Impaired Speech Recognition

A thesis presented in partial fulfilment of the
requirements for the degree of

Master of Information Sciences
in
Computer Science

Massey University, Albany, New Zealand.

Mohammed Nasser Almujil

2015

Abstract

The purpose of this thesis is to present a novel mobile health application that can recognize impaired speech (using audio signals) and turn it into understandable speech. The system is developed to help Dysarthria of Speech patients communicate better with others in their everyday life. It will provide some background information about motor speech disorders, Dysarthria of Speech and the technical aspects of this application. It will then explain and test the algorithms to recognize impaired speech using audio fingerprinting technology. Finally it will discuss the test results and recommends some future work to improve the current algorithms.

Acknowledgements

I would like thank my project supervisor, Associate Professor Ruili Wang, for his advice and support throughout the course of this project, and for giving me the opportunity to write about this interesting topic.

Contents

List of Figures

1 Introduction

In this section we will introduce the motivation and objectives of this thesis. Sections 1.1 and 1.2 will give some background information about motor speech disorders, Dysarthria of speech and mobile applications. Section 1.3 will explain the communication problem currently faced by Dysarthria of speech patients. Finally, section 1.4 outlines the thesis main objectives and finally section 1.5 proposes a solution to the current issue and an approach to meet the thesis objectives.

1.1 Moto speech disorders

Motor speech disorders (MSDs) are a result of neurological impairment affecting the programming, planning, control and execution of speech [3]. Figure 1 illustrates the different types of acquired neurologic communication disorders recorded in the Division of Speech Pathology in the Department of Neurology at the Mayo Clinic from 1993 to 2008. It can be clearly seen that Dysarthria is responsible for 53% of the primary diagnoses and that it is far more dominant than any other category. Therefore, this thesis will focus on providing a query-by-example (QBE) mobile health speech assistance service to Dysarthria of speech patients. Mobile health is identified by The Global mHealth Initiative at Johns Hopkins University as ‘‘the use of mobile information and communication technologies to leverage health outcomes.’’ [42]. Mobile health applications can improve the situation of the individual and also can reduce the cost of treatments as it has been proven across a range of clinical areas all over the world [36][38][39][44] and in New Zealand [40]. For example, the use of

mobile health applications to collect information from patients and send them to doctors instead of the traditional paper based technique is very cost-effective and time saving especially in developing countries [46].

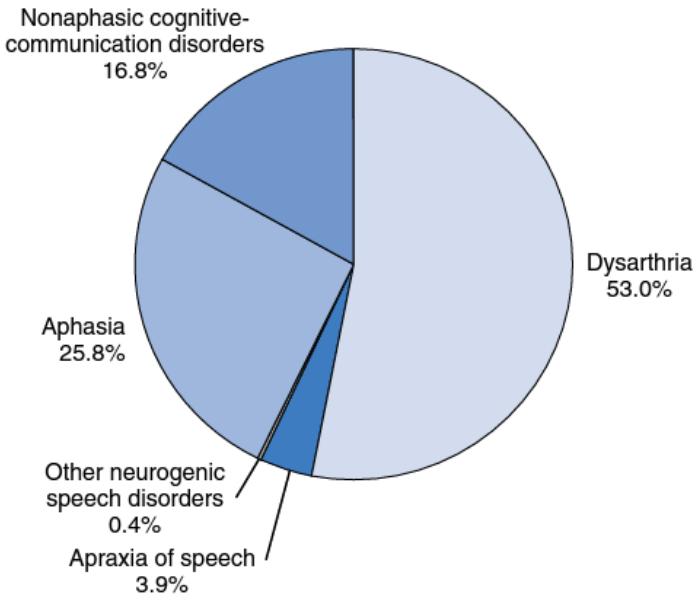

Figure 1: Neurologic communication disorders. Retrieved from: [3]

1.2 Dysarthria

Dysarthria of speech is a MSD and it represents a group of neurologic speech disorders that are associated with abnormalities in the strength, speed, range, steadiness, tone, or accuracy of muscle movements required for speech production. As a result, Dysarthria patients produce impaired or difficult to understand speech [3]. Getting involved into conversations is a difficult task faced by Dysarthria of speech patients and it is their primary communication concern. The result of this impairment at a personal level range from anxiety to total social withdrawal [4] [6]. Furthermore, the type of Dysarthria this

thesis is targeting is not related to any issues with understanding cognitive language and Dysarthria patients are linguistically normal [3].

1.3 Current situation

The output of an appropriate text-to-speech synthesizer increases motivation between Dysarthria speakers and their conversational partners [6]. There is evidence to suggest that positive attitudes toward Dysarthria individuals are influenced by the use of text-to-speech synthesizers rather than other communication devices. A text-to-speech synthesizer can replicate, more closely, normal speech that conversation partners are more familiar with. In some cases where produced speech is completely impaired, the text-to-speech synthesizer will become the voice of the individual. The text-to-speech synthesizer differs from traditional communication techniques performed in hospitals and clinics. Patients don't need to spend time in an unfamiliar place with limited privacy which may affect their health and mood [35]. Instead, they can communicate with people anywhere they like. However, the limitation of the text-to-speech synthesizer technology is that it restricts how long the conversation will last, due to issues such as the time taken to produce speech [6].

1.4 Objectives

The main objective of this thesis is to develop a faster technology that will help speed up the process of composing and producing understandable speech by using impaired speech as an input using a mobile phone or tablet microphone. Another objective is to examine whether or not Musicg [11] audio fingerprinting algorithms can detect

similarities between two audio files by only examining their waveform and their generated fingerprint. Final objective is to investigate if it is possible to use Musicg fingerprinting algorithms to speed up the process of composing and producing understandable speech. If not, this thesis will propose a different fingerprinting algorithm approach for future work to find a way to satisfy the main objective.

1.5 Speech Assistant Introduction

We will attempt to develop a faster technology that will help speed up the process of composing and producing understandable speech. The speech assistance service retrieves the identity of audible prerecorded impaired speech, by sampling a few seconds of audio, using a mobile phone as a recording device. Once the audible prerecorded impaired speech is identified, we can then know the corresponding text of the impaired speech, which can be turned into understandable speech by a text-to-speech synthesizer. This technology will significantly reduce the time taken to type the required sentence.

Query by example is a more flexible impaired speech recognition modality than the one provided by text-to-speech mobile applications. It is faster to construct understandable speech from a sample recording than by requiring the patient to key in the phrase. The speech assistance service can be implemented using a combination of two computer science fields, namely, Speech Processing and Machine Learning [5]. Audio fingerprinting in Speech Processing has attracted a lot of attention for its audio

monitoring and identification capabilities. It is used in leading QBE music search service Shazam, provided by Shazam Entertainment Ltd. [5]

Motor speech disorders, Dysarthria of speech, mobile health, the current communication issue and a proposed solution to the communication issue were the main aspects discussed in this section.

2 Literature review:

Section 2.1 introduces audio fingerprinting technology. Section 2.2 compares and contrasts audio fingerprinting with its alternatives in an effort to find out which technology is more suitable for this assistance application. Section 2.3 describes the properties of any audio fingerprinting system. The next section discusses most suitable audio fingerprint extraction and similarity matching algorithms for this application. Section 2.5 explains possible data storage challenges and the most suitable database for this assistance application. Section 2.8 discusses EEG signal systems and why Audio fingerprinting technology was used to recognize impaired speech over EEG signal systems. Finally, human computer interaction with regards to mobile health applications is reviewed in section 2.9

2.1 Audio Fingerprinting

Audio fingerprinting or Content-based audio identification (CBID) systems extract a perceptual digest of a piece of audio content, i.e. the voice fingerprint. The main objective of extracting a voice fingerprint is to be able to compare two audio files for similarities without having to compare the actual large audio files, by using a summarized short textual representation of the large audio content instead. [1]. Summarized short textual representation is usually stored in a database. When presented with unlabeled audio, its fingerprint is calculated and matched against those stored in the database. Using fingerprints and matching algorithms, distorted versions of a recording can be identified as the same audio content [8]. Database storage space

is not big since they are only textual representations of the actual audio file, so the fingerprint only contains the important part of the audio, making it easy to search and store [1].

Audio Fingerprinting technologies have recently attracted attention since they allow the monitoring of audio independently of its format, and without the need of meta-data or watermark embedding [9]. It became increasingly popular since music is easily copied and distributed illegally on the internet by music pirates [2]. Audio fingerprinting technologies and algorithms have been successfully applied to various applications but notably by commercial application developers. Shazam Entertainment, Ltd, helps people connect to music through recognizing music in their environment by using their mobile phones. The algorithm can recognize a short audio sample of music that had been broadcast, mixed with noise, audio distortion in its surroundings and audio filters, all before arriving at their servers. The algorithms performs the recognition quickly over a large database of music with nearly 2M tracks, and has a low number of false positives while having a high recognition rate [10]. It extracts acoustic relevant characteristics of song content and stores them in a database. When presented with unidentified song content (for instance, a user wanting to know what song they are listening to, played in the car radio), characteristics of that piece are calculated and matched against those stored in the database. The song title and a download link to the song is then provided to the user [8].

2.2 Audio Fingerprinting alternatives

The technology revolution in the recent past allowed for illegal distribution of copy-righted music on the internet. Copy-right holders needed a system that can identify music distributed illegally in the internet in order to issue a copyright infringement notice. Research has been active in this area ever since. Watermark audio identification systems were proposed as a solution to this problem. It adds a mark, the watermark, to the original audio signal. The watermark does not affect the audio quality and is permanently embedded in the audio signal. The detection system should be able to find the watermark in a broadcasted audio signal and retrieves watermark data. Fingerprinting systems were also later proposed as a potential solution to the problem [2].

Both systems work well in different types of applications but we are interested in comparing both systems with regards to our impaired speech recognition application. The following table compares and contrasts both systems with regards to impaired speech recognition [2].

	Fingerprinting	Watermarking
Audio signal	any	Watermarked audio signal only
Database	Yes	no
Robustness	Yes	no

Table 2.1: Fingerprinting and watermarking

The following sections will discuss both systems in detail with regards to this assistance application.

2.2.1 Audio signal

Fingerprinting systems don't need to previously process all audio signals in order to be able to detect them. A voice fingerprint does not change the audio signal at all but rather analyzes it and constructs a hash (a Fingerprint) uniquely associated with this signal. On the other hand, watermarking systems must previously process input audio and add a watermark in the audio signal. For this application, where impaired speech is provided by the user through a mobile phone microphone, audio is generated as a query from the user. Every audio query will not be the same and audio signal must be processed on the fly. We will need to rely on analysis of the audio signal and therefore watermarking will not work in this application [2].

2.2.2 Database

Audio fingerprinting systems need to acquire some knowledge about the audio signals so it will be able to detect them. It will only require the audio signal during the identification phase. A database must be built which will hold knowledge about audio signals from the user input. In this application, the database contains the fingerprints

of all phrases the system is supposed to detect. During detection, the fingerprint is extracted from an input audio signal and the matching algorithm compares it to all fingerprints in the database. The database must be updated with new audio signals from the user, for each new phrase. As the number of fingerprints in the database grows, the detection process becomes more complex as memory requirements and computational costs also grow [2].

In contrast, Watermark systems require no database. Instead, it analyzes the audio signal in order to identify a watermark. Having found one, it retrieves watermark data. The data is embedded in the signal itself and the system's job is to look for a watermark and extract its data. This gives it a huge advantage, as the complexity of the detection process remains unchanged even when new songs are released [2].

2.2.3 Robustness

Watermark data in the audio signal occupies a tiny area of the audio signal and is much weaker compared to the original audio signal due to such constraints as limiting the effects on audio quality. Also, MP3 compression and piracy attacks (in case of music) can add noise to the audio signal which has even worse effects on the embedded watermarked data. Consequently, the watermark may be no longer detectable [2].

Fingerprint systems, on the other hand, rely on the audio signal only during the detection process and this is why they are inherently more robust. Distorted and tampered with audio signal can still be detected, depending on the fingerprint extracting

process and algorithms. As long as the fingerprinted audio signal in the database sounds “approximately” the same, the input audio is regarded “approximately” the same, too. The degree of proximity depends on the system’s audio signal processing algorithms and robustness [2].

2.3 Audio fingerprinting system properties

Fingerprinting system requirements vary depending strongly on the application but these properties are useful when comparing various audio fingerprinting systems [8]. The following table illustrates and explains audio fingerprinting system properties.

Property	Description
Accuracy	The application must be accurate. This can be done by noting the number of phrases correctly identified, wrong phrase identifications (false positive) and no-results-found identifications.
Reliability	The system should not provide false positive results. Positive results can only be provided if a strong match is found otherwise the result should always be false negative.
Robustness	Algorithms should be able to detect impaired speech regardless of surrounding audio signal noise and distortion. For instance, it should be able to detect impaired speech in places such as a café.
Security	The system should protect the application from being hacked or tampered with particularly by anything that will influence the detection algorithms. Also, it should protect the user's private information by encrypting all information about the user [45]. 95.63% of apps in the market could cause some damage through information security or privacy infringement [49]. Table 2.3 illustrates possible security risks and their potential solution.
Versatility	The system should have the ability to detect audio given a wide variety of formats.
Scalability	The system should have scalable algorithms that can work fast with large number of records.
Complexity	This includes computational costs of the fingerprint extraction process, the fingerprint size, the complexity of the look up algorithms, the complexity of the fingerprint comparison algorithms and the cost of adding new items to the database [8] [2].

Table 2.2: Audio fingerprinting system properties

2.4 Audio fingerprint extraction and matching algorithms

There are two solutions available to extract and identify audio fingerprints. The first solution is to use an open source library that provides both functionalities. The second solution is to use fingerprint extraction and identification algorithms from available

audio fingerprinting systems. Both solutions are reviewed to identify which one best suits this application.

2.4.1 Open source library: MusicG

MusicG is a lightweight audio analysis library, written in Java, with the purpose of helping developers work with audio. It offers various operations such as reading, cutting and trimming audio easily while recording is still in progress. Most importantly, it extracts a fingerprint from an InputStream or from a wave file. It also provides the ability to compare two fingerprints for similarities. It returns a score from Zero to one, where Zero means no similar feature is found and one means on average there is at least one match in every frame from both audio from which the fingerprints were originally extracted. It provides tools for digital signal processing, which renders the waveform or spectrogram for research and development purposes. It is freely available to modify and use even for commercial applications. Access to all source code is available, so changes can be made when needed to adjust to the assistance application needs [11].

2.4.2 Audio fingerprinting systems

There are multiple papers such as [1], [9], [12] and [18] that propose and review fingerprinting algorithms as well as complete fingerprinting systems. Most of these papers are domain specific. For instance, some are designed to identify music with a gigantic knowledge database. Also, most of them do not fully fit with the requirements of this application. Instead, some parts are relevant but others are irrelevant. For

example, in [1] the search algorithm is irrelevant because it is designed to search songs with an average size of five minutes length using a modern personal computer. In contrast, for this application we search a few seconds of audio using a mobile phone/table. Mobile phones are far less powerful than personal computers. On the other hand, we can experiment with the fingerprint extraction algorithms which might be helpful and relevant.

For some of these frameworks, one would need access to a database with millions of fingerprinted music data in order to perfectly test the algorithms. Working with such systems will require a combination of algorithms from different papers because one size does not fit all the application requirements. This is possible to implement but it is also time consuming.

For this application, it is best to use MusicG open source library for its powerful features and ease of use. Also, since access to source code is available, fingerprint extraction and detection algorithms can be improved in further development of this product. Since MusicG is written in Java, it means it is android compatible and can be used in an android powered mobile phone. Finally, using it gives this application a head start with the possibility to develop its algorithms further to deliver a novel purpose. Fingerprint extraction and similarity comparison are the only two features used from this library. Other functionalities in this application such as recording, comparison algorithms and knowledge storage are developed for this application and explained in section three of this thesis.

2.5 Knowledge storage

The knowledge database of a fingerprint system is traditionally remotely hosted by service providers. For instance, Shazam Entertainment, Ltd store fingerprints obtained from millions of songs in a remote knowledge database. In addition, they provide a song detection service through mobile phones where users can query their knowledge database [10].

Since the impaired speech detection service is specific to a single user, local knowledge storage is used instead of a remote database. Using local knowledge storage is faster to query than from a remote server. Also, no internet connection required for the service to be fully functional.

Instead of using a traditional database, fingerprints for each phrase are stored in JavaScript Object Notation (JSON) file format. Each phrase will have its own JSON file as will be illustrated in the methodology section of this thesis. JSON is described by json.org as "a lightweight data-interchange format. It is easy for humans to read and write. It is easy for machines to parse and generate. JSON is a text format that is completely language independent but uses conventions that are familiar to programmers of the C-family of languages, including C, C++, C#, Java, JavaScript, Perl, Python, and many others. These properties make JSON an ideal data-interchange language."[13]

In further development of this product where there is a need for a remote database, the local knowledge database can be easily converted to a remote database. This is because the storage format (JSON) is highly scalable.

2.6 Audio filters

Accurate and efficient retrieval of impaired speech from audio data is one of the most important issues that need to be resolved especially in noisy environments. The application is intended to be used by patients in different environments especially in noisy environments. Capturing only the parts of the audio needed by the application will dramatically improve the fingerprint quality resulting in a reasonably high similarity and matching accuracy. [17]

2.7 Android

The target platform is the Android operating system. Android powers hundreds of millions of devices in over 190 countries around the globe and the numbers are growing, fast. The application store for Android devices is Google Play store. Google play store has over 8000 medical applications. According to the Mobile Health Market Report 2013-2017, around 500 million people will have adopted medical applications by 2015 [27]. Moreover, it has been predicted that mobile health applications will have a market value of over 11 billion USD by 2018 [29]. This application was developed using Android Studio and it is the official Android application integrated development environment (IDE). The programming language used is Java programming language.

Android, Android Studio and Java programming language are freely available for development and use, even for commercial applications [14] [15].

2.8 EEG

Electroencephalogram signals (EEG) transforms activities in the human brain to signals which is analyzed using signal processing and machine learning techniques and then translated into commands to an end user application. EEG gained popularity in applications that help patients with disability such as Amyotrophic Lateral Sclerosis (ALS), locked-in syndrome and other neurological disorders. Electroencephalographic signal based applications main objective is to help patients with disability communicate better with others. There are various papers that experiment with EEG technology to help people with disability. For example, [19] is using EEG technology to implement a synthesizer application for the disable using eye blinks as input. Others have tried to recognize unspoken speech using EEG signals such as the research provided in [20]. Unspoken speech is described by [20] as “the process in which a subject imagines speaking a given word without moving any articulatory muscle or producing any audible sound”.

EEG could be used to recognize impaired speech but there are few issues. The focus of this thesis is to recognize impaired speech for Dysarthria of speech patients. Dysarthria of speech patients, we are targeting, have no issues related to understanding cognitive language and Dysarthria patients are linguistically normal. The majority are able to walk, talk and perform activities like other able people. EEG is usually suitable for

patients with severe disability such as full or half paralysis. Another issue is that EEG is not portable, patients will need to wear an EEG cap which works with a separated device that receives EEG signals. Also, an EEG system is expensive to buy and only serves a single purpose. Although, it is important to note that there are some EEG systems that are portable as they use mobile devices. Those devices are, for example, used in the fitness and well-being business field. Also, these devices are increasingly becoming more affordable. For example, see http://neurosky.com/biosensors/eeg-sensor/ [51]

Although a mobile EEG can be a good supplement to our system, for this research we chose to recognize impaired speech using a mobile device and audio fingerprinting technology only. Firstly, smart phone and tablets penetration is increasing dramatically. 46% of people in New Zealand used their smartphones in the past seven days. People carry smartphones with them wherever they go and as a result it became a central part of their lives [21]. According to Research New Zealand's key findings from their Report on a Survey of New Zealanders' Use of Smartphones and other Mobile Communication Devices 2015, One-in-two smartphone users prefer their smartphone over any other device and that smart phones are now an "everyday device". Also one of their most important key findings to this research is that almost all smartphone and tablet users use apps for key functions [22].

Recognizing impaired speech using mobile phones for this thesis research is the best option because most patients are using smartphones already and don't need to obtain a new device. Also, mobile phones are cheaper to buy comparing to EEG systems. Finally, mobile phones are multi-purpose devices. Users can use them to recognize impaired speech and for other tasks in their everyday lives.

2.9 Human Computer Interaction

Dysarthria of Speech patients must be taken into consideration when developing the application [43][50]. In this first prototype, users' technical abilities should taken into considerations. Since this assistant application is intended to be used regularly, the user interface design should be designed for mobile screen size. The application should also have fast navigation. Buttons should be bigger so they are clear in small smartphone screens especially for elderly users. The application design shouldn't have any screen gestures because some elderly users may not be able to reach functionalities triggered by gestures such as screen swipe. Such changes to the user interface design can help make the application easier to use [31]. Having a simple user interface design shouldn't result in a boring application. It should have a good modern design while maintaining its simplicity. The application should also engage its users immediately otherwise they won't return to the application in the future. There is no simple formula for increasing users' retention and engagement [32]. The use of personalization for in-app user interface design may help increase retention. For example allowing users to change the text color or the buttons sizes [33]. The results in [41] suggested that users wanted a more engaging application and one their feedback was that they wanted a "more

graphically appealing user display". Developers could also use the concept of gamification to improve users' retention. Gamification is described by [47] as "A process by which the desirable features of games are used in non-game contexts to leverage a person's interest in competition to achieve results in another realm" However, taking this into consideration, following a mobile health development framework and using quality of experience tools during the development process may help make an engaging and effective application.

Many mobile health applications are published to the public with little understanding of their end users. Such applications are hard to use and most likely will fail [25]. To avoid this, development team should conduct usability tests during the development process. Investigators with suitable roles should present the application to its end users and test its efficiency and user satisfaction. Feedback is then returned to the development team for improvements [26]. Collaboration between doctors and application developers maybe challenging to manage. Luckily, there are frameworks such as the one proposed in [28] that can help manage the communication between researchers, doctors and application technical developers. The framework were especially made for mobile health applications development. Before the application deployment, developers can measure the quality of user experience. Researchers at [30] and [37] developed a tool specifically designed to measure the quality of experience for mobile health applications. It can significantly help developers understand how the application is doing in various aspects such as quality of content, security, ease of use, availability, performance and appearance. Context-based user interface development is

also important. Developers should take into consideration how the application will perform in different environments. For example, developers should investigate the best sound level when the application is used in a slightly noisy environment [34].

2.10 Summary

In this section we discussed Audio fingerprinting, database challenges, audio filters, Android operating system , human computer interaction and EEG signal.

3 Methodology:

In this section we explain application usage as well as backend algorithms and processes for this assistant application. There are two different processes, both processes depend on each other to make up the final product. The first process is the training process, which will be explained along with its backend processes. The second process is the detection process, which is where the application attempts to recognize impaired speech.

The first is the training process, during which, users select and read a phrase and concurrently record a few seconds of impaired speech using an Android phone or tablet as a recording device. The application then extracts a perceptual digest of the recorded audio content, i.e. the fingerprint. The fingerprint is saved locally alongside with other fingerprints in a JSON file named after the selected phrase.

The second process is the detection process. It retrieves the identity of audible prerecorded impaired speech by sampling a few seconds of audio using an Android phone or tablet as a recording device. The fingerprint is extracted from sampled audible content and compared against a prerecorded collection of fingerprints. Identified prerecorded impaired speech is turned into understandable speech then “spoken” by a text-to-speech synthesizer. Figure two illustrates an overview of the assistance application framework highlighting each process.

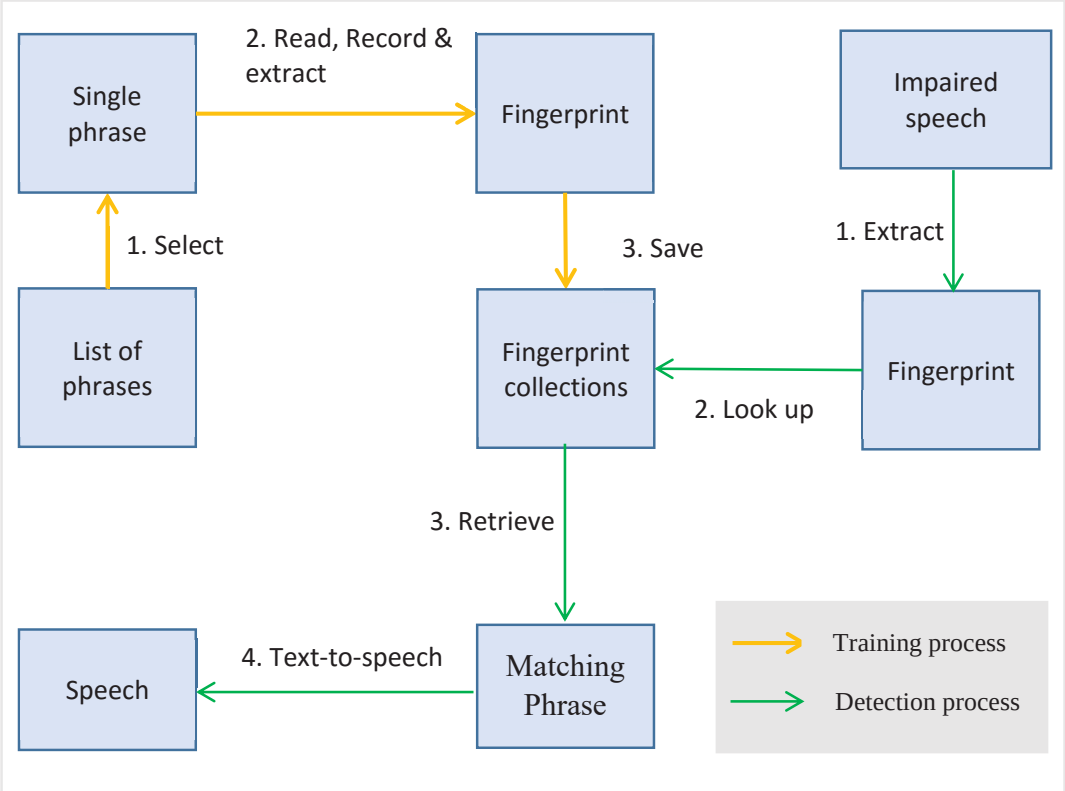

Figure 2: Application framework

The next section will discuss both processes thoroughly, starting with the training process and moving to the detection process.

3.1 Training process

During the training process, the user teaches the application to understand their voice, preparing it for the detection process. Detailed description of the technical aspects in the training process is explained in the following sections

3.1.1 Phrase selection

Users are firstly presented with a list of phrases that are commonly used in a daily basis. Phrases such as “How are you?” and “I am good, thank you”. They are presented as a list of radio buttons. Users can select a phrase and only one at a time. Figure 3 illustrates the presentation of the phrase selection step.

Figure 3: Phrase selection

3.1.2 Read selected phrase

Having selected a phrase, users can begin the next step by pressing the “Record” button as shown in Figure 3. This will change the page view to the recording page as illustrated in figure 4. The recording will start as soon as the view appear. Users then should read the selected phrase while their voice is being captured and analyzed.

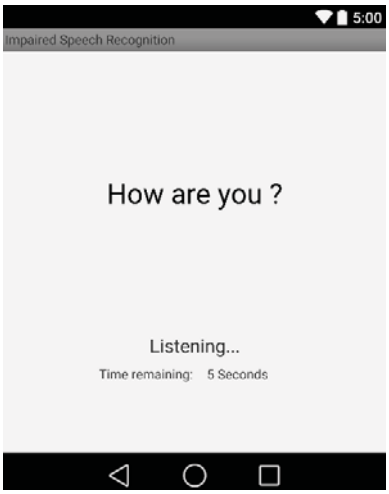

Figure 4: Listening view.

3.1.3 Audio recording.

While the phrase is being read, the voice of the individual is recorded and examined concurrently in order to extract a fingerprint. Android offers two applications programming interfaces (APIs) for audio recording. They are AudioRecord class and MediaRecorder class. The AudioRecord class allows for uncompressed recording where MediaRecorder automatically records compressed audio into a file. AudioRecord class provides more control over recorded audio. I am able to analyze and apply audio filters such as noise removal while audio is being captured. Processing audio while recording is still in progress is also faster. One disadvantage of using AudioRecord is that it is hard to manage where MediaRecorder on the other hand is easier.

The AudioRecord class manages the audio resources for Android applications to record audio from the audio input hardware of the device. Upon creation, an associated audio

buffer is constructed that it will fill with the new audio data. The size of this buffer, specified during the construction, determines how long an AudioRecord can record before "over-running" data that has not been read yet. Data should be read from the audio hardware in chunks of sizes lower than the total recording buffer size [7]. The AudioRecord object is highly configurable; the following code snippet is the AudioRecord class constructor. Table 3.1 represents parameters used to construct AudioRecord object in this application.

```
public AudioRecord(int audioSource, int sampleRateInHz, int
channelConfig, int audioFormat, int bufferSizeInBytes)
```

Parameters	Value
audioSource	***AudioSource.MIC***
sampleRateInHz	***44100Hz***
channelConfig	***AudioFormat.CHANNEL_IN_MONO***
audioFormat	***AudioFormat.ENCODING_PCM_16BIT***
bufferSizeInBytes	***8820***

Table 3.1: Recording configuration

As shown in Table 3.1, the AudioRecord object is constructed with *bufferSizeInBytes* set to 8820. However, Audio data is written to a buffer size of 4410, which is *bufferSizeInBytes/2*. Upon recording completion, written data is analyzed further to extract a fingerprint.

3.1.4 Fingerprint extraction and organization.

While recording is still in progress. AudioRecord object reads audio data from the audio hardware for recording into a buffer. A full buffer represents a new chunk of

audio used to extract a fingerprint. Figure 5 illustrates algorithms used to extract and organize fingerprints for a single recording session.

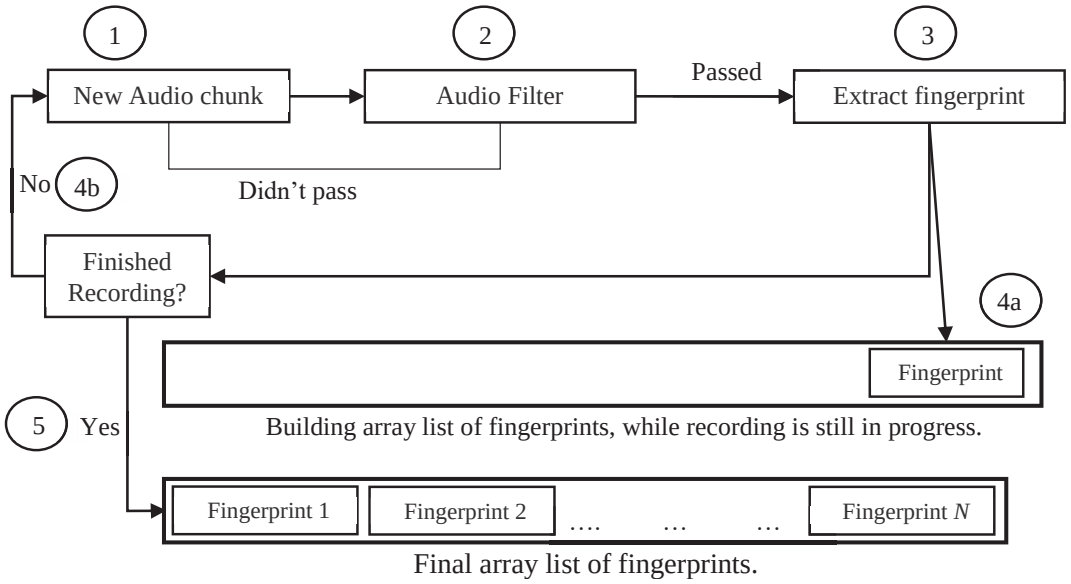

Figure 5: Fingerprint extraction and organization.

As illustrated in Figure 5, while recording is still in progress, a new chunk of audio is sent from the AudioRecord object as a full buffer of audio data. In the second step the application examines audio data to determine if it is noisy or not. Valid audio data is passed to the third step for fingerprint extraction. Failed Audio data is not used and instead the application waits for a new full buffer of audio data in step 1.

Having extracted a fingerprint from valid audio data in step 3, the application appends the extracted fingerprint to an array list of fingerprints specific to the current recording session as illustrated in step 4a. The application then determines whether the recording has reached an end or not, and if the recording is not finished a new audio chunk is obtained and the process is repeated. Upon completion, a final array list of fingerprints for every chunk of audio from the recording process is released as shown in step 5. The final array list is then saved in a JSON file named after the currently recorded phrase.

3.1.5 Fingerprint storage.

The final array list of fingerprints from 3.1.4 is converted into a JSON object and then pushed to the end of a JSON array of recording objects. Every recording object has two values, a field which holds metadata about the object itself and a JSON array which represents a single collection of fingerprints from a single recording session.

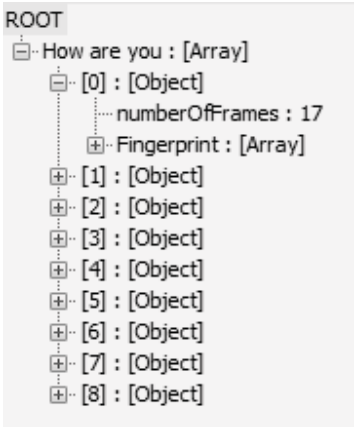

Figure 6: Fingerprint storage

Figure 6 illustrates the storage structure used to store fingerprints by phrase. It can be seen that ROOT element of the JSON file is a JSON array of objects named “How are you: [Array]”. Each object in this array contains an impaired speech recording session. Let’s take for example “[0] : [Object]”, it’s a single impaired speech recording session of the phrase “How are you?” and it has two elements. The first element is “numberOfFrames: 17”, which is metadata about the second element “Fingerprint: [Array]”. It tells us that “Fingerprint: [Array]” length is 17. Each array element of the “Fingerprint: [Array]” was extracted from the audio buffer (audio chunk) while recording was in progress. Each phrase has its own file and they all follow the same structure. Files are located in the device’s external storage. For instance, in Figure 6 the associated file is called “How are you.json” in the device file system.

Fingerprint storage marks the final step in the training process. A collection of fingerprints for every phrase is now available to be examined for the detection process.

3.2 Detection process

The detection process is where the application attempts to understand impaired speech. It examines previous recordings against input audio data in an effort to find a matching phrase. Upon success, the found phrase is “spoken” using a text-to-speech synthesizer. The following sections will give some insight into the detection algorithms used in this assistance application.

3.2.1 Start detection process

On the main page of the assistant application, users can press a “Start” button to initiate the detection process as illustrated in Figure 7.

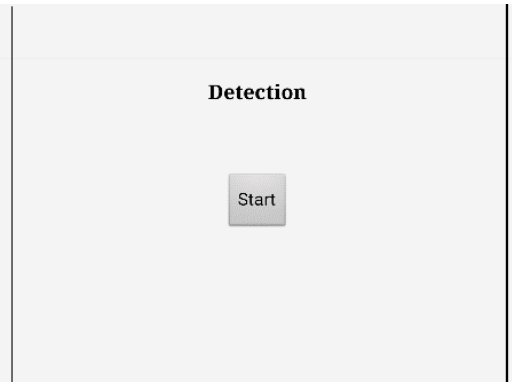

Figure 7: Start Detection Process

Page view is changed instantly to the listening view as shown in figure 8. Audio capture starts immediately after the listening view appears.

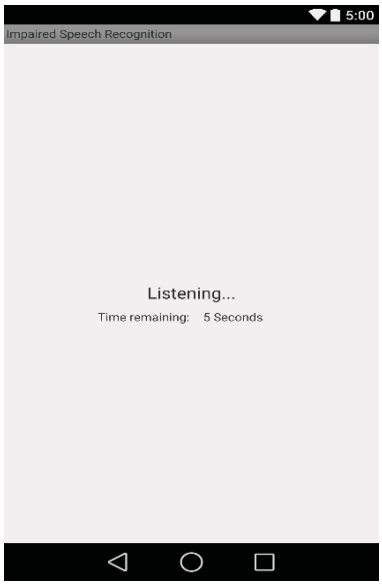

Figure 8: Audio capture in the detection process.

3.2.2 Recording and Fingerprint extraction.

Audio recording and Fingerprint extraction in the detection process are explained in section 3.1.3 and 3.1.4, respectively. Training and detection processes must have the same recording and fingerprint extraction algorithms to ensure consistent audio input quality. Having finished extracting a fingerprint of recorded audio in Figure 5 step 5, the final array list of fingerprints of every chunk of audio is compared with previous recordings to identify the input impaired speech.

3.2.3 Detection algorithms

Input speech is compared against previous recordings from all phrases once recording is finished. Please consider the following pseudo-code which demonstrates the detection algorithms.

```
1       For each JSON file representing a collection of fingerprints for a phrase
2           For each recording JSON object
3               Get array list of fingerprints
4               Loop through array list of fingerprints
5                   Compare fingerprint with queried impaired speech fingerprint
for similarities
6                   Get similarity result
7               End loop
8               Get similarities average of single recording.
9           End loop
10          Get similarities average of all recordings
11      End loop
12      Determine which phrase has the highest average.
13      Share results with user.
```

Firstly the application loops through all JSON files saved during the training process. Each JSON file is named after a phrase such as “How are you.json”. (Please refer to 3.1.5 for a detailed description of the file content and structure.) Line 2 loops through all JSON objects and each object represents a recording session. Line 3 is where the application iterates through the current JSON object and retrieves the stored array list of fingerprints. The application will loop through this array list since it contains a fingerprint for every chunk of audio from its recording session as illustrated in line 4. The next operation is to compare the queried impaired speech list of fingerprints with the impaired speech fingerprint list retrieved from the previously recorded session. Figure 9 visually illustrates an example of how queried impaired speech and a saved impaired speech compared for similarities, in order to obtain a final similarity score in lines 5, 6 and 8.

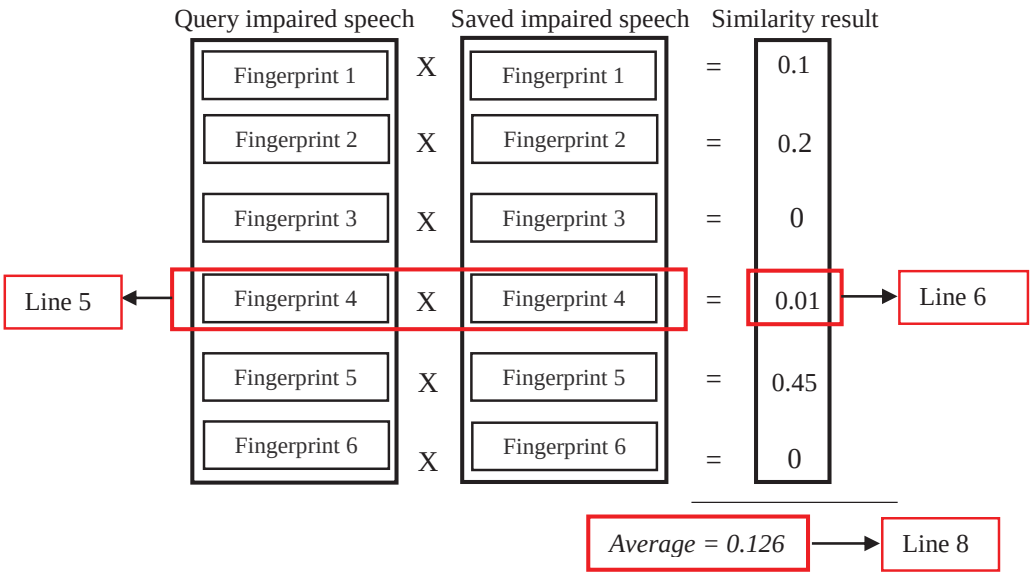

Figure 9: Fingerprint comparison

Comparison is linear which means audio content is compared periodically from the start of the recording to the end. Comparison of fingerprints returns a similarity from Zero to one, where Zero means no similar feature is found and one means on average there is at least one match in every frame from both chunks of audio from which the fingerprints were originally extracted. Similarity result values are accumulated and the similarity result average value is calculated

Line 10 sums all similarity result average values from all recording session in order to get the average similarity across all recordings for the currently examined phrase. In other words, it sums all values obtained from line 8 and calculates the final similarity score for the whole phrase. It gives a good indication of how queried impaired speech compares to different recording sessions for a particular phrase. Higher similarity

results across different recording sessions means this phrase is most likely to be a match to the queried impaired speech.

Finally, Line 12 determines which phrase has the highest score and it is then displayed to the user on a different view called the result view. Given the correct result, the user can use provided text-to-speech synthesizer to speak the final phase. Figure 10 shows an example of the result view.

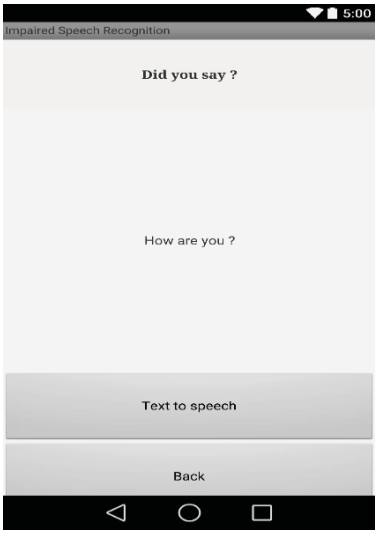

Figure 10: Result view

3.3 Summary

The technical aspects and algorithms of the impaired speech recognition service application were discussed in this section. The user interface, training process and detection process were explained in detail.

4 Application test

This section tests the application detection process. The application is tested in two different environments. The first test takes place in a quiet place and the second test is in a slightly noisy environment.

Algorithms and application detection ability were tested using a Samsung Galaxy S4 model number SHV-E330S running Android Lollipop 5.0.1 (API 21). Four commonly used phrases were used in this test. They are “How are you?”, “I am good, thank you”, “Can I have a cup of tea, please?” and “See you later”. The purpose of this test is to be able to detect the audio signal provided in the training process. The application was trained previously. Training process was done in a quiet room and also in a slightly noisy room. Both results for each test are provided separately.

The person testing was me. A dysarthria patient was not used to test the application because it not robust enough to be tested on Dysarthria of Speech patients yet. It needs further development as it is stated in the discussion section.

The selected phrases were trained 10 times, each. The application was queried 10 different times using phrases randomly selected from the ones used for recording in the training process. The following tables illustrate the similarity results acquired during the detection process in a quiet place and, in comparison, a slightly noisy place. Results presented in bold represent the highest similarity score out of the other phrases. Green

highlight determines if the phrase with the highest similarity score is the correct match and red signals a false positive result.

4.1 Quiet place

A quiet place is a place that has no noise or sound, especially no disturbing sound. It is free, or comparatively free, from noise. The following table shows the results obtained when the application is queried with randomly selected phrases in a quiet place.

Queried phrase	How are you?	I am good, thank you	Can I have a cup of tea, please?	See you later
How are you	**0.044052307**	0.03014845	0.02631579	0.005263158
I am good, thank you	0.016148327	**0.02890867**	0.01	0.014285715
Can I have a cup of tea, please?	0.022727273	0.017812178	**0.11991582**	0.0
See you later	0.012121213	0.020000001	0.013333334	**0.074761905**
How are you	**0.030303031**	0.021025643	0.026666667	0.0
See you later	0.033756685	0.0055555557	**0.03888889**	0.019365082
I am good, thank you	0.0056818184	**0.07284591**	0.01904762	0.0071428576
Can I have a cup of tea, please?	0.033459596	0.010526316	**0.15880303**	0.031681225
See you later	0.0	0.0	0.0071428576	**0.05**
I am good, thank you	0.004784689	**0.03113529**	0.015789473	0.018014705

Table 4.1: Quiet place results

Table 4.1 has shown the similarity scores acquired from 10 queried phrases and every other previously recorded and fingerprinted phrase. The highest similarity score is highlighted in green and red. Green highlight means the highest similarity score matches the queried phrase. Red highlight means the highest similarity score does not match the queried phrase and is considered false positive result. It can be clearly seen that 9 out of 10 phrases were correctly identified in a quiet place.

4.2 Slightly Noisy place

A slightly noisy place is a place that has small in degree; small noise. For example, areas of the library where talking is allowed. The following table shows the results obtained when the application is queried with randomly selected phrases in a slightly noisy environment.

Queried phrase	How are you?	I am good, thank you	Can I have a cup of tea, please?	See you later
How are you	**0.049570788**	0.015789473	0.015789473	0.019691879
I am good, thank you	**0.038807187**	0.012955466	0.024712121	0.023277864
Can I have a cup of tea, please?	0.017857142	0.013333334	**0.053552628**	0.033560924
See you later	0.0	**0.014358975**	0.014285715	0.0
How are you	0.005050505	0.012955466	0.019111112	**0.034453783**
See you later	0.017045455	0.0125	0.03125	**0.05220238**
I am good, thank you	**0.02832365**	0.005263158	0.02	0.0071428576
Can I have a cup of tea, please?	0.027777778	0.013574662	**0.06349417**	0.019275209
See you later	0.017045455	0.0	0.03125	**0.033035714**
I am good, thank you	0.05594406	0.030769233	**0.1**	0.007692308

Table 4.2: Noisy place results

Results have illustrated that it is possible to record, fingerprint and then detect audio signals, regardless of its content, captured from a mobile phone microphone with a small margin of error in quiet places. 9 out of 10 phrases were correctly identified in a quiet place. In contrast, the detection process showed poor results in slightly noisy

places. 5 out 10 phrases were detected in a mildly noisy place. Given results show that detection robustness is directly affected by noise presence. The application is mostly likely to report false positive results in a noisy place.

4.3 Summary

Results have illustrated that it is possible to record, fingerprint and then detect audio signals captured from a mobile phone microphone with a small margin of error in quiet places. The application did not do well in a slightly noisy place. There various reasons to why this happened which will be discussed in the discussion section.

5 Discussion

In this section, the application test results as well as important aspects of the application framework are discussed. Machine learning, recording quality, false positive result threshold, scalability, fingerprint extraction, audio filters, application test results and human computer interaction are the main aspects in this discussion.

The system in its current state is not ready for deployment despite the fact that it can detect audio signals. Realistically, most conversations take place in different kinds of places. Some of these places are noisy and hence detection will not work. Instead, it will report false positive results. There are various aspects of the fingerprint system that can be improved to increase its robustness.

5.1 Machine learning

Detection algorithms are designed to work better with large sets of data. That is, training the application by acquiring more fingerprints for each phrase will increase detection accuracy. For example, training the application for the phrase “How are you?” hundreds of times in places with different levels of noise will increase its chances of being detected.

This is because users will not be able to replicate the same audio signal pattern for a given phrase every time. Therefore, having more recordings of a phrase will teach the application different patterns and ways the phrase is pronounced by the user. Secondly,

the surrounding environment may affect the recording even if the audio signal pattern is replicated perfectly. Again, more recording will teach the application how phrases are pronounced in different environments.

Tests samples implemented in the result section are considered too small for the application to pick up any patterns through its machine learning algorithms. The application must be tested with at least 500 recordings, in different environments, for each phrase. Thus, machine learning capabilities were not fully explored.

5.2 Recording quality

Recording quality can drastically affect the entire framework. Recording unnecessary audio data and wrong recording are two important causes of low recording quality. Current recording set up allows users to press a “Record” button and the recording will start right after the button is pressed. The application will go on recording for 5 seconds. The issue is that the user may not need all 5 seconds. In this case, some recorded audio data are unnecessary and they are misleading the application in the detection process. Chunks of unnecessary audio will drive the average down or in some cases may result in false positive results.

Also, wrong recording of a phrase can drive its average down, for example, when a user tries to train the application to detect the phrase “How are you” but instead forgets to speak the words, resulting in a silent recording added to the phrase list.

A proposed solution for both issues is to provide a “press and hold” button to record audio rather than a single press button. This way the application will only capture required audio data.

5.3 False positive threshold

Currently detection algorithms will show the phrase with the highest similarity score, regardless of whether it is the correct phrase or not. One of the properties of any fingerprinting system is reliability. A reliable system will not report false positive results. To address this issue, results below a threshold must be excluded. Since similarity scores are a number between Zero and one, the application can report a false negative “No-matching-phrase” result, if the highest score is below 0.4 threshold.

5.4 Scalability

Constantly training the application could only result in more data and as a result storage will get bigger in size. Fortunately, as fingerprints are only textual presentation of the initial audio data, storage size is very small. Files that contain 10 recordings have an average size of *37.75 kilobytes (KBs)*. A file with 1000 recordings has a size of approximately *3.775 Megabytes (MBs)*. Therefore, 200 files with 1000 recordings have a storage size of approximately *755MBs*. It is considered small for the amount of information stored. While storage size is not a threat at this stage, more data cause performance issues.

The application has not shown any performance issues during the test conducted in the result section. One possible reason is that the number of audio files is small. However, performance will be affected once the application deals with thousands of files. Training and detection processes need more efficient search algorithms which can handle a vast amount of data. In future development of this product, training and detection algorithms must be improved. One way is to implement efficient search algorithms that are specific to this application. Another approach is to implement robust algorithms proposed in literature such as the one proposed in [16].

5.5 Fingerprint extraction and audio analysis

Currently fingerprints are extracted using open source library MusicG. Fingerprint similarity comparison is also implemented by MusicG. The application doesn't work well in slightly noisy environments as illustrated in 4.2 in the test section. It could be as a result of MusicG's algorithms or any other system component such as audio recording quality. Essential components of any fingerprinting application work together to achieve the end goal. However, with excellent robust system components except for one crucial component, that one will affect the entire system.

MusicG is an open source library and its algorithms need to be improved along with other system components such as pre-processing and post-processing of captured audio data (Audio filters). There are two approaches to achieving a relatively high similarity and matching accuracy. The first approach is to improve and build upon the current

algorithms. This means improving fingerprint extraction and similarity matching algorithms as well as audio filter algorithms of the current system.

The second approach is to try and implement relevant algorithms proposed in other papers, for example, fingerprint extraction and matching algorithms proposed in [12] or [1] and audio filters for fingerprinting systems proposed in [17]. Using any approach is possible and results will achieve a relatively high similarity and matching accuracy. However, both approaches are time consuming to implement and the later approach has some issues as discussed in section 2.4.2.

5.6 Application Test

The application test is considered short and may not be sufficient to provide an accurate detection rate. However, it is a proof of concept and research and development in this field should continue. An ideal test would be conducted by five to ten Dysarthria of Speech patients. Users would have the ability to enter an unlimited number of sentences and each sentence is one to eight words long. The test can be carried out in a similar fashion to the test suggested by [24] or by [37]. Both test the application usability to ensure it is user friendly and works as it should. Also, having thousands of sentences may affect the application's performance and some problems may rise when the application is working at its peak. To address this, [23] suggested an automatic graphical user interface crawling-based practice that can be set up to test the application for possible bugs.

5.7 Human computer interaction

Audio signal processing has been the focus of this thesis. However, human computer interaction is a crucial component of any digital application. The user interface must suit Dysarthria of speech patients who will be using it on a daily basis. In some cases where speech is totally impaired, the application will be the voice of the individual. [7] has suggested important issues to consider when building a graphical user interface for Dysarthria of speech patients. Also, test-driven user interface development is important as developers could get some feedback directly from the application users. Human computer interaction issues were reviewed in section 2.9 and in future development of this product we should follow a mobile health software development framework. It can help manage the development process between coders, doctors and patients. Also when the application is finished we should conduct a quality of experience test the application's efficiency and user satisfaction.

5.8 Summary

To sum up, significant aspects of this application were discussed in this section. The importance of machine learning, audio recording quality, false positive result and its effects on the detection process, scalability, fingerprint extraction, audio filters, application test results and human computer interaction were the main aspects in this discussion.

6 Conclusion and future work

We have developed a novel mobile health application that can recognize audio signals in a very quiet environment using audio fingerprinting technology. Audio fingerprinting, its alternatives and properties, fingerprinting algorithms and EEG signal systems were reviewed in this thesis. The technical aspects and algorithms of the impaired speech recognition service application were also discussed. The main technical aspects were the application's user interface, the training process and the detection process.

6.1 Discussion

Results have illustrated that it is possible to record, fingerprint and then detect audio signals captured from a mobile phone microphone with a small margin of error in quiet places. It made the process of turning impaired speech into understandable speech much faster which is the main objective of this thesis. Also, this proves that fingerprinting algorithms can be used to compare two audio files for similarity. Unfortunately, the application did not do well in a slightly noisy place. As a result, it is not robust enough and ready to be deployed or tested on Dysarthria of Speech patients. It defeats the third objective of this thesis which is to examine whether Musicg can be used to turn impaired speech into understandable speech efficiently. The fingerprinting algorithms must be improved. The detection and similarities matching algorithms must be improved, too. This can be one by improving the current fingerprint extraction, detection and similarity matching algorithms (Musicg) or by adapting a

ready-made audio fingerprinting framework. Another option is to create a new fingerprinting algorithms from the scratch.

6.2 conclusion

To sum up, recognizing impaired speech or, audio signals using a mobile phone microphone is possible as illustrated in this thesis. The application was able to recognize audio signals but only in quiet environments. Recording quality, audio filters and fingerprint extraction and similarity algorithms are the main aspects that must be improved to increase its robustness.

References

[1] Haitsma, J., & Kalker, T. (2003). A highly robust audio fingerprinting system with an efficient search strategy. *Journal of New Music Research*, *32*(2), 211-221.

[2] Gomes, L. D. C., Cano, P., Gomez, E., Bonnet, M., & Batlle, E. (2003). Audio watermarking and fingerprinting: for which applications?. *Journal of New Music Research*, *32*(1), 65-81.

[3] Duffy, J. R. (2013). *Motor speech disorders: Substrates, differential diagnosis, and management*. Elsevier Health Sciences.

[4] Griffiths, S., Barnes, R., Britten, N., & Wilkinson, R. (2011). Investigating interactional competencies in Parkinson's disease: The potential benefits of a conversation analytic approach. *International Journal of Language & Communication Disorders*, 46(5), 497-509. doi:10.1111/j.1460-6984.2011.00012.x

[5]Wang, A. (2006). The Shazam music recognition service. *Communications of the ACM*, *49*(8), 44-48.

[6] Creer, S., Green, P., Cunningham, S., & Yamagishi, J. (2010). Building Personalized Synthetic Voices for Individuals with Dysarthria using the HTS Toolkit. In J. Mullennix, & S. Stern (Eds.) *Computer Synthesized Speech Technologies: Tools for Aiding Impairment* (pp. 92-115). Hershey, PA: . doi:10.4018/978-1-61520-725-1.ch006

[7] AudioRecord. (2015, June 26). Retrieved June 30, 2015, from http://developer.android.com/reference/android/media/AudioRecord.html

[8] Cano, P., Batlle, E., Gómez, E., de CT Gomes, L., & Bonnet, M. (2005). Audio fingerprinting: concepts and applications. In *Computational intelligence for modelling and prediction* (pp. 233-245). Springer Berlin Heidelberg.

[9] Cano, P., Batle, E., Kalker, T., & Haitsma, J. (2002, December). A review of algorithms for audio fingerprinting. In *Multimedia Signal Processing, 2002 IEEE Workshop on* (pp. 169-173). IEEE.

[10] Wang, A. (2003, October). An Industrial Strength Audio Search Algorithm. In *ISMIR* (pp. 7-13).

[11] musicg - Lightweight Java API for audio analysing, Android compatible - Google Project Hosting. (n.d.). Retrieved from https://code.google.com/p/musicg/

[12] B.kekre, H., Bhandari, N., Nair, N., Padmanabhan, P., & Bhandari, S. (2013). A Review of Audio Fingerprinting and Comparison of Algorithms. *International Journal of Computer Applications IJCA*, 24-30.

[13] Introducing JSON. (n.d.). Retrieved June 18, 2015, from http://json.org/

[14] Android, the world's most popular mobile platform. (n.d.). Retrieved June 30, 2015, from https://developer.android.com/about/android.html

[15] Download. (n.d.). Retrieved June 18, 2015, from https://developer.android.com/sdk/index.html

[16] Yu, C., Wang, R., Xiao, J., & Sun, J. (2014). High performance indexing for massive audio fingerprint data. *Consumer Electronics, IEEE Transactions on*,*60*(4), 690-695.

[17] Wang, Y., Yu, X., Wang, W., Wan, W., & Swaminathan, R. (2012). Noise removal of audio clips for fingerprint matching. *2012 International Conference On Audio, Language & Image Processing*, 942. doi:10.1109/ICALIP.2012.6376749

[18] Malekesmaeili, M., & Ward, R. K. (2014). A local fingerprinting approach for audio copy detection. *Signal Processing*, 98308-321. doi:10.1016/j.sigpro.2013.11.023

[19] Soman, S., & Murthy, B. (2015). Using Brain Computer Interface for Synthesized Speech Communication for the Physically Disabled.*Procedia Computer Science*, *46*(Proceedings of the International Conference on Information and Communication Technologies, ICICT 2014, 3-5 December 2014 at Bolgatty Palace & Island Resort, Kochi, India), 292-298. doi:10.1016/j.procs.2015.02.023

[20] Porbadnigk, A., Wester, M., & Jan-p Calliess, T. S. (2009). EEG-based speech recognition impact of temporal effects.

[21] Our Mobile Planet: New Zealand. (2012). *Understanding the Mobile Consumer*. Retrieved from http://www.iab.org.nz/images/uploads/Google_Our_mobile_planet_NZ%20May%202012.pdf

[22] A Report on a Survey of New Zealanders' Use of Smartphones and other Mobile Communication Devices 2015. (2015). *Research New Zealand.* Retrieved from http://researchnz.com/pdf/Special%20Reports/Research%20New%20Zealand%20Special%20Report%20-%20Use%20of%20Smartphones.pdf

[23] Amalfitano, D., Fasolino, A., & Tramontana, P. (2011). A GUI Crawling-Based Technique for Android Mobile Application Testing. *2011 IEEE Fourth International Conference On Software Testing, Verification & Validation Workshops (ICSTW)*, 252. doi:10.1109/ICSTW.2011.77

[24] Black, W. S. (2015). *Current Practices for Product Usability Testing in Web and Mobile Applications* (Honors Theses, University of New Hampshire,

Durham, Concord, and Manchester, New Hampshire, U.S.). Retrieved from http://scholars.unh.edu/cgi/viewcontent.cgi?article=1228&context=honors

[25] Ben-Zeev, D., Kaiser, S. M., Brenner, C. J., Begale, M., Duffecy, J., & Mohr, D. C. (2013). Development and usability testing of FOCUS: A smartphone system for self-management of schizophrenia. *Psychiatric Rehabilitation Journal*, 36(4), 289-296. doi:10.1037/prj0000019

[26] Rajput, Z., Mbugua, S., Amadi, D., Chepngeno, V., Saleem, J., Anokwa, Y., & ... Were, M. (2012). Evaluation of an Android-based mHealth system for population surveillance in developing countries. *Journal Of The American Medical Informatics Association*, 19(4), 655-659. doi:10.1136/amiajnl-2011-000476

[27] Zapata, B. C., Fernandez-Aleman, J. L., Idri, A., & Toval, A. (2015). Empirical Studies on Usability of mHealth Apps: A Systematic Literature Review. *Journal Of Medical Systems*, (2), 1.

[28] Mata, P., Chamney, A., Viner, G., Archibald, D., & Peyton, L. (2015). A development framework for mobile healthcare monitoring apps. *Personal And Ubiquitous Computing*, (3-4), 623.

[29] Pongpisit, W., Sineerat, R., & Therdpong, D. (2015). mHealth: A Design of an Exercise Recommendation System for the Android Operating System. *Walailak Journal Of Science & Technology*, 12(1), 63-82.

[30] Martínez-Pérez, B., Torre-Díez, I., Candelas-Plasencia, S., & López-Coronado, M. (2013). Development and Evaluation of Tools for Measuring the Quality of Experience (QoE) in mHealth Applications. *Journal Of Medical Systems*, 37(5), 1-8. doi:10.1007/s10916-013-9976-x

[31] Dunsmuir, D. T., Payne, B. A., Cloete, G., Petersen, C. L., Görges, M., Lim, J., & ... Ansermino, J. M. (2014). Development of mHealth applications for pre-eclampsia triage. *IEEE Journal Of Biomedical And Health Informatics*, 18(6), 1857-1864. doi:10.1109/JBHI.2014.2301156

[32] McCurdie, T., Taneva, S., Casselman, M., Yeung, M., McDaniel, C., Ho, W., & Cafazzo, J. (2012). mHealth consumer apps: the case for user-centered design. *Biomedical Instrumentation & Technology / Association For The Advancement Of Medical Instrumentation, Suppl*49-56. doi:10.2345/0899-8205-46.s2.49

[33] Daihua X., Y., Parmanto, B., Dicianno, B. E., & Pramana, G. (2015). Accessibility of mHealth Self-Care Apps for Individuals with Spina Bifida. *Perspectives In Health Information Management*, 1-19.

[34] Alnanih, R., Ormandjieva, O., & Radhakrishnan, T. (2013). Context-based and Rule-based Adaptation of Mobile User Interfaces in mHealth. *Procedia Computer Science*, 21(The 4th International Conference on Emerging Ubiquitous Systems and Pervasive Networks (EUSPN-2013) and the 3rd International Conference on Current

and Future Trends of Information and Communication Technologies in Healthcare (ICTH), 390-397. doi:10.1016/j.procs.2013.09.051

[35] Hamida, S. T., Hamida, E. B., & Ahmed, B. (2015). A new mHealth communication framework for use in wearable WBANs and mobile technologies. *Sensors (Basel, Switzerland)*, 15(2), 3379-3408. doi:10.3390/s150203379

[36] Erbes, C. R., Stinson, R., Kuhn, E., Polusny, M., Urban, J., Hoffman, J., & ... Thorp, S. R. (2014). Access, Utilization, and Interest in mHealth Applications Among Veterans Receiving Outpatient Care for PTSD.*Military Medicine*, 179(11), 1218-1222. doi:10.7205/MILMED-D-14-00014

[37] Brown, W. 3., Yen, P., Rojas, M., & Schnall, R. (2013). Assessment of the Health IT Usability Evaluation Model (Health-ITUEM) for evaluating mobile health (mHealth) technology. *Journal Of Biomedical Informatics*, 46(6), 1080-1087. doi:10.1016/j.jbi.2013.08.001

[38] Sieverdes, J. C., Raynor, P. A., Armstrong, T., Jenkins, C. H., Sox, L. R., & Treiber, F. A. (2015). Attitudes and perceptions of patients on the kidney transplant waiting list toward mobile health–delivered physical activity programs. *Progress In Transplantation*,25(1), 26-34. doi:10.7182/pit2015884

[39] Alwin van, D., Cécile RL, B., Hynek, H., Jos WR, T., Tjabe, S., & Allard J van der, B. (2014). Evaluation of an Mhealth Intervention Aiming to Improve Health-Related Behavior and Sleep and Reduce Fatigue among Airline Pilots. *Scandinavian Journal Of Work, Environment & Health*, (6), 557.

[40] Ni Mhurchu, C., Whittaker, R., McRobbie, H., Ball, K., Crawford, D., Michie, J., & ... Myers, K. (2014). Feasibility, acceptability and potential effectiveness of a mobile health (mHealth) weight management programme for New Zealand adults. *BMC Obesity*,

[41] Kazemi, D. M., Cochran, A. R., Kelly, J. F., Cornelius, J. B., & Belk, C. (2014). Integrating mHealth Mobile Applications to Reduce High Risk Drinking among Underage Students. *Health Education Journal*, 73(3), 262-273.

[42] Gleason, A. W. (2015). mHealth — Opportunities for Transforming Global Health Care and Barriers to Adoption.*Journal Of Electronic Resources In Medical Libraries*, 12(2), 114-125. doi:10.1080/15424065.2015.1035565

[43] Price, M., Yuen, E. K., Goetter, E. M., Herbert, J. D., Forman, E. M., Acierno, R., & Ruggiero, K. J. (2014). mHealth: A Mechanism to Deliver More Accessible, More Effective Mental Health Care. *Clinical Psychology & Psychotherapy*, (5), 427.

[44]Chib, A., van Velthoven, M. H., & Car, J. (2015). mHealth adoption in low-resource environments: A review of the use of mobile healthcare in developing

countries. *Journal Of Health Communication*, 20(1), 4-34. doi:10.1080/10810730.2013.864735

[45] Brian, R. M., & Ben-Zeev, D. (2014). Mobile health (mHealth) for mental health in Asia: Objectives, strategies, and limitations. *Asian Journal Of Psychiatry*, 1096-100. doi:10.1016/j.ajp.2014.04.006

[46] Medhanyie, A. A., Moser, A., Spigt, M., Yebyo, H., Little, A., Dinant, G., & Blanco, R. (2015). Mobile health data collection at primary health care in Ethiopia: a feasible challenge. *Journal Of Clinical Epidemiology*, (1), 80. doi:10.1016/j.jclinepi.2014.09.006

[47] Dicianno, B. E., Parmanto, B., Fairman, A. D., Crytzer, T. M., Yu, D. X., Pramana, G., & ... Petrazzi, A. A. (2015). Perspectives on the Evolution of Mobile (mHealth) Technologies and Application to Rehabilitation. *Physical Therapy*, 95(3), 397-405. doi:10.2522/ptj.20130534

[48] Arora, S., Yttri, J., & Nilsen, W. (2014). Privacy and Security in Mobile Health (mHealth) Research. *Alcohol Research: Current Reviews*, 36(1), 143-150.

[49] Dehling, T., Gao, F., Schneider, S., & Sunyaev, A. (2015). Exploring the Far Side of Mobile Health: Information Security and Privacy of Mobile Health Apps on iOS and Android. *JMIR Mhealth And Uhealth*, 3(1), e8. doi:10.2196/mhealth.3672

[50] Gkatzidou, V., Hone, K., Sutcliffe, L., Gibbs, J., Tariq Sadiq, S., Szczepura, A., & ... Estcourt, C. (2015). User interface design for mobile-based sexual health interventions for young people: Design recommendations from a qualitative study on an online Chlamydia clinical care pathway. *BMC Medical Informatics & Decision Making*, 15(1), 1-13. doi:10.1186/s12911-015-0197-8

[51] D. Kuda, personal communication, February 29, 2016