information which is now available over the world wide web. In this context, the term *"speech enabling"* mobile devices refers to enhancing the usability of existing applications on mobile devices to include speech recognition functionalities e.g. When sending an SMS, instead of using the keypad as an input mode, use voice (speech) to enter the message. In addition this will improve productivity by allowing easier interactions between a user and their device. Recent advances in speech technology have enabled researchers to design automatic speech recognition (ASR) systems that have excellent performance under ideal conditions. Unfortunately, real-world conditions are far from ideal which leads to the degradation in performance of these systems. Users expect that these architectures

will perform reliably across telephone/cellular networks. Consider that a conventional ASR system, which uses a high-powered processor for signal processing, is still subject to degradation when the speech signal is transmitted over a telephone network. Smart mobile devices being developed today are far more powerful than their predecessors, yet the recognition performance suffers because of complex recognition algorithms.

This study investigates available technologies that can be used to implement a speech recognition system on a mobile device and forms a baseline for future mobile speech research.

1.1 Brief introduction to Speech Technology

Speech technology is the broad term which has been given to a whole host of speech applications.

These include:

- Automatic Speech Recognition (ASR) - which converts spoken words into text
- Speech Synthesis - which converts written text into speech (TTS) using a synthesized voice
- Speaker Verification and Identification - these two techniques can be explained by answering the following two questions, (i) who is speaking? and (ii) is the person who is speaking really who they claim to be?

In the area of speech technology, speech synthesis, speaker verification, speaker identification and speaker recognition are already well researched and established. Attempting to implement all of these speech technologies onto a mobile device would require extensive investigation by a number of researchers. These are two of the main reasons why this study focuses only on techniques available to implement an automatic speech recognition system.

The utopia in a speech recognition system is a machine which has the capability to fully recognize and comprehend the spoken language on any subject by any speaker in any environment. Even though research in this field has been on-going

for many years we are still far from realising this vision. According to Rabiner and Juang, automatic speech recognition is defined as the conversion of spoken words into written text [4]. Speech recognition systems can be used to allow humans to control and manipulate computers to perform specific functions, for example; dictation programs or command and control of applications.

The following steps describe an overview of the speech recognition process, starting from the time a user enters a phrase (speech signal) until the machine returns with an appropriate result (phrase) or specific action [4, 5]. Figure 1.1 on the following page shows a block diagram of the ASR process.

- Step 1:
  - *Input*: User enters a phrase into a microphone or telephone and the system captures this acoustic signal.
- Step 2:
  - *Digitization*: Conversion by the system from an analog to digital signal. This converts the input signal into something which approximates the acoustic properties of the human ear.
- Step 3
  - *Phonetic Breakdown*: The specific speech recognition algorithms break down the converted words into basic components of speech. These basic building blocks of speech correspond to vowel and consonant sounds.
- Step 4:
  - *Statistical Modelling*: The system does comparisons between these models and the templates which it stores in its database and tries to find a match.
- Step 5:
  - *Matching*: The specific algorithm then tries to map the possible phonetic representation to phrases or words which are described in its grammar (database).

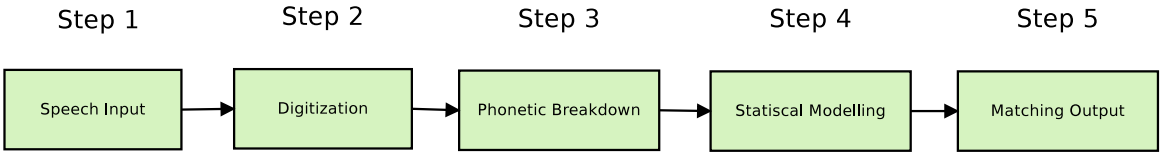

Figure 1.1: Block diagram of the automatic speech recognition process.

1.2 Factors Affecting Speech Recognition Performance on Mobile Devices

ASR and TTS systems are currently well-established. Systems using this technology are operating at accuracies in excess of 90% like the new Dragon Naturally Speaking software from Nuance [6,7], however we need to be cognisant of the fact that most are being run on high-powered desktop and server machines. There are many challenges when implementing ASR on mobile devices, and the following is a list of the main constraints related to the device itself:

- Physical size
  - Dimensions of the phone determine how many integrated circuits (ICs) can be included on the main-board.
  - For example in, speech recognition, it would be useful if the device had a dedicated arithmetic logic unit (ALU) for the speech processing since it requires numerous calculations to occur as the system processes and samples the speech in "real-time" .
- Processing power
  - The functionality and design of the digital signal processor (DSP) to a large extent determined the success or failure of an ASR system. Since ASR places an even greater demand on the DSP, systems require higher-performing DSPs than their predecessors. A good DSP performance requires a high processing speed, low power consumption and high code density.

- Battery power consumption
  - Batteries are important for the mobility and portability of mobile devices and usually run out quickly in most handheld devices. There is however an ongoing effort to reduce consumption of power and increase battery life for mobile devices. Due to the number of calculations which are required in ASR, this could negatively impact on the battery life
- Limited memory
  - In the past, most mobile devices had a limited memory capacity. ASR requires a large amount of free memory for storage the of templates. This is however set to change as newer model phones now have memory card slots where additional memory can be added in the form of secure digital (SD) memory cards.
- Technologies and Standards
  - Until very recently, vendors provided mobile devices without adhering to any particular standards and thus a technique used on one handset would not necessarily work on other. In May 2002, Nokia and Siemens Information and Communication Mobile announced that the companies agreed on a framework of collaboration to create and drive the implementation of mobile terminal software based on open standards [8].
- Connectivity
  - Connectivity is very important to mobile users who are constantly on the move. In the event that connectivity is lost, the ASR implementation on mobile devices should have the ability to save and minimize the loss of data especially in a client-server model where data is being exchanged over the network.

All of these factors contribute greatly to the speech enabling of these devices and thus we have to be careful when implementing an appropriate algorithm to factor in

all of its computational requirements. In addition to the above list, environmental factors should be considered as it could adversely affect the performance.

1.3 Applications for ASR on Mobile Devices

There are many applications for speech recognition on mobile devices, below is a list of some of these:

- SMS entry - sending an SMSs while driving, without having to use key presses
- Dictation - writing notes or documents while on the move
- Command and Control - issuing commands to your phone to access applications without having to touch the device

1.4 Problem Statement

Today, mobile phones are used for much more than just talking. Sending short text messages (SMS), multimedia messages (MMS), checking e-mails (Electronic Mail), weather reports and sports results, are but a few of the applications for which this device is used. Users are searching for faster and easier ways to interact with their devices. Conventional methods use relatively small keypads for user input or a stylus in the case of most PDAs (Personal Digital Assistant). This poses a significant problem especially for disabled persons and for persons who have larger than average sized fingers. This is commonly referred to as the *fat-finger* problem.

Since speech remains the most natural form of communication for human beings, it is logical to conclude that speech recognition will be the best suited form of interaction with smart mobile devices. Implementing a speech recognition system on a mobile device has many limitations and even with the advancements in mobile technology, these devices still suffer from the above-mentioned constraints. The physical size of the device usually limits the maximum screen size which is used to display information to users. Processing power and memory size are crucial when

implementing an ASR system on mobile devices, added to this the application will have to have a low power requirement.

1.5 Research Objectives

The main research objective is to investigate all relevant techniques for implementing an ASR system on mobile devices and to make recommendations as to which provides the best possible solution. This statement is quite broad and can be narrowed down to the following:

- Provide a comprehensive review of how speech recognition works and investigate the associated methodologies, making recommendations of best technique to implement.
- Provide a comprehensive review of the techniques which can be used to implement ASR systems on mobile devices and compare the two most popular techniques.
- Develop a modular testing framework which can be used to compare the two ASR techniques, drawing conclusions and making recommendations about the better one.
- This framework must also form a baseline for future mobile speech research.

1.6 Contribution to Knowledge

The main aim of this study is to report on ASR techniques which are currently available for mobile devices. This study compares two of the current most favourable techniques providing a good platform for future research in this area.

It provides a solid overview of the Sphinx4 speech recognition system together with the European Telecommunications Standards Institute (ETSI) front-end which is a standard for speech recognition front-ends on mobile devices and contributes to further research in the fascinating field of Speech Technology in South Africa.

1.7 Scope and Limitations

This study concentrates on the implementation of an automatic speech recognition system for mobile devices. Text-to-speech processing, speaker identification, speaker verification and speaker recognition were all out of the scope of this study since research in this area is already well-established. All experiments conducted in this study were executed on a university desktop machine and where needed, a normal headset with microphone. The database used for testing was a subset of the TIMIT testing database, a well-known test database for speech research. Acoustic models were trained using TIMIT training database. All tests were conducted in the university laboratory under normal operating conditions (no sound proofing). No special recording equipment was used.

1.8 Plan of Development

The remainder of this thesis is organised as follows:

- Chapter 2 is an introduction to the fundamentals of automatic speech recognition.
- Chapter 3 looks at the current techniques available for implementing ASR on mobile devices. Both the advantages and disadvantages are investigated and the limitations of these techniques are also presented.
- Chapter 4 explores the Sphinx-4 speech recognizer which was developed at the Carnegie Mellon University in Pittsburgh USA. We also investigate the ETSI defined front-end for Distributive Speech Recognition (DSR) which was developed by the European Telecommunications Standards Institute (ETSI) Aurora Working Group.
- Chapter 5 describes the experimental framework which was setup to test the various speech recognition techniques.
- Chapter 6 discusses all the experimental results which were obtained and presents the analysis of the results.

- Chapter 7 provides a summary of all the research done, the achievements of this research, as well as a discussion for future research relating to the concepts explored in this study. Conclusions are drawn and recommendations are made of the best technique(s) to be used for enabling ASR on mobile devices.

1.9 Summary

In this chapter, an introduction into the field of automatic speech recognition was provided. All the objectives and aims of this research were discussed, as well as the scope and limitations of the work presented in the remainder of this thesis. The next chapter discusses the fundamental concepts of automatic speech recognition to give the reader a basic understanding of the processes involved in automatic speech recognition.

Chapter 2

Automatic Speech Recognition Fundamentals

In this chapter a review of the key concepts involved in automatic speech recognition is provided. It is important to understand the history of speech recognition before explaining its concepts, which is why this chapter begins with a brief background followed by an introduction to the human speech production system, and then describes speech signal processing techniques that are used in speech recognition.

2.1 Background of Speech Recognition

Automatic speech recognition or ASR as it is commonly known, is the process by which a computer maps an acoustic speech signal into a text format. ASR allows a machine to convert spoken words into written text or to respond to verbal commands. Research in speech recognition can be traced back to the days of Alexander Graham Bell, who in the early 1870's discovered how to convert air pressure waves (sounds) into electrical impulses [9].

Speech recognition has inspired movies such as the famous *2001 –A Space Odyssey* featuring the computer HAL and the robot R2D2 in the *Star Wars* series. These films illustrate what is commonly known as the *"Holy Grail"* of speech research, showing systems with flawless synthesis and recognition. Appendix A shows an approximate time-line for speech recognition [10].

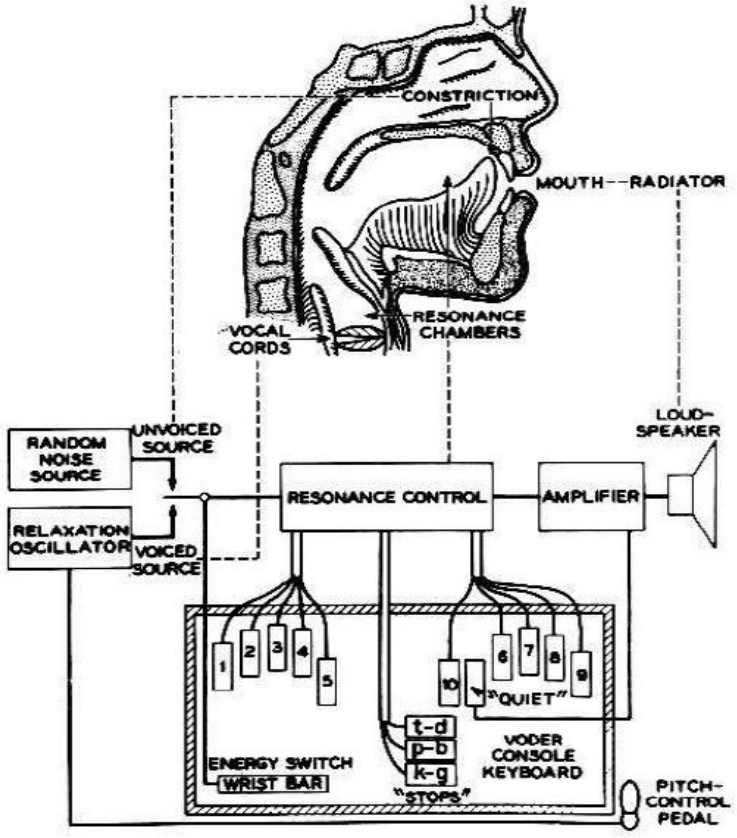

Figure 2.1: A schematic of the VODER system.

The earliest practical implementation was in the early 1930's in AT&T's Bell Labs [10–12] where they produced the first electronic speech synthesizer called the *VODER (Voice Operating Demonstrator).* The initial device was actually called the VOCODER (VOICE CODER) which later became the inspiration for the VODER system [12]. The VOCODER system analyzed speech into varying parameters which drove a synthesizer to do reconstruction of the approximated speech signal [12]. The VODER system, shown in figure 2.1 explains that this system made use of a wrist bar for selecting a voice or noise source, and used a foot pedal for selecting the fundamental pitch of the output. The voice, sometimes referred to as a noise source was then passed through ten bandpass filters, operated by the fingers to produce the final output [12, 13]. Even though this system displayed poor performance in terms of the resulting speech quality, it led to more interest in producing high-quality speech synthesizers.

IBM was the first company to release a dictation system based on speech recog-

nition in 1994 [14]. Since then speech recognition systems have been implemented in Embedded Systems, Telephony Applications and in Multimedia Applications with many improvements being made over the years. With more than thirty years of research and development, today we have reasonable quality and commercially viable speech recognition applications.

2.2 The Human Speech Production System

When trying to understand speech recognition it is important to first understand the process and characteristics of human speech production. This process is more critical when performing speaker verification or speaker identification which is why for the purposes of this study, a broad outline of the process is given in this section.

Figure 2.2 shows the human speech production system. The lungs and trachea, larynx and vocal tract are the three commonly recognized components of the production of human speech [15–17]. Kivimaki describes in his masters thesis, [13], that human speech production as a fluctuation of air pressure, where the air is forced out from the lungs and diaphragm through the epiglottis and passed through the vocal tract. The major organs responsible for speech can be seen in figure 2.2.

The diaphragm and the lungs supply the system with the air-flow needed to initiate the speech process [13]. The trachea is an air flow passage reinforced with rings of cartilage through which the air produced by the lungs and the diaphragm reaches the larynx [18]. The larynx is a complex system of cartilage and muscles containing and controlling the vocal chords. The vocal cords modulate the air flow by opening and closing causing vibrations. These vibrations create the sounds that are finally delivered as speech. In the speech production process, the loudness of the resultant speech is controlled by the lungs and trachea but they make little audible contribution to speech [18].

The glottis is the open space between the upper end of the trachea and vocal cords. The larynx forms an air passage to the lungs and it holds the vocal cords [18]. [12] describes the epiglottis as a cartilage at the root of the tongue that protects the trachea from liquid or food in the swallowing process. When you swallow the

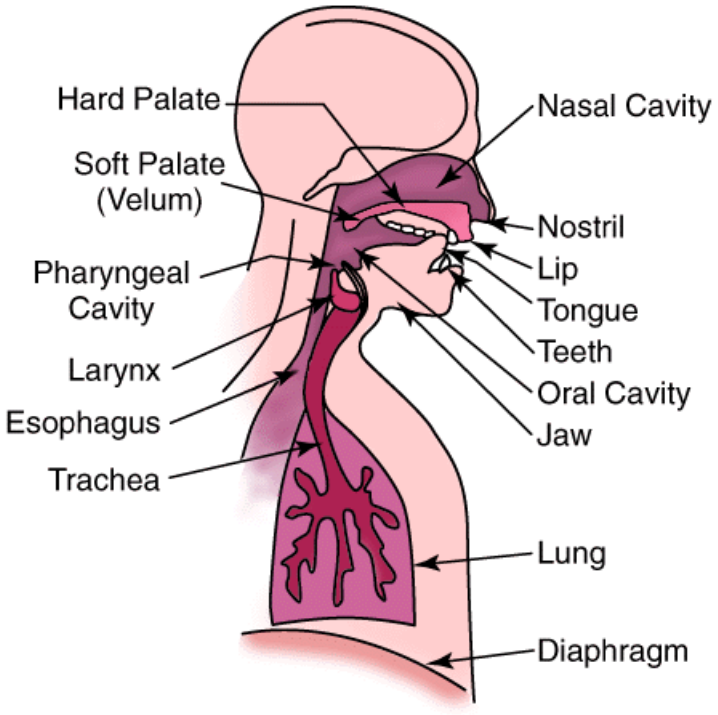

Figure 2.2: The human speech production system.

cartilage is depressed, blocking food or liquid from entering your trachea. The throat or pharynx, is the lined cavity behind the mouth and the nose. The velum or (soft palate) controls the air flow from the pharynx to the oral and nasal cavities. The oral cavity is the main source of speech output and has changeable dimensions [13]. The dimensions are shaped by the movements of the lips, tongue and the velum. The nasal cavity has a fixed shape and dimension and is responsible for producing the nasalized sounds of speech [13].

Understanding how the human speech production system works, improves our understanding of how speech recognition systems should be modeled to achieve greater recognition accuracy.

2.3 Traditional Speech Recognition Methodologies

Automatic speech recognition techniques are classified as either continuous or discrete and can be speaker independent or speaker dependent. This section gives a brief overview of these concepts.

2.3.1 Speaker Dependence and Speaker Independence

Speaker dependence means that the recognition system is trained using only a single voice, preferably that of the person who will use the system. The system will therefore only understand the accent of that particular person who has trained it. Unlike its counterpart, speaker dependence, speaker independence uses a database of voices while training. This system is more favourable as it allows more than one person to interact with it. As you can imagine this form of speech recognition is more error prone as everyone has their own accent and no one person pronounces a word in the same way. When you are sick or have a cold your voice will also change and this adversely affects these kinds of systems.

2.3.2 Context Sensitive and Context Insensitive

Context sensitive means that the system is implemented using a limited dictionary, i.e. only the limited number of words entered are recognized by it. Context insensitive allows for any English word in the dictionary to be spoken because it implements a much larger dictionary than that of the context sensitive system.

2.3.3 Discrete Speech Recognition

In a discrete system, a user will say a sentence using brief pauses between words. This method has proved very successful mainly owing to its clear, defined boundaries and in addition it uses the least amount of processing power. The main problem with this kind of input is that it is not very natural and thus not user friendly. A technique called Dynamic Time Warping (DTW) is used in this type of recognition which will be described in further detail later in this chapter.

2.3.4 Continuous Speech Recognition

Contrary to discrete speech recognition, continuous speech allows for input without pauses, giving it more of a natural feel. The trade-off however is that this kind of system requires more complex computations and uses more processing power than its predecessor. Hidden Markov Models (HMMs) are generally used to implement

this kind of recognition and this technique will be discussed later in the chapter. This technique is more error prone due to the speaker not pausing between each word, altering the speaking rate and possibly overlapping words, which does not occur in discrete speech recognition.

2.3.5 Keyword Spotting

Keyword spotting is used as a bridge between discrete and continuous speech recognition. Systems based on this model are able to identify a word or a group of words in a sentence which correspond to some command. This technique is commonly used in Interactive Voice Response (IVRs) Menus such as the one Multichoice (DSTV) uses on its customer service line.

Traditional IVRs will prompt you for key presses, e.g. "Press 1 for account enquiries or 2 for technical problems". In this case you would be prompted for a word, e.g. "technical problem" and the system will automatically route you to the closest match.

2.4 Speech Recognition Algorithms

This section looks at the most popular speech recognition algorithms being used today. Dynamic Time Warping, Hidden Markov Models and Neural Networks.

2.4.1 Dynamic Time Warping

This algorithm is a technique which is applied in ASR to cope with different speaking speeds, for example the duration of one utterance of speech could be different to another utterance of the same word [19]. More specifically it is a method that allows us to find an optimal match between two given sequences with certain restrictions. There are two main concepts in dynamic time warping (DTW), i.e. features and distances. Features, because the information in each signal needs to be represented in some form or another and distances because we need some metric which will help us in the matching process [19]. There are two types of distances which will be used, (i) local distances which are a computational difference between a feature

of one signal and a feature of the other [19] and (ii) global distances which are the overall computational differences between an entire signal and another signal of possibly different length. This algorithm uses frame based feature extraction where the feature analysis consists of calculating a feature vector at regular periods. The feature vectors contain multiple elements and therefore we need a way of calculating the local distance.

This local distance is measured by using the Euclidean distance metric shown below [20]:

$$d(x,y) = \sqrt{\sum_{j} (x_j - y_j)^2} \tag{2.1}$$

This metric is known to be more computationally expensive than others but gives more weighting to larger differences in a single feature.

2.4.1.1 Symmetrical DTW

There are two types of DTW, symmetrical and asymmetrical. For the purpose of this project we will only be concerned with symmetrical DTW since it is simpler and works for our application. In order to obtain a global distance between two speech patterns a time alignment needs to be performed. In figure 2.3, this problem is illustrated with the input speech running along the x-axis and the template speech up the y-axis [19]. This figure also indicates that at position (i,j), the previous position could only have been (i-1,j), (i-1,j-1) or (i,j-1) due to the applied restrictions.

The global distance score is calculated by simply adding the local distances, the recursive formula is shown in equation 2.2:

$$D(i,j) = min[D(i-1,j-1), D(i-1,j), D(i,j-1)] + d(i,j) \tag{2.2}$$

Since we need an efficient algorithm we impose certain restrictions, [19] listed below: ·

- Matching paths cannot move backwards in time.
- Every input frame must be used in a matching path.

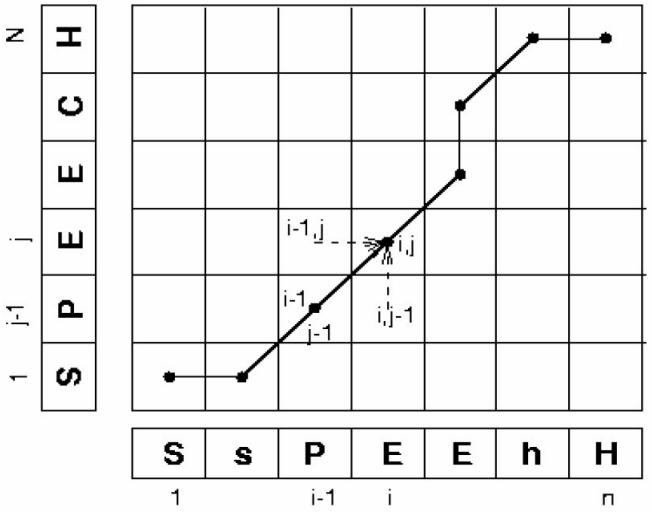

Figure 2.3: Illustration of a noisy signal "SSPEEHH" being compared to the clean template "SPEECH".

- Local distance scores are combined to give a global distance.

This algorithm is better known as dynamic programming (DP), but when applied to template-based speech recognition it is known as dynamic time warping. In general DP is guaranteed to find the lowest distance path through the matrix, while minimising the amount of computation. This algorithm is computationally expensive but has proven in the past to produce good results. This function reads in a speech sample and then performs a time alignment with that speech signal and every template in the database and then selects the template with the overall lowest distance cost. This, unlike the other algorithms, has to be completed at run-time and therefore has a certain amount of latency associated with it.

2.4.2 Hidden Markov Models

The Hidden Markov Model or HMM is one of the latest speech recognition technologies currently being used [4, 21–24]. This section gives a brief explanation of HMMs and about how they work. This technique is defined as *"a finite set of states, each of which is associated with a (generally multidimensional) probability distribution*" [23]. Transition probabilities are the set of probabilities which govern the transitions among the states. In a particular state an outcome or observation can be generated, according to the associated probability distribution. Only the outcome,

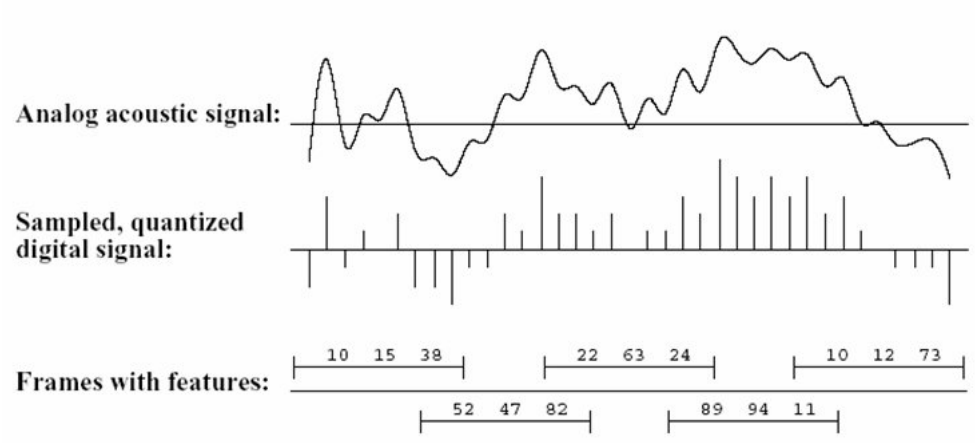

Figure 2.4: The three stages used in HMMs.

and not the state, is visible to an external observer and therefore the states are said to be "hidden" to the outside; hence the name Hidden Markov Model [23, 24]. Hidden Markov Models are, as mentioned before, a recently developed technique that is now broadly used in the categorization of noisy sequences into a set of discrete categories [22].

Some applications of HMMs are found in speech recognition and in the comparison of protein sequences in *chemistry*. The following is a list of more divergent applications of HMMs [21, 22]:

- Modelling Control of Cellphone Networks
- Computer Recognition of American Sign Language
- The Timing of Trading in the Financial Market

A HMM is a variation on the well-known Markov chain model, one of the most widely studied stochastic models of discrete events as described by Bartholomew in 1975 [22].

2.4.2.1 How HMM's Work

Hidden Markov Models as explained by [4, 25], says that HMMs consists of three components: *priors* (initial state probabilities), *state* transition model and an *evidence* observation model. The transition and observation models never change.

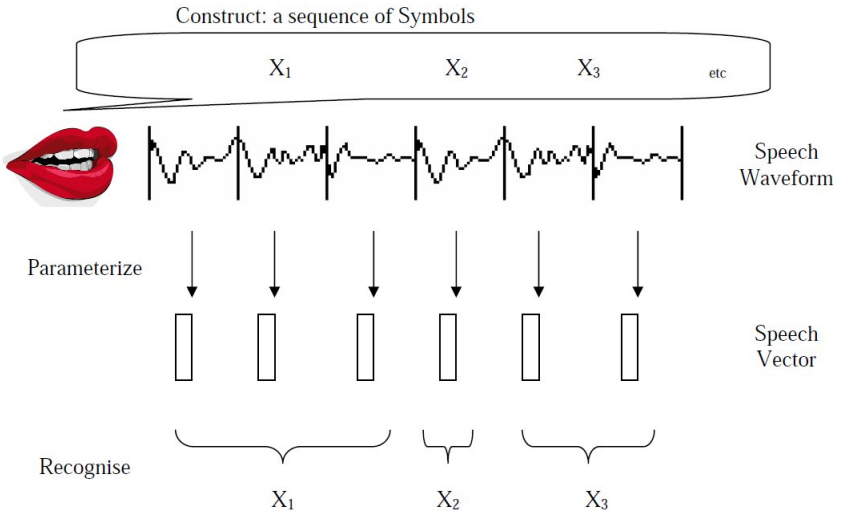

Figure 2.5: Sentence Message Encoding and Decoding.

The only changes are assumed to be caused by stationary processors. HMMs have many uses which include: filtering, prediction, smoothing, and most likely explanation, with prediction being of most interest to us. From figure 2.4, the raw signal is the microphone displacement as a function of time which is processed into overlapping frames which is described by features. The frame features are typically formants which are peaks in the power spectrum. Another representation of how HMMs relate to speech recognition is shown in figure 2.5 which was adapted from [21]. This figure illustrates how a sentence would be broken up into symbols.

The incoming waveform first has to be converted into a sequence of equally spaced parameter vectors. This discrete set of parameter vectors is assumed to form an exact representation of the speech waveform. This is on the basis that for the duration covered by a single vector the speech waveform is assumed as being stationary. The purpose of a recognizer is to effectively produce a mapping between speech vectors and the underlying symbols given in a spoken utterance. There are however, two main problems which make this task a difficult one [4, 22]:

- *"Mapping from symbols to speech is not one-to-one, since different underlying symbols can give rise to similar sounds.*
- *It is difficult to identify the boundaries between symbols from speech waveforms. Therefore, this makes it impossible to treat speech waveforms as a connected*

sequence of static waveforms."

It is not possible to treat a speech waveform as a chain of concatenated static patterns due to the large variations in realised speech (e.g. mood, environment and speaker inconsistency) and locating the boundaries between symbols. As can be seen from above, the Hidden Markov Model is quite a complex system and can be researched in great detail to understand how it works. In this study a speech recognizer which uses this technique will be implemented.

2.4.3 Artificial Neural Networks

Historically, neural networks referred to a circuit of biological neurons e.g. a brain processing information. Artificial Neural Networks or ANNs, are composed of artificial nodes or programming constructs which mimic the biological neurons. ANN is used in fields such as the detection of medical phenomena, stock market prediction, credit assignment, monitoring of machinery or even engine management [26]. ANN is traditionally a pattern recognition technique which has recently been applied to speech recognition with success.

2.4.3.1 How Neural Networks Work

ANNs have the ability to derive meaning from imprecise or complex data. This allows it to extract patterns that are far too complex to be noticed by other computing techniques [27]. Unlike conventional computing which follows a strict set of instructions to solve a given problem, ANNs, learn by example, similar to the human brain. They cannot be programmed to follow a set of instructions. Advantages of a neural network are listed below [4, 27]:

- Ability to learn how to perform a task based on training data (Adaptive Learning)
- Creates their own organisation of the information they receive during the learning phase (Self-Organization)
- Computations can be carried out in parallel (Real-Time operation)

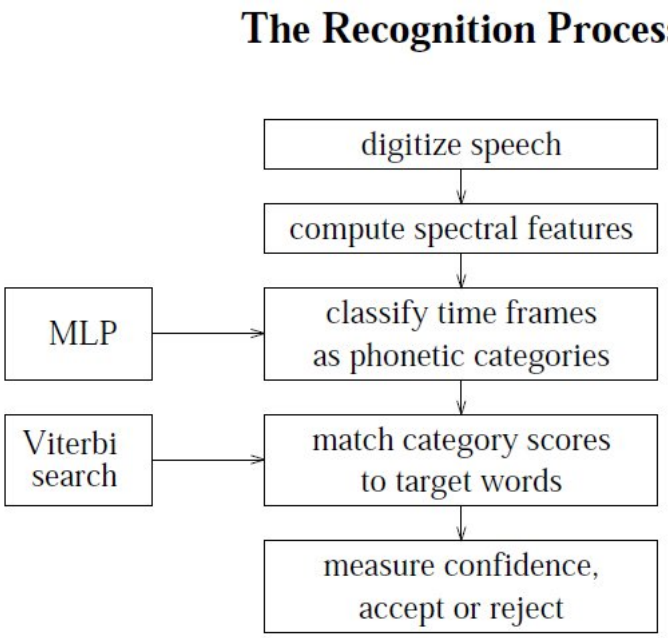

Figure 2.6: The recognition process using ANNs.

- They are fault tolerant

Due to the nature of a neural network, resolving an issue has the disadvantage that its outcome can be very unpredictable. As Stergiou quotes in [27], *"Neural networks do not perform miracles. But if used sensibly they can produce some amazing results. "*

2.4.3.2 How ANNs can be used in Speech Recognition

There are four basic steps to follow when performing speech recognition [4]:

- Convert the speech from analog to digital
- Compute the features that represent the spectral domain of the speech
- ANN is used to classify a set of these features into phonetic-based categories
- A viterbi search is used to match the output scores to the target words to check which is the most likely word that was spoken

This process is shown in block diagram form in figure 2.6 obtained from [28].

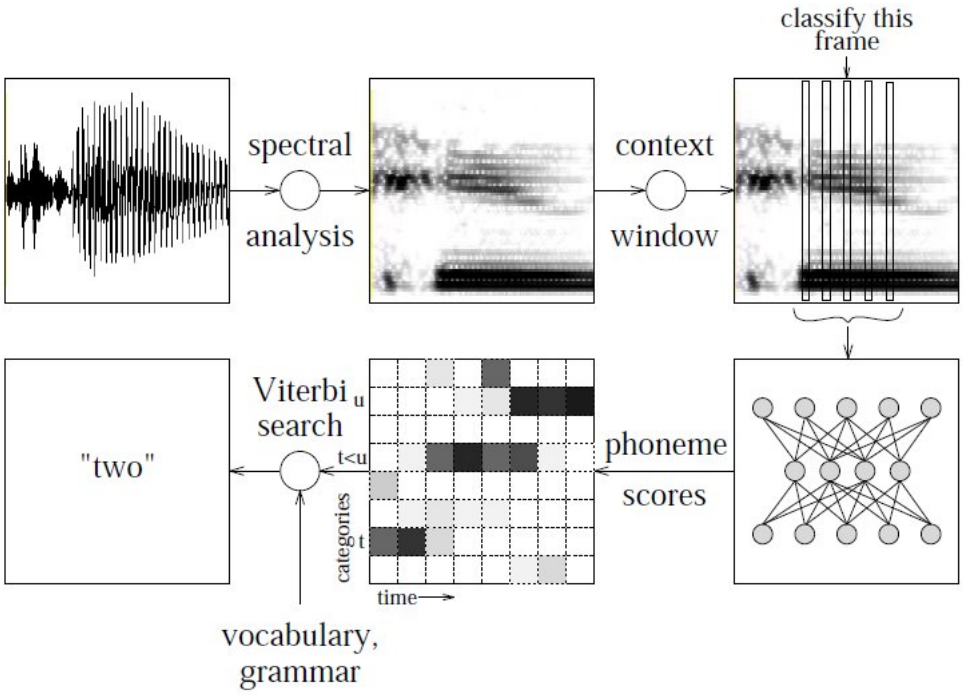

Figure 2.7: Graphical view of Recognition Process using ANNs.

Figure 2.7 can be explained as follows [4, 28, 29]:

The digitized waveform is converted into a spectral-domain representation. From this representation, either one of two sets of features can be used, which is entirely dependent on the speech recognizer. It has been suggested that we use twelve mel-frequency cepstral coefficients (MFCCs), those MFCC delta features that indicate the degree of spectral change plus one energy feature, and one delta-energy feature (for a total of 26 features per frame). It is important to note that these values are not applied in all cases of ASR but are specific to the referenced literature. Cepstral-mean-subtraction (CMS) of the MFCC coefficients is done to remove some of the effects of noise.

To get feedback regarding the acoustic content, a context window of features are taken, which is illustrated by the vertical lines in the first row, third block in figure 2.7. This means that we take the frame of interest together with frames which lie -60, -30, 30 and 60 ms away from it, in order to take into consideration the dynamic nature of speech. These features are then sent as a context window to the neural network for classification. The output of the neural network which is a classification of each input frame, is measured in terms of the probabilities of phoneme-based

categories. These context windows are used to build a matrix of the probabilities of phoneme-based categories over time.

The word to be recognised in figure 2.7 is *"two"*, the darker regions in the t, $t<u$, and u categories indicate the higher probabilities of those classes at those specified times. A viterbi search is used to find the best path through the matrix of probabilities for each legal string. The word string that corresponds to this best path will be outputted.

2.5 Summary

This chapter reviewed some of the key concepts involved in automatic speech recognition. It showed a brief history of speech recognition and also investigated the human speech production system. Key speech recognition methodologies together with current algorithms which are used to perform speech recognition were also reviewed. The next chapter focuses on speech recognition techniques which are used for implementations on mobile devices.

Chapter 3

Automatic Speech Recognition Techniques for Mobile Devices

Today there are a wide variety of techniques which can be used to perform speech recognition and it takes many components to advance from human speech to a corresponding output text on a computer. This section describes three distinct ASR techniques which are used to implement speech recognition on mobile devices, *Embedded Speech Recognition, Network Speech Recognition (NSR) and Distributive Speech Recognition (DSR).*

3.1 The Basics of ASR

Chapter 2 laid a good foundation for us to go ahead and describe the basics of an automatic speech recognition system. In an ASR system, the goal is to produce output of the most likely sequence of words being spoken (input). That is to find the the most probable sequence of words shown in equation 3.1, which belongs to a fixed vocabulary given a set of acoustic observations shown in equation 3.2 [4, 30]. Rabiner shows that the best estimation for the word sequence is by following the Bayesian approach applied to ASR [4]. This is shown in equation 3.3.

$$W = (w_1, w_2, ...) \tag{3.1}$$

$$O = (o_1, o_{2,\ldots,}o_T) \tag{3.2}$$

$$W^* = arg\,max_W P(W|O) = arg\,max_W \frac{P(O|W)P(W)}{P(O)} \tag{3.3}$$

A speech recognizer performs the following four operations in order to generate an output:

1. Extracts features (acoustic observations) from the spoken utterance
2. Estimates *P(W)* - which defines the probability of individual word sequence happening
3. Estimates *P(O/W)* - which defines the likelihood that the particular set of features originates from a certain sequence of words
4. Finds a word sequence that delivers the maximum of equation3.3.

3.2 Embedded Speech Recognition

In embedded speech recognition or client-based speech recognition, all the components required live on the terminal device. The ideal ASR system would be an embedded speech recognition system where no external components would be required. This is certainly the architecture of choice when it comes to Personal Digital Assistants (PDAs) which are more powerful and have well-established operating systems (OS) [30].

3.2.1 How Embedded Automatic Speech Recognition Works

Figure 3.2 shows the pre-processing steps which take place at the front-end for feature extraction [31]. The basic procedure which is followed is shown in the steps below:

- Audio signal is captured on the device
- Feature extraction is performed on the incoming speech signal

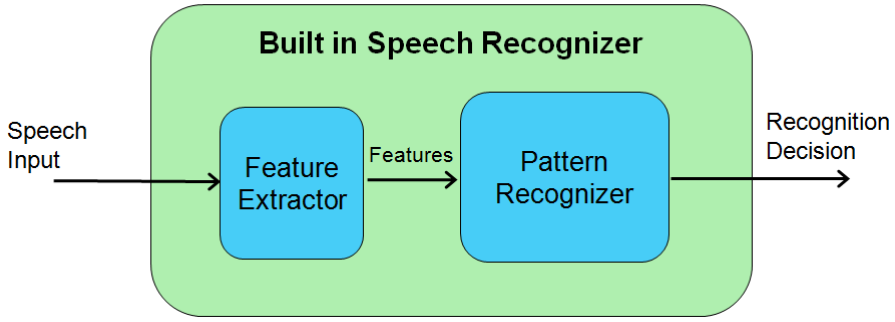

Figure 3.1: A functional block diagram, illustrating components in an embedded speech recognition system.

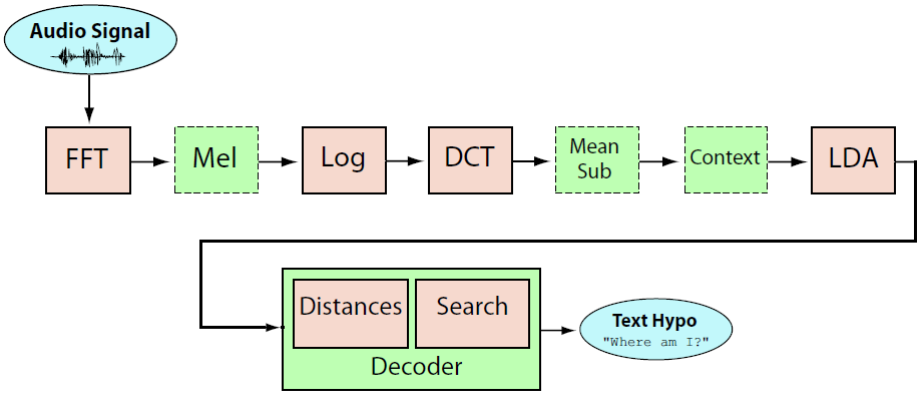

Figure 3.2: Illustration of the pre-processing which takes place at the front-end feature extraction block as well as the back-end decoder block which performs the recognition.

- Features are sent to the decoder which scores the features
- Those with the highest scores are selected and sent as output

3.2.2 Advantages of Embedded Automatic Speech Recognition

- The main advantage of PDAs is that they have well-known processor architectures which have development kits which are optimised for certain platforms [32].
- Embedded ASR also has no latency costs as all the processing is done in the client as can be seen in figure 3.1

3.2.3 Disadvantages of Embedded Automatic Speech Recognition

- As previously mentioned the important factor to consider in this type of implementation are the limited resources which are available and in order to achieve good recognition rates, every effort must be put into creating algorithms which are efficient and cost effective. Novak notes in [33], the two most important resources as the memory size and execution speed of the algorithms.
- This type of implementation is mostly successful in voice dialing, command and control applications, because it only requires a limited dictionary and can be achieved by implementing a simple energy or signature function.
- This form of recognition requires further exploration and development before it becomes a viable option for applications which require larger dictionaries.

3.3 Network Speech Recognition

Network speech recognition takes a client-server approach as shown in figure 3.3. By using this type of methodology, moving all the complex processing to a back-end server, avoids the complications caused by resource limitations on the terminal device. NSR can be used not only on PDAs but also on thin terminal clients such as mobile phones which historically have less processing power than PDAs.

3.3.1 How Network Speech Recognition Works

The main difference between embedded ASR and NSR is that all the processing is moved from the terminal device to a back-end server which performs both the pre-processing as well as the recognition.

The audio is still captured on the terminal and a speech coder is used to compress the audio and send it over the network to a back-end server that has the capability of servicing multiple clients simultaneously [30]. Zaykovskiy suggests in [30], that there are several other successful implementations of this method using different codecs;

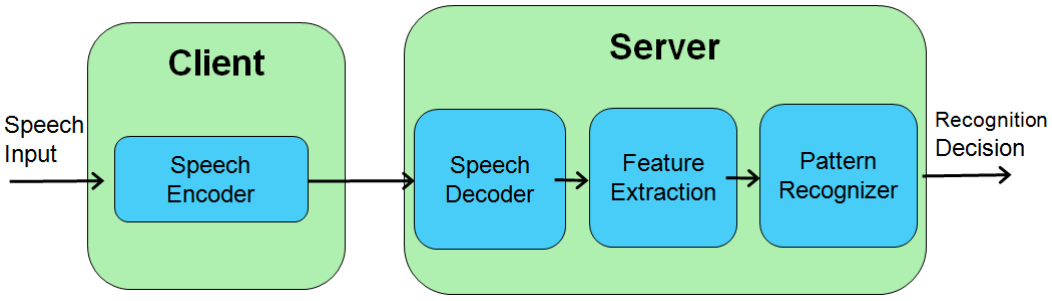

Figure 3.3: A functional block diagram of network speech recognition.

ETSI GSM 06.10 [34], FS1015 and FS1016 [35] and ITU-T G.723.1 [36]. These implementations were out of the scope of this study and thus not implemented as it was not the primary focus.

3.3.2 Advantages of Network Speech Recognition

- As previously mentioned the client will not have to perform any complex computations
- Having a powerful back-end server provides access to the recognizers based on the different grammars and even different languages
- A seamless upgrade and modification can be done on recognition engine, without intervention from the end-user [30]

3.3.3 Disadvantages of Network Speech Recognition

The main disadvantage is performance degradation, when using low bit-rate codecs which is exaggerated by background noise together with transmission errors.

3.4 Distributive Speech Recognition

Distributive speech recognition or DSR is one of the more recent techniques being researched to enable mobile devices with speech recognition. This method like NSR uses a client-server approach but, instead of outsourcing all the processing to a back-end server, it spreads the load across the client and server shown in figure 3.4.

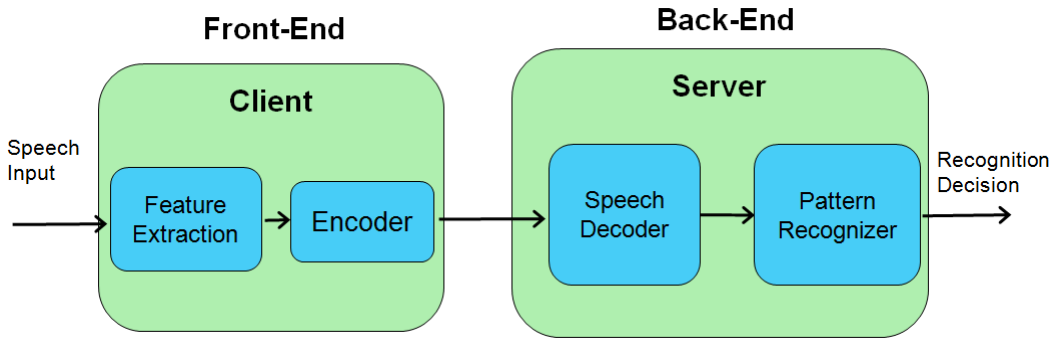

Figure 3.4: A functional block diagram of a DSR system.

3.4.1 How Distributive Speech Recognition Works

A typical DSR system is based on decoupling the front-end processing from the rest of the recognition mechanism using a client-server model over the data network [37]. The name distributive speech recognition is given because the feature extraction and the speech recognition are distributed across the network. This setup allows a terminal device (client) to only be responsible for the feature extraction and speech coding part, while the back-end server (central host) handles the decoding and computational intensive recognition part. Figure 3.5 shows an outline of a DSR system. On the client side, features are extracted using a mel-cepstrum based feature extraction technique (MFCC) from the input speech and then compressed and sent over the wireless channel to a large server where the features are decompressed and sent to a state-of-the-art Hidden Markov Model (HMM) based classifier. The classifier will return a recognition result back to the client.

3.4.2 ETSI Front-End for Distributive Speech Recognition

A standardized front-end for DSR has been specified by the Aurora Working Group within the European Telecommunications Standards Institute (ETSI) for use on mobile phones and other communication devices which connect to speech recognition servers [38]. This has been done so that all front-end clients are identical regardless of the type of device the client is. A detailed diagram of the front-end standard can be seen in figure 3.6.

This section describes this standard in more detail and was extracted from

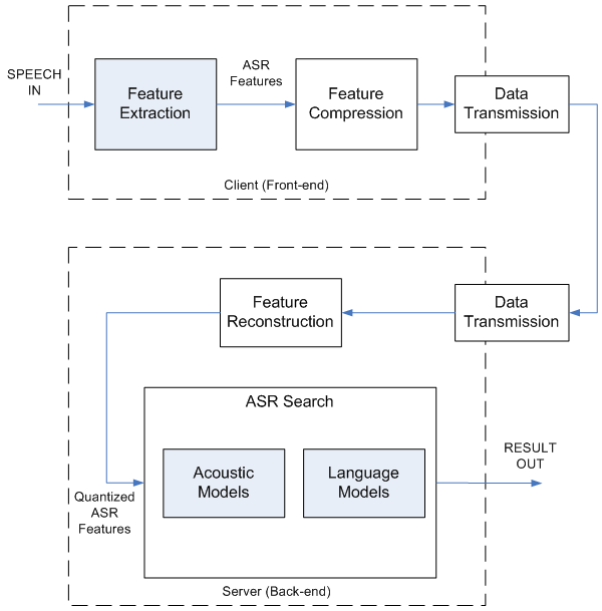

Figure 3.5: Client-Server based ASR system.

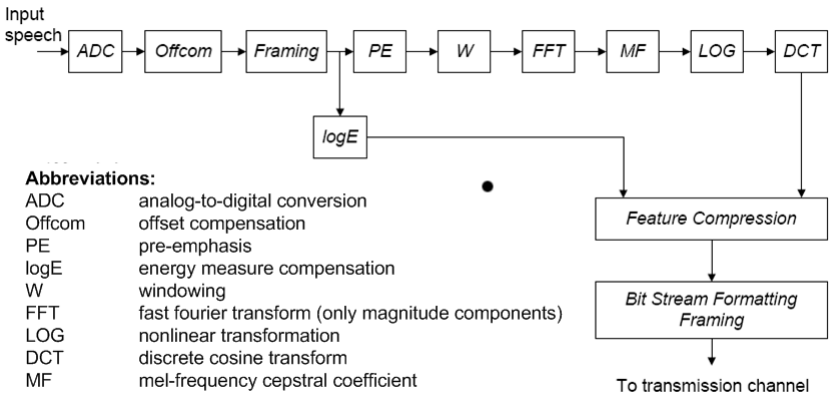

Figure 3.6: Block diagram of the ETSI front-end algorithm.

Aurora-ETSI Standards document [38]. The standards specified by ETSI are briefly shown below [38, 39]:

- Mel-Cepstrum feature set consisting of 12 cepstral coefficients logE and C_0
- Data transmission rate of 4.8 kbps
- Low computational and memory requirements
- Low latency
- Robustness to transmission errors

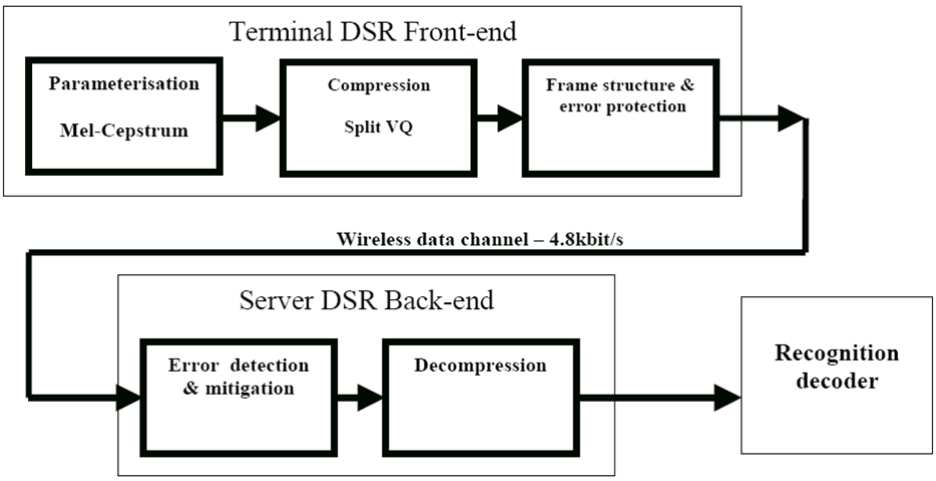

Figure 3.7: Block diagram of DSR system

DSR Mel-Cepstrum Front-end Standard

Figure 3.7, taken from [39] takes a closer look at the processing stages for a DSR front-end.

The algorithm which is used in [38] can be described as follows:

- The speech signal is sampled and parametrized using the mel-cepstrum algorithm
- This generates 12 cepstral coefficients (see equation 3.4) along with C0 and logE (see equation 3.5)

$$C_i = \sum_{j=1}^{23} f_j \cdot cos\left(\frac{\Pi \cdot i}{23}(j - 0.5)\right),\ 0 \leqslant i \leqslant 12 \tag{3.4}$$

$$logE = ln\left(\sum_{i=1}^{N} S_{of}(i)^2\right) \tag{3.5}$$

Symbol	Meaning
Ci	ith cepstral coefficient
f_j	log filter bank output for the jth Mel channel
N	frame length
S$_{of}$	offset-free input speech signal

- This is compressed to obtain lower data rate (4.8 kbps) for transmission

- The compressed parameters are formatted into a defined bit stream
- This is then transmitted over wireless/wireline transmission link to a remote server
- The parameters are then checked for transmission errors
- Front-end parameters are decompressed to reconstruct the DSR Mel-Cepstrum features
- These are then passed to the recognition decoder sitting on the central server

Mel-Cepstrum Parametrization

The block diagram shown in figure 3.6 represents the specification of the Mel-Cepstrum DSR Front-end standard submitted by Nokia. The feature vector, as mentioned above, consists of 12 cepstral coefficients (C_1 - C_{12}) together with C_0 and logE (log energy) parameter, making up a total of 14 components. It is suggested by Pearce in [39] that the reason for incorporating C_0, was to support algorithms which would be requiring it at the back-end (such as noise adaption).

Further details of the cepstral analysis are shown below [40]:

- Signal offset compensation with notch filtering
- Pre-emphasis with a factor of 0.97
- Application of Hamming window
- FFT based mel filterbank with 23 frequency bands in the range from 64Hz up to half of the sampling frequency

Feature Compression Algorithm

The standard for the compression algorithm was designed by Motorolla and the specifications are discussed in this section. The compression algorithm was designed to take feature parameters for each short-time analysis frame of speech data (see equation 3.6) where m denotes the frame number plus C_0 and logE (see equation 3.7)

$$c(m) = [c_1(m), c_2(m), ..., c_{12}(m)]^T \tag{3.6}$$

$$y(m) = \begin{bmatrix} c(m) \\ c_0(m) \\ log[E(m)] \end{bmatrix} \tag{3.7}$$

Symbol	Meaning
m	frame number
y*(m)*	Feature vector with 14 components

A split vector quantization (VQ) algorithm is used to obtain a final data rate of 4.8 kbps of speech. The closest VQ centroid is found using a weighted Euclidean distance to determine the index (see equations 3.8 and 3.9).

$$d_j^{i,i+1} = \begin{bmatrix} y_i(m) \\ y_{i+1}(m) \end{bmatrix} - q_j^{i,i+1} \tag{3.8}$$

$$idx^{i,i+1}(m) = \underset{0 \le j \le (N^{i,i+1}-1)}{argmin} \left\{ \left(d_j^{i,i+1}\right) W^{i,i+1} \left(d_j^{i,i+1}\right) \right\}, i = 0, 2, 4, ..., 12 \tag{3.9}$$

Symbol	Meaning
$y(m)$	Feature vector with 14 components
m	Frame number
$q_j^{i,i+1}$	denotes the jth codevector in the codebook $Q^{i,i+1}$
$idx^{i,i+1}(m)$	codebook index
$Q^{i,i+1}$	compression codebook
$W^{i,i+1}$	weight matrix
$N^{i,i+1}$	compression: size of the codebook

A codebook of size 64 is used for each pair of cepstral coefficients from C_1 - C_{12} and 256 vectors are used for C_0 and logE (see table 3.1 [38]). A quantization scheme is used to code the relevant coefficients which result in 44 bits per speech frame.

Codebook	Size	Weight	Element 1	Element 2
	$(N^{i,i+1})$	$(W^{i,i+1})$		
$Q^{0,1}$	64	1	C_1	C_2
$Q^{2,3}$	64	1	C_3	C_4
$Q^{4,5}$	64	1	C_5	C_6
$Q^{6,7}$	64	1	C_7	C_8
$Q^{8,9}$	64	1	C_9	C_{10}
$Q^{10,11}$	64	1	C_{11}	C_{12}
$Q^{12,13}$	256	Non-Identity	C_0	LogE

Table 3.1: Split-Vector Quantization Feature Parings.

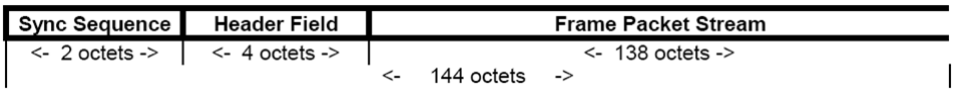

Figure 3.8: Multi-frame format used in DSR.

Error detection bits are added (4 bits of CRC for each pair of speech frames) to the compressed data and the compressed speech frames are grouped into multiframes for transmission and decoding (see figure 3.8), this is done to counteract the presence of channel errors. For a more detailed description of this algorithm, see [38].

Enhancements for DSR Front-End

The previous sections have described the basic standard of the Aurora-ETSI front-end feature extraction and compression algorithms. It is not imperative that this exact standard be used, but merely a guideline of how it should be implemented. Researchers have proposed to improve this method, as in [41] where a novel approach to improve the computational efficiency is described. They use a structure of two-stage mel-warped Wiener filtering algorithm which is the main part for AFE. Using this approach discards the convolution operations in time-domain and the calculation of the power spectrum, thereby saving on a large number of computations.

Another interesting approach is shown in [42] whereby the authors present an integration of hybrid acoustic models using tied-posteriors in the distributive environment. Their results show that a hybrid technique is able to out-perform standard continuous systems. Their approach proves to be useful since changes can be made to the client without affecting the server and vice versa.

In [43], Paliwl made use of Gaussian Mixture Models (GMMs) for the coding of Mel frequency-warped cepstral coefficient (MFCC) features in DSR. In this paper a comparison between vector quantizers and a GMM-based block quantizer, which has relatively low computational and memory requirements, is made. Their studies showed an improvement in recognition performance due to the simple computations and bit-rate scalability.

Another example is mentioned in [44], where the authors propose to reduce the recognition complexity by compressing the extracted features using scalable encoding techniques which provides a multi-resolution bitstream together with a scalable recognition system. In this research paper it was shown that using speech coders optimized for recognition, rather than perceptual distortion, provides a better recognition rate performance.

3.4.3 Traditional Back-End for Distributive Speech Recognition

The back-end of the DSR system is where the recognition occurs and is done on a high-powered desktop machine. This machine takes advantage of its processing power and memory capacity, which is the main advantages giving DSR the edge over other methods to implement ASR systems on mobile devices.

A traditional back-end server in a DSR system contains a feature reconstruction part and a recognizer as shown in figure 3.9. The feature reconstruction model would be responsible for decoding the feature vectors coming from the front-end and checking them for transmission errors. The reconstructed cepstral features would be sent to the recognizer which then presents its results.

The two most popular state-of-the-art speech recognizers are the Hidden Markov Model Toolkit (HTK) and the SPHINX IV recognizer. Just like the front-end, it is only a template of components which the server should contain. An IBM research report [45], discussed the server-side speech reconstruction in more detail but was out of the scope of this research. There have also been proposals in [46] to reduce the recognition complexity at the server-side. Their research investigates a scalable recognition system to perform this task, that is to reduce both the computational

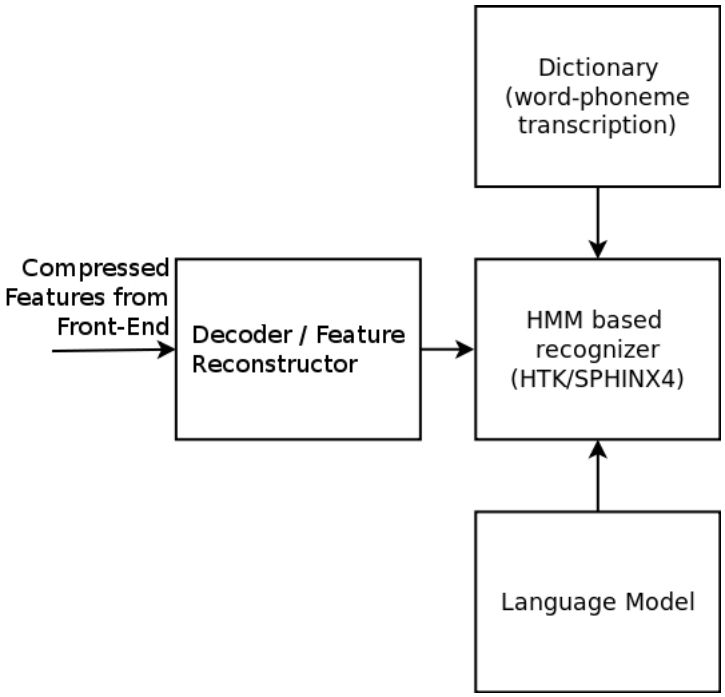

Figure 3.9: Block diagram of a back-end recognizer for speech recognition.

load and the bandwidth requirement at the server by using a low complexity pre-processor to eliminate unlikely classes.

3.4.4 Advantages of Distributive Speech Recognition

The main advantages of distributive speech recognition are explained below [39]:

- Improved recognition performance over wireless channels
- Multimodal speech applications operate over a single wireless data-channel
- Ease of integration of combined speech and data applications
- Ubiquitous access with guaranteed recognition performance levels

The greatest benefit of DSR is the fact that the system is implemented on an error-protected data channel which minimizes the impact of speech codec and channel errors. This historically improves the performance of a recognition system over mobile speech channels. The use of DSR also enables multi-modal speech applications to operate over a single wireless data channel as opposed to having separate speech

and data channels. DSR also offers the promise of a guaranteed level of recognition performance over every network

3.4.5 Disadvantages of Distributive Speech Recognition

In Chapter 1, section 1.2 the limitations which are experienced when implementing an ASR system on a mobile device are described. Since DSR has a back-end server performing the recognition, most of the issues listed in that section are alleviated.

Having said this, they would still need to be taken into account when implementing our feature extraction algorithm and speech coding on the client side. In [47], the authors performed an extensive study on the issues related to DSR and also explained that when using the mobile voice network, there is a degradation in performance due to low bit rate speech coding and channel transmission errors. Manolis also suggested in [47] that using an error-protected data channel will result in a higher recognition performance.

3.5 Summary

In this chapter different techniques which can be used to implement speech recognition on mobile devices was described. Both the advantages and disadvantages of each technique was discussed, and for the purpose of this study looked in greater detail at DSR, focussing on its front-end, and briefly describing the back-end.

Chapter 4

CMU Sphinx-4 Speech Recognizer

This chapter describes the Sphinx-4 speech recognizer which was a joint project between the Carnegie Mellon University (CMU), Sun Microsystems Laboratories and Mitsubishi Electric Research Laboratories (MERL). At the start a brief history of traditional speech recognition systems is presented, after which previous Sphinx recognizers are investigated and finally, the latest in the family, the Sphinx-4 system architecture and how it works, will be discussed.

4.1 A brief History of Speech Recognition Systems

Traditionally in speech research, systems have often had to be developed completely from scratch [48], even if researchers wanted to investigate one facet of the technology, and in most cases the system was optimized around a particular methodology. This type of approach did have a major benefit in that it provided foundational methods for speech recognition research.

Examples of such systems are listed below [48]:

- In the Dragon speech system, Baker introduced hidden markov models (HMMs) [49]
- The earlier Sphinx systems explored variants of HMMs such as discrete HMMs [50], semi continuous HMMs [51], and continuous HMMs [52]
- Other systems explored specialized search strategies such as using lex tree

searches for large N-Gram models [53]

This kind of single approach to system design hinders progress in the field of speech research, which is not its purpose, as these implementations were often hardwired to a large degree. In some cases these systems have licensing agreements which makes research for non-academics more difficult. Examples of open source speech recognition systems are: Hidden Markov Model Toolkit (HTK) [54], Audio-Visual Continuous Speech Recognition (AVCSR) [55] and the Institute of Signal and Information Processing (ISIP) speech-to-text system, *"ISIP has been committed to providing the research community with free software tools for digital information processing via the Internet to facilitate worldwide synergistic development of speech recognition technology"* [56].

4.2 The CMU Sphinx Suite of Applications

The Sphinx system is a state-of-the-art HMM based recognition system, and is one of the most widely used recognition systems. Like all other HMM based systems, it works by first learning the characteristics of speech units and then uses the information it gathers to find the most likely sequence of words or sound units for a given speech signal. Training is referred to as the process of learning about the speech signal. The process of using the information which it learned from the training phase to hypothesize the most probable sequence of words is called decoding.

In the Sphinx suite there exists the following trainer and recognizers:

- *SphinxTrain*
  - This application is used to train speech databases for Sphinx
- *PocketSphinx*
  - This is a lightweight speech recognition engine specifically tuned for handheld mobile devices [57]
- *Sphinx-1*

  - This was one of the first user independent high performance ASR systems
  - It offered support for simple word pair grammars

- *Sphinx-2*
  - This version of Sphinx is written in the C programming language (designed for speed)
  - It's main features include continuous speech decoding
  - It's a speaker independent system
  - It uses semi continuous acoustic models and either bi-gram or tri-gram language models [58]

- *Sphinx-3*
  - This implementation was developed primarily for flexibility

- *Sphinx-4*
  - This is the latest in the Sphinx suite and was written in the Java programming language, and is described later in more detail

4.3 Introduction to Sphinx-4 Speech Recognizer

Lamere describes in [59], a distributed, cross-discipline team which was formed to create the Sphinx-4 speech recognizer which is defined in the paper as *"an open source platform that incorporates state-of-the art methodologies and also addresses the needs of emerging research areas"*. This system uses different operating systems on different machines and since the Sphinx-4 was written in the Java programming language, it is available on a large variety of development platforms. The Sphinx-4 system project is open source and available for download from SourceForge [60] since its inception.

When investigating the Sphinx-4 system it was found that its modular and pluggable framework incorporating design patterns from existing systems, with the flexibility to support the latest areas of research interest was exactly the platform needed.

This framework is also modular (it comprises separable components dedicated to specific tasks) and it is pluggable (modules can be easily replaced at run-time) [61]. Sphinx-4 also comes pre-packaged with a variety of modules which implement state-of-the-art speech recognition techniques which is another reason why it was chosen for research.

To give an idea of the systems modularity, any particular Sphinx-4 module which researchers wish to use can be specified at run-time - this means code does not need to be re-complied. Other examples of its modularity are listed below [48, 59, 61]:

- Change the language model (by modifying the *linguist* model)
- Use either continuous, semi-continuous or discrete state output distributions (modify the *acoustic scorer*)
- Use any level of context in the definition of basic sound units (*phonemes*)
- Decide between depth first or breadth-first search module.

The next section looks at the system architecture of the Sphinx-4 decoder in greater detail.

4.4 Sphinx-4 System Architecture

Figure 4.1 shows the complete system architecture of the Sphinx-4 speech recognizer or decoder as it is also known [48]. From the graphic it is clear that there are two main blocks, the front-end and the decoder. The front-end is primarily responsible for parameterizing the incoming speech signal and passing it onto the decoder. The decoder has three parts, the search manager,the linguist and the acoustic scorer all of which work together to perform decoding on the extracted speech features.

4.4.1 Front-End

As previously mentioned the main purpose of the front-end is to parameterize the incoming speech signal into a sequence of features. Figure 4.2 shows a more detailed representation of the front-end module which consists of several communicating

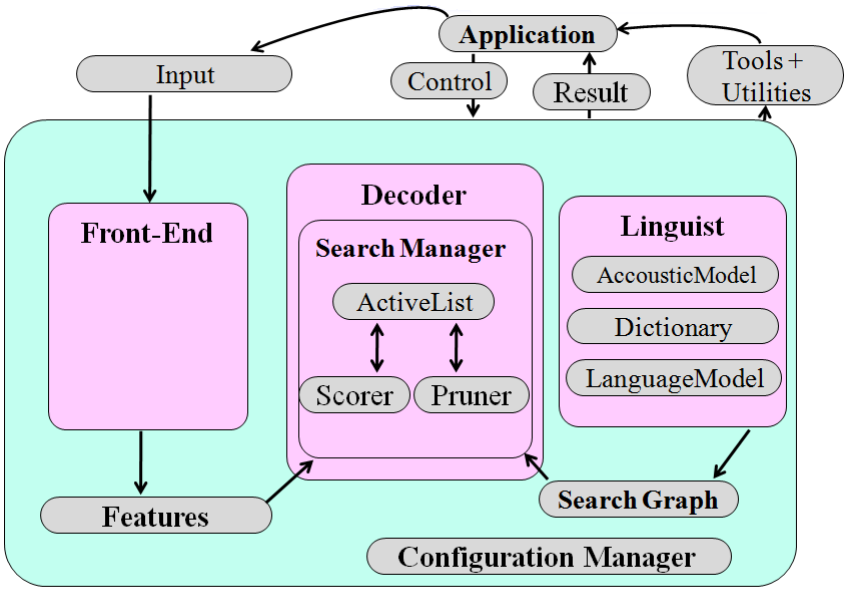

Figure 4.1: Sphinx-4 decoder framework.

blocks each with its respective input and output [61]. What needs to be noted is that each block is independent of the others and can easily be replaced as needed. Also to be noticed is that the input of each block is linked to the output of its predecessor and interprets it to ascertain if it is speech data or control signals (this may indicate start points or end points of speech or data which were dropped, etc).

If the incoming data is indeed speech, it is then processed and the output is buffered, waiting for the successor block to request the data. Since the front-end can handle control signals, the design allows this system to be used in live scenarios

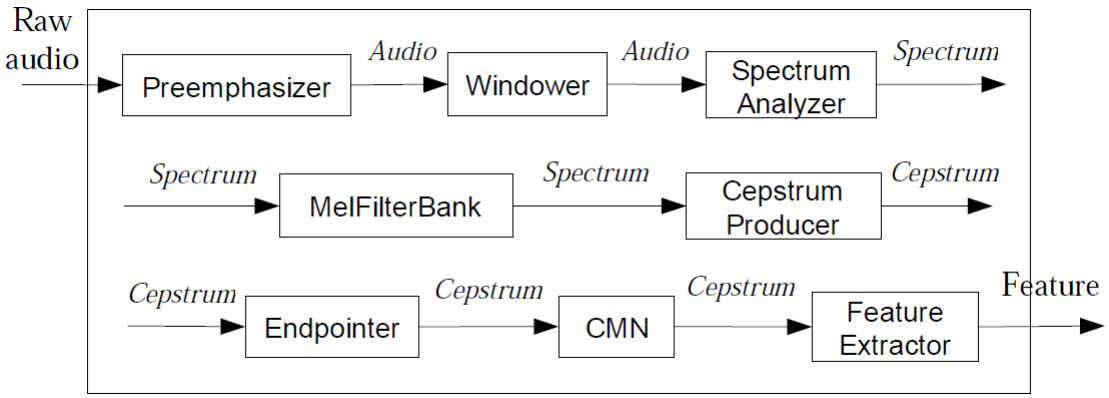

Figure 4.2: Detailed implementation of the Sphinx-4 front-end.

without modification. The Sphinx-4 system is capable of running continuously from a stream of input speech. Lamere describes the three modes of operation which can be used for input speech [59]:

- Fully end-pointed
  - the system performs explicit end-pointing, determining both beginning and ending endpoints of a speech segment automatically
- Click to talk
  - where the speaker indicates the beginning of a speech segment but the system determines the end point
- Push to talk, where the user indicates both the beginning and the end of the speech segment

A simple energy level function is currently being used to perform the endpoint detection. Since only signals detected as speech are sent to the decoder, it does therefore not waste any unnecessary time processing non-speech segments.

4.4.2 Decoder

The features from the front-end are sent to the decoder and scored against an acoustic model. This score will then indicate the likelihood that a particular set of features belongs to the phoneme of the corresponding acoustic model. The ultimate goal of speech recognition will be to find the best possible sequence of words which will match the input speech signal. The decoder block is composed of three modules; the *search manager*, the *linguist* and the *acoustic scorer*. The Sphinx-4 decoder uses the output features from the front-end, together with the search graph from the *linguist* to generate a result hypotheses. These will be described in greater detail in this section.

4.4.2.1 The Linguist

Firstly, the *linguist* is a pluggable module in the decoder which allows users to dynamically insert different *linguist* implementations. The *linguist* generates a search graph for the decoder but hides all the complexities involved in this process [48, 59, 61]. The *linguist* module consists of three other pluggable modules, i.e. the *language Model*, the *dictionary* and the *acoustic model* which is further described in this section. A typical implementation constructs the search graph by using the structure of the *language model* and the structure of the *acoustic model*. The *linguist* can also use a *dictionary* to map words from the *language model* into sequences of *acoustic model* elements

Language Model

In short the *language model* provides a word level language structure which is represented either by graph-driven grammars or stochastic N-Gram models [48]. Note that again these are pluggable models which can be replaced. In Sphinx-4, there are a variety of *language models* which are listed below [48]:

- SimpleWordListGrammar: defines a grammar based upon a list of words.
- JSGFGrammar: supports the Java Speech API Grammar Format (JSGF) [62],
- LMGrammar: defines a grammar based upon a statistical language model.
- FSTGrammar: supports a finite-state transducer (FST) [63]in the ARPA FST grammar format.
- SimpleNGramModel: provides support for ASCII N-Gram models in the ARPA format.
- LargeTrigramModel: provides support for true N-Gram models generated by the CMU-Cambridge Statistical Language Modeling Toolkit [64].

Dictionary

As the name suggests the *dictionary* provides pronunciations of the words which are found in the *language model*. These pronunciations are broken down into phonemes found in the *acoustic model*.

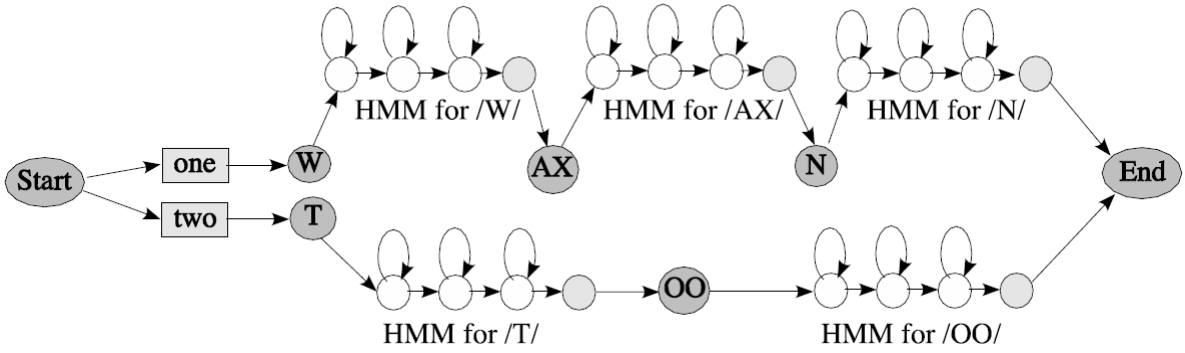

Figure 4.3: Example of a *search graph*.

Acoustic Model

The *acoustic model* provides the mapping between a unit of speech and an HMM that can be scored against incoming features, which as previously mentioned is received from the front-end. As described in [48] the *linguist* breaks each word in the active vocabulary into a sequence of context-dependent sub-word units. The units then get passed to the *acoustic model* which retrieves the HMM graphs associated with those units. It uses these HMM graphs together with the *language model* to construct the *search graph*.

It is important to note that the Sphinx-4 HMM is merely a directed graph of objects (see figure 4.3), unlike most speech recognition systems who store the HMM graphs as a fixed structure [48]. In this graph, each node corresponds to an HMM state and each arc represents the probability of transitioning from one state to another in the HMM.

4.4.2.2 The Search Manager

The main purpose of the *search manager* is to construct and search a tree of possibilities for the best possible hypotheses [48]. In order to create this tree it uses information obtained from both the linguist as well as data from the *acoustic scorer*. It makes use of a token tree shown also in [65] to be used by other recognitions systems. Lamere and Kwok describes this tree as consisting of a set of tokens containing information about the search and providing complete historical data of all active paths in the search, i.e. each token contains the overall acoustic and language scores

of the path at a given point (a Sentence-HMM reference), an input feature frame identification, and a reference to the previous token, thus allowing back-tracing [61].

The search manager also provides a beam pruner. This component constrains the scores to a configurable minimum which is relative to the best score, while keeping the total number of active tokens to a configurable maximum. New implementations can easily be created that provide alternative methods of storing and pruning the active list [61].

The active list, shown in figure 4.1, is available as part of the final recognition result. Applications will use the active list to inspect the highest scoring paths.

4.4.2.3 The Acoustic Scorer

The task of this component is to compute state output density values for various states for any given vector [48, 61]. This scoring mechanism provides the scores on demand to the search module. As stated in [61], in order to calculate these scores, the scorer module must also communicate with the front-end module to obtain the features for which the scores must be computed. It also has the capabilities to use parallel input features, and in this case the *acoustic scorer* must score the incoming features against the proper set of HMMs.

4.5 Summary

In this chapter the architecture of the Sphinx-4 recognizer, the different modules of the decoder and their methodologies was investigated. It is important to note that since the front-end is pluggable, it will be exploited in this study by substituting the original with an alternate front-end, that of the one described in Chapter 3. The next chapter focuses on the experimental framework which was set up to create the baseline system.

Chapter 5

Experimental Framework for Evaluating the ETSI Front-End with the CMU Sphinx-4 Back-End Decoder

The purpose of this chapter is to describe the experimental framework design, implementation and evaluation. The two speech recognition techniques being compared in this thesis are the *Network Speech Recognition* system and *the Distributive Speech Recognition* system. The experimental design will provide a baseline framework for evaluating these two techniques and also a platform for future research in this area.

The rest of this chapter is structured as follows; Section 5.1 describes the design criteria of the system. Section 5.2 describes the databases which are used to evaluate the baseline system together with its procedure. Section 5.3 describes the basic design and implementation of the system which is evaluated in section 5.4.

5.1 Design Criteria for Experimental Framework

The basic design criteria for this framework are as follows:

- The implementation should be based on techniques which are well-documented in literature. These techniques have been discussed in Chapters 3 and 4. This

will create a baseline system which incorporates techniques that are regarded as standard implementations for this type of recognition system.

- The performance of the system should be in line with that of systems being developed and evaluated under similar conditions.
- Any code which supplements the current implementation must be well commented and in-line with the Sphinx-4 system in its modular approach.

5.2 Experimental Database

The speech corpus used in this thesis is the TIMIT Acoustic-Phonetic Continuous Speech Corpus [66]. Below is a description of the two different instances of the TIMIT corpus which have been used in this thesis:

- The original TIMIT corpus
- A modified TIMIT corpus. This is a VodacomTIMIT database. It is based on the TIMIT database and was passed through the Vodacom GSM network. Two mobile phones were used at stationary positions to capture the data, this procedure was conducted between 19/10/2004 and 8/10/2005 [67].

The TIMIT corpus of read speech has been designed to provide speech data for the acquisition of acoustic-phonetic knowledge and for the development and evaluation of automatic speech recognition systems [68]. This database contains a total of 6300 sentences of which 630 speakers are from 8 major dialect regions of the United States (US), each speaking 10 sentences. Table 5.1 shows the breakdown by dialect region and gender of speakers [68].

This corpus provides both training and testing sets which will be used for comparisons.

5.3 System Design and Implementation

The design of this system was carefully planned and structured in its particular manner so as to make it easy to understand in order for others to reproduce and

Dialect Region (dr)	No. of Males	No. of Females	Total
1	31	18	49
2	71	31	102
3	79	23	102
4	69	31	100
5	62	36	98
6	30	16	46
7	74	26	100
8	22	11	33

Table 5.1: Dialect distribution of speakers.

improve upon it for future research. It is worthwhile to mention that all implementations and simulations were performed on an Intel Pentium desktop machine, 2.8GHz CPU, 1GB RAM, 80GB HDD in the university laboratories. This machine was running a Linux kernel, using the Ubuntu 8.04 distribution OS.

In order to implement their baseline system using the Sphinx-4 decoder, the researchers had to follow a tutorial written by CMU [69] describing the protocols related to a Sphinx-4 implementation.

5.3.1 Training the Sphinx-4 Decoder with TIMIT Database

For the purpose of the implementation, the more important procedure which needed to be followed was that of training the TIMIT database in order to create trained acoustic models for the decoder. This section briefly describes the components which are required for training:

- Trainer source code - *SphinxTrain*
- The acoustic signals - TIMIT Training Database - The trainer uses a set of sample speech signals (in this case the TIMIT DB) and learns the parameters of the models of sound units
- Transcript file - this file contains the sequence of words exactly as they appear in the speech signal and it informs SphinxTrain which sound units to learn
- Language Dictionary - contains legitimate words of the language mapped to its sub-word units

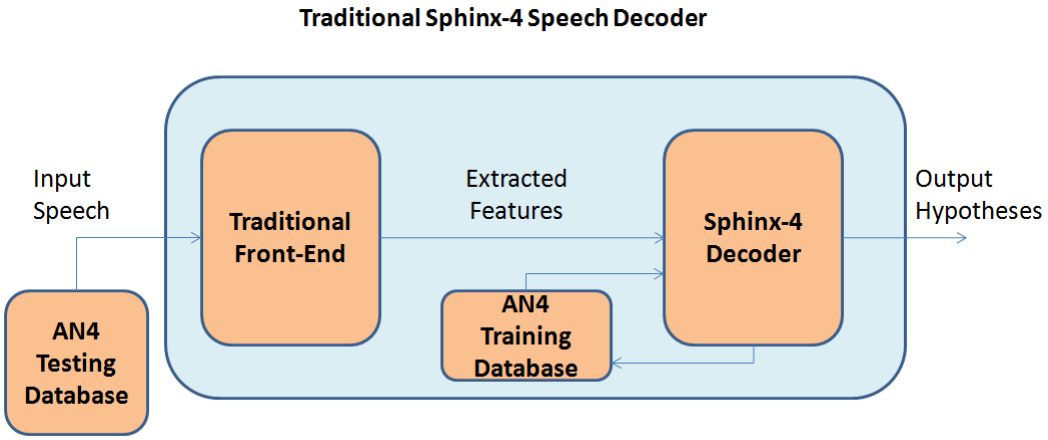

Figure 5.1: Block diagram of the traditional implementation of the Sphinx-4 recognizer.

- Filler Dictionary - contains non-speech sounds mapped to non-speech sub-word units

5.3.2 Traditional Sphinx-4 Speech Decoder with built in Front-End

Before describing this system, it is important to first understand which components are required for decoding speech. These are listed below:

- The Decoder Source Code - in this case *Sphinx-4* decoder
- The Language Dictionary - must be provided by the trainer
- The Filler Dictionary - must be provided by the trainer
- The Language Model - must be provided by the trainer
- The Test data, this will differ depending on the implementations, from AN4 database, TIMIT database or VodacomTIMIT database.

The baseline implementation was the default Sphinx-4 speech recognizer, using the built-in front-end with the AN4 database speech corpus. This is implemented as a cross-reference to ensure the results are in-line with those results noted by CMU [69]. This can be seen in a block diagram shown in figure 5.1.

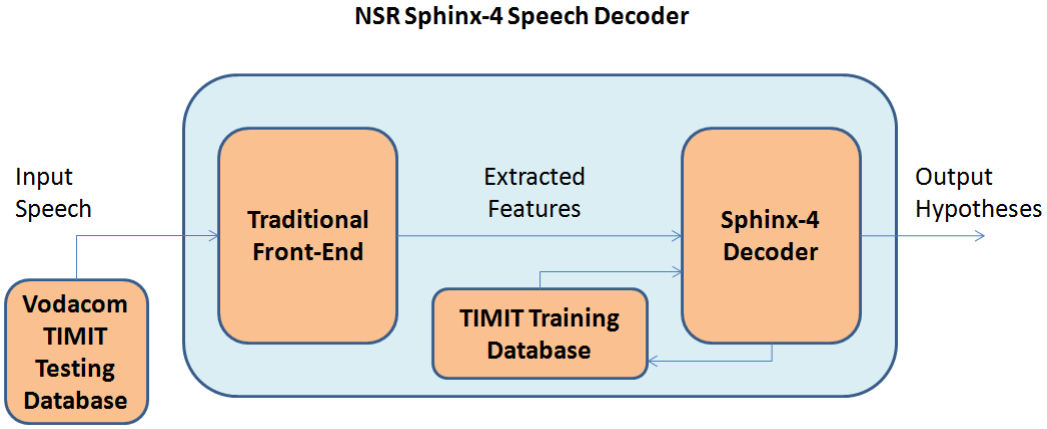

Figure 5.2: Block diagram of the NSR implementation using the Sphinx-4 recognizer.

5.3.3 NSR using the Sphinx-4 Speech Decoder

In the implementation for *Network Speech Recognition* the VodacomTIMIT database is used to simulate the entire acoustic signal being sent over the network to a back-end server, where feature extraction is then performed on the degraded speech data. A block diagram is shown in figure 5.2 of this type of implementation.

5.3.4 ETSI Front-End with DSR Sphinx-4 Speech Decoder (without network degradation)

In order to implement a successful distributed speech recognition system, the acoustic front-end and the DSR back-end must assume the same standardization procedure for feature extraction [70, 71]. The STQ-Aurora DSR Working group has established a series of standards (*publicly available C implementations*) which can be used:

- ES 201 108 [72]
  - This standard describes the widely used mel-cepstrum based feature extraction together with compression and transmission error mitigation algorithms. This is the standard which is used in this study to implement the front-end. In this specific implementation the compression and transmission error mitigation algorithms were excluded.

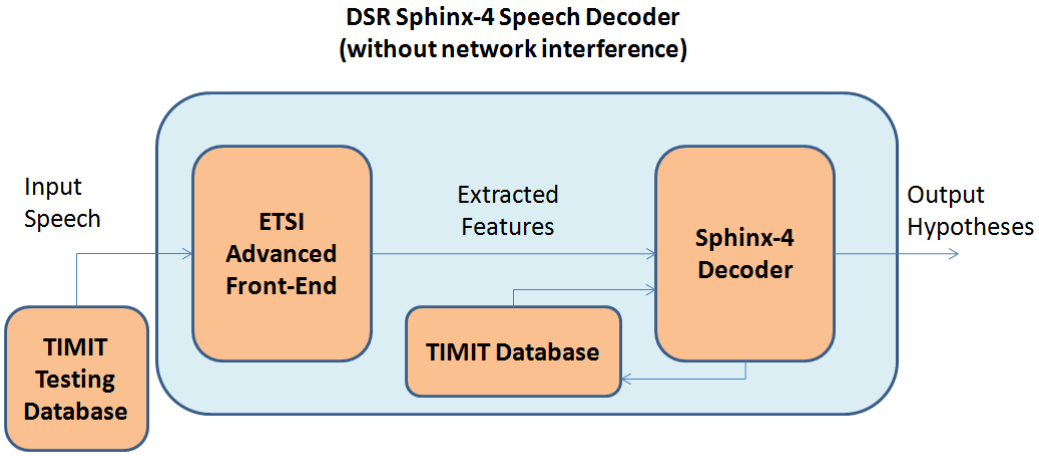

Figure 5.3: Block diagram of the DSR implementation using the Sphinx-4 recognizer without any network degradation.

- ES 202 050 [73]
  - This standard improves on the previous one, by implementing a noise robust version for counteracting the effects of a noisy environment. This is called the advanced front-end (AFE).
- ES 202 211 [74] and ES 202 212 [75]
  - These are improvements on ES 201 108 and ES 202 050. They allow for an additional 0.8kbit/s reconstruction of intelligible speech from the feature stream.
- TS 126 243 [76]
  - This standard uses only fixed-point arithmetic which is used in the extended advanced front-end.

Figure 5.3 shows a block diagram of the implementation for the ETSI front-end with DSR Sphinx-4 speech decoder without any network interference.

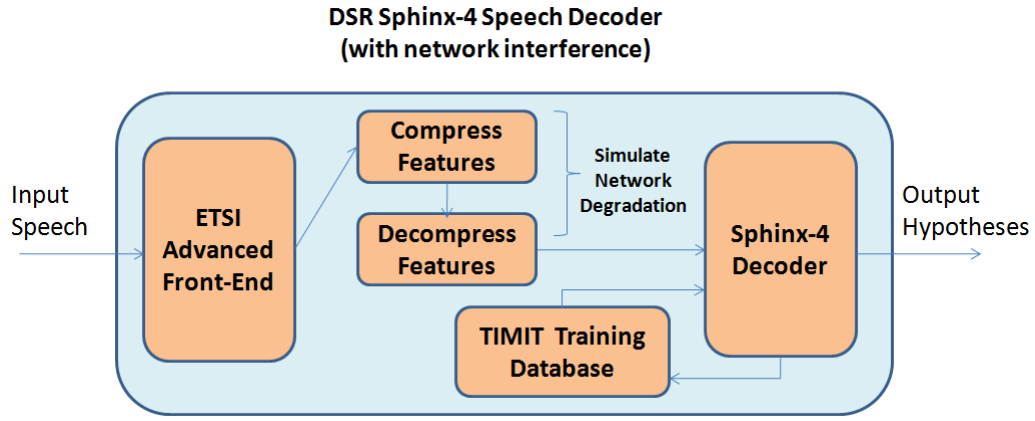

Figure 5.4: Block diagram of the DSR implementation using the Sphinx-4 recognizer including a simulation of network degradation.

5.3.5 ETSI Front-End with DSR Sphinx-4 Speech Decoder (including network degradation)

As stated in the section above, the ES 202 108 standard is used to implement the front-end for the system. This setup includes the compression and transmission error mitigation algorithms which were excluded in the previous setup. This will give a better indication as to how the system should perform under *"real"* operating conditions. The outline of this implementation can be seen in figure 5.4.

5.4 System Evaluation

A very simple approach was taken to evaluate this system using performance measures specified in section 5.4.1. Sections 5.3.3, 5.3.4, 5.3.5 are each compared by the performance measures specified in 5.4.1.

5.4.1 Performance Measures

The performance measures which are of most interest are listed below:

- Accuracy
  - For the purpose of these experiments, the accuracy is defined as the number of correct recognized words out of the total number of words

used. More specifically it is defined by [77] in the following steps:

* Speech is pre-loaded in a dialog system
* Establish a speech script comprising of test content corresponding to the speech pre-loaded in the dialog system
* Comparing the test content with the speech pre-loaded in the dialog system

- Word Error-Rate
  - The word error-rate (WER) is a common metric which is used in speech recognition [78, 79]. This can be described as follows [78]:
    * It is the edit distance between a reference word sequence and its automatic transcription which is normalized by the length of the reference word sequence
    * The normalization is applied to allow a comparison between different systems on different tasks
  - Equation 5.1, represents the above description.
    * S is the number of substitutions
    * D is the number of the deletions
    * I is the number of the insertions
    * N is the number of words in the reference.
    * r refers to the reference transcription as opposed to automatic transcription

$$WER = \frac{S + D + I}{N_r} \tag{5.1}$$

- Sentence Error-Rate (SER)
  - This is similar to the WER except that instead of comparing words, the sentence as a whole is compared as being recognized or not.

5.5 Summary

This chapter reviewed the design and implementation of the baseline system which was implemented in this study. It described the criteria which were used when designing this system, the components which were used, such as the speech corpus, different front-end standards, together with a back-end design. An overview was given of the performance measure which would be used to evaluate the system in the next chapter.

Chapter 6

Experimental Results and Analysis

In this chapter the results of the experiments obtained in Chapter 5 are presented. First a brief overview of the testing procedure is discussed after which the results and comments are given. The remainder of this chapter is broken into the following sections; 6.1 Traditional Sphinx-4 Speech Decoder results, 6.2 NSR using the Sphinx-4 Speech Decoder, 6.3 ETSI Front-End with DSR Sphinx-4 Speech Decoder (without network degradation), 6.4 ETSI Front-End with DSR Sphinx-4 Speech Decoder (including network degradation).

6.1 Traditional Sphinx-4 Speech Decoder Results

This experiment was set up according to the CMU tutorial [69]. This system was replicated on a test desktop PC in the university laboratory and the same tests were run in order to ascertain if the results were in-line with that of CMU, to give confidence that the baseline system was on the right track.

Test Database	Vocabulary Size	CMU - WER	Eval - WER
TIDIGITS	11	0.549%	0.581%
AN4	80	1.192%	3.290%
RM1	1000	2.880%	6.980%

Table 6.1: Comparison of results between CMU and the replicated system.

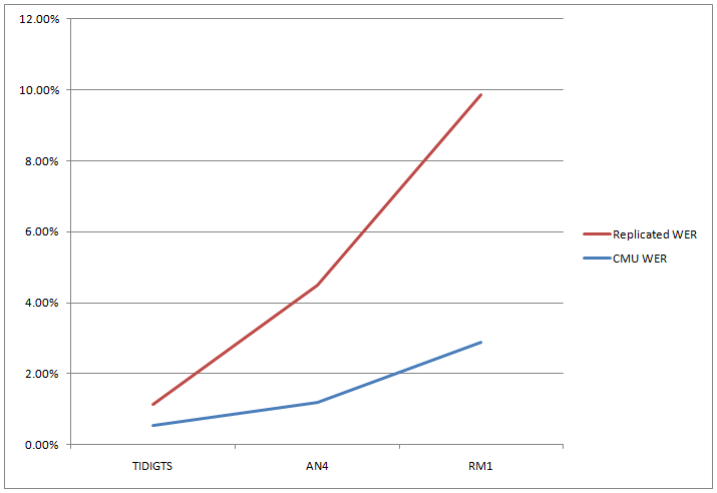

Figure 6.1: A graphical representation of the results.

6.1.1 Results of the Evaluation

These results are shown in table 6.1. Since access to all the speech databases which were initially used by CMU was unavailable owing to licensing, table 6.1 only shows a subset of the results. The AN4 and RM1 databases were made available for download and could be used in this test procedure [69]. The TIDIGTS however, is a subset of the TIMIT database which is why it could be included in the results. These results are also represented graphically in figure 6.1.

6.1.2 Analysis of Results

From table 6.1, it can be concluded that the replicated system is aligned with that of CMU even though the WER is somewhat higher than the ones noted by CMU. In the experiment conducted, the recommended parameters by [69] were used, it is possible that the results which are obtained at CMU are as a result of different parameter values being used. In this experiment were attempted a combination of different parameters in order to better match the WERs noted by [69] and the

results in table 6.1 show the results of the best combination. In the case of the TIDIGITS, the WER is only increased by 0.032% which is marginal. When we compare the AN4 database results, the WER has increased by 2.09% against the evaluation framework, this is higher than the previous TIDIGTS database but it remains relatively insignificant. When evaluating the RM1database the WER has jumped by 4.1%. This can be explained by the size of the database, the bigger the data-set becomes the more likely it is to increase the WER.

6.2 NSR using the Sphinx-4 Speech Decoder

As discussed in Chapter 5, before any experiments could be run for the network speech recognition part, the TIMIT database first needed to be trained. To do this the *transcription* files for each speech utterance had to be manually created in their database as well as manually construct the *language* and *filler* dictionaries.

Since this is quite a time-consuming task and with the TIMIT database having approximately 6300 sentences it was decided to condense the data-set and use 20 speakers from dialect region 1 (described in Chapter 5, section 5.2), with each speaker providing 10 selected sentences totaling 200 sentences which is more than double the amount of data that the AN4 database provides. For convenience, we will call this subset of data the TIMIT (200).

For this experiment the VodacomTIMIT (200) data is used to test with in order to simulate the entire speech signal being sent over the network. There is a built in function in Sphinx-4 which automatically gives the SER. Since the WER was a requirement, a tool called sclite [80] was used, which is an evaluation tool that can be downloaded from the National Institute of Standards and Technology (NIST) [81]website.

6.2.1 Results of the Evaluation

Table 6.2 shows the initial results when running a regression test. This experiment was run in triplicate and the average accuracies, WER and SER are shown. This method is more likely to produce a true reflection of the recognition rates achieved.

Test Database	Accuracy	WER	SER
VodacomTIMIT (200)	49.21%	20.16%	51.92%

Table 6.2: The initial results for the standard front-end and Sphinx4 decoder using default parameters.

Test Database	Accuracy	WER	SER
VodacomTIMIT (200)	56.27%	16.19%	44.15%

Table 6.3: The results for the standard front-end and Sphinx4 decoder using modified parameters.

6.2.2 Analysis of Results

The initial results illustrated in table 6.2 were achieved when the decoder was using the default parameter settings. These settings are shown below:

- $CFG_STATESPERHMM, this parameter specifies the number of states which represent an HMM - *default = 5*
- $CFG_SKIPSTATE, which allows the HMMs to skip states and handles the topology of the HMMs - *default = yes*
- $CFG_FINAL_NUM_DENSITIES and $CFG_INITIAL_NUM_DENSITIES, this variable determines the amount of gaussians in this mixture - *default = 8*

Through a process of trial and error, varying the settings by changing each of the parameter values one at a time and in various combinations, better results were achieved with the following set of parameters:

- $CFG_STATESPERHMM = *3*
- $CFG_SKIPSTATE = *yes*
- $CFG_FINAL_NUM_DENSITIES = 6
- $CFG_INITIAL_NUM_DENSITIES = 1

The improvement in the results can be explained because of the fact that the sample set used was significantly reduced and it is suggested by [69] using fewer gaussians in an HMM could be beneficial.

Test Database	Accuracy	WER	SER
TIMIT (200)	75.63%	9.79%	28.15%

Table 6.4: The results for the ETSI front-end and Sphinx-4 decoder using modified parameters.

6.3 ETSI Front-End with DSR Sphinx-4 Speech Decoder (without network degradation)

For this experiment the ETSI front-end was used as described by [72], together with the TIMIT (200) test database. The same training models were used as discussed in 6.2 since they have the same training database. For this experiment, the parameter settings were unchanged from that of the modified ones shown in section 6.2.2.

6.3.1 Results of the Evaluation

The results for the regression test performed on the ETSI front-end and Sphinx-4 back-end are shown in table 6.4. As with the above-mentioned experiments, these tests were conducted in triplicate and averaged out to provide more accurate results.

6.3.2 Analysis of Results

As seen in table 6.4, the DSR system with an accuracy of 75.63% and a WER of 9.79% outperforms the NSR system which only scored 56.27% recognition accuracy and a WER of 16.19%. This meant that there was an overall improvement of 34.40% in the recognition accuracy and a 39.53% improvement in the WER. This result is expected since this implementation does not send the entire voice signal through the voice network. At the front-end, the features are extracted from clean speech samples, not samples which have been sent over a network and picked up errors which explains this improvement.

From these results, it can be concluded that DSR is a better solution for speech enabling mobile devices than that of NSR. The next experiment which was conducted is described in the section below, it allows one further step to be taken by including an algorithm which will compress and decompress the features after they have been

Test Database	Accuracy	WER	SER
TIMIT (200)	76.10%	9.69%	28.05%

Table 6.5: The results of including a network phase in the experiment.

extracted at the front-end to simulate the process of the features being sent over a real voice network.

6.4 ETSI Front-End with DSR Sphinx-4 Speech Decoder (including network degradation).

In this experiment, the goal is to simulate the extracted features being sent over a voice network which would make it vulnerable to errors and then decompress it before decoding. The results of this experiment is shown in table 6.5.

6.4.1 Results of the Evaluation

Table 6.5 shows the result of incorporating a simulated network phase between the feature extractor and the decoder.

6.4.2 Analysis of Results

Interestingly enough the results which were achieved are very similar, if not slightly better than the results which were achieved for DSR without including channel interference. This is not what was expected and we can only speculate that (i) either the algorithm which we used is flawed and not introducing any channel interference or (ii) it does not sufficiently simulate the real environment.

6.5 Summary

In this chapter all the results of the experiments conducted in this study are presented. The baseline system proved to be in-line with that of the Sphinx-4 performance statistics which indicate that the implementation was correct. This chapter also showed that DSR proves to be a better performing system than that of NSR,

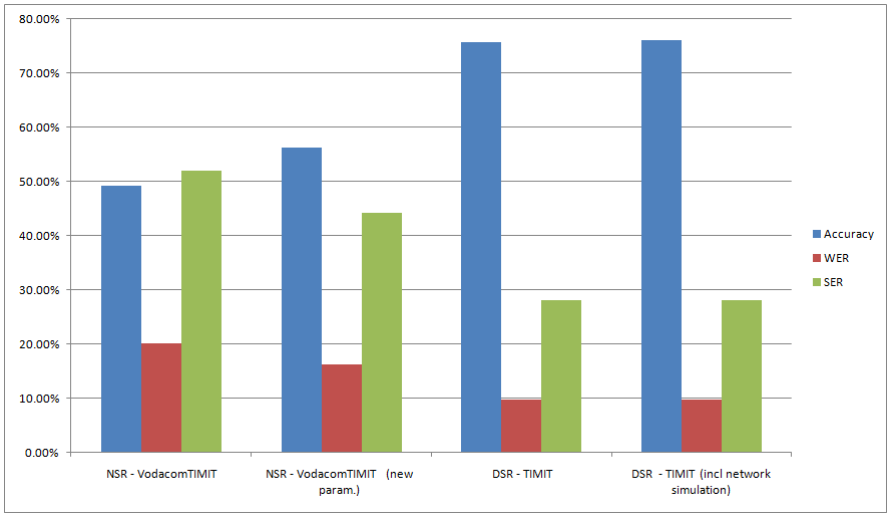

Figure 6.2: A snapshot of all results presented in this chapter.

and the surprise find was that simulating networking interference in our DSR implementation did not reduce the recognition results. A summary of all the results can be seen in figure 6.2. The next chapter will present conclusions and recommendations for future research in this area.

Chapter 7

Conclusions and Recommendations

This chapter provides a brief review of all topics covered in this thesis. All the results are reiterated and conclusions based on the research and experimental work done are drawn. Lastly, directions for future work are presented.

7.1 A Summary of the Research Conducted

As described in the problem statement in Chapter 1, mobile devices are being used for more than just initiating or receiving calls and sending SMSs. Today's smart phone's have more powerful CPUs (Central Processing Units) and larger memory capacities which allow them to run more complex applications. One of the major drawbacks is still the way in which users are forced to interact with their devices, using clumsy keypads or stylus's or even touch screens. It is conceivable that all these interfaces have improved a user's experience, but there remain challenges to using them, for example, while driving, in a situation where the driver's hands need to be free and more importantly if a person is physically challenged and can use speech to make life more manageable. Since speech is still the most natural form of communication, it is logical to implement an interface which exploits this. One of the main purposes of this thesis is to suggest a technique which could be used to implement this technology.

A review of Chapter 1, the main objectives of this study:

- The first one was to provide a comprehensive review of speech recognition and

its associated methodologies. This objective was fulfilled in Chapter 2 which focussed on the human speech production system, different speech methodologies as well as state-of-the-art speech recognition algorithms which are currently in use.

- The second objective was to review techniques which have been suggested in literature as having had success in implementing an ASR system on a mobile device. This was covered in Chapter 3.

- The third objective was covered in Chapters 4 and 5. Chapter 4 provided an overview of the Sphinx speech recognition system and Chapter 5 described the experimental framework which was implemented in using the ETSI front-end, reviewed in Chapter 3 and the Sphinx-4 back-end which was described in Chapter 4.

- The fourth objective was reached through the analysis of the results described in Chapter 6.

7.2 Conclusions

Based on the research and experiments which were conducted in this study, the following conclusions are drawn:

- The two more popular approaches to implementing an Automatic Speech Recognition (ASR) system on a mobile device is Network Speech Recognition (NSR) and Distributive Speech Recognition (DSR). Each of these systems have their associated advantages and disadvantages.

- In NSR the client device need not have a powerful processor as the feature extraction will take place at the back-end server. The client is only responsible for capturing the audio and compressing it before it is sent across the network. The disadvantage of NSR is the performance degradations which are due to low bit-rate codecs, exaggerated by background environmental noise together with transmission errors.

- DSR has the advantage that the load is spread evenly across the client and the server and only speech features are sent across an error-protected data channel which reduces the likelihood of transmission errors. The only negative is that feature extraction must take place on the mobile client, implying that it needs to have the necessary processing power and memory capacity.

- Secondly, different techniques were compared for the implementation of a speech recognition system in this study. Recall from Chapter 2, the speech recognition techniques which can be used are Dynamic Time Warping (DTW), Hidden Markov Models (HMMs) and Artificial Neural Networks (ANNs). DTW is a technique which is applied in ASR to cope with different speaking speeds, for example the duration of one utterance of speech could be different to another utterance of the same word [19]. The ANN technique is traditionally a pattern recognition technique which has only recently been applied to speech recognition with success. ANNs primary disadvantage is that its outcome can be very unpredictable. Literature suggests that HMMs are currently the standard in speech recognition systems [4,24,61]. Since it was decided upon in this study to use a DSR approach, which can use a powerful server to perform the recognition, HMMs are chosen because of their track record in producing high quality recognition systems. Since Sphinx-4 makes use of HMMs to perform its recognition, it was chosen as the baseline for this framework.

- From the results obtained in Chapter 6, when comparing NSR and DSR there is a remarkable improvement in performance of the DSR system with an accuracy of 75.63% versus NSR which achieved only 56.27%. The surprise find was that introducing network interference actually improved the recognition accuracy from 75.63% to 76.10%. The most logical conclusion is that the algorithm which was used does not simulate the real environment accurately enough.

- Finally, since a modular approach in design was taken, similar to that of the Sphinx-4 system, the framework which is shown in this study provides a good platform for future research in this area.

7.3 Recommendations for Future Work

Based on the scope and limitations of this study the following recommendations for future work which involves speech recognition on mobile devices can be made:

- This study can be taken further by implementing the baseline framework on an actual mobile device for the client-side and a server for the back-end. Comparisons can then be made between the techniques and compared to the results obtained here. This will clarify the effect which channel interference has on the recognition accuracy.

- The ETSI front-end used in this study is the ES 202 108. This standard has been improved upon as described in Chapter 5, ES 202 050, ES 202 211, ES 202 212 and TS 126 243. An implementation of the improved standards can be compared to the results in this study. This will further enhance this baseline system.

- Since most mobile devices are java enabled, another interesting investigation would be to port the ETSI standards from C to Java [70], thereafter implementing the Java versions on different client devices and evaluating the recognition accuracy and WER. Porting these standards to Java will allow testing on a wider range of mobile devices.

- CMU has released a stripped down version of the Sphinx speech recognizer called PocketSphinx [57]. It will be interesting to run this system on an embedded platform and compare its results with those of NSR and DSR obtained in this study.

- Another improvement on this baseline system could be to use a South African Speech database in different languages. This will be a mammoth task as South Africa has 11 official languages and each language is unique in its composition. By doing this researchers will be able to report on how the recognizer responds to using different language models.

Bibliography

[1] Mobile web: Latest facts and stats forecast a rosy outlook. Online Resource [Last Accessed: 2010-02-11]. http://mobithinking.com/blog/latest-mobile-stats.

[2] The number of mobile subscribers in south africa surpassed 51.9mn by the end of q1 2009. Online Resource [Last Accessed: 2010-02-11]. http://www.officialwire.com/.

[3] Portio research. Online Resource [Last Accessed: 2010-02-11]. http://www.portioresearch.com.

[4] Lawrence Rabiner and Biing-Hwang Juang. *Fundamentals of Speech Recogntion*. 1993.

[5] Giving voice to business. Online Resource [Last Accessed: 2010-02-11], 2001. http://www.midcosystems.com.

[6] Nuance. Online Resource [Last Accessed: 2010-02-11].

[7] Nuance - dragon naturally speaking. Online Resource [Last Accessed: 2010-02-12]. http://www.consumersearch.com/voice-recognition-software/dragon-naturallyspeaking-10-preferred.

[8] Nokia and Siemens. Nokia and siemens to collaborate on mobile software, application development and implementation based on open and common standards. Online Resource [Last Accessed: 2010-05-18], 2002.

[9] Anonymous. The history of voice recognition technology. *Information Management Journal*, 2004.

[10] Garfinkel. History of speech recognition and transcription software. Online Resource [Last Accessed: 2010-02-11], 1998. http://www.dragon-medical-transcription.com/.

[11] B.H. Juang and Lawrence R. Rabiner. Automatic speech recognition: A brief history of the technology development. Master's thesis, University, 2004.

[12] S.Lemmetty. Review of speech synthesis technology. Master's thesis, Helsinki University of Technology, 1999.

[13] J. Kivimaki. Very low bit rate speech coding using speech recognition analysis and synthesis. Master's thesis, Tampere University of Technology, Finland, 2000.

[14] O. Deroo. A short introduction to speech recognition. Online Resource [Last Accessed: 2010-02-11], 2004. http://www.babeltech.com/.

[15] J. P. Campbell. Speaker recognition: A tutorial. In *Proceedings of the IEEE*, volume Proceedings of the IEEE, pages 1437–1462, 1997.

[16] T. Kinnunen. Spectral features for automatic text-independent speaker recognition. *Department of Computer Science: University of Joensuu, Finland*, 2003.

[17] Masaaki Honda. Human speech production mechanisms. Technical report, NTT Communication Science Laboratories, 2003.

[18] T.W. Parsons. *Voice and Speech Processing.* McGraw-Hill, 1986.

[19] S.Wrigley. Speech recognition by dynamic time warping. Online Resource [Last Accessed: 2010-02-11], 1998. http://www.dcs.shef.ac.uk/ stu/com326/.

[20] M.W. Kadous. Dynamic time warping. Online Resource [Last Accessed: 2010-02-11], 2002.

[21] P.A. Schrodt. Early warning of conflict in southern lebanon using hidden markov models. Technical report, New York St. Martins Press, 1999. http://web.ku.edu/ keds/papers.dir/AIED.C6.pdf.

[22] P.A. Schrodt. Pattern recognition of international crises using hidden markov models. Technical report, Ann Arbor University of Michigan Press, 1999. http://web.ku.edu/ keds/papers.dir/AIED.C6.pdf.

[23] N. Warakagoda. Hidden markov models. Online Resource [Last Accessed: 2010-02-11], 1996. http://jedlik.phy.bme.hu/ gerjanos/HMM/node2.html.

[24] L.E. Baum. hidden markov model. Online Resource [Last Accessed: 2010-02-11], 1972. http://www.itl.nist.gov/div897/sqg/dads/HTML/hiddenMarkovModel.html.

[25] T.R. Iortger. A practical introduction to hidden markov models. Online Resource [Last Accessed: 2010-02-11]. Http://faculty.cs.tamu.edu/ioerger/HMMs.ppt.

[26] Electronic Stats Textbook. Neural networks. Online Resource [Last Accessed: 2010-02-11]. http://www.statsoft.com/textbook/neural-networks/.

[27] C.Stergiou and D.Siganos. Neural networks. Online Resource [Last Accessed: 2010-02-11].

[28] J.P. Hosom, R.Cole, and M.Fanty. Speech recognition using neural networks at the center for spoken language understanding. Online Resource [Last Accessed: 2010-01-20], July 1999. http://www.cslu.ogi.edu/tutordemos/.

[29] A.Marshall. Artificial neural network for speech recognition. Online Resource [Last Accessed: 2008-09-10], march 2005.

[30] D. Zaykovskiy. Survey of the speech recognition techniques for mobile devices. Technical report, Department of Information Technology, University of Ulm, Germany, 2006.

[31] T.W.Kohler, C.Fugen, S.Stuker, and A.Waibel. Rapid porting of asr-systems to mobile devices. In *The 9th European Conference on Speech Communication and Technology*, 2005.

[32] Intel. Intel performance libraries. Technical report, Intel, 2006.

[33] M.Novak. Towards large vocabulary asr on embedded platforms. In *Interspeech 2004 ICSLP*, 2004.

[34] J.M. Huerta. *Speech recognition in mobile environments*. PhD thesis, Carnegie Mellon University, 2000.

[35] B.Raj, J.Migdal, and R.Singh. Distributed speech recognition with codec parameters. In *Mitsubishi Electric Research Laboratories*, 2001.

[36] C. Pelaez, A.Gallardo-Antolin, and F.Diaz de Maria. Recognizing voice over ip: A robust front-end for speech recognition on the world wide web. *IEEE Trans*, 2001.

[37] M. Perakakis. Distributed speech recognition. *http://www.telecom.tuc.gr/ perak/speech*, 2001.

[38] ETSI. Speech processing, transmission and quality aspects (stq); distributed speech recogniton; front-end feature extraction algorithm; compression algorithms. Document ETSI ES 201 108 V1.1.2 (2000 - 2004), European Telecommunications Standards Institute, http://www.etsi.org, 2000.

[39] D. Pearce. Enabling new speech driven services for mobile devices: An overview of the etsi standards activities for distributed speech recognition front-ends. In *AVIOS: The Speech Aplications Conference*, Motorola Limited, Jays Close, Viables Industrial Estate, Basingstoke, HANTS, RG22 4PD, United Kingdom, May 22 - 24 2000. Motorola Labs and Chairman ETSI STQ-Aurora DSR Working Group.

[40] D. Pearce and H. Hirsch. The aurora experimental framework for the performance evaluation of speech recognition systems under noisy conditions. In *ICSLP 2000(6th Int Conf on Spoken Lang Processing)*, Beijing, China, October 2000.

[41] J. Li, B. Liu, R. Wang, and L. Dai. A complexity reduction of etsi advanced front-end for dsr. In *ICASSP*, volume 1. iFlytek Speech Lab, Univeristy of Science and Technology, China, 17-21 May 2004.

[42] J.Staderman, G.Rigoll, and eds. Hybrid nn/hmm acoustic modeling techniques for distributed speech recognition. Technical report, Technische Universitat Munchen, 2006.

[43] S.So, K.K. Paliwl, and eds. Scalable distributed speech recogniton using gaussian mixture model-based black quantization. vol. 48 of speech communication, School of Microelectronic Engineering, 2006.

[44] N. Srinivasamurthy, A. Ortega, and S. Narayanan. Efficient scalable speech compression for scalable speech recognition. In *IEEE International Conference on Multimedia and Expo*. Integrated Media Systems Centre, Dept of EE-Sytems, Univeristy of Southern California, Los Angeles, CA 90089-2654, 2000.

[45] T. Ramabadran, A. Sorin, M. McLaughlin, D. Chazan, D. Pearce, and R. Hoory. The etsi extended distributive speech recognition (dsr) standards: Server-side speech reconstruction. Research Report H-0200, IBM, October 22, 2003 2003.

[46] N. Srinivasamurthy, A. Ortega, S. Narayanan, A. Ortega N. Srinivasamurthy, and S. Narayanan. Efficient scalable encoding for distributed speech recognition. *IEEE Transactions on Speech Audio Processing*, 2003.

[47] P. Manolis. Distributed speech recogniton issues. Technical report, Department of Electronics and Computer Engineering, Technical Univeristy of Crete, June 2001.

[48] W.Walker, P.Lamere, P.Kwok, B.Raj, R.Singh, E.Gouvea, P.Wolf, and J.Woelfel. Sphinx-4: A flexible open source framework for speech recognition. White paper, Sun Microsystems, 2004.

[49] J.K.Baker. The dragon system - an overview. *IEEE Transactions on Acoustic, Speech and Signal Processing*, 1975.

[50] K. F. Lee, H. W. Hon, and R. Reddy. An overview of the sphinx speech recognition system. *IEEE Transactions on Acoustics*, (38), Jan 1990.

[51] X. Huang, F. Alleva, H. W. Hon, M. Y. Hwang, and R. Rosenfeld. The sphinx-ii speech recognition system: an overview. *Computer Speech and Language*, (7), 1993.

[52] P. Placeway, S. Chen, M. Eskenazi, U. Jain, V. Parikh, B. Raj, M. Ravishankar, R. Rosenfeld, K. Seymore, M. Siegler, R. Stern, and E. Thayer. The 1996 hub-4 sphinx-3 system. In *DARPA Speech Recognition Workshop*. Online Resource [Last Accessed: 2010-01-05].

[53] M.Ravishankar. Some results on search complexity vs accuracy. In *DARPA Speech Recognition Workshop*, 1997. Online Resource [Last Accessed: 2009-11-05].

[54] S.Young. The htk hidden markov model toolkit: Design and philosophy. Technical report, Cambridge University Engineering Department, 1994.

[55] X.X.Li, Y.Zhao, X.Pi, L.H.Liang, and A.V.Nefian. Audio-visual continuous speech recognition using a coupled hidden markov model. In *7th International Conference on Spoken Language Processing*, Denver, Sep 2002.

[56] N. Deshmukh, A. Ganapathiraju, J. Hamaker, J. Picone, and M. Ordowski. A public domain speech-to-text system. In *6th European Conference on Speech Communication and Technology*, number 5, Sep. 1999.

[57] Pocketsphinx - sphinx for handhelds. Online Resource [Last Accessed: 2010-02-08]. http://www.speech.cs.cmu.edu/pocketsphinx/.

[58] C.Liao. Understanding the cmu sphinx speech recognition system. *National Chenglichi Univeristy*, 2002.

[59] P.Lamere, P.Kwok, W.Walker, E.Gouvea, R.Singh, B.Raj, and P.Wolf. Design of the cmu sphinx-4 decoder. In *8th European Conference on Speech Communication and Technology*, 2003.

[60] SOURCEFOURGE. Sourceforge. Online Resource [Last Accessed: 2010-05-18]. http://sourceforge.net/.

[61] P. Lamere, P. Kwok, E. Gouvea, B. Raj, R. Singh, W. Walker, M. Warmuth, and P. Wolf. The cmu sphinx- 4 speech recognition system. In *ICASSP*, 2003.

[62] ORACLE. Java speech api grammar format (jsgf). Online Resource [Last Accessed: 2010-02-11]. http://java.sun.com/products/java-media/speech/forDevelopers/JSGF/JSGF.html.

[63] M.Mohri. Finite-state transducers in language and speech processing. *Computational Linguistics*, 1997.

[64] P. Clarkson and R. Rosenfeld. Statistical language modeling using the cmu-cambridge toolkit. In *5th European Conference on Speech Communication and Technology*, 1997.

[65] S. J. Young, N. H. Russell, and J. H. S. Russell. Token passing: A simple conceptual model for connected speech recognition systems. Technical report, Cambridge University Engineering Dept, 1989.

[66] J.S. Garofolo, Lori F. Lamel, William M. Fisher, Jonathan G. Fiscus, David S. Pallett, Nancy L. Dahlgren, and Victor Zue. Timit acoustic-phonetic continuous speech corpus. Online Resource [Last Accessed: 2010-02-11], 1993. http://www.ldc.upenn.edu/Catalog/CatalogEntry.jsp?catalogId=LDC93S1.

[67] Timit database passed through the vodacom network. Internal Project at the Speech Technology and Research Group, University of Cape Town, 2005.

[68] Defense. *The DARPA TIMIT Acoustic-Phonetic Continuous Speech Corpus (TIMIT)*. DARPA-ISTO, speech disc cd1-1.1 edition, 1990.

[69] CMU, Sun Microsystems, and Mitsubishi Electric Research Laboritories. Sphinx-4 tutorial. Online Resource [Last Accessed: 2010-02-11], Dec 2008. http://cmusphinx.sourceforge.net/sphinx4/doc/ProgrammersGuide.html.

[70] Dmitry Zaykovskiy and Alexander Schmitt. Java (j2me) front-end for distributed speech recognition. In *AINAW '07: Proceedings of the 21st International Conference on Advanced Information Networking and Applications Workshops*, pages 353–357, Washington, DC, USA, 2007. IEEE Computer Society.

[71] Dmitry Zaykovskiy and Alexander Schmitt. Deploying dsr technology on today's mobile phones: A feasibility study. In *PIT*, pages 145–155, 2008.

[72] Speech processing, transmission and quality aspects (stq); distributed speech recognition; front-end feature extraction algorithm; compression algorithm. ETSI Standard ES201 108 v1.1.3, Sep 2003.

[73] Speech processing, transmission and quality aspects (stq); distributed speech recognition; advanced front-end feature extraction algorithm; compression algorithm,. ETSI Standard ES 202 050., Oct 2002.

[74] Distributed speech recognition; extended front-end feature extraction algorithm; compression algorithm, back-end speech reconstruction algorithm. ETSI Standard ES 202 211., Nov 2003.

[75] Distributed speech recognition; extended advanced front-end feature extraction algorithm; compression algorithm, back-end speech reconstruction algorithm. ETSI Standard ES 202 212, Nov 2003.

[76] Digital cellular telecommunications system (phase 2+); universal mobile telecommunications system (umts); ansi c code for the fixed-point distributed speech recognition extended advanced front-end. ETSI Technical Specification TS 126 243, Dec 2004.

[77] Jesse Huang and Jia-Fu Chen. Method of verifying accuracy of a speech. Online Resource [Last Accessed: 2010-01-05]. http://www.faqs.org/patents/app/20080262840.

[78] Iain McCowan, Darren Moore, John Dines, Daniel Gatica-Perez, Mike Flynn, Pierre Wellner, and Herv e Bourlard. Retrieval measures for speech recognition evaluation. Technical report, IDIAP, 2005.

[79] Y.Y. Wang, A. Acero, and C. Chelba. Is word error rate a good indicator for spoken language understanding accuracy. In *IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, 2003.

[80] Nist evaluation tools. Online Resource [Last Accessed: 2010-02-08].

[81] NIST. National institute of standards and technology. Online Resource [Last Accessed: 2010-05-18]. http://www.nist.gov/index.html.

Appendix A

Timeline of Speech Recognition

- 1936
  - AT&T's Bell Labs produced the first electronic speech synthesizer called the Voder (Dudley, Riesz and Watkins). This machine was demonstrated in the 1939 World Fairs by experts that used a keyboard and foot pedals to play the machine and emit speech.
- 1969
  - John Pierce of Bell Labs said automatic speech recognition will not be a reality for several decades because it requires artificial intelligence.
- Early 1970's
  - The Hidden Markov Modeling (HMM) approach to speech recognition was invented by Lenny Baum of Princeton University and shared with several ARPA (Advanced Research Projects Agency) contractors including IBM.
  - HMM is a complex mathematical pattern-matching strategy that eventually was adopted by all the leading speech recognition companies including Dragon Systems, IBM, Philips, AT&T and others.
- 1971

  - DARPA (Defense Advanced Research Projects Agency) established the Speech Understanding Research (SUR) program to develop a computer system that could understand continuous speech. Lawrence Roberts, who initiated the program, spent $3 million per year of government funds for 5 years. Major SUR project groups were established at CMU, SRI, MIT's Lincoln Laboratory, Systems Development Corporation (SDC), and Bolt, Beranek, and Newman (BBN). It was the largest speech recognition project ever.

- 1978

  - The popular toy "Speak and Spell" by Texas Instruments was introduced. Speak and Spell used a speech chip which led to huge strides in development of more human-like digital synthesis sound. 1982 Covox founded. Company brought digital sound (via The Voice Master, Sound Master and The Speech Thing) to the Commodore 64, Atari 400/800, and finally to the IBM PC in the mid '80s.

- 1982

  - Dragon Systems was founded in 1982 by speech industry pioneers Drs. Jim and Janet Baker. Dragon Systems is well known for its long history of speech and language technology innovations and its large patent portfolio.

- 1984

  - SpeechWorks, the leading provider of over-the-telephone automated speech recognition (ASR) solutions, was founded.

- 1993

  - Covox sells its products out to Creative Labs, Inc.

- 1995

  - Dragon released discrete word dictation-level speech recognition software. It was the first time dictation speech recognition technology was available to consumers. IBM and Kurzweil followed a few months later.

- 1996

  - Charles Schwab is the first company to devote resources towards developing a speech recognition IVR system with Nuance. The program, Voice Broker, allows for up to 360 simultaneous customers to call in and get quotes on stock and options... it handles up to 50,000 requests each day. The system was found to be 95% accurate and set the stage for other companies such as Sears, Roebuck and Co., and United Parcel Service of America Inc., and E*Trade Securities to follow in their footsteps.

- 1996

  - BellSouth launches the world's first voice portal, called Val and later Info By Voice.

- 1997

  - Dragon introduced "Naturally Speaking", the first "continuous speech" dictation software available (meaning you no longer need to pause between words for the computer to understand what you're saying).

- 1998

  - Lernout & Hauspie bought Kurzweil. Microsoft invested $45 million in Lernout & Hauspie to form a partnership that will eventually allow Microsoft to use their speech recognition technology in their systems.

- 1999

  - Microsoft acquired Entropic, giving Microsoft access to what was known as the "most accurate speech recognition system" in the world.

- 2000
  - Lernout & Hauspie acquired Dragon Systems for approximately $460 million.
  - TellMe introduces first world-wide voice portal.
  - NetBytel launched the world's first voice enabler, which includes an online ordering application with real-time Internet integration for Office Depot.
- 2001
  - ScanSoft Closes Acquisition of Lernout & Hauspie Speech and Language Assets.
- 2003
  - ScanSoft Ships Dragon NaturallySpeaking 7 Medical, Lowers Healthcare Costs through Highly Accurate Speech Recognition.
  - ScanSoft Closes Acquisition of SpeechWorks International, Inc. 2003 ScanSoft closes deal to distribute and support IBM ViaVoice Desktop Products.