UNIVERSITY OF THE WESTERN CAPE

Handwritten alphabet character recognition using audio signatures and machine learning

by

Bruce Beck

A thesis submitted in fulfilment for the
degree of Master of Science

in the
Faculty of Natural Sciences
Department of Computer Science

Supervisor: Dr Mehrdad Ghaziasgar

January 2023

Declaration of Authorship

I, Bruce Beck, declare that this thesis "*Handwritten alphabet character recognition using audio signatures and machine learning*" is my own work, that it has not been submitted before for any degree or assessment at any other university, and that all the sources I have used or quoted have been indicated and acknowledged by means of complete references.

Signed: _____

Date: _____

To my dad

Abstract

This research investigates the creation of an audio-based character recognition system that is able to segment, process and recognise uppercase English letters continuously drawn by the user on a given writing surface such as a table-top using a generic writing implement. The aim is to make use of the microphones on a single smartphone to capture the acoustic signal generated by the user as they draw letters on the writing surface, followed by the application of audio segmentation to subdivide the audio signal into segments corresponding to each letter, and finally the application of a combination of the Mel-Frequency Cepstral Coefficients feature descriptor and Support Vector Machines to recognise the segmented letters.

The aim is to also host the system in a cloud-based environment that can be easily accessible by any device via a web browser. Such a system would be light-weight, portable and easily accessible. Most importantly, it would provide smartphone users with an additional means of interacting with the device, namely, the ability to write/draw on the surface next to the smartphone and have the characters captured and used on the smartphone in various ways.

The main challenges to the creation of such a system are: the limitation on the number of microphones available on the given smartphone; the transmission of the acoustic signals through the air, which is less efficient than transmission through solids; and the difficulty to isolate and insulate ambient noise. Therefore, this research aims to determine whether the proposed techniques are sufficient to achieve high accuracy character recognition when using the two microphones on a single smartphone as an audio capture source.

Acknowledgements

I would like to thank my supervisor Dr Mehrdad Ghaziasgar for continued support and involvement during my study and not giving up on me during this long and hard journey.

I would also like to thank Retail Capital for the continuous support throughout the years and encouragement to complete my studies.

I would also like to thank my friends and my partner for encouraging me and supporting me.

Finally, and most importantly, I'd like to thank my parents for their support because without them none of this would be possible.

Contents

Declaration of Authorship i

Abstract iii

Acknowledgements iv

List of Figures viii

List of Tables x

Abbreviations xi

1 Introduction 1
- 1.1 Background and Motivation . . . 1
- 1.2 Research Question . . . 6
- 1.3 Research Progression and Research Objectives . . . 6
- 1.4 Premises . . . 7
- 1.5 Thesis Outline . . . 7

2 Literature Review 9
- 2.1 Overall Objectives . . . 9
- 2.2 Audio Capture . . . 11
  - 2.2.1 Recording Devices . . . 12
  - 2.2.2 Impact of Writing Implements and Surfaces on Audio Capture . . . 15
  - 2.2.3 Writing location . . . 16
  - 2.2.4 Discussion on Audio Capture . . . 16
- 2.3 Audio Segmentation . . . 17
  - 2.3.1 Discussion on Audio Segmentation . . . 19
- 2.4 Feature Extraction Techniques . . . 19
  - 2.4.1 Mel-Frequency Cepstral Coefficients Feature Extraction . . . 19
  - 2.4.2 Spectrogram Feature Extraction . . . 21
  - 2.4.3 Discussion on Feature Extraction Techniques . . . 22
- 2.5 Classification and Recognition . . . 22
  - 2.5.1 Template Matching . . . 22
  - 2.5.2 Support Vector Machines . . . 23

2.5.3 Neural Networks . . . 27
2.5.4 Discussion on Classification and Recognition . . . 29
2.6 Summary . . . 29

3 Tools and Techniques for Handwriting Recognition 32
3.1 Audio Signal Capture . . . 32
3.2 Audio Segmentation . . . 34
3.3 Feature Extraction . . . 35
3.3.1 Pre-emphasis . . . 37
3.3.2 Framing and Windowing . . . 37
3.3.3 Fast Fourier Transform . . . 39
3.3.4 Mel Scale Filtering . . . 41
3.3.5 Logarithmic function . . . 42
3.3.6 Discrete Cosine Transform . . . 42
3.3.7 Delta Features . . . 43
3.4 Classification Using Support Vector Machines . . . 43
3.4.1 Optimization . . . 45
3.5 Summary . . . 47

4 Design and Implementation 48
4.1 Audio Capture . . . 48
4.1.1 Recording Device . . . 48
4.1.2 Recording Application . . . 49
4.1.3 Environment, Surface and Recording Implement . . . 51
4.2 Audio Processing . . . 52
4.2.1 Audio Segmentation . . . 52
4.2.2 Feature Extraction . . . 54
4.3 Classification . . . 54
4.3.1 Classes and Data . . . 55
4.3.2 Training and Testing Datasets . . . 56
4.3.3 Optimization . . . 57
4.4 Model Hosting . . . 59
4.5 Summary . . . 60

5 Experimental Results and Analysis 61
5.1 Audio Segmentation Results and Analysis . . . 62
5.2 Letter Recognition Results and Analysis . . . 64
5.2.1 Semi-Seen Testing Results and Analysis . . . 65
5.2.2 Unseen Testing Results and Analysis . . . 68
5.3 Comparison of the Proposed Approach to Related Studies . . . 72
5.4 Summary . . . 73

6 Conclusion 77
6.1 Future Work . . . 78
6.2 Concluding Comments . . . 78

A Additional Results **79**

Bibliography **82**

List of Figures

2.1 Impact of different surfaces and writing implements depicted by Haishi *et al.* [9] . . . 16
2.2 Results of various writing locations conducted by Chen *et al.* [5]. . . . 17
2.3 MFCC feature extraction as described by Luo *et al.* [18] . . . 20
2.4 Results of different combinations of feature extraction and classification by Chen *et al.* [5]. . . . 21
2.5 Common misclassified characters found by Li and Hammond [15]: left column represents the ground-truth character and the right column represents the corresponding incorrectly predicted character. . . . 23
2.6 SVM Classification as illustrated by Meyer [20]. . . . 24
2.7 The idea of SVMs as explained by Hearst *et al.* [11] which shows the creation of a non-linear hyperplane in input space by mapping training data onto a higher-dimensional feature space via a kernel function. . . . 24
2.8 Word clusters by Yu *et al.* [31] . . . 26
2.9 Recognition accuracy by Yu *et al.* [31] in 3 different scenarios. . . . 27
2.10 Neural Network configurations by Schrapel *et al.* [23] . . . 28

3.1 Overview of the proposed handwriting recognition system. . . . 33
3.2 Visualization of the raw audio signal of 10 consecutive handwritten 'B's as captured by a single microphone. The regions of the audio signal corresponding to the first four 'B's have been indicated by the read region lines. . . . 34
3.3 An example of a single B extracted from the audio signal in Figure 3.2 . . . 36
3.4 MFCC feature extraction process as described by Luo *et al.* [18] . . . 36
3.5 Example of the audio signal in Figure 3.3 after pre-emphasis. . . . 37
3.6 Framing on top of the pre-emphasized signal in Figure 3.5. . . . 38
3.7 Example of a Hamming window that is applied to an audio signal: (Top) The original audio signal; (Bottom) Result of applying the Hamming window to the signal. . . . 39
3.8 a) Shows the individual frame after a Hamming window has been applied b) the resulting frequency spectrum after FFT c) an illustration of the spectral power of the frame. . . . 40
3.9 Illustration of Mel filterbanks where $N_q = 10$. . . . 42
3.10 Example of linear SVM classification . . . 44
3.11 Example of grid search to find optimal training parameters as explained by Syarif *et al.* [24]. . . . 46
3.12 Performance comparison between standard LIBSVM and LIBSVM-GPU as illustrated by Athanasopoulos *et al.* [2] . . . 47

4.1 Various possible writing locations, adapted from Chen *et al.* [5] 49
4.2 Screenshot of the recording application created for the purposes of this research. 50
4.3 Data collection process used in this research. 50
4.4 Environment used for data collection. 51
4.5 Close-up of the writing surface used in data collection. 52
4.6 Writing implement used by the test subjects in data collection. 52
4.7 Example of 10 'B's writing by a user as shown previously in Figure 3.2 . . 53
4.8 Each individual 'B' extracted from the signal in Figure 4.7. 53
4.9 Resultant feature vector of MFCC feature extraction applied to a single 'A' character. 54
4.10 Deaf-blind stroke orderings required by users in data collection and testing. 55
4.11 Original contour plot from grid-search utilising the conventional C and γ optimization range. 59
4.12 Contour plot from grid-search utilising the adjusted range of C and γ. . . 59

5.1 Deaf-blind stroke orderings required by users in training and testing. . . . 62
5.2 Results of semi-seen testing per letter across all test subjects, sorted in descending order of accuracy. 65
5.3 Confusion matrix of semi-seen testing results across all subjects. 66
5.4 Results of semi-seen testing per subject across all classes. 67
5.5 Results of unseen testing per letter across all test subjects, sorted in descending order of accuracy. 69
5.6 Confusion matrix across all subjects in unseen testing. 70
5.7 Results of unseen testing per subject across all classes. 71

List of Tables

2.1 Success rate results for Moreira *et al.* [21] . . . 12
2.2 Success results for Li *et al.* [15] using different writing implements. . . . 15
2.3 Comparison of KNN and SVMs as conducted by Luo *et al.* [18]. . . . 25
2.4 Results of letter recognition by Haishi *et al.* [9]. . . . 25
2.5 Results of letter recognition by Yu *et al.* [31]. . . . 26
2.6 Results of the four network configurations by Schrapel *et al.* [23]. . . . 29

4.1 Training, semi-seen and unseen datasets. . . . 57
4.2 Cross-validation accuracy of each C and γ parameter pair evaluated in the grid-search optimisation procedure. . . . 58

5.1 Training, semi-seen and unseen datasets. . . . 62
5.2 Results from audio segmentation per unseen test subject. . . . 63
5.3 Letters in the unseen testing experiment which did not meet the acceptable recognition accuracy of 60% (left column), along with letters they were most confused with (right column). . . . 71
5.4 Comparison of proposed system to other implementations in the literature as described in Chapter 2 . . . 74

A.1 Number of correctly recognised samples (out of 3) for each subject for each letter class on the semi-seen dataset. . . . 79
A.2 Number of correctly recognised samples (out of 8) for each subject for each letter class on the unseen dataset. . . . 80
A.3 Percentage of correctly recognised samples (out of 8) for each subject for each letter class on the unseen dataset. . . . 81

Abbreviations

ANN	**A**rtificial **N**eural **N**etwork
AWS	**A**mazon **W**eb **S**ervices
CC	**C**epstral **C**oefficients
CFCC	**C**ochlear **F**ilter **C**epstral **C**oefficients
CNN	**C**onvolutional **N**eural **N**etwork
COTS	**C**onsumer **O**ff **T**he **S**helf
CPU	**C**entral **P**rocessing **U**nit
dBFS	**d**ecibels relative to **F**ull **S**cale
DCT	**D**iscrete **C**osine **T**ransform
DTW	**D**ynamic **T**ime **W**arping
FFT	**F**ast **F**ourier **T**ransform
GPU	**G**raphics **P**rocessing **U**nit
HCI	**H**uman **C**omputer **I**nteraction
HMM	**H**idden **M**arkov **M**odel
KNN	**K**-**N**earest **N**eighbours
LSTM	**L**ong **S**hort **T**erm **M**emory
MFCC	**M**el **F**requency **C**epstral **C**oefficients
PC	**P**ersonal **C**omputer
RAM	**R**andom **A**ccess **M**emory
RBF	**R**adial **B**asis **F**unction
STE	**S**hort **T**erm **E**nergy
stPSD	**s**hort **t**ime **P**ower **S**pectral **D**ensity
SVM	**S**upport **V**ector **M**achine
TDoA	**T**ime **D**ifference **o**f **A**rrival
VRAM	**V**ideo **R**andom **A**ccess **M**emory
ZCR	**Z**ero **C**rossing **R**ate

Chapter 1

Introduction

1.1 Background and Motivation

Research in the field of Human-Computer Interaction (HCI) investigates alternative and novel means of interacting with computers and computational devices in a less traditional way i.e. alternatives to using e.g. a mouse and keyboard.

HCI research encompasses research into various ways that humans may interact with computers. These can be broken down into two broad areas. The first area of HCI is visual-based HCI which involves the use of a camera or multiple cameras, various visual technologies, combined with computer vision methods to send data, perform an action or elicit some response on a computer or device that the user is interacting with. Two examples of this area of HCI are hand tracking and facial recognition [13].

Audio-based HCI is the second area of HCI research that is of interest. Audio-based HCI makes use of microphones to capture one or more audio signals, combined with audio processing techniques to be able to interpret the signal that it is receiving. Finally this can be used to elicit a response on a computer or device. The most common example of audio-based HCI is a personal assistant on modern smartphones. In this case, the signal received and processed is speech-based audio. Speech-based HCI is a sub-field of audio-based HCI. Apart from speech-based HCI, audio-based HCI can also take the form of non-speech-based HCI which involves research into capturing, processing and utilising non-speech audio signals to elicit some response from a computer. An example of this is the use of audio signals used to determine touch location / pattern / shape [27, 28].

This research builds on previous research done by Wu et al. [27, 28] which proposed

the use of transducer microphones connected to a personal computer (PC) to capture and process sound emitted by a hard surface such as a table, when drawn on by a hard instrument, such as a chopstick or pencil. In their case, the transducers were physically attached to the writing surface with glue. Their system then used the audio signals emitted by the hard surface and captured by the microphones to carry out processing to be able to successfully recognise all 26 uppercase letters of the Roman alphabet when physically drawn on the writing surface by the writing instrument.

Their system proved to be robust and beneficial in providing a novel means of non-speech-based HCI. One limitation of the system was that its use is limited to PCs which have the required input port(s) for the microphone transducers used. The system also assumes that each recording consists of the audio signal of a single drawn character, and doesn't include any strategy to automatically segment audio signals into constituent letters; this was left to future work.

Smartphones have become commonplace in the world today, so much so that it is rare to find someone that does not own, or is not using, one. In a 2014 study by the World Bank, it was seen that 97% of the world population was using a mobile phone [22]. The main ways of interacting with these devices include: touch such as typing on the touch screen or pressing buttons; vision in the form of unlocking the device with facial recognition; and audio—specifically speech-based audio—in the form of speaking to the virtual assistant or other speech-based applications. There are also other sensors on a smartphone such as: GPS; accelerometer; a light sensor to name a few. There is currently no standard way to control or interact with smartphone devices by means of non-speech audio input.

Research into ways of leveraging the microphones on smartphones for non-speech-based HCI can prove to be beneficial and leverage the microphone sensors on these devices more comprehensively.

Based on the above, this research focuses on extending Wu et al.'s work by developing a non-speech-based HCI system focused around smartphones. The proposed system aims to capture and process sound emitted by a hard surface—"the writing surface"—such as a table, when drawn on by a hard instrument, such as a chopstick or pencil. The aim is to use the microphones already built into a smartphone to capture the audio signals emitted by the hard surface and carry out processing to first segment letters in an audio signal potentially consisting of multiple letters, and then recognise all 26 uppercase letters of the Roman alphabet. As such, this research constitutes a significant advancement over Wu et al.'s work.

For the purposes of this research, the smartphone used will be a Samsung Galaxy Note

10 plus, which is the smartphone that is available. This smartphone, like many others, consists of a microphone on the bottom end of the device, and another on the top end of the device. These microphones are both capable of a recording quality as high as 256kbps at a rate of 48kHz. Given the ubiquity of smartphones in today's world means that the proposed system and its associated novel means of HCI will be more widely and readily available, without the need for any extra hardware to be carried around. The approach will have a low barrier to entry because of these factors to users which do not have the extra hardware required.

An ideal use-case of such a system is a user being provided the ability to write a piece of intended text on the table next to the smartphone e.g. their full name and surname, or a message, and have the hand-written text subsequently processed and appropriately captured on the smartphone as input. This would provide users with a new flexible form of interaction with smartphones.

One important aspect to this research problem is the question of the number of microphones that are required to ensure effective recognition by the system. Two microphones provide two positions from which to capture the sound source, and the underlying machine learning apparatus can then automatically triangulate and infer the two-dimensional location of the strokes in the sound source in time. This is important when trying to distinguish between sounds with the same number of strokes but different shapes e.g. "A" and "H".

To this end, Wu et al. [27, 28] investigated and compared the use of one-, two- and three-microphone configurations and found that two microphones were sufficient to obtain high accuracy recognition even on data from unseen test subjects, which was a phenomenal result. However, it is important to bear in mind differences in the hardware setup that Wu et al.'s used versus the setup proposed in this research, all of which increase the complexity of the research problem in this research:

1. **Limitation on the number of microphones:** Wu et al.'s setup made it possible to readily add or remove microphones, whereas the proposed setup is limited to using a maximum of two microphones embedded into the mobile phone.

2. **Medium for the propagation of sound:** Wu et al.'s project made use of microphones that were physically attached to the writing surface by means of adhesive. This was a major advantage since sound is known to propagate through solids more effectively and efficiently than through the air [26]. In contrast, the setup proposed in this research makes use of the microphones on a mobile phone that has simply been placed on—and not attached to—the writing surface. As such the sound from the writing surface will mostly be captured through the air.

So while the proposed setup is significantly more natural, flexible and convenient, it is expected to make the recognition problem proportionally more challenging.

3. **Noise insulation:** Wu et al. attached a relatively large amount of cotton wool to the top of their microphones in order to isolate sources of noise in the environment above and around the writing surface. This was possible due to the fact that their microphones were attached to the writing surface. This kind of noise insulation will not be possible with the proposed setup since, as mentioned, the sound from the writing surface will be captured by the microphones on the mobile phone primarily through the air. Noise is therefore unavoidably expected to affect the recognition accuracy of the system, with the benefit of a natural, flexible and convenient hardware setup.

Based on this discussion, it remains to be seen whether the use of the two microphones on the single smartphone will be sufficient to obtain high accuracy handwritten character recognition. High accuracy recognition in this context takes two forms. The first form involves accurately recognizing newly drawn characters by test subjects whose writing is familiar to the system due to the system having been trained on other samples from these test subjects i.e. semi-seen test samples. In practice, this would mean that a new user using the system would be required to carry out a once-off pre-training procedure in order to allow the system to learn these patterns and provide a high accuracy recognition. This is not unrealistic provided that the pre-training procedure is once-off and relatively short.

The second, and more advantageous, form of high accuracy recognition involves the system accurately recognizing characters drawn by test subjects that have been previously completely unseen to the system i.e. unseen test samples. This will mean that there will be no need for a pre-training procedure.

It remains to be seen whether the proposed hardware setup will be sufficient for high accuracy on unseen test samples, or whether it will require a pre-training procedure. Either way, the system would be considered to be useful since a once-off pre-training procedure would not be unreasonable or unrealistic if the final system is sufficiently accurate and provides a novel enhanced means of interacting with the mobile phone.

The audio processing component of the proposed system is broken up into 3 parts: a letter segmentation algorithm which splits words into letters for classification; a feature descriptor which is able to extract a standardised feature vector that is ready for classification; and classification using a classifier that has been trained to recognize letters based on the features extracted from the segmented letters.

For the purposes of this research, users will be allowed to continuously write letters, without manual intervention with the system. To allow a user to be able to write continuous words, a segmentation algorithm needs to be developed. The algorithm involves detecting the spaces between the letters that have been written. To this end, the assumption made in this research is that the pause between letters will be longer than the pause between strokes within a single letter.

For the purposes of this research, a dataset will have to be collected and used. The size of this dataset will be relatively small i.e. on the order of a few 1000 samples. While deep learning techniques were considered for use, it was determined that such techniques require medium-to-large datasets in order to properly generalize. Trained on small datasets, such as the one collected in this research, they are expected to overfit the datatset and fail to generalize. Therefore, when making use of small datasets, it is more appropriate to make use of "traditional" classification techniques which make use of a feature descriptor and a classifier.

According to Logan[16], the mel-frequency cepstral coefficients (MFCC) feature descriptor has been successful in many audio processing applications due to the way that it is able to represent the speech amplitude spectrum in a compact form. The MFCC is considered to be very robust when it comes to speech and non-speech classification and is well suited to this research.

Once the features have been extracted, a classifier is trained on these features and subsequently used for classification of testing/unseen samples on an on-going basis. There are numerous classifiers that have been used in related works including: Support Vector Machines (SVMs), Convolutional Neural Networks (CNN) and K-Nearest Neighbours (KNN)[9, 18, 27, 28, 30]. All of these papers achieved similar accuracy.

For the purposes of this research, and following on the phenomenal success of Wu et al.'s system, it was decided to use SVMs, which have been shown to combine with the MFCC very well. SVMs are highly flexible classifiers capable of non-linear classification. They come with the significant benefit of always converging to a minimum and having a very low parameter complexity as compared to many other classification techniques, including neural networks [25]. They have been utilised in a wide variety of fields using small-to-medium sized datasets including: facial recognition and classification, text classification.

1.2 Research Question

Based on the discussion in Section 1.1, the following research question can be formulated: "How accurately can handwritten words be segmented and recognized using audio captured by microphones on a smartphone using the proposed techniques i.e. MFCC coupled with SVMs?" This main research question can be broken down into research sub-questions as below:

1. How effectively can letters be segmented and extracted from an audio signal?
2. How accurately can the segmented letters be recognized?
3. Is it sufficient to capture audio input by the microphones on a single smartphone in order to achieve accurate segmentation and recognition of handwritten letters using the proposed techniques i.e. MFCC coupled with SVMs?

1.3 Research Progression and Research Objectives

The following research objectives will guide the progression of the research towards answering each of the research sub-questions towards answering the main research question:

1. Implement an audio capturing application to record audio from a smartphone as a user draws handwritten letters.
2. Implement an audio segmentation algorithm to analyze the captured audio to locate and segment the letters drawn.
3. Implement the MFCC feature descriptor and apply it to the audio signal of each segmented letter.
4. Collect a handwritten data set of uppercase characters "A" to "Z" as performed by various users.
5. Train a SVM classifier on a training sample of the handwritten data set to recognize the upper-case letters using MFCC features.
6. Test the classifier on a testing sample of the handwritten data set to determine the effectiveness of the letter recognition approach.

1.4 Premises

The following assumptions are made beforehand to be able to create a realistic scope for the research and provide key pointers and ideas for future work.

1. Recognition will be done on upper case Roman alphabet letters.
2. A single fixed surface will be used for the collection of the data set.
3. Fixed stroke patterns will be used by all training and testing subjects.
4. A relatively consistent stroke speed will also be used by all training and testing subjects.
5. The same room will be used for gathering the datasets and testing the system.

1.5 Thesis Outline

The rest of the thesis is structured as the below.

Chapter 2: *Related Work.* This chapter provides an in depth look into the previous work that has been conducted in line with the main research question. It describes the techniques and classifiers implemented and their accompanying results are compared to provide context for the current research.

Chapter 3: *Tools and Techniques for Handwriting Recognition* This chapter will go into detail on each individual step in this research and the details behind each. The steps that will be explained include: audio signal capture; audio segmentation; feature extraction; and classification.

Chapter 4: *System Design and Implementation.* This chapter will provide a comprehensive and in-depth discussion into the implementation of each part of the proposed system i.e. audio capture, letter segmentation, feature extraction, letter recognition and dictionary lookup. In each part, a detailed discussion on the algorithms, processes and techniques used relevant to that part will be described. Therefore, this chapter also details the data set collected and methods used to train and optimize the SVM classifier. In so doing, this chapter describes how research objectives 1–5 are met.

Chapter 5: *Experimental Results and Analysis.* This chapter describes the experiments conducted, results obtained and subsequent analyses carried out to meet the remaining research objective 6 towards arriving at answers to all of the research sub-questions 1–3, and ultimately arriving at an answer to the main research question.

Chapter 6: *Conclusion.* This chapter concludes the thesis by summarizing the findings and providing directions for future work.

Chapter 2

Literature Review

The aim of this chapter is to provide context to this research and to go into more detail on the information discussed in the previous chapter. The research objectives 1 to 6 detailed in the previous chapter involved the implementation of a sequence of components that collectively form the system proposed in this research, capturing audio data, finding an effective and accurate way of segmenting the audio signal into relevant parts; finding the most effective way of extracting features from the segmented audio; and devising a method to accurately recognize the segmented audio.

Accordingly, after providing an overview of the studies selected for inclusion in this research in Section 2.1, the literature survey presented in this chapter is then sub-divided, and corresponds to, these objectives, whereby Sections 2.2 to 2.5 the following aspects of the related studies, respectively: audio capture; audio segmentation; feature extraction; and classification and recognition. Given that various research studies have generally mixed and matched different methods and techniques for each of these objectives, structuring this chapter in this way provides a framework to organize and analyze the studies systematically. The chapter concludes with a summary of the related work and recap of what has been discussed.

2.1 Overall Objectives

To aid in providing context of each piece of literature reviewed, the overall objective of each piece of literature will be covered below.

Wu *et al.* [27, 28] created a novel non-speech-based HCI system which aimed to capture and process sound emitted by a generic stylus to be able to recognise seven fundamental

shapes, digits 0–9 and the uppercase letters 'A' through 'Z'. They did not use a smartphone, but instead carried out recognition of audio captured by transducer microphones attached to the writing surface and insulated from noise by means of cotton wool. A similar setup was used by Moerira *et al.* [21] where they did not opt to use a smartphone for audio capture but rather a low-cost hardware system which focused more on sensory perception of objects and surfaces, which was trained to recognize 3 fundamental shape gestures: *circle*, *square* and *triangle*.

With reference to the above papers and the additional hardware, which was required, Schrapel *et al.* [23] developed a specialised sensory writing pen, Pentelligence, which was able to sense the motions; sound emissions on the pen tip while various shapes were drawn by the pen; and pressure of the pen when impacting the surface. The reasoning behind this was the finding that users often prefer to write on physical paper as opposed to tablets with styluses [4]. They therefore believed that there was a gap between analog and digital. They tested their system on digits 0–9.

Li and Hammond [15] identified, at the time, that a secondary device such as a stylus or writing tablet, either of which are costly and not readily available, was required to capture drawing gestures and interact with the computer in this way. They identified the possibility of using a smartphone, which was a more readily available alternative, to interact with a computer via sound, however they utilised an Apple Macbook Air microphone for collecting training data and tested their system with a standard iPhone and Android phones. They implemented their proposed smartphone-based system to recognize 26 hand-sketched characters ('A'–'Z').

The work of Li and Hammond was further extended by Chen *et al.*[5], the latter of which shared the sentiment of utilising technology which was readily available to perform tasks which otherwise required extra pieces of hardware. Chen *et al.* created a software called *Ipanel* which allowed them to control their smartphones with common gestures on a surface adjacent to their smartphone. The gestures they were able to recognise were *click*, *flip*, *scroll* and *zoom*. In addition to these gestures, the application also aimed to recognise the 10 handwritten digits and the 26 handwritten alphabet characters.

Luo *et al.* [18] identified a limitation with current smartphones which is that, when they are being used, part of the screen is blocked from view by the user's hand as it interacts with the screen. They therefore identified that creating an innovative means of interacting with the phone is required to overcome this limitation. As with other studies in this review, they proposed smartphone interaction through audio signals, which allows control of the smartphone without the need to touch the smartphone itself, and this overcomes the limitation of blocking the user's view of the screen. They created

a system built for the Android smartphone operating system which was able to recognize 7 common gestures, namely: *click*, *flip*, *left/right*, *scroll up/down* and *zoom in/out*.

Zhou *et al.* [30] believed that the shrinking size of smartphone screens would eventually become an issue and that utilising a smartphone via additional means separate to the body of the smartphone, such as voice and handwriting input, would be more convenient and useful. They further believed that handwriting input is more suitable for drawing and writing mathematical expressions. Therefore, they proposed to create a system to recognize 26 free-style letters of the alphabet in lowercase drawn on the right-hand side of a smartphone located 20cm away.

Finally, Yu *et al.*'s research [31] focused on creating a system to explore the possibility of eavesdropping on a user's handwriting using a smartphone placed on the writing surface. This problem was identified because of the research done by Asonov and Agrawal [1], Zhuang *et al.* [32], Berger *et al.* [3] and Halevi and Saxena [10], all of whom identified the possibility of the sound of typing on a keyboard being prone to eavesdropping via acoustic signals. Accordingly, Yu *et al.* focused on using the audio captured by a smartphone placed on the same desk on which a user (eavesdropping victim) was writing and using the audio signals captured to recognise the user's handwriting. Their implementation aimed to recognize all 26 uppercase characters.

The sections that follow provide greater detail on these studies in terms of their means of: audio capture; audio segmentation (if applicable); feature extraction; and classification and recognition.

2.2 Audio Capture

A wide range of audio capture mechanisms have been used, which are detailed in this section. There are several factors to discuss when it comes to audio capture, including: the actual device used to capture audio (covered in Section 2.2.1), the implement/device used to generate the sound (covered in Section 2.2.2), the surface on which the implement/device generates the sound (covered in Section 2.2.2), and the mechanism/-physical setup in which the capturing device and sound emitting implement are arranged (covered in Section 2.2.3). A discussion of the findings of the section are summarised in Section 2.2.4.

Signal	Success Rate (%)
Raw time signal	33.0
FFT signal	15.0
Envelope signal	65.0
Time scaled envelope signal	90.0

TABLE 2.1: Success rate results for Moreira *et al.* [21]

2.2.1 Recording Devices

This section explains the various recording devices that were used in previous works and the benefits and shortcomings of each. The devices will get progressively more advanced as the discussion progresses.

Wu *et al.* [27, 28] utilised piezoelectric transducer microphones connected to a computer to capture audio. The justification of the approach was the low cost of the implementation, as well as the small size and portability of the setup which made it relatively mobile. One of the research questions investigated by Wu *et al.* was to find the optimal number of microphones that are ideal in recognising handwritten characters with a high accuracy. To this end, they investigated the use of one, two and three microphones. They found that a larger number of microphones would be able to help in differentiating the trajectories of strokes in characters, but that even a single microphone was sufficient to provide a high accuracy recognition on even unseen data and users. Increasing the number of microphones from one to two to three saw an increase of ±2% per microphone on semi-seen data, and ±1% for unseen data.

Moreira *et al.* [21] mirrored Wu *et al.* in designing a low-cost system which was able to produce acceptable results. Instead of piezoelectric microphones, they utilised a microphone connected to a stethoscope. The stethoscope was used to produce a less noisy signal from the environment. The stethoscope had a dual lumen design while the microphone was a mini-Electret microphone with dimensions of 0.16 × 0.06 inches. The microphone array was then connected to a conditioning circuit by Adafruit MAX4466 which acts to filter and amplify the signal. As seen in Table 2.1, the results were highly dependent on the different audio processing algorithms that were used which included: the raw time signal; Fast Fourier Transformed (FFT) signal; envelope signal; and time-scaled envelope signal. They found that the limitation with their microphone setup was the impact that any small variations in microphone array placement made significant changes to the signal characteristics, which required extra placement care to be able to repeat their results.

Another possible recording device which is readily available is the single microphone on

a laptop. Li *et al.* [17] investigated using the microphone embedded on a laptop to turn a laptop screen into a writing pad. To achieve this, an acoustic signal of varying frequencies is emitted by the two speakers on the laptop. Then, by utilising the Doppler frequency shift of acoustic signals, the distortion in the original acoustic signal captured by the microphone is used to track movements on the screen. Li *et al.* [17] were able to utilise the two speakers on the laptop combined with the single microphone to achieve a recognition accuracy of over 90%. This indicates that there is a possibility of using a single microphone to triangulate the direction and location of a writing implement to recognise a handwritten character. It was stated that a limitation to this implementation is the requirement of constantly having a sound emitted by the speakers of the laptop for the approach to work, thereby rendering the speakers occupied and unavailable for any other use.

The microphones on smartphones were a predominant means of audio capture in the literature, as they are commonplace and readily available. Therefore, the upcoming paragraphs focus on the recording devices in the literature that provide more context on the viability of smartphones in this research.

With the reason for using a smartphone being as stated above, being able to allow anyone to use the system becomes quite important. This leads to questioning whether any smartphone would be applicable and able to record audio effectively enough to answer the research questions. Smartphones have different microphones that record at different audio qualities. These factors were taken into consideration by Zhou *et al.* [30], who identified that modern smartphones have two microphones, a primary microphone on the bottom and a secondary microphone on the top of the smartphone. However, they indicated that they would be utilising the primary microphone for processing of handwritten sound. They left the investigation of whether the secondary microphone can be utilised to eliminate noise to future work. They tested multiple smartphones recording at a sampling rate of 44100Hz. Sampling rate is defined as the number of samples recorded per second from a continuous audio signal by a recording device. They found that a high enough accuracy (90% and above) was achieved when using their proposed Neural Network implementation for classification.

Chen *et al.* [5] shared this sentiment and set out to identify finger strokes on a surface with a consumer-off-the-shelf (COTS) smartphone. They compared a ZTE nubia, Motorola MT887 and Samsung G3568V and achieved 92.1%, 89.5% and 90.5% accuracies, respectively. The small differences in accuracy were attributed to potential differences in microphone qualities. There was no mention of utilising both microphones on the ZTE Nubia Z9 mini that was used, so it is assumed the phone was utilised in its default configuration i.e. bottom microphone as primary and top microphone for noise

cancellation. They discussed the potential of utilizing both microphones independently combined with time-difference-of-arrival (TDoA) to track the user's strokes more accurately, but this was left to future work. They tested multiple combinations of feature descriptors and machine learning techniques and found that the single microphone on the devices yielded sufficient results with the best combination of techniques achieving a 92.25% accuracy, consisting of Convolutional Neural Networks (CNN) and spectrogram images.

Yu *et al.* [31] took the smart device approach even further when attempting to recognize handwritten strokes using a smartwatch. Smartwatches are equipped with multiple sensors, with particular focus being placed on the use of the accelerometer and microphone. The acceleration parameters of a stroke pattern provided by the accelerometer are used to combat near field noise i.e. the accelerometer is used for noise reduction. The microphone is used as the primary audio capture device in a similar manner to other studies mentioned thus far. The system captures audio signals and acceleration data from the smartwatch and transmits them to the smartphone for processing. The smartwatch sampling rate is 22050Hz, which is half the sampling rate of other recording devices in the literature. Yu *et al.* [31] concluded that under certain conditions they were able to achieve an accuracy of 50-60%, which showed an indication that it is possible to use a smartwatch and smartphone combination to capture, process and recognize handwritten audio signals with an acceptable accuracy.

Haishi *et al.* [9] expanded on Yu *et al.*'s work and attempted to modify the hardware setup to potentially improve the results by removing the need to use the gyroscopic signals from the smartwatch. Like Yu *et al.*'s, they chose the use of a smartwatch because of the additional complexity and hardware features provided by smartwatches over smartphones. They stated that a smartphone has two microphones which help in improving the localization accuracy, also stating that a smartphone can be placed closer to the origin of the acoustic signals therefore potentially further improving on accuracy. A seen accuracy of 81.03% and an unseen accuracy of 74.83% were achieved.

To summarise the outcome of this section: a single smartphone can achieve sufficiently high accuracy, although the physical setup and the processing component play a vital role in the accuracy obtained; and the smartphone type does affect the accuracy but does appears to be far less of a contributing factor in this regard than the physical setup and the processing component.

Writing Implement	Success
Key	0.859
Pen	0.868
Finger	0.783

TABLE 2.2: Success results for Li *et al.* [15] using different writing implements.

2.2.2 Impact of Writing Implements and Surfaces on Audio Capture

The previous section explained the different audio capture devices used. This section will go into detail on each of the different writing implements used in the literature. Writing implements may include the following: finger, pen, pencil, keys etc. The writing implement may have an impact on the recognition accuracy of the audio recognition system, as may the writing surface used.

The most common implement used in the literature is a pen, as it serves multiple use cases such as: capturing one's own writing on a desk; or snooping on another person as they are writing. Li and Hammond [15] explored the performance of their system with different writing implements: a key; a pen; and a fingernail. Table 2.2 shows the results for each writing implement.

It can be seen in the table that the key and pen had very similar results, whereas the use of a finger had somewhat lower accuracy. This finding may be explained by the fact that a finger produces a much quieter sound than a key or a pen and therefore emits less information that can be captured by the microphone and processed. Given that the pen and key yield similar results, it can be said that a hard writing tool combined with a hard surface is beneficial in yielding successful results in this kind of application.

Haishi *et al.* [9] tested their system with four different types of pens on three different surfaces, with the results shown in figure 2.1. Observing the figure, the impact of the surface is very clear and consistent where it is observed that surfaces which are hard, i.e. wooden pad and plastic pad, provide a higher accuracy than surfaces which absorb some of the acoustic signal emitted i.e. notebook. This trend is consistent regardless of the type of pen used.

With regards to differences in the pens used, it can be observed in the figure that the three pens to the right of the graph performed very similarly, whereas the 0.38 gel pen provided a slightly higher overall accuracy than all three others pen used.

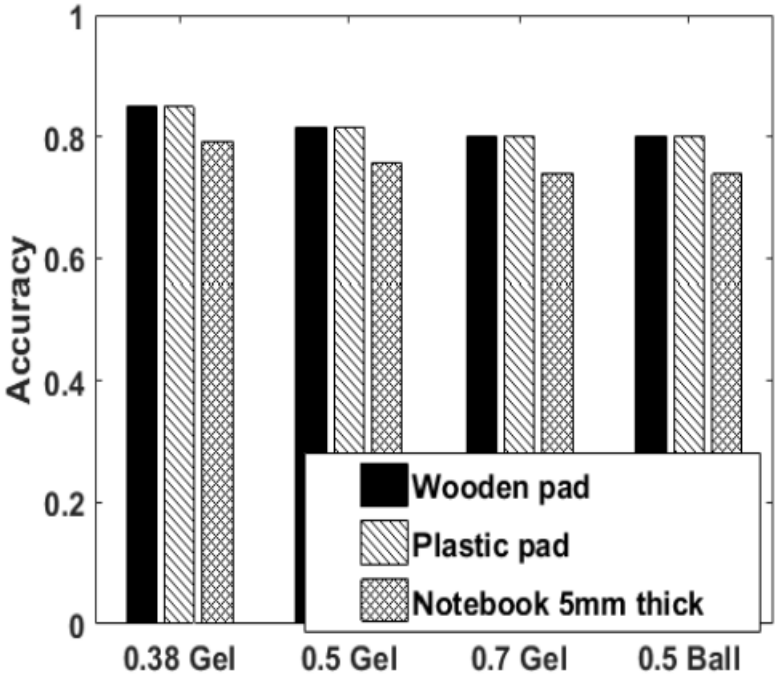


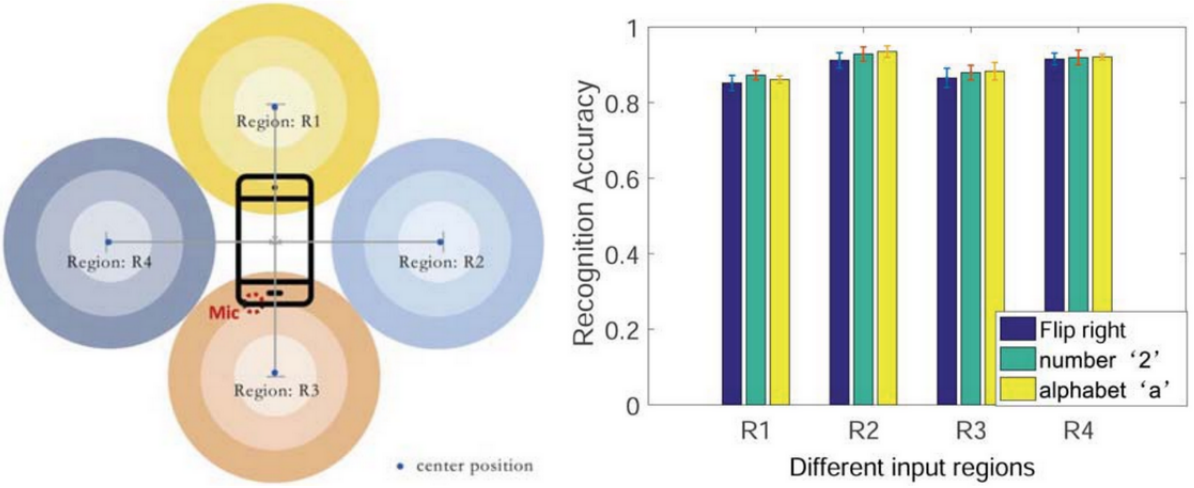
FIGURE 2.1: Impact of different surfaces and writing implements depicted by Haishi *et al.* [9]

2.2.3 Writing location

Writing location can be defined as the region near the audio capture device one which a user writes. Writing location becomes increasingly important when using a smartphone due to the placement of the microphones. Chen *et al.* [5] compared the different writing locations respective to the position of the smartphone, namely: top; left side; right-side; and underneath the smartphone. The results are illustrated in Figure 2.2 and it is illustrated that the best positions are "R2" and "R4". This could be attributed to the fact that most English characters are written top to bottom, which means that the microphone is able to deduce the trajectory of the stroke instead of only the sound moving away from the microphone.

2.2.4 Discussion on Audio Capture

Most of the literature focuses on utilising a smartphone to capture audio the user is writing on a wooden surface or on a hard notebook with a pen. Smartphones can provide sufficiently high accuracy recognition. The number of microphones used does affect the recognition accuracy, although some studies appear to have successfully leveraged only a single smartphone microphone. The greatest impact on audio capture is the writing implement and writing surface, both of which affect the loudness and sharpness of the



(a) Different positions and regions of operation input.

(b) Impact of input region.

FIGURE 2.2: Results of various writing locations conducted by Chen *et al.* [5].

acoustic signal, which in turn affects the ability to accurately recognize handwriting. Finally, the writing location is equally important, where the best location appears to be on either side of the smartphone.

2.3 Audio Segmentation

This section will go into detail on implementations of audio segmentation in the literature where applicable. Audio segmentation aims to extract portions of an audio signal corresponding to individual characters.

Chen *et al.* [5] developed a segmentation algorithm to be able to split the acoustic signal effectively on each letter. The algorithm they developed is shown in Algorithm 1. A sliding window W_a set to $20ms$ is used which yields 884 sample points from a $44kHz$ signal. The expected stroke period—the minimum gap to be observed between strokes—is assumed to be fixed at $600ms$ which is assumed to be the typical maximum period for a single stroke.

An empirical noise threshold $A_\epsilon = 0.02$ is used for a typical office environment and it was stated that this value is environment-specific and needs to be updated according to the environment of the testing. The sliding window W_a is used to calculate the accumulated energy $A(t)$ which is compared to A_ϵ to determine a touch peak which is taken to be the approximate start of a stroke. A segment of $50ms$ before and $950ms$ after each touch peak is extracted for processing and recognition. Gesture input tests were conducted

to evaluate their segmentation algorithm and it was found that the system was able to achieve a low false-alarm rate of 1.15% even with an increasing level of noise up to $61.5dB$ in a cafe environment, which is a noisy environment.

Algorithm 1 Algorithm for segmentation by Chen *et al*[5].

Input: The acoustic signal $x_(t)$, the width of moving average window W_a, the width of signal piece W_g and three temporal thresholds T_1, T_2, T_3
Output: The segment of each operation $\overrightarrow{t_g}$.

```
for t ∈ x(t) do
    A(t) = Σ_{n=t}^{t+W_a} E(n)
    if A(t) > A_ε then
        for i > t & i < t + T_1 do
            if A(t) < A(i) then
                break;
            end if
            i++;
        end for
        if i == t + W_g then
            t is the time of a Peak;
            t_g→ = (t - 50ms, t + 950ms), record t_g→;
            t = t + W_g;
        end if
        t++;
    end if
end for
```

Zhou *et al.* [30] implemented a thresholding approach but included three thresholds in their implementation: the maximum length of a burst noise segment; the time gap between letters; and the time gap between words.

Luo *et al.* [18] took the thresholding approach even further by implementing a dual-threshold scheme. This involved combining zero-crossing rate (ZCR) and short time energy (STE) to perform the segmentation. The acoustic signals were divided into 256 samples, each $32ms$ in length. Luo *et al.* [18] defined short term energy E_n of an acoustic signal $x(n)$ as in Equation 2.1.

$$E_n = \sum_{m=-\infty}^{\infty} [x(m) \cdot w(n-m)]^2 \tag{2.1}$$

where w is the window function and n and m are the end and start indices of a given window as it slides over the audio signal. ZCR is used to indicate the number of times a frame of a signal crosses the zero level of a signal, and it was defined as in Equation 2.2.

$$Zn = \frac{1}{2} \sum_{m=-\infty}^{\infty} |sgn[x(m)] - sgn[x(m-1)] \cdot w(n-m) \tag{2.2}$$

where *sgn* is a function denoted as in Equation 2.3.

$$sgn[x(n)] = \begin{cases} 1 & x(n) \geq 0 \\ -1 & x(n) < 0 \end{cases} \tag{2.3}$$

Chen *et al.* [29] proposed an approach similar to Luo *et al.* [18] with a dual-threshold scheme and found similar success with their results.

2.3.1 Discussion on Audio Segmentation

The audio segmentation techniques discussed make use of threshold schemes to mark the start and end of salient segments in an audio stream, with the number and type of thresholds varying. One main method observed was to use a threshold as an upper-bound to ambient noise, and this threshold was applied to a sliding window. Furthermore, making an assumption about the gaps between strokes and the gaps between letters drawn is necessary to separate and segment the individual letters.

In this research, the audio segmentation used will make use of these thresholding concepts.

2.4 Feature Extraction Techniques

This section will go into detail on the feature extraction techniques that were used in the related studies. Feature extraction can be described as the process applied to a sample to obtain key salient characteristics of that sample which can be used to recognize or classify it by means of a classifier.

The following subsections describe prominent feature extraction techniques used in the related studies. A discussion on these techniques follows.

2.4.1 Mel-Frequency Cepstral Coefficients Feature Extraction

An effective audio feature extraction method is the MFCC. This technique is one of the most popular in audio recognition due to its robustness and the way it attempts

to mimic human hearing. As a brief description of its function: MFCC is a means of representing the short-term power spectrum of sound, based on a linear cosine transform of a log spectrum on a nonlinear Mel scale of frequency [18]. The extraction process is shown in Figure 2.3. The steps in the figure are explained below [18]:

- Pre-emphasis: A process of passing the acoustic signal through a high-pass filter to raise the high-frequency portion and flatten the spectrum of the acoustic signal.
- Windowing: A sliding window/frame that is passed over the audio signal. A Hamming window is applied to the window/frame to increase the continuity of the left and right ends of the frame.
- Fast Fourier Transform: FFT is applied to each frame of the signal, in order to transform the distribution of energy into frequencies and calculate the periodogram of the power spectrum.
- Mel-filter bank: A process of applying the Mel filterbank to the power spectrum, and then summing the energy in each filter.
- Logarithm energy: All filterbank energies are then passed through a logarithmic function to convert them from the linear spectrum to the logarithmic spectrum.
- Discrete cosine transform (DCT): DCT is applied to the log filterbank energies to obtain and return the DCT coefficients.

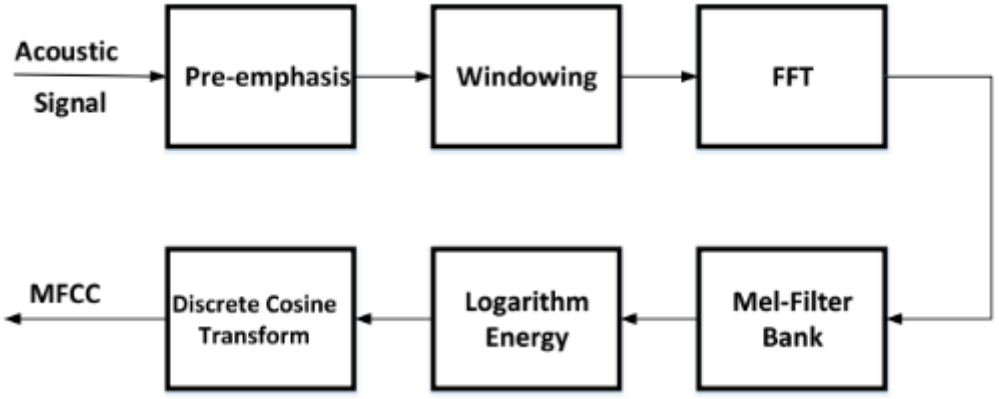


FIGURE 2.3: MFCC feature extraction as described by Luo *et al.* [18]

According to Luo *et al.* [18], the performance of the MFCC feature extractor decreases as the ambient noise increases. This led them to implementing the Cochlear filter cepstral coefficients (CFCC) and creating a hybrid of MFCC and CFCC which improved their

accuracy. For the purposes of this research, MFCC will be sufficient as the ambient noise will remain low in an office environment based on Premise 5 set out in Chapter 1.

Yin *et al.* [29] implemented MFCC with their Ubiquitous Writer system which allowed them to recognise cursive handwriting. The approach involved frequency components in the range of 0Hz to 10kHz, which is where the Mel filter banks were applied. The number of MFCCs for each sound frame was set to 12 and the logarithmic energy was calculated per frame. These coefficients were then combined with a K-Nearest Neighbours (KNNs) algorithm for classification. This ultimately led to achieving an average accuracy of greater than 90% in environments with an ambient noise level of 45dB and below. As the ambient noise increased and exceeded levels of 70dB, the accuracy decreased to 65%. This confirmed the findings of Luo et al. [18] who stated that the performance of the MFCC would likely decrease as the ambient noise increased.

Li and Hammond[15] also implemented the MFCC feature descriptor. Their implementation used a window size of 256 and the coefficient values used were the first 12 values, except for the first coefficient which was removed as it proved to have a negative effect on their results. An 80% accuracy was achieved with their combination of MFCC and dynamic time warping (DTW).

2.4.2 Spectrogram Feature Extraction

Chen *et al.*[5] identified that an acoustic signal represented as a Spectrogram is highly distinguishable across various writing operations using a finger as the writing implement. This led to Chen *et al.* creating Spectrogram images of the acoustic signal and combining them with a CNN, thereby achieving an accuracy of 92.25%. The Spectrograms were 64 × 64 in size, which can readily be processed by a graphics processing unit (GPU). They compared their Spectrogram and CNN approach to various combinations of feature extraction and classification. Their results are shown in Figure 2.4.

KNN and ASD	SVM and ASD	SVM and MFCC	CNN and MFCC	CNN and Spectrogram
14.57%	46.53%	62.16%	82.17%	92.25%

FIGURE 2.4: Results of different combinations of feature extraction and classification by Chen *et al.* [5].

Observing the figure, it is seen that their approach yields better results than other techniques, but they state that latency can be an issue when used on a smartphone that does not have a powerful GPU to support the Spectrogram and CNN approach.

2.4.3 Discussion on Feature Extraction Techniques

The above section explained two common feature extraction techniques used in the previous literature. The discussion showed that the MFCC descriptor has been tried and tested and is therefore a good option. The MFCC descriptor is commonplace among audio recognition applications because of its robustness and ability to mimic the human hearing mechanism. The alternative presented in the above findings is the use of an image-based approach i.e. Spectrograms, which display the processed signal as an image in the time–frequency domain. While this option may potentially provide a higher accuracy, it suffers from latency issues in the absence of a capable GPU. It may therefore not be suitable for use on smartphones at the current time.

As such, the intention is to use the MFCC feature descriptor in this research.

2.5 Classification and Recognition

Classification is the final step in the process of audio recognition. This section will go into detail on the classification methods that were implemented previous studies. Many different approaches were used for classification. These approaches along with studies that used them are covered in each of the subsections below.

2.5.1 Template Matching

Template matching is a generic term for a group of methods that match an input sample to a series of pre-registered templates in the same format and determine the template that best matches the input. This is used to determine the class to which the input belongs.

Li and Hammond [15] implemented template matching using DTW, which allowed them to match 26 handwritten Roman characters with a fixed stroke pattern with varying speeds. The DTW algorithm was used to calculate the distance D between the input A and each previously computed template B to determine the best match. This is shown below in Equation 2.4.

$$D(A,B) = \frac{1}{N} \min_{F} [\sum_{k=1}^{k} d(c(k)) * w(k)] \tag{2.4}$$

where $c(k)$ is the mapping between the candidate and the template at time index k, $w(k)$ is the weighting function, N is the normalized value which in the case of Li and Hammond [15] is the mean amplitude of the input signal, and the aim is to find the optimal path of F to minimize $D(A, B)$.

They achieved results of 80% and above in their testing and stated that future work could include a greater noise removal system as their system performed poorly in noisy and/or public environments. The characters that were the most common source of misclassification errors can be seen in Figure 2.5, where the left column represents the ground-truth character and the right column represents the incorrectly predicted character. They stated that the factors contributing to these misclassified cases could be similarities in the number of strokes, combined with the stroke ordering, of letters.

C	U
D	P
A	H
J	T
X	Y

FIGURE 2.5: Common misclassified characters found by Li and Hammond [15]: left column represents the ground-truth character and the right column represents the corresponding incorrectly predicted character.

2.5.2 Support Vector Machines

Support vector machines (SVMs) are commonly used in the previous works in acoustic signal recognition which makes them a focal point in this research. SVMs were originally developed by Cortes and Vapnik [7]. SVMs were originally intended to be used for binary classification but they were later adapted to multi-class classification problems. According to Meyer [20], SVMs compute the optimal hyperplane between classes by maximizing the margin between the points of either class closest to the opposing class. These points lie on the margin and are called support vectors, and the mid-point of the margin is the optimal hyperplane. This can be shown in Figure 2.6.

As explained by Hearst *et al.* [11], a key idea of Support Vector Machines is to map training data non-linearly onto a higher-dimensional feature space via a kernel function. This makes it possible to compute the separating hyperplane in the input space without explicitly calculating the mapping in the feature space. This was illustrated by Hearst *et al.* [11] in Figure 2.7.

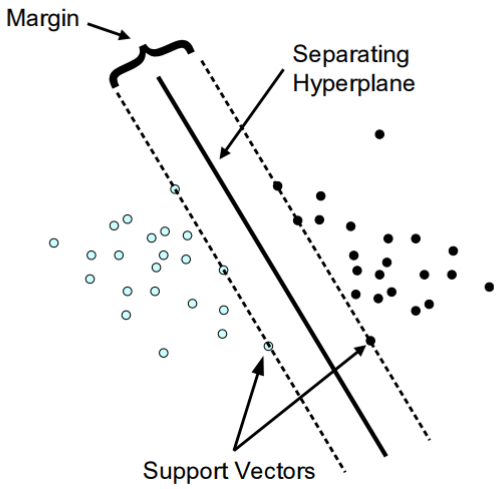


FIGURE 2.6: SVM Classification as illustrated by Meyer [20].

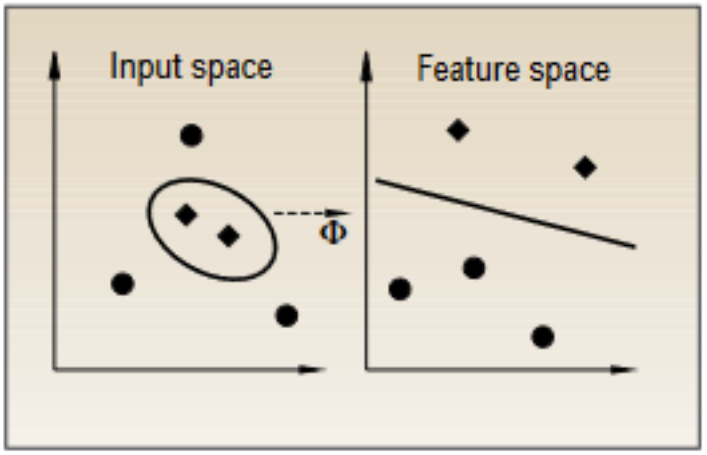


FIGURE 2.7: The idea of SVMs as explained by Hearst *et al.* [11] which shows the creation of a non-linear hyperplane in input space by mapping training data onto a higher-dimensional feature space via a kernel function.

In the literature, there are various different applications of SVMs, either as the primary classification method or as a comparative technique to other classification methods. The following paragraphs will go into detail of the results of previous studies and the comparisons of SVMs to other classification methods.

Wu *et al.*[27, 28] implemented an SVM for their audio-based digit recognition system, which aimed at recognizing the 26 uppercase letters which were written on a surface next to a set of transducer microphones as mentioned in Section 2.2.1. The system yielded a 91% accuracy on a semi-seen dataset with 5 test subjects.

Another set of testing was carried out with 3 additional subjects, each of which wrote each letter 6 times. This was called the unseen test dataset as these subjects were not utilised in the training phase at all. This dataset yielded a 77% accuracy across all

Classification Method	Training Samples	Accuracy (%)	Time (s)
SVM (C=0.5, γ=0.08)	5	73.00	0.34
	10	84.00	0.50
	15	91.30	0.66
	5	93.20	0.78
KNN (k=7)	5	57.00	0.09
	10	80.00	0.16
	15	90.00	0.29
	5	91.10	0.36

TABLE 2.3: Comparison of KNN and SVMs as conducted by Luo *et al.* [18].

unseen test subjects.

Luo *et al.* [18] implemented an SVM as their primary classification method. An RBF kernel was chosen to be able to achieve a nonlinear mapping. The RBF kernel was noted as being particularly suitable due to having a small number of optimizable parameters. They compared an SVM to KNNs, and found that they were able to achieve a higher accuracy of 93.2% with the SVM as compared to 91.1% with KNNs, when training on 7 gestures with 20 samples each.

The results of the comparative experiment can be seen in Table 2.3. They concluded that SVMs have a better performance, especially when the number of samples is relatively small, and the training time for SVMs increases at a slower rate than that of KNNs, even if the training time of KNNs is lower to begin with.

Haishi *et al.* [9] implemented a smartwatch as their primary recording device as mentioned previously. In their research they decided to conduct a comparison between popular classification methods: KNNs, SVMs and CNNs. Given that their system was designed for smartwatches, they required the classification process to be particularly efficient. They therefore determined the time cost of letter recognition along with the accuracy. Their dataset consisted of 10 subjects writing an essay consisting of 300 words in uppercase. The results of their experimentation can be seen in Table 2.4.

Table 2.4 shows that CNNs and SVMs are very efficient when compared to KNNs when comparing time in seconds, but KNNs and CNNs provide a higher accuracy compared to SVMs in this application.

Method	Accuracy (%)	Time (s)
KNN	80.00	3.00
SVM	60.00	0.30
CNN	82.00	0.10

TABLE 2.4: Results of letter recognition by Haishi *et al.* [9].

$$C_1 = \{\text{'C'}, \text{'G'}, \text{'L'}, \text{'M'}, \text{'O'}, \text{'S'}, \text{'U'}, \text{'V'}, \text{'W'}, \text{'Z'}\},$$
$$C_2 = \{\text{'B'}, \text{'D'}, \text{'J'}, \text{'K'}, \text{'P'}, \text{'Q'}, \text{'R'}, \text{'T'}, \text{'X'}\},$$
$$C_3 = \{\text{'A'}, \text{'E'}, \text{'F'}, \text{'H'}, \text{'I'}, \text{'N'}, \text{'Y'}\}.$$

FIGURE 2.8: Word clusters by Yu *et al.* [31]

Yu *et al.* [31] proposed the use of "recognition clusters" whereby letters were clustered into three groups, each of which is recognized separately to improve on accuracy. The clusters can be seen in figure 2.8. The clusters allowed them to build specialized SVMs per cluster of letters. This was done by segmenting the audio signal not only by words but also calculating the number of strokes in each letter. This allowed them to select which SVM to use for prediction. For training, they had 2 subjects write 26 Roman characters 20 times, and for testing 3 subjects wrote each character 20 times. During training it was stated that the location of writing and smartphone were fixed.

The system was tested in three different ways: testing on the training data in the same location that training data was collected; testing with unseen data in the same location that training data was collected and labelled; and testing with unseen data in a different location to the one training data was collected. The results obtained by the system are summarised in Figure 2.9 which shows: 26.79% for unseen data with different writing locations; 43.01% for seen data, with a different writing location (Case 1); and 64.94% for seen data in the same writing location (Case 2).

Observing Table 2.5, it can be seen that using either unseen data or using an unseen location results in a significant reduction in accuracy. A complete breakdown of the accuracies per recognised letter is provided in Figure 2.9, and the figure shows a large amount of variation in accuracy across letters, as a result of similarities in the number or pattern of strokes in various letters.

Zhou *et al.* [30] compared SVMs, KNNs and a custom-built classification algorithm consisting of an Inception-LSTM network, which is a neural network combined with time series modelling. The inputs into their Inception-LSTM comprised of short-time power spectral density (stPSD) features. They obtained a 28.7% accuracy with the SVM

Method	Accuracy (%)
Data Unseen + Loc Unseen	26.79
Data Seen + Loc Unseen	43.01
Data Seen + Loc Seen	64.94

TABLE 2.5: Results of letter recognition by Yu *et al.* [31].

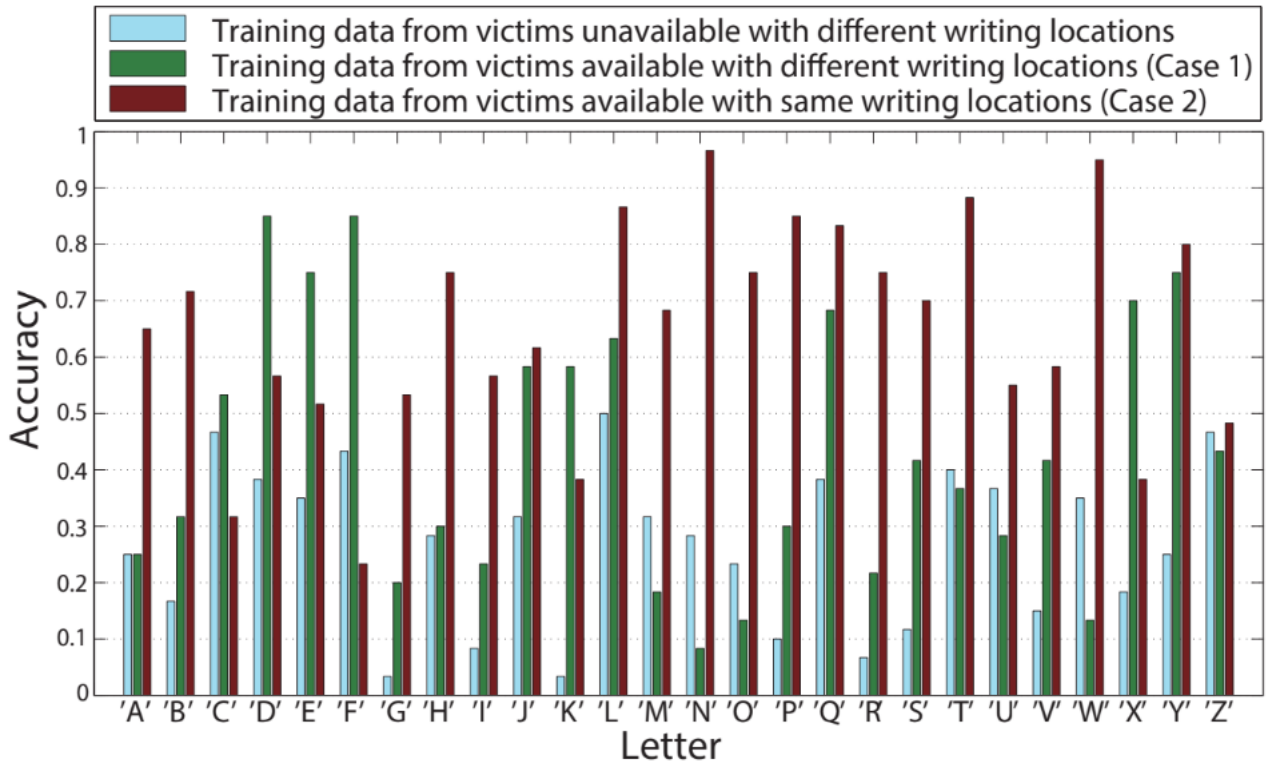


FIGURE 2.9: Recognition accuracy by Yu *et al.* [31] in 3 different scenarios.

as opposed to 16.5% and 73.8% for their KNN and Inception-LSTM implementations respectively. The big difference in accuracy can be attributed to the fact that their users were allowed to write with differing stroke patterns; the SVM struggled to differentiate between single stroke versus multi-stroke characters and the ordering of the characters.

Chen *et al.* [5] implemented their *IPanel* system which uses the images of spectrograms combined with a CNN for classification. This method was first compared to a range of other implementations including an SVM and MFCC combination. These results were shown previously and are illustrated in Figure 2.4. One of the conclusions in that application was that the SVM and MFCC combination may not be able to effectively distinguish between a very large number of classes, in that case 43 different types of actions (10 digits, 26 alphabetical characters and 7 gestures) that they were attempting to recognise, and in such cases a different setup may be required.

2.5.3 Neural Networks

This section explains the approaches in the literature that utilised Neural Networks, and their potential pros and cons.

Neural Network topologies

Layer		Network: *Motion & Write*	*Audio*	*Audio & Write*	*All*
Input		1050	3500	3650	4400
hidden	*1*	1050	3500	3650	4400
	2	790	2802	2952	3442
	3	530	2104	2254	2484
	4		1406	1556	1406
Output		10	10	10	10

Neurons

FIGURE 2.10: Neural Network configurations by Schrapel *et al.* [23]

Schrapel *et al.* [23] utilised a neural network for their Pentelligence handwriting recognition system as their classification method of choice. Their reasoning was that a template matching approach would not be sufficient as there are multiple different ways that people would write various characters, with a different number of strokes and/or a differing stroke pattern. A good example is the number '5' which can be written in many ways. Another factor in their decision was the fact that it is challenging to conduct feature selection for audio because the microphone can provide noisy data. A neural network, they argued, would be able to solve this problem by automatically weighting and selecting features.

A training dataset of 6169 handwritten digit samples was collected. The different features of the writing included audio, motion, and handwriting. They compared four different neural network configurations, with different input features, namely: audio only; motion and pen pressure; audio and pen pressure; and all three features combined. The configuration of the four networks, along with the types of input and the number of neurons on the hidden layers of each associated network, can be seen in Figure 2.10. The average accuracy achieved by each configuration is summarised in Table 2.6.

Schrapel *et al.* identified the potential of overfitting and therefore utilized dropout which randomly removes a set of selected neurons and their connections. A 25% dropout rate achieved the greatest results which implies there was an element of overfitting that was being corrected.

As shown in Table 2.6, utilising only Motion Pressure achieved the greatest accuracy. This led to their final classification system first recognizing an unknown digit using the motion and pressure data and utilizing the audio data to validate the prediction. They were able to achieve an overall accuracy of 78.3%. It is important to note that these results were achieved using a specialized writing device with several sensors which

allowed the utilisation of more features than with a simple writing implement such as a pen combined with a microphone.

2.5.4 Discussion on Classification and Recognition

This section aimed to introduce the various methods of classifying and recognizing hand-written characters. The discussion showed that a range of different classifiers have been used in previous studies, with varying levels of success. Ultimately, it appears that the context of use is important to consider, as are aspects of the hardware setup such as the writing implement, writing surface and audio capture device, as well as the feature descriptor used. Based on this and based on the success registered by Wu *et al.*, this research aims to use SVMs coupled with MFCCs and the hardware setup explained previously, to investigate the extent to which this can be successful.

As discussed, SVMs have been used in a variety of studies. The results are mixed and vary between 90% and 60% accuracy. This depends on the context of use and a range of factors as described previously. For fixed stroke patterns and a moderately large number of classes, as is the case in this research, they hold promise, and are therefore used in this research.

2.6 Summary

This chapter aimed to explain previously implemented systems which utilise similar techniques that would be required in this research.

After providing an overview of the studies selected for inclusion in this research, the literature survey detailed the following aspects of the related studies: audio capture; audio segmentation; feature extraction; and classification and recognition.

With respect to audio capture, it was found that many related studies focused on utilising a smartphone to capture audio as written on a hard surface with a pen. Smartphones, as an audio capture device, were found to be capable of providing sufficiently high accuracy

Network	Accuracy (%)
Motion & Pressure	79.2
Audio only	58.4
Audio & Pressure	60.4
All features	60.6

TABLE 2.6: Results of the four network configurations by Schrapel *et al.* [23].

recognition. It was also seen that the greatest impact on audio capture is the writing implement and writing surface, both of which affect the loudness and sharpness of the acoustic signal, which in turn affect the ability to accurately recognize handwriting. Finally, the writing location was found to be equally important, where the best location appears to be on either side of the smartphone.

In terms of audio segmentation, the audio segmentation techniques discussed were found to make use of thresholding schemes to mark the start and end of salient segments in an audio stream, with the number and type of thresholds varying. One main method observed was to use a threshold as an upper-bound to ambient noise, and this threshold was applied to a sliding window. Furthermore, assuming about the gaps between strokes and the gaps between letters drawn is necessary to separate and segment the individual letters. In this research, the audio segmentation used will make use of these thresholding concepts.

The discussion on feature Extraction techniques explained two common feature extraction techniques used in the previous literature. The discussion showed that the MFCC descriptor has been tried and tested and is therefore a good option. The MFCC descriptor is commonplace among audio recognition applications because of its robustness and ability to mimic the human hearing mechanism. The alternative presented was the use of an image-based approach i.e. Spectrograms, which displays the processed signal as an image in the time–frequency domain. While this option may potentially provide a higher accuracy, it suffers from latency issues in the absence of a capable GPU and is therefore not suitable for use on smartphones at the current time. As such, the intention is to use the MFCC feature descriptor in this research.

Finally, the discussion on classification and recognition aimed to introduce the various methods of classifying and recognizing handwritten characters. The discussion showed that a range of different classifiers have been used in previous studies, with varying levels of success. SVMs were shown to have been used in a variety of studies with results varying between 60% and 90% accuracy.

Ultimately, the discussion made it apparent that the context of use is important to consider, as are aspects of the hardware setup such as the writing implement, writing surface and audio capture device, as well as the feature descriptor used. Based on this, and based on the success registered by Wu *et al.*, this research aims to use SVMs coupled with MFCCs and the hardware setup explained previously, to investigate the extent to which this can be successful. For fixed stroke patterns and a moderately large number of classes, as is the case in this research, they hold promise, and are therefore used in this research.

The next chapter discusses the different tools and techniques required and used in this research to implement the proposed handwriting recognition system.

Chapter 3

Tools and Techniques for Handwriting Recognition

This chapter will outline the different tools and techniques required and used in this research to implement the proposed handwriting recognition system, inline with the research question and sub-questions posed in Chapter 1.

Similar to the previous chapter, the organization of this discussion on tools and techniques corresponds to tools and techniques pertaining to the research objectives set out in Chapter 1, where each objective can be considered to be a task to be implemented towards realizing the final proposed smartphone-based handwriting recognition system. Accordingly, the tasks in the overall system are depicted in Figure 3.1 and the rest of the chapter will be sub-divided according to the tasks in the figure, where each section will delve into the tools and techniques used to achieve each of these tasks. Sections 3.1 through 3.4.1, respectively, discuss tools and techniques towards the following tasks: audio signal capture; audio segmentation; feature extraction; and classification.

At the end of this chapter, the reader will be able to understand the techniques and tools used in this research ahead of the implementation in the next chapter.

3.1 Audio Signal Capture

The first step in the process of recognising handwritten characters is capturing audio emitted on an intended writing surface. This section will showcase an example of what a handwritten signal looks like when depicted as a sound wave. This provides a baseline

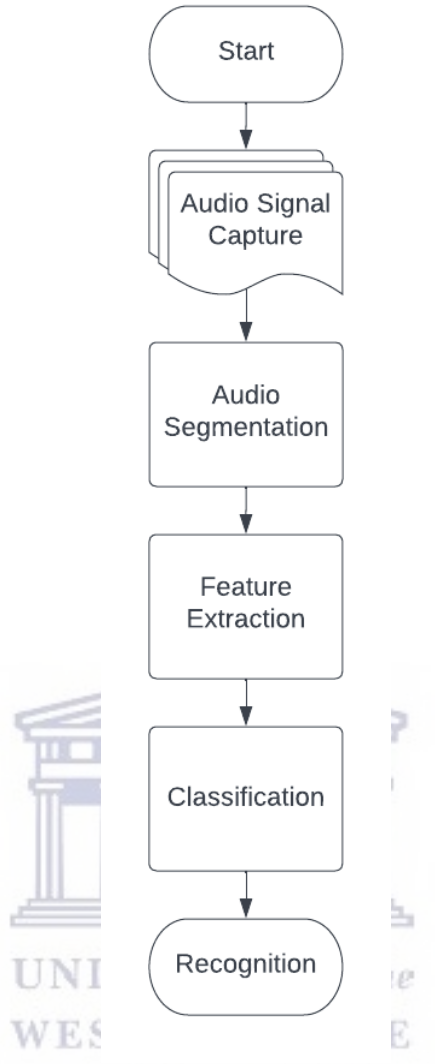


FIGURE 3.1: Overview of the proposed handwriting recognition system.

of what the audio signal starts off as before it is processed in subsequent steps of the process in the handwriting recognition system.

Figure 3.2 is a visualization of the raw audio signal of 10 consecutive handwritten 'B's as captured by a single microphone. The 'B's in the signal are represented by spikes in the audio signal, the first four of which have been demarcated by the red region lines, separated by brief regions of flat or near-flat signal between the bulges, representing the brief silences between the letters. The audio gap is left between each letter in order to make it easier to segment the audio into 10 separate audio signals. The reason for this is to allow a user to write a continuous string of letters without stopping in order to make the system more readily usable in a real-world scenario.

It is also important to note, by observing the figure, that there is a wide visual variation between the acoustic patterns of each 'B' drawn, although there are also some common

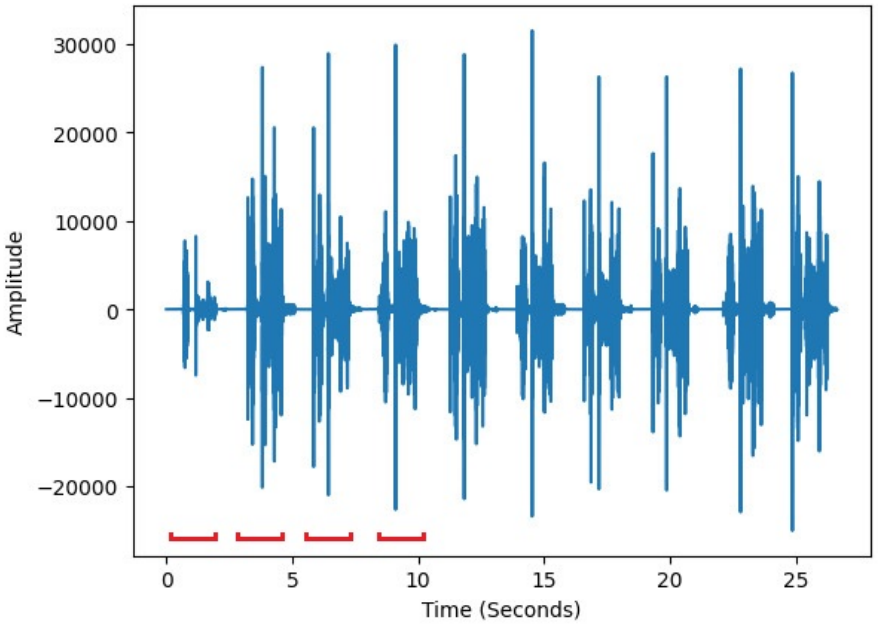


FIGURE 3.2: Visualization of the raw audio signal of 10 consecutive handwritten 'B's as captured by a single microphone. The regions of the audio signal corresponding to the first four 'B's have been indicated by the read region lines.

features.

3.2 Audio Segmentation

This section explains how audio segmentation is carried out. The goal of audio segmentation in this context is to split the audio signal into separate salient parts, in this case the parts of the input audio signal corresponding to each of the hand-drawn letters, which can then be individually classified and recognized. The audio segmentation algorithm utilised in this research was proposed by Robert [12]. This method will be explained below.

An overview of the algorithm follows. The audio segmentation strategy devises a means of separating letters by assuming that the user will observe a brief silence between written letters. The algorithm then works by detecting key points of silence in the input audio signal as determined by an empirically determined silence threshold measured in decibels-relative-to-full-scale (dBFS)[19] which is applied to a sliding window that the algorithm slides over the audio signal. The key points of silence are combined into ranges that specify ranges of silence in the input signal. Points of non-silence, in this case corresponding to written letters, are then taken as ranges in which silence is not observed in the input signal.

Based on the above description, assume that the dBFS of the input signal S is represented by D which is a set of dBFS values corresponding to each sample in the input signal given by $\{D_i | i \in \{1, 2, \ldots, N_s\}\}$ with N_s representing the number of samples in S. The first step in the process is then to determine the set of indices K of key samples at which silence is observed in the signal; an index k is in K if the root-mean-square of the signal dBFS starting from k over a window of length L falls below a pre-determined silence threshold T_l as follows:

$$\begin{aligned} K &= \left\{ k \quad \text{if} \quad \left[\sqrt{\frac{1}{L} \sum_{i=l}^{k+L} D_i^2} \right] < T_l \quad \right\} \\ \forall \quad k &\in \left\{ 1, (s_L + 1), (2 \cdot s_L + 1), (3 \cdot s_L + 1), \ldots, \left(\left\lfloor \frac{(N_s - L)}{s_L} \right\rfloor \cdot s_L + 1 \right) \right\} \end{aligned} \tag{3.1}$$

Where s_L is a step size by which to increment the position of the sliding window, the value of which is typically set to the number of samples corresponding to 1ms in the audio signal.

Once the points K are detected, the indices in K corresponding to the start and stop indices of continuous silent ranges $R = \{(l_1, r_1), (l_2, r_2), \ldots, (l_{N_R}, r_{N_R})\}$ are determined. This is done by scanning through indices $k \in K$ and taking any index k to be the start $l_z = k$ of a new range number z, and $(k - 1)$ to be end point $r_{(z-1)} = (k - 1)$ of the previous range number $(z - 1)$ if $k - (k - 1) > s_L$ i.e. if there a jump in index greater than the sliding window step size.

Having done this, the start and stop indices of silent portions in the input audio signal have been obtained in R. Accordingly, the portions of the audio signal that correspond to non-silence N are taken to be the samples that start at the end of each silent range and stop at the beginning of the next silent range i.e. $N = \{(r_1 + 1, l_2), (r_2 + 1, l_3), \ldots, (r_{N_R}, N)\}$. These indices are used to subdivide the original audio signal into the constituent non-silent segments $\left\{ (S_{r_1+1}, \ldots, S_{l_2}), (S_{r_2+1}, \ldots, S_{l_3}), \ldots, (S_{r_{N_R}}, \ldots, S_{N_s}) \right\}$ for further processing. An example of the audio signal of a segmented 'B' is shown in Figure 3.3.

3.3 Feature Extraction

The feature extraction technique used in this research is the Mel-Frequency Cepstral Coefficients (MFCC). The purpose of the MFCC process is to convert a signal from the

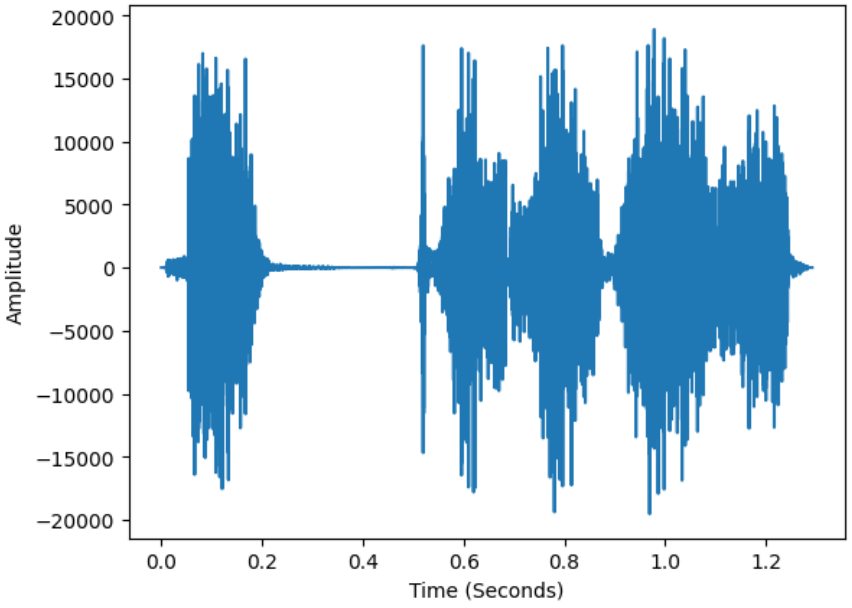


FIGURE 3.3: An example of a single B extracted from the audio signal in Figure 3.2

time domain to the frequency domain in the form of a frequency spectrum by means of a Fourier transform, followed by converting the frequency spectrum into a spectrum of spectrums, known as a cepstrum. Figure 3.4 was provided in a previous chapter but has been repeated here for convenience. The figure depicts the main steps involved in computing the MFCC feature descriptor given an audio signal. The following subsections follow the progression through the steps in the figure and go into detail to explain each step and provide examples of the output of each step.

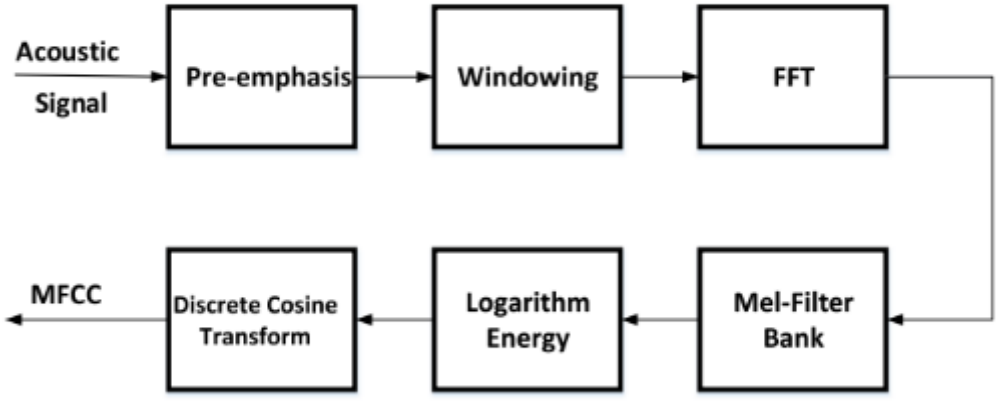


FIGURE 3.4: MFCC feature extraction process as described by Luo *et al.* [18]

3.3.1 Pre-emphasis

Pre-emphasis is the process of passing the acoustic signal through a high-pass filter in order to emphasize the higher frequencies of the audio signal. Assuming that $S^{(0)}$ is the audio input signal corresponding to a single handwritten letter segmented from the original audio recording as previously described—with the newly introduced superscript (0) indicating that the audio signal segment is initially unprocessed—and assuming that $S^{(0)}$ consists of N_s equally spaced sampling points indexed by the symbol n given by $S^{(0)} = \{S_n^{(0)} | n \in \{1, 2, \ldots, N_s\}\}$, the result of applying pre-emphasis to $S^{(0)}$ is $S^{(1)}$ as expressed in Equation 3.2.

$$S_n^{(1)} = S_n^{(0)} - \beta S_n^{(1)} \quad \forall \quad n \in \{0, 1, \ldots, (N_s - 1)\} \tag{3.2}$$

Where β represents a pre-emphasis factor typically set to a value between 0.9 and 1.0, with a value of $\beta = 0.97$ used in this case. An example of pre-emphasis applied to Figure 3.3 can be seen in Figure 3.5

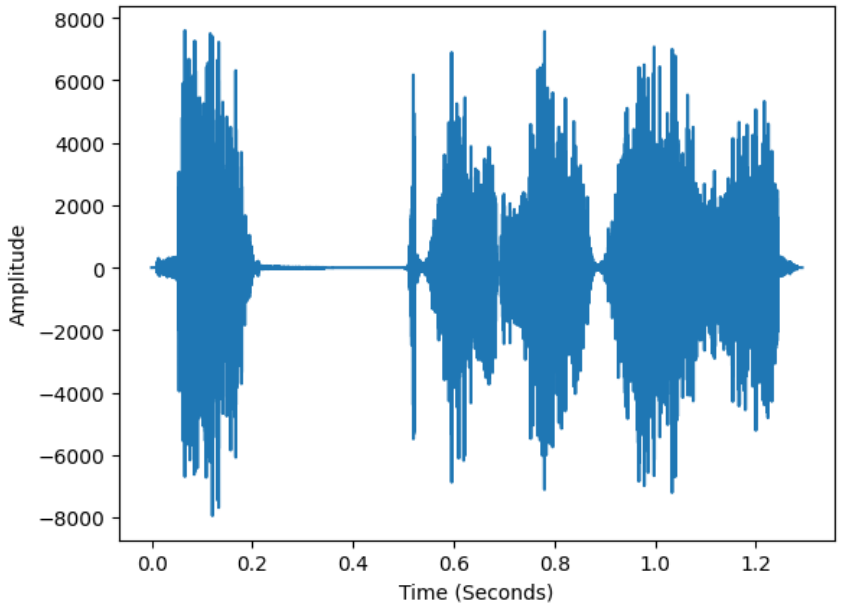


FIGURE 3.5: Example of the audio signal in Figure 3.3 after pre-emphasis.

3.3.2 Framing and Windowing

The pre-emphasised audio signal is segmented into smaller overlapping blocks/frames of typically $25ms$ in length, where consecutive frames typically overlap each other by $10ms$. While these values may be adjusted, they are the most commonly used values with the MFCC feature descriptor and are therefore used in this research.

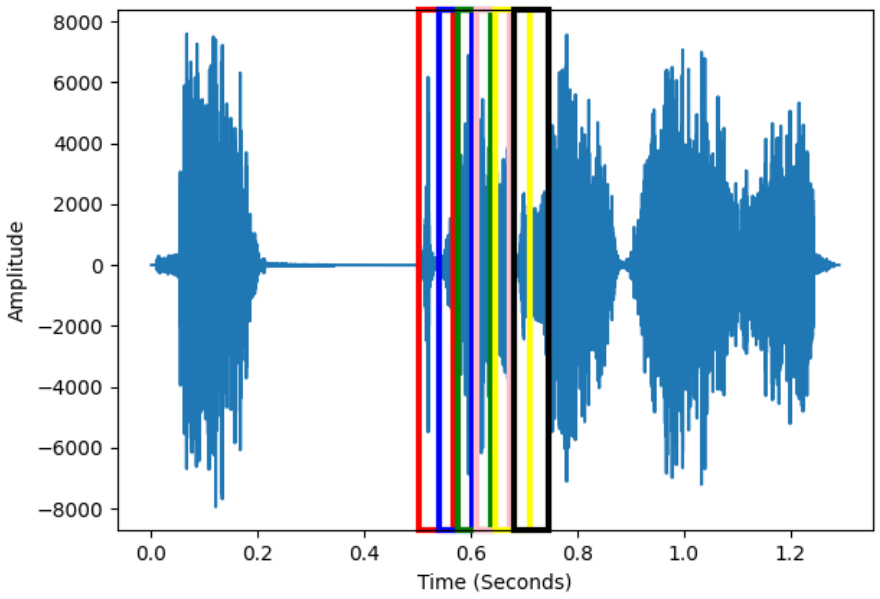


FIGURE 3.6: Framing on top of the pre-emphasized signal in Figure 3.5.

An example of framing can be seen in Figure 3.6. The figure illustrates how the signal is split into individual overlapping frames, visually represented by the different coloured rectangles, each of which has a length of $25ms$. To be precise, it should be noted that the resulting frame length denoted by N_s, given an input signal sampled at 44.1kHz, is given by $N_s = \lceil 0.025s \times 44100Hz \rceil = 1103$ samples. Accordingly, the notation $F_{j,k}$ will be used to refer to the kth sample in frame index j, noting that each frame consists of N_s samples.

Each frame is then convolved with a Hamming window. This windowing operation is designed to taper the start and end of the signal to 0, and keep the continuity of the signal, in order to produce a signal that conforms to the requirements of subsequent steps in the process. The Hamming window function W can be found in equation 3.3.

$$W(k) = 0.54 - 0.46\cos\left(\frac{2k\pi}{N_s - 1}\right), \quad \forall \quad k = \{0, 1, 2, \ldots, (N_s - 1)\} \tag{3.3}$$

where the length of the Hamming window is set to N_s in order to align with the frames with which it will be convolved, and k is the sample index within the Hamming window given by $k \in \{0, 1, \ldots, (N_s - 1)\}$. The following equation, Equation 3.4 describes the application of the Hamming window to obtain the windowed frames $S^{(2)}_{j,k}$ for all frames j:

$$S_{j,k}^{(2)} = W(k) * F_{j,k}, \quad \forall \quad k \in \{0, 1, 2, \ldots, (N_s - 1)\} \tag{3.4}$$

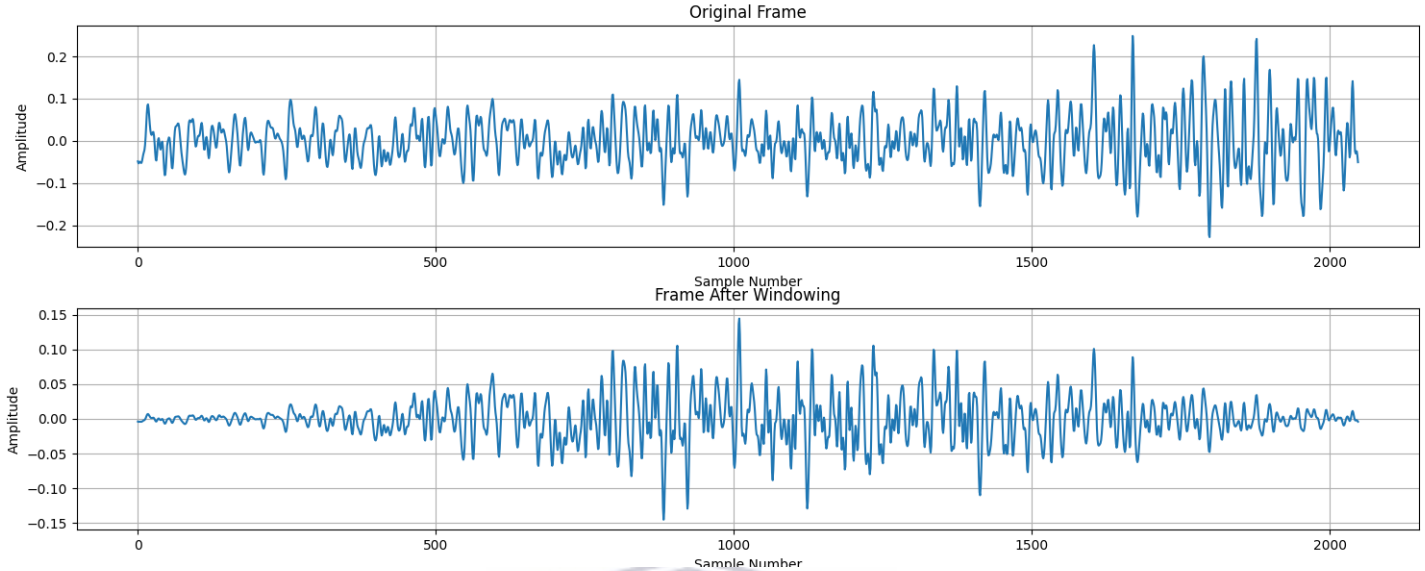


FIGURE 3.7: Example of a Hamming window that is applied to an audio signal: (Top) The original audio signal; (Bottom) Result of applying the Hamming window to the signal.

The result of applying this process to the audio signal on the top of Figure 3.7 is shown in the bottom of the same figure.

3.3.3 Fast Fourier Transform

The Fast Fourier Transform (FFT) is an efficient implementation of the discrete Fourier transform (DFT) for converting an audio signal from the time domain to the frequency domain. Converting the signal into the frequency domain is required in subsequent steps of the MFCC feature descriptor and also corresponds to human hearing. The FFT is applied to each frame resulting from the previous step in the process. The application of the FFT to each frame j is defined in Equation 3.5. Of note is the change in indexing of the resulting S from k to f indicating a change from the time domain to the frequency domain.

$$S_{j,f}^{(3)} = \sum_{k=0}^{N_s-1} f_p \exp\left[-i2\pi \frac{pk}{N_s}\right] \quad \forall \quad k \in \{0, 1, 2, \ldots, N_k\} \tag{3.5}$$

where f_p is the p-th frequency and N_k is the number of FFT bins to use in the computation which is typically set to $N_k = \left(\frac{N_s}{2} - 1\right)$. This is followed by a computation

of estimates of the spectral power at each frequency in each frame ȷ which is given by Equation 3.6:

$$S_{j,f}^{(4)} = \frac{1}{N_s} \mid S_{j,f}^{(3)} \mid^2. \tag{3.6}$$

The result of applying this process to the audio signal in Figure 3.8a can be seen in Figures 3.8b and 3.8c.

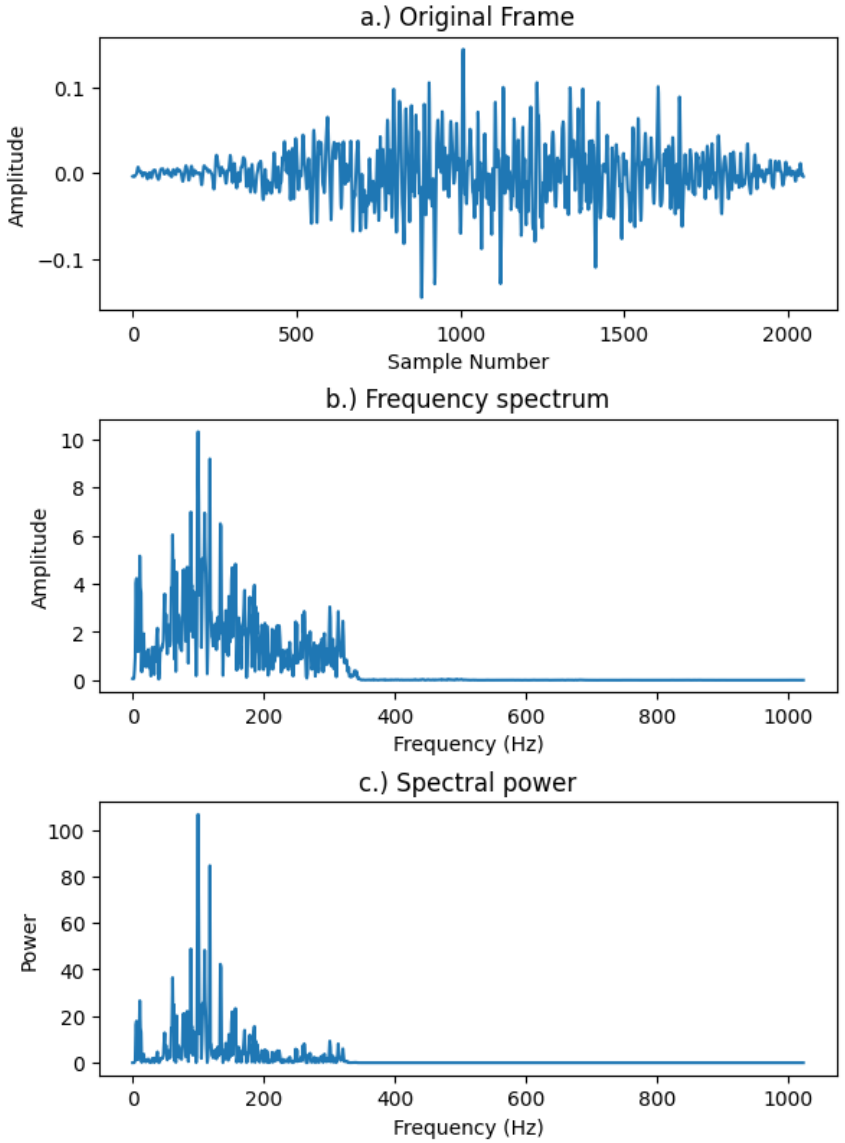


FIGURE 3.8: a) Shows the individual frame after a Hamming window has been applied b) the resulting frequency spectrum after FFT c) an illustration of the spectral power of the frame.

3.3.4 Mel Scale Filtering

Mel-scale filtering uses a series of triangular filters which mimic the way humans perceive sound. The filtering scheme is based on a non-linear frequency scale called the Mel-scale, which is a psychoacoustic measure of pitch as judged by humans [6].

The first step in Mel-scale filtering is to assign each frequency of the power spectrum into sets of weighted bins on the Mel scale. This is done by taking the smallest and largest frequencies in the spectrum and converting them to the Mel scale, followed by segmenting the converted frequency range into a desired number of equally spaced segments called bins, also referred to as Mel filters. To transform a frequency f into the Mel-scale, the following formula shown in Equation 3.7 is used.

$$m = 2595log(1 + \frac{f}{700}) \tag{3.7}$$

The Mel-scale is logarithmic which means that equally sized bin widths in the Mel-scale, once converted back into the frequency domain, yield actual bin widths that increase proportional to the frequency.

To convert from the Mel-scale back to the frequency domain, Equation 3.8 is used.

$$f = 700(10^{\frac{m}{2595}} - 1) \tag{3.8}$$

Assuming that a total of N_q Mel filters are desired to be used, this gives rise to exactly $(N_q + 2)$ bin boundaries which demarcate the extent of each Mel bin. The boundaries can be denoted as $\{d_0, d_1, \ldots d_{N_q}, d_{N_q+1}\}$. Accordingly, the following piece-wise defined function $MF_q(f)$ defines the N_q Mel filters:

$$MF_q(f) = \begin{cases} 0 & \text{if} \quad f < d_{q-1} \\ \dfrac{f - d_{q-1}}{d_m - d_{q-1}} & \text{if} \quad d_{q-1} \leq f < d_q \\ \dfrac{d_{q+1} - f}{d_{q+1} - d_m} & \text{if} \quad d_m < f \leq d_{q+1} \\ 0 & \text{if} \quad f > d_{q+1} \end{cases} \tag{3.9}$$

The image in Figure 3.9 illustrates the resulting filterbanks when using $N_q = 10$ filters,

and it can be observed that the width of the filters increase as the frequency of the spectrum increases to the right. Practically speaking, a typical value of $N_q = 26$ is used in most applications.

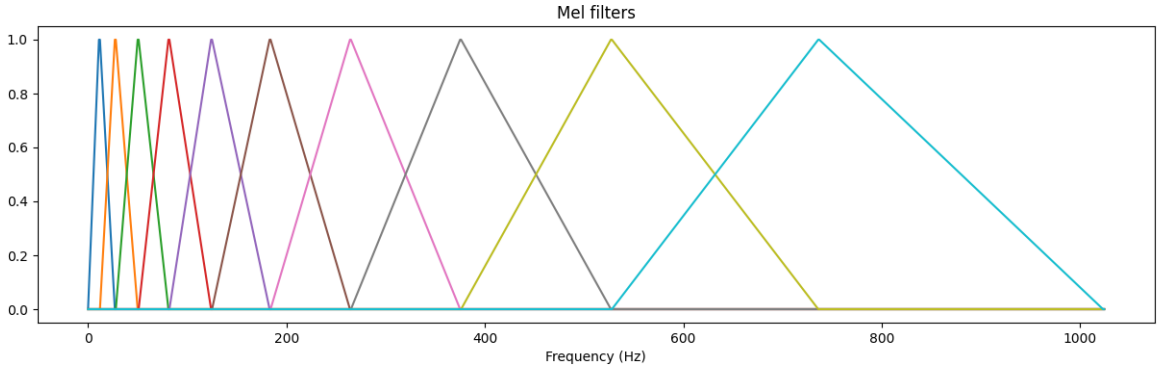


FIGURE 3.9: Illustration of Mel filterbanks where $N_q = 10$.

3.3.5 Logarithmic function

Each calculated filter is then applied to the power spectral energy $S_{j,f}^{(4)}$ of each frame j produced in Equation 3.6 and the result is compressed to a logarithmic scale as shown in Equation 3.10. This has the effect of compressing the filter energy of each filter, thereby obtaining the perceived loudness of each filter on a logarithmic scale.

$$S_{j,q}^{(5)} = \log\Big[\sum_{f=0}^{f} {}_m S_{j,f}^{(4)} \cdot MF_q(f)\Big] \quad \forall \quad q \in \{1, \ldots, N_q\} \tag{3.10}$$

It is once again useful to note the change of indexing from f in $S^{(4)}$ to q in $S^{(5)}$, indicating that the result corresponds to Mel filters N_q.

3.3.6 Discrete Cosine Transform

The next step is to apply the Discrete Cosine Transform (DCT) to each log-filter energy to obtain a spectrum of the log-filter spectrum i.e. a spectrum of the spectrum, referred to as "cepstrum". Equation 3.11 summarizes this process.

$$S_{j,c}^{(6)} = \sum_{q=1}^{N_q} S_{j,q}^{(5)} \cos[\frac{c(2q-1)\pi}{2N_q}], \quad \forall \quad c \in \{0, 1, 2, \ldots, N_q\} \tag{3.11}$$

where c is the cepstrum index. It should be noted that the newly formed $S_{j,c}^{(6)}$ comprises of a series of cepstrum values or coefficients, which are the cepstral coefficients (CCs) in the MFCC feature descriptor. In practice, a value of $N_q = 26$ yields 26 CCs, and of these only the first 13 are used, since the second set of 13 CCs represent extremely fast-paced changes in the underlying audio signal which are not considered to be useful.

3.3.7 Delta Features

The previously calculated CCs describe only the power spectrum within each individual frame. It has been found that speech audio also contains the trajectories of the CCs over time i.e. across frames. Therefore, a means of computing a representation of these trajectories and appending them to the previously calculated CCs provides a feature vector containing more information that is potentially useful. One representation of the dynamic changes between frames are the first and second derivatives denoted as $\Delta S_{j,c}^{(6)}$ and $\Delta^2 S_{j,c}^{(6)}$, respectively. Assuming a large enough sampling rate, these derivatives can be approximated by:

$$\Delta S_{j,c}^{(6)} = S_{j+1,c}^{(6)} - S_{j-1,c}^{(6)}, \quad \Delta^2 S_{j,c}^{(6)} = \Delta S_{j+1,c}^{(6)} - \Delta S_{j-1,c}^{(6)} \tag{3.12}$$

The final feature vector V_j for a given frame j is then constructed by concatenating the CCs and the first and second derivatives as shown below:

$$\vec{V_j} = [S_{j,c}^{(6)},\ \Delta S_{j,c}^{(6)},\ \Delta^2 S_{j,c}^{(6)}] \ \ \forall \ \ \ c \in \{0, 1, 2, \ldots, N_q\} \tag{3.13}$$

Finally, the feature vector for the entire original audio sequence $\vec{V}$ is constructed by concatenating the feature vectors $\vec{V_j}$ for all frames j as follows:

$$\vec{V} = [\vec{V_j}] \ \ \forall \ \ \ j \tag{3.14}$$

3.4 Classification Using Support Vector Machines

Support Vector Machines (SVMs) are a classification algorithm which aim to determine a decision boundary with the greatest margin between points of two classes in a given feature space. Figure 3.10 shows a basic binary linear support vector classification problem. Assuming that the black and white circles in the figure represent data samples

expressed in terms of two arbitrary features X_1 and X_2, and that the samples represent two target classes—black and white—belonging to a positive class and negative class, respectively. There are numerous potential decision boundaries that can be used to separate the points of the two classes. Three example decision boundaries have been depicted in the figure and denoted as lines H_1, H_2 and H_3. There are two important factors to consider when evaluating decision boundaries. The first is that some decision boundaries may effectively separate samples of the two classes, such as H_2 and H_3, while other may not, such as H_1. The second factor to consider with respect to decision boundaries is that the degree of separation achieved by some boundaries may be better, e.g. H_3, than others, e.g. H_2.

It can be seen that H_3 separates the points of the two classes more effectively than H_2. In this example, H_3 is, in fact, known as the maximum margin classifier which because it separates the two classes while ensuring the maximum possible distance between points of the two classes. The premise of SVMs is to achieve this maximum margin decision boundary. To achieve this, SVMs determine the data samples of either class that are closest to the opposing class, known as support vectors, and draw a boundary that maximally separates these points, hence the name Support Vector Machine. The support vectors are therefore defined as the the subset of data points in each class which assist in the definition of the maximum margin classifier.

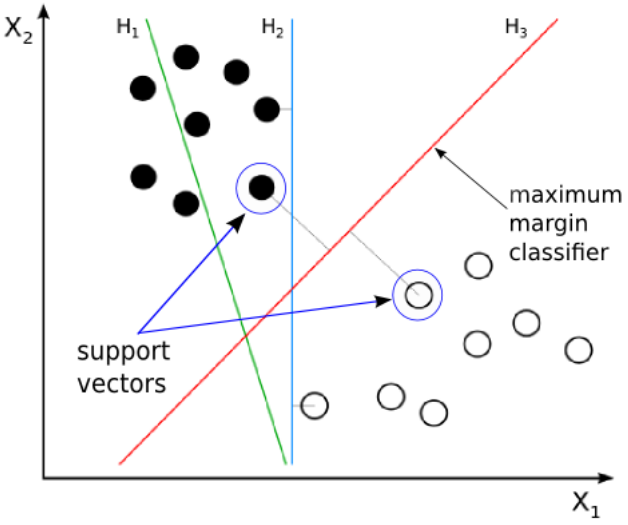


FIGURE 3.10: Example of linear SVM classification

The example in Figure 3.10 is a simple linear classification problem as the points in the two classes are linearly separable. In a real world example, data is not always readily linearly separable. One of the strengths of SVMs is the ability to readily map the data onto a higher-dimensional feature space in which data which is normally linearly

inseparable becomes separable. This is done by means of SVM kernels. The four most common kernel functions are defined as:

Linear Kernel:

$$k(x, y) = (x^T y) \tag{3.15}$$

Polynomial Kernel:

$$k(x, y) = (\gamma X^T y + b)^d, \gamma > 0 \tag{3.16}$$

Radial Basis Function (RBF) or Gaussian kernel:

$$k(x, y) = exp(-\gamma ||x - y||^2), \gamma > 0 \tag{3.17}$$

Sigmoid Kernel:

$$k(x, y) = tanh(\gamma(x^T y) + b), \gamma > 0 \tag{3.18}$$

In the above equations, γ and d are kernel parameters and the performance of SVMs depend on utilizing an appropriate kernel and corresponding parameters. In previous works, the RBF kernel was shown to outperform the linear kernel, and it is generally better than the sigmoid kernel [14]. The RBF kernel is comparable to the polynomial kernel but because of the optimization complexity of the polynomial kernel, the RBF kernel is the optimum choice for this research, because of its performance combined with its efficiency.

3.4.1 Optimization

Not all datasets are created equal, therefore parameter optimization is required to acclimatize the classifier to the dataset being used to ensure the best results. The optimization method in this research is grid search. Grid search is used to optimize RBF kernel parameters γ and C using cross validation to find the best hyper-parameter combination that provides the best training cross-validation accuracy.

K-fold cross validation is done by splitting the dataset into K subsets. Iteratively, one subset is used as a test set and it is then tested against the classifier training on the data of the remaining $(K - 1)$ training sets. A final cross-validation accuracy across all K subsets is then obtained by taking an average.

Combining the concepts of grid search optimization and K-fold cross validation, the optimization process then involves determining cross-validation accuracies for a range of γ and C pairs on a grid and determining the pair with the highest cross validation accuracy which is selected for training the final SVM classifier model. A typical optimization parameter range for γ and C is $C \in (2^{-5}, 2^{-3}, ..., 2^{15})$ and $\gamma \in (2^{-15}, 2^{-13}, ..., 2^{3})$, which is the range used in this research. In terms of K-fold cross-validation, in this research K=10 was used.

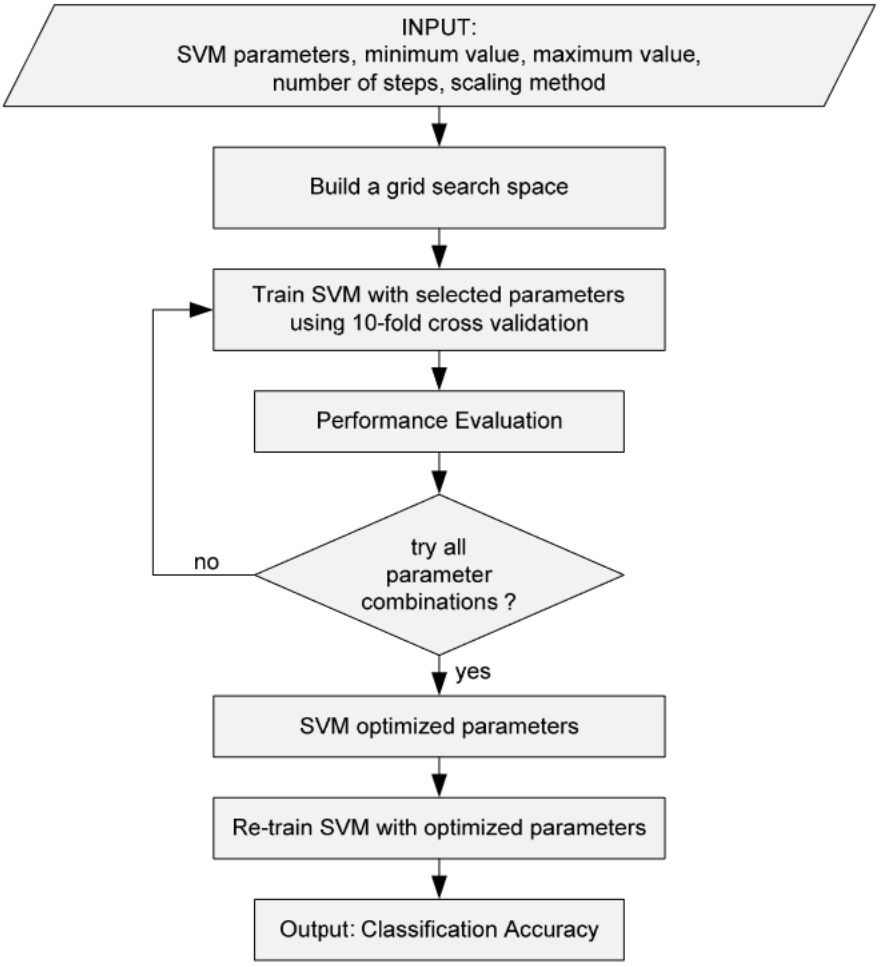


FIGURE 3.11: Example of grid search to find optimal training parameters as explained by Syarif *et al.* [24].

Figure 3.11 shows the process of grid search optimization and at the end of the process, the optimal γ and C are obtained and can then be used to train the final model. Grid search optimization is generally beneficial and is mostly capable of locating good optimum parameters. The technique is however weighed down by processing speed as it is resource intensive. When deployed on a CPU, the procedure can be extremely time-consuming. However, with the introduction of the LIBSVM-GPU library [2], the speed of the procedure is increased considerably by leveraging parallel processing on Nvidia

GPUs. The improvement is visually illustrated in Figure 3.12 in which it can be seen that the CPU version of LIBSVM takes exponentially longer to process as the number of training samples increases, whereas the GPU version experiences a slight linear increase in time, due to the ability to parallel process samples simultaneously. With the increased optimization performance in grid search cross validation when using a GPU, iterating across parameter pairs becomes easier, which makes it possible to explore a larger number of options and find a more optimal set of parameters in a reasonable amount of time.

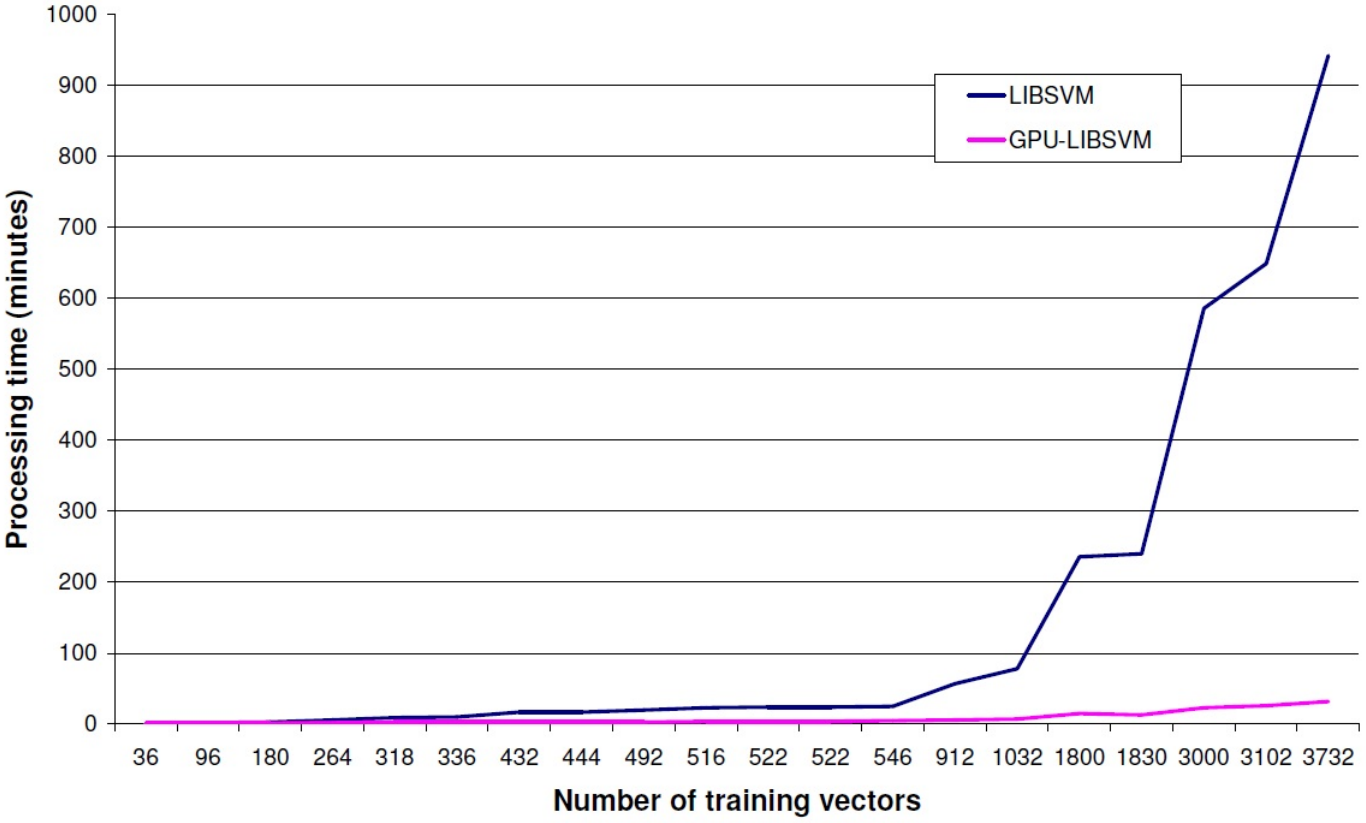


FIGURE 3.12: Performance comparison between standard LIBSVM and LIBSVM-GPU as illustrated by Athanasopoulos *et al.* [2]

3.5 Summary

This chapter explained the different techniques that are utilised in this research to be able to segment audio signals from subjects writing multiple Roman alphabet letters at a time, extract features from each segmented letter and classify each of them.

Chapter 4

Design and Implementation

This chapter details the steps that were taken to implement the system in this research to meet the research objectives set out in Chapter 1. The goal of this chapter is to assist the reader in understanding the steps that were taken with the possibility of replicating the results that were obtained in the following chapter.

The rest of the chapter is organized as follows: Section 4.1 describes the setup used to capture audio, including the recording device, the recording application and the environment used to collect data for experimentation; Section 4.2 discusses the procedures used to process the audio, including segmenting audio signals into constituent words, followed by the feature extraction; Section 4.3 describes the dataset collected and the process of training and optimizing the SVM classifier on the dataset; the chapter is then concluded.

4.1 Audio Capture

The audio capture task in the proposed system is carried out by means of a smartphone. Most modern smartphones have 2 microphones—one on the bottom, and another on the top—to remove ambient noise from the signal. The following subsections provide further detail on the audio capture of the proposed system.

4.1.1 Recording Device

For the purposes of this research, a Samsung Galaxy Note 10 Plus smartphone was selected and utilised as the recording device. Along with the recording device, another important factor is the writing location. Referring to Figure 4.1, the writing location for

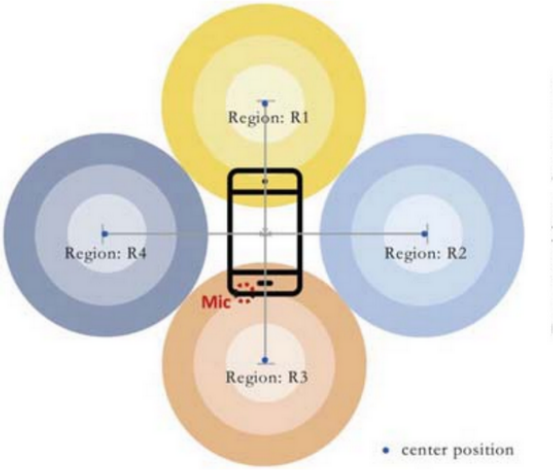


FIGURE 4.1: Various possible writing locations, adapted from Chen *et al.* [5]

this research was on the right-hand side of the smartphone, indicated as Region "R2" in the figure. As explained in Chapter 2, Chen et al. found that this location is suitable for this task. Furthermore, this location facilitated the writing task because all of the test subjects were right-handed, which led to a more natural writing position relative to the position of the smartphone.

4.1.2 Recording Application

To complete research objective 1 set out in Chapter 1, a mechanism to utilise the microphones on the smartphone for further processing by the proposed system needed to be developed. To be able to do this effectively, the default recording app on a smartphone would not be sufficient as it is a stand-alone app that can not be easily integrated into the proposed system. Recording with this app would also result in recording latency making it generally unsuitable. For this reason, a web app was developed which can be accessed from the mobile browser on the smartphone. A screenshot of the web app is provided in Figure 4.2.

The web app consists of an HTML web page, with a JavaScript recording framework which requests access to the microphone of the device on which the web app is running. To process the audio data, there is a Flask Python application hosted using Amazon Web Services (AWS) Lambda with AWS API Gateway to allow *.wav* files to be uploaded to AWS S3 which is a cloud storage system. Once the data file is uploaded to AWS S3, an AWS Lambda function is triggered which takes the audio recording and splits it into the individual letters, and these are then saved into a folder labelled with the appropriate letter. This is how the data collection was conducted in an efficient manner. The data collection and pre-processing process can be seen in Figure 4.3.

FIGURE 4.2: Screenshot of the recording application created for the purposes of this research.

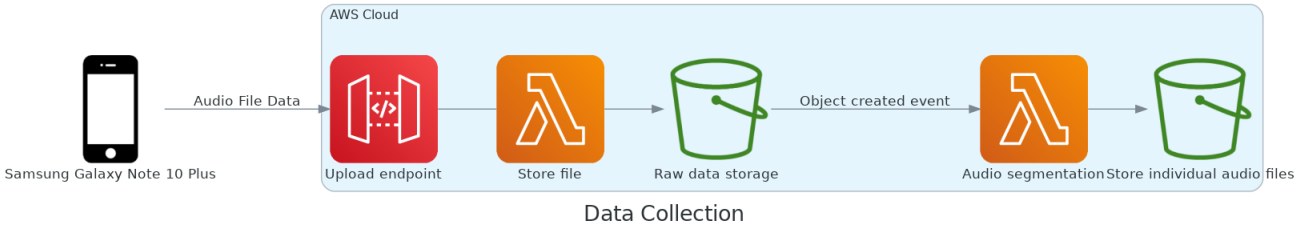


FIGURE 4.3: Data collection process used in this research.

Having completed the development of this application, research objective 1 outlined in Chapter 1 was successfully achieved.

4.1.3 Environment, Surface and Recording Implement

This subsection describes the conditions under which the datasets for this research were collected. The first aspect of this is the environment in which the data was collected. The main factors for choosing an environment to collect data were ease of access and accessibility to test subjects. Therefore the most logical place to collect data would be the workplace of this researcher. The meeting room at the researcher's workplace, seen in Figure 4.4, was used. The room is quiet and large with curtains against the walls which absorb a small amount of the outside noise. It is important to note that the room was not completely insulated from outside noise in order to keep the conditions as natural to a real-world usage as possible.

FIGURE 4.4: Environment used for data collection.

Section 2.2.2 in Chapter 2 discussed the fact that the writing surface has an impact on the outcome of the results and it was illustrated that the surface with the best results is one that provides the greatest decibel level from writing on its surface. A close-up image of the writing surface used in this research—the meeting room table—can be seen in Figure 4.5.

The writing implement used by all the test subjects was required to give off an audible sound signature which can be adequately received by the smartphone microphones. Therefore a wooden chopstick as shown in Figure 4.6 was used and the test subjects would use the back of this chopstick to write. The back of the chopstick emits a deeper more audible sound which is not as sharp as the thinner tip.

FIGURE 4.5: Close-up of the writing surface used in data collection.

FIGURE 4.6: Writing implement used by the test subjects in data collection.

The conditions under which each test subject was required to record data were setup to ensure a consistent recording process, but there were some shortcomings. Multiple attempts were required to allow the subjects to correctly and accurately write each letter multiple times in the correct stroke pattern without stopping the recording.

4.2 Audio Processing

Audio processing is the next step in the process and it is arguably the most important step once the dataset has been collected. This section will go into the implementation details of the each audio processing technique explained in Chapter 3.

4.2.1 Audio Segmentation

The process of audio segmentation is essential because of the continuous recording method that is implemented in this research. Different people have different speeds of writing. An example of a signal captured using the system can be seen in Figure 4.7 which consists of 10 'B's written consecutively with brief gaps between them.

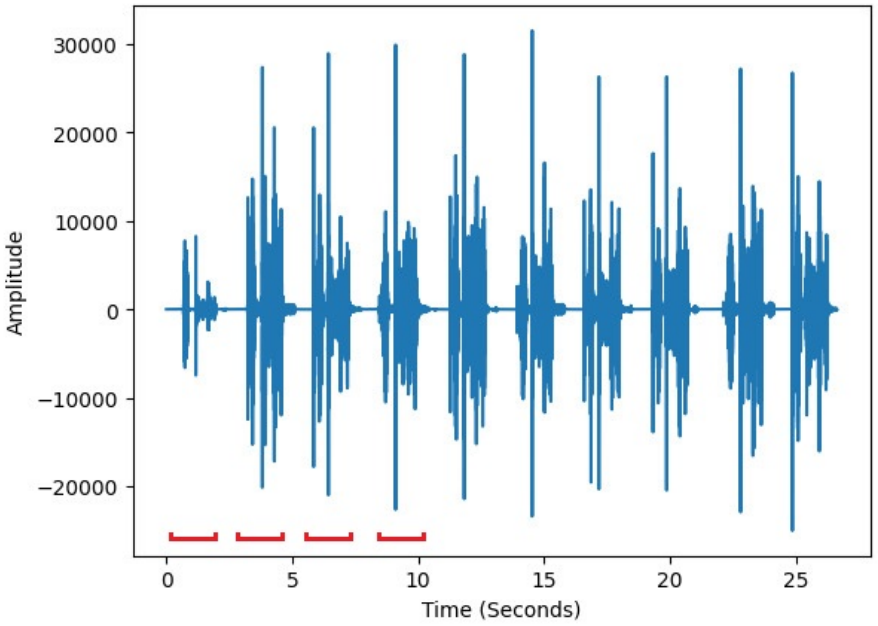


FIGURE 4.7: Example of 10 'B's writing by a user as shown previously in Figure 3.2

The audio segmentation is conducted using the algorithm explained in Section 3.2. To accurately extract each individual character from a single audio signal, writers were required to observe a brief pause between letters. This allows the audio segmentation algorithm to be able to differentiate between silence between strokes and silence between letters. To detect silence, a threshold value $T_l = -16$ dBFS, minimum silence length $L = 1000$ ms, and step size $s_L = 1$ ms, were used. The output of the segmentation is depicted in Figure 4.8.

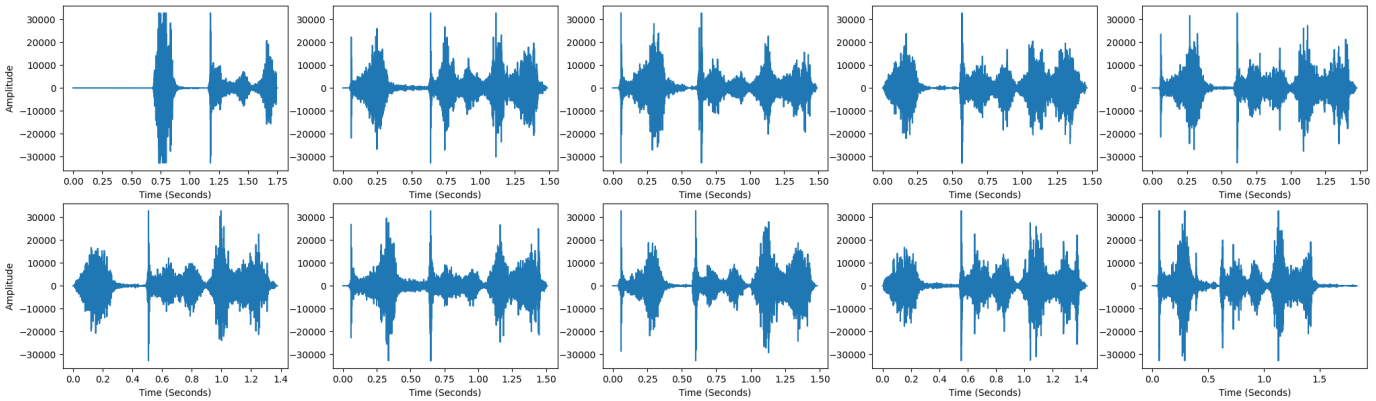


FIGURE 4.8: Each individual 'B' extracted from the signal in Figure 4.7.

The process of splitting the audio signal on silence is implemented upon data collection as explained earlier, and also in real-time during testing and in the final live system implementation steps to ensure consistency. Once a subject has completed writing a series of characters, the file is then automatically sent to a process which attempts to create individual audio files.

At this stage, it can be said that Objective 2 set out in Chapter 1 has been achieved with a high segmentation success.

4.2.2 Feature Extraction

The feature extraction portion of the audio processing procedure utilises the MFCC feature extraction process explained and outlined in the previous chapter. For this implementation of MFCC, an FFT size of $N_k = 2048$ was used. Furthermore, the number of Mel filters $N_q = 26$ was used. As an example, the resultant MFCC feature vector for an individual letter 'A' can be seen in Figure 4.9.

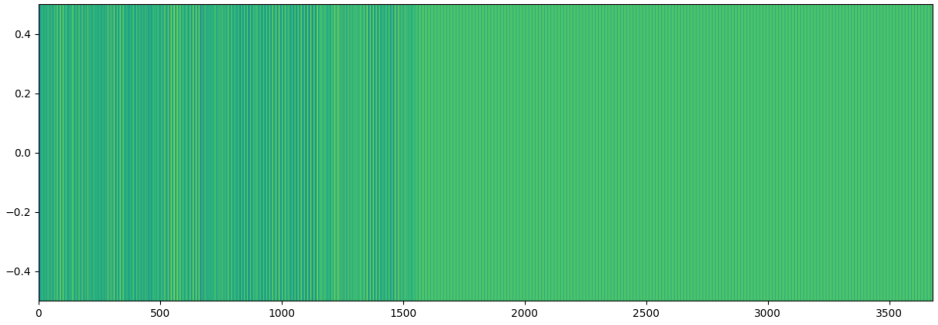


FIGURE 4.9: Resultant feature vector of MFCC feature extraction applied to a single 'A' character.

As such, at this stage Objective 3 set out in Chapter 1, to be able to successfully implement MFCC on segmented letters, has been achieved.

4.3 Classification

Once all of the MFCC feature vectors for the training dataset have been created, the next step is to begin classification using the SVM classifier. The following subsections describe the process and details of the dataset; optimization implementation; the training phase; and the testing phase of the system. A description of the dataset and how it was collected will be provided; the optimization method and any shortcomings therein; the results of the training phase; and finally the results of the initial testing phase which is designed to provide a benchmark for whether the resultant model is able to generalise and perform well.

4.3.1 Classes and Data

The objective of this research is to be able to recognize handwritten characters. In the case of this research the classes are defined as the 26 different handwritten English upppercase letters which are written by different users. In accordance with Premise 3 set out in Chapter 1, the stroke patterns of letters used by users to record the dataset are pre-defined and have been taken from the English Deaf-blind Alphabet Manual, which is used to write English characters on a deaf-blind person's hand [8]. The stroke patterns can be seen in Figure 4.10.

FIGURE 4.10: Deaf-blind stroke orderings required by users in data collection and testing.

It can be observed in Figure 4.10 that there are groups of letters which are drawn similarly and can cause complexity in classification. One example are the letters 'I' and 'J' which are drawn in a single stroke, with the only difference being a curve at the end of 'J'.

4.3.2 Training and Testing Datasets

Initially, a dataset consisting of 5 different subjects (Subjects 1–5) was collected. Each subject was required to write each of the 26 letters of the alphabet 10 times i.e. 5 subjects × 26 letters × 10 samples = 1300 samples in total initially.

It should be noted that the 10 samples per letter written by each subject were written continuously in a single recording and then automatically segmented into 10 constituent letters as explained before. With the 10-letter batches, a limitation was uncovered with the first and last letter of each recording. The limitation of the first letter is the possibility that the subject started writing before the recording had actually started thus possibly resulting in the first letter being only partially recorded. The limitation of the last letter is the possibility that the subject made an extra noise or pushed the phone when attempting to stop the recording possibly resulting in added background noise and/or a deformed audio signature of the last letter. To counter these possible limitations, the first and last letter of each 10-batch recording were removed thus resulting in a total of 5 subjects × 26 letters × 8 samples = 1040 samples.

The dataset was then split into 2 groups; 5 of the 8 samples of each of the 5 subjects per letter were used to train the classifier i.e. 5 subjects × 26 letters × 5 samples = 650 recordings for training. The remaining 3 samples per subject per letter were used as a testing set. As explained in Chapter 1, this testing set can be considered to be a "semi-seen" testing set since the samples are completely unseen to the classifier, but other samples of the same test subjects and letters have been seen by the classifier during training. The results on the semi-seen testing set provides an indication of the ability of the classifier to recognize characters drawn by a user after carrying out a once-off pre-training procedure to allow the system to learn the user's drawing patterns.

An additional dataset consisting of 3 completely new test subjects (Subjects 6–8) was collected. The collection of the data was done in the same way as with the training / semi-seen dataset i.e. the subjects were required to write each of the 26 characters 10 times, and the first and last character of each recording were ignored. This resulted in a new dataset with 3 subjects × 26 letters × 8 samples = 624 audio recordings. This dataset was used entirely as an unseen dataset to test the robustness of the resultant classifier to completely new and unseen users without any pre-training procedure. This is summarized below:

1. Training data: 26 classes × 5 subjects × 5 recordings per class = 650 samples.
2. Semi-seen data: 26 classes × 5 subjects × 3 recordings per class = 390 samples.

Dataset	Subjects	Classes	Samples	Total
Training	1–5	26	1–5	650
Semi-Seen	1–5	26	6–8	390
Unseen	6–8	26	1–8	624

TABLE 4.1: Training, semi-seen and unseen datasets.

3. Unseen data: 26 classes × 3 subjects × 8 recordings per class = 624 samples.

The 3 datasets are summarized in Table 4.1.

At this stage, Objective 4 set out in Chapter 1, to collect a dataset of uppercase handwritten characters, can be considered to be achieved.

4.3.3 Optimization

The optimization step is required to find the best parameters, C and γ, for the RBF kernel. Optimization was carried using 10-fold cross validation via grid search, as explained in Section 3.4.1 in the previous Chapter. Optimization was carried out on an Nvidia Geforce GTX 1080ti GPU with 11GB VRAM at a frequency of 1582MHz.

The optimization method was to use the default parameter optimization range for grid search i.e. $C \in \{2^{-5}, 2^{-3}, 2^{-1}, \ldots, 2^{11}, 2^{13}, 2^{15}\}$ and $\gamma = \{2^{-15}, 2^{-13}, 2^{-11}, \ldots, 2^{-1}, 2^{1}, 2^{3}\}$. The initial grid search optimization yielded a contour plot which can be seen in Figure 4.11. Upon observation, it was seen that highest-performing parameter pairs were saturated on the bottom-right of the graph. It is was therefore decided to adjust the ranges of both parameters to explore combinations extending downwards and to the right where a possibly greater set of optimal results may be found. Therefore, the C and γ parameter search range was adjusted to $C \in \{2^{0}, 2^{1}, 2^{2}, \ldots, 2^{11}, 2^{12}, 2^{13}\}$ and $\gamma = \{2^{-20}, 2^{-19}, 2^{-18}, \ldots, 2^{-7}, 2^{-6}, 2^{-5}\}$, and this led to obtaining the optimization contour plot which can be seen in Figure 4.12.

For completeness, the results of the grid search parameter optimization can be seen in Table 4.2, which shows the C, γ pairs and their respective cross validation accuracy. Ultimately, the optimal parameter pair obtained with the optimization procedure were $C = 2^{5}$ and $\gamma = 2^{-13}$ with a cross validation accuracy of 83.2%. Observing Table 4.2, it can be seen that a number of parameter pairs corresponding to $\log_2 \gamma = -13$ and $\log_2 C \geq 5$ provide the optimal cross validation accuracy of 83.2%. The pair with the smallest value of C was selected as this represents the parameter with the most regularisation, ensuring a sufficiently complex underlying model.

$\log_2 C$	$\log_2 \gamma$																			
	-20	-19	-18	-17	-16	-15	-14	-13	-12	-11	-10	-9	-8	-7	-6	-5
0	31.1	31.1	31.2	31.2	31.4	31.2	33.7	42.3	54.3	65.4	68.6	66.2	56.3	44.6	25.4	8.0
1	31.1	31.1	31.2	31.2	32.0	34.3	44.0	59.1	72.0	76.2	76.2	70.0	58.8	46.0	29.4	8.3
2	31.1	31.1	31.2	31.7	34.5	45.5	60.5	74.3	78.6	81.2	77.1	70.0	58.8	46.0	29.4	8.3
3	31.1	31.2	31.7	34.3	46.0	61.2	74.9	79.4	82.3	81.4	77.1	70.0	58.8	46.0	29.4	8.3
4	31.2	31.7	34.2	46.6	62.3	75.5	79.8	82.9	82.5	81.4	77.1	70.0	58.8	46.0	29.4	8.3
5	31.7	34.0	46.5	62.3	75.4	80.0	82.8	**83.2**	82.5	81.4	77.1	70.0	58.8	46.0	29.4	8.3
6	34.2	46.5	62.5	74.8	79.8	82.8	83.1	83.2	82.5	81.4	77.1	70.0	58.8	46.0	29.4	8.3
7	46.6	62.5	74.9	79.5	82.3	83.1	83.1	83.2	82.5	81.4	77.1	70.0	58.8	46.0	29.4	8.3
8	62.5	74.8	79.7	81.7	82.5	83.1	83.1	83.2	82.5	81.4	77.1	70.0	58.8	46.0	29.4	8.3
9	74.8	79.5	81.7	82.2	82.5	83.1	83.1	83.2	82.5	81.4	77.1	70.0	58.8	46.0	29.4	8.3
10	79.5	81.5	82.3	82.2	82.5	83.1	83.1	83.2	82.5	81.4	77.1	70.0	58.8	46.0	29.4	8.3
11	81.5	82.2	82.3	82.2	82.5	83.1	83.1	83.2	82.5	81.4	77.1	70.0	58.8	46.0	29.4	8.3
12	82.2	82.2	82.3	82.2	82.5	83.1	83.1	83.2	82.5	81.4	77.1	70.0	58.8	46.0	29.4	8.3
13	82.2	82.2	82.3	82.2	82.5	83.1	83.1	83.2	82.5	81.4	77.1	70.0	58.8	46.0	29.4	8.3

TABLE 4.2: Cross-validation accuracy of each C and γ parameter pair evaluated in the grid-search optimisation procedure.

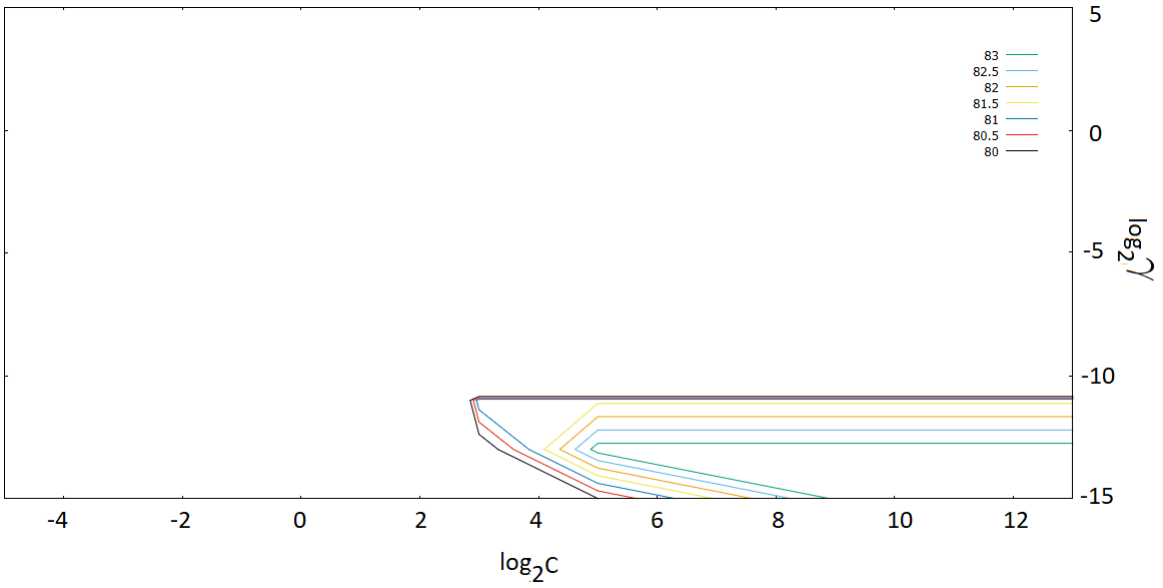


FIGURE 4.11: Original contour plot from grid-search utilising the conventional C and γ optimization range.

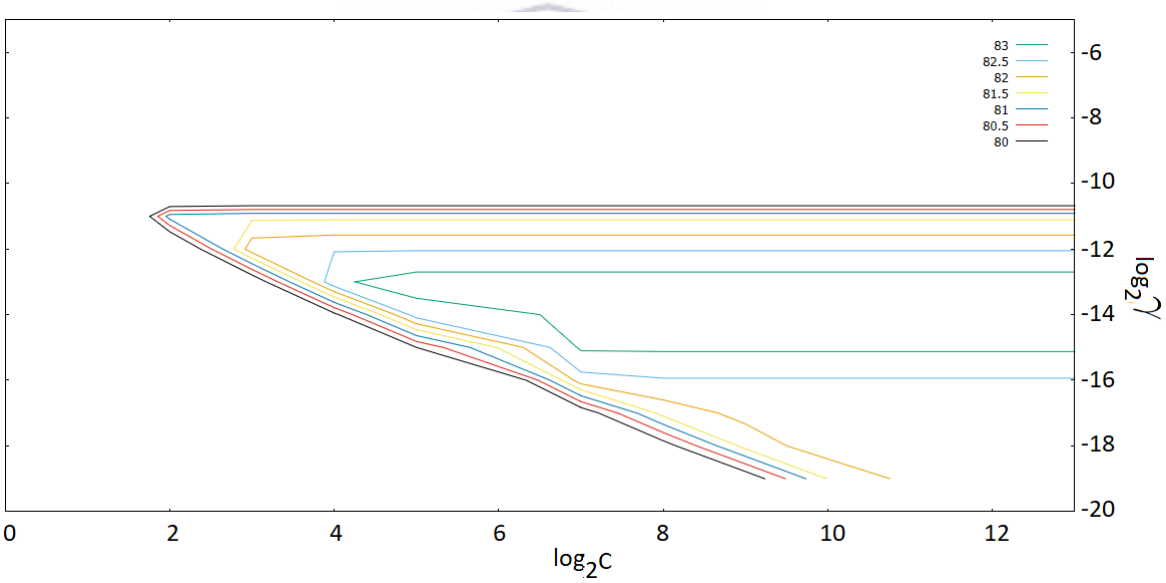


FIGURE 4.12: Contour plot from grid-search utilising the adjusted range of C and γ.

Accordingly, Objective 5 has been met at this stage.

4.4 Model Hosting

One additional benefit of the system implemented in this research is that once the model has been generated, it is placed on the cloud to make it accessible by the smartphone.

In the interest of keeping costs down, a low cost solution was implemented which incorporated storing the raw model file on Amazon S3 and retrieve it on demand when a request is sent to the web server with an audio file ready to be classified. With this approach the model is available to anyone that wishes to classify audio signals, although this has not been tested on smartphones other than the one selected for use in this research. A shortcoming to this approach is that the latency for the request to be sent to the server and back to the smartphone is around 5–10 seconds, which can be improved in future works.

4.5 Summary

This chapter described the implementation of the proposed audio letter recognition system. This was done in a number of steps, including the method of capturing the input audio, the procedures used to segment the audio and extract features from it, the process of training the classifier, including a description of the datasets used for training and testing, and finally the setup used to host the system on the cloud. In so doing, Research Objectives 1–5 have been achieved.

The next chapter focuses on meeting the remaining objectives, thereby concluding the research.

Chapter 5

Experimental Results and Analysis

This chapter details the experiments carried out to assess key components of the proposed system, along with analysing the results obtained to be able to complete the remaining research objective 6, thereby answering the research sub-questions and the main research question set out Chapter 1.

All experiments were conducted using the Python programming language and environment on a PC running Windows 10, with the following hardware: Intel Core i7 8700k running at 3.70GHz with 16GB DRR4 3200MHz RAM and an Nvidia Geforce GTX 1080ti GPU with 11GB VRAM at a frequency of 1582MHz, noting that the GPU was not used during testing but rather during the optimization step during implementation explained in the previous chapter. Testing was conducted on the recording device mentioned previously, namely the Samsung Galaxy Note 10 plus, using the web application explained previously, utilizing the model hosted on the cloud. Results were then saved as the unseen test subjects were conducting the tests.

For ease of reference, Table 4.1 from the previous chapter has been repeated here in Table 5.1 since the table summarises the datasets that will be used throughout this chapter. These datasets have been explained in the previous chapter in detail. Similarly, the stroke patterns illustrated in Figure 4.10 have been repeated in Figure 5.1 for ease of reference.

The chapter is organised as follows: Section 5.1 outlines the results of the audio segmentation component of the research to determine whether the system is able to segment audio files consisting of multiple letters into their constituent letters; Section 5.2 provides the results of the letter recognition experimentation on the semi-seen and unseen

FIGURE 5.1: Deaf-blind stroke orderings required by users in training and testing.

datasets as explained in the previous chapter; Section 5.3 then compares the proposed system to related studies; the chapter is then concluded.

5.1 Audio Segmentation Results and Analysis

To be able to measure the performance of the audio segmentation, it is first required to outline the objective behind this component of the system in this research. Audio segmentation is required to allow the user to be able to write freely without having to start and stop the recording between letters in a word; where the audio segmentation will be tasked at subdividing the audio recording into the constituent letters.

Dataset	Subjects	Classes	Samples	Total
Training	1–5	26	1–5	650
Semi-Seen	1–5	26	6–8	390
Unseen	6–8	26	1–8	624

TABLE 5.1: Training, semi-seen and unseen datasets.

Letter	Segmentation Success Rate (%)		
	Subject 6	Subject 7	Subject 8
A	100	100	100
B	100	100	100
C	100	100	100
D	100	100	100
E	100	100	100
F	100	100	100
G	100	100	100
H	100	100	100
I	100	100	100
J	100	100	100
K	100	100	100
L	100	100	100
M	100	100	100
N	100	100	100
O	100	100	100
P	100	100	100
Q	100	100	100
R	100	100	100
S	100	100	100
T	100	100	100
U	100	100	100
V	100	100	100
W	100	100	100
X	100	100	100
Y	100	100	100
Z	100	100	100

TABLE 5.2: Results from audio segmentation per unseen test subject.

It was decided to configure the audio segmentation component to subdivide an audio recording approximately, and allow the classifier to learn and adapt to any variations in this regard as well.

Accordingly, for the purposes of this research, the success of the audio segmentation component will be determined by comparing the number of letters written by a subject, and the number of audio files that are generated from such a recording i.e. if a subject writes 10 letter A's, there should be a resulting 10 ".wav" files generated for processing and classification thus yielding an audio segmentation hit rate of 100%. This was done for all samples of Subjects 6–8 of the Unseen data set.

The results of the segmentation success rate are shown in Table 5.2 and it can be seen that the audio segmentation process achieved 100% success across all subjects in the experiment.

This means that data collection and final testing can take place efficiently as each audio

sample can be split up without any manual intervention, and without having to manually and frequently carry out stoppages between letters.

Accordingly, it can be stated that Objective 2 set out in Chapter 1 has been successfully achieved. In response to research sub-question 1, it can be stated that the proposed audio segmentation strategy is 100% effective at segmenting audio recordings into constituent letters.

5.2 Letter Recognition Results and Analysis

Letter recognition in this context can be described as the ability to pass the audio recording of a single letter, as segmented by the audio segmentation component, into the classifier model and return a prediction corresponding to the correct ground truth of that letter.

To determine what is deemed an acceptable recognition accuracy for both test datasets, it is crucial to first contextualise the classification problem in terms of the number of classes and the associated probability of correctly predicting a class. As such, it should be considered that the probability of correctly guessing a given letter—out of the total 26 letters—at random is given by $\frac{1}{26}$ which approximates to 4%. Statistically speaking, therefore, any result obtained above this percentage would result in the classifier being considered to be better than random guessing and therefore effective. However, for this research, a recognition accuracy of greater than 60% will be considered to be acceptable. This is significantly higher than the random guessing accuracy, but also means that the resultant classifier is correct the majority of the time.

The following subsections describe the results and corresponding analysis on the semi-seen set in Section 5.2.1 and the unseen set in Section 5.2.2. In each case, the results of the classifier will be evaluated by carrying out the following analyses in order:

- An analysis of the overall accuracy, precision, recall and f_1 score of the classifier, across all subjects and letters.
- An analysis of the accuracy per letter class in order to determine the robustness of the classifier to variations across classes.
- An analysis of the accuracy per subject in order to determine the robustness of the classifier to variations across test subjects.

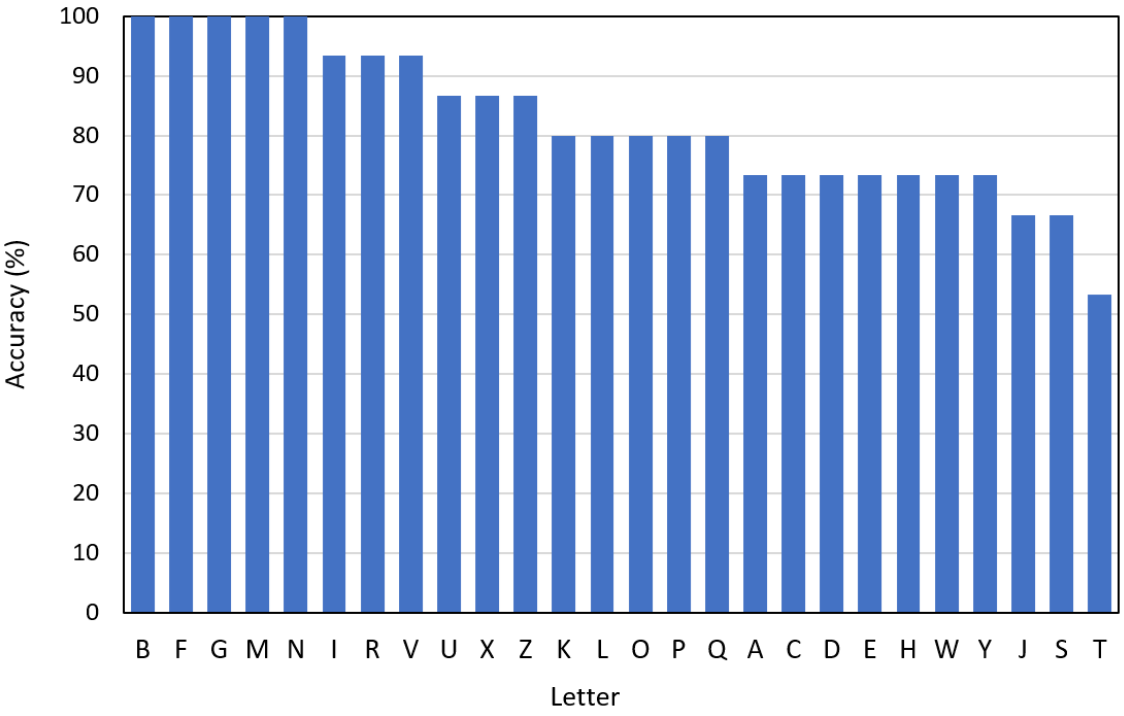


FIGURE 5.2: Results of semi-seen testing per letter across all test subjects, sorted in descending order of accuracy.

5.2.1 Semi-Seen Testing Results and Analysis

The semi-seen test set consists of unseen samples of subjects of whom other samples were used during training. Practically speaking, these results provide an indication of the ability of the classifier to recognize characters drawn by a user after carrying out a once-off pre-training procedure to allow the classifier to adapt to the user's drawing patterns.

The complete set of results obtained for every subject for every letter is provided in Table A.1 in Appendix A. The results have been provided in the form of the number of correctly recognised samples given that the number of samples per subject-letter entry was 3.

Overall, the results of the semi-seen testing for precision, recall and f_1 score were 83.6%, 82.3% and 82.9%, respectively, with an overall accuracy of 82.3%. These results are extremely encouraging, and significantly surpass the 60% criterion mentioned previously.

Figure 5.2 illustrates the accuracy of each letter across all test subjects, sorted in descending order of accuracy. It can be seen that: five letters achieve a perfect 100% accuracy, with a further three letters exceeding 90% accuracy; eight letters achieve between 70-80% accuracy; only three letters achieve below 70% accuracy. While there is variation across letters, overall, 23 of the 26 letters exceed 70% accuracy, which is an

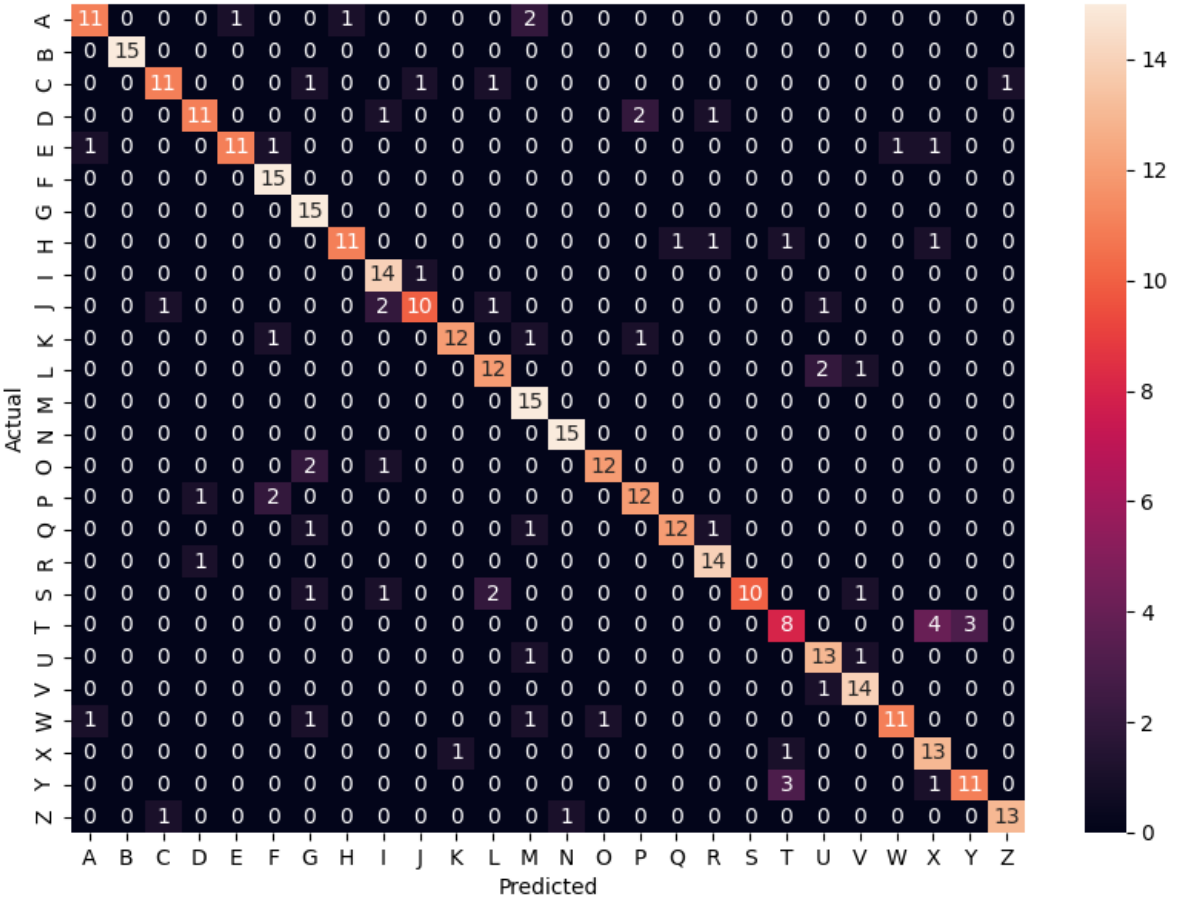


FIGURE 5.3: Confusion matrix of semi-seen testing results across all subjects.

excellent result. This indicates strong robustness to the letters in general. Only one letter drops below the 60% criterion which is the letter "T".

To analyse this further, Figure 5.3 provides a confusion matrix for the experiment. The main feature of note in the matrix is the observation that "T" was consistently incorrectly predicted as "X", and "Y" in all incorrectly predicted cases, 4 and 3 times respectively. This is not surprising given that the stroke patterns of the three letters are similar in direction as well as number of strokes as shown in Figure 5.1. This is further confirmed by the observation that "X" and "Y" are also confused with "T" in all of their misclassified samples. It is encouraging to note that this challenge only significantly affected one letter "T", which indicates that the system can still provide excellent overall usability.

Figure 5.4 depicts the average accuracy for each subject across all letters. It is very

encouraging to note that, on average, no outlier is observed and the accuracies across subjects are very close, with a range of only about 8%. The accuracies ranged from 85.9% for Subject 4 to 78.2% for Subject 2, all significantly higher than the criterion accuracy of 60% decided on earlier. This indicates strong robustness in the proposed system to variations in test subjects, given that pre-defined stroke patterns from the English deaf-blind Alphabet Manual [8] were used.

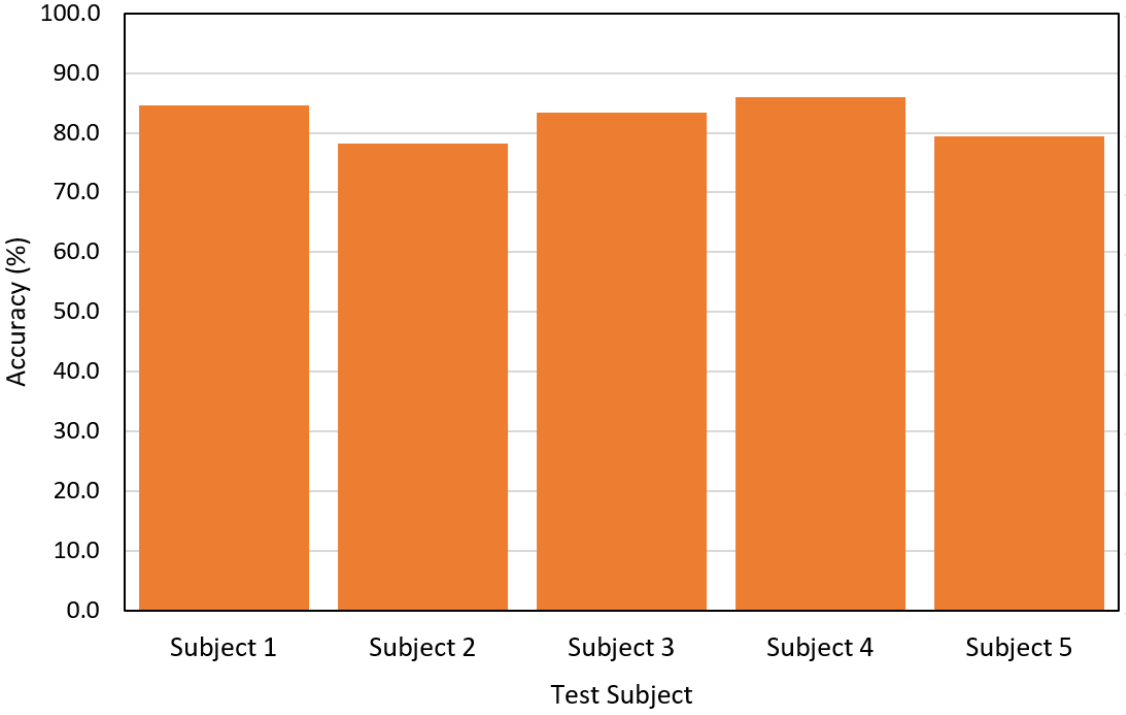


FIGURE 5.4: Results of semi-seen testing per subject across all classes.

As it stands, it can be stated that Research Objective 6 has been partially achieved because the tests so far have only been against semi-seen data. Furthermore, as a partial response to Research Sub-question 2, it can be stated that, if the user carries out a pre-training / handwriting registration procedure with the classifier, the proposed system can provide very high accuracy recognition that is robust to variations in test subjects and letters.

Furthermore, as a partial response to Research Sub-question 3, it can be stated that if the user carries out a pre-training / handwriting registration procedure with the classifier, with the feature extraction and classification techniques used, the microphones on a single smartphone are sufficient to achieve accurate segmentation and recognition of handwritten letters.

5.2.2 Unseen Testing Results and Analysis

Unseen testing is utilised to test how well the model can generalize to previously unseen conditions, in this case new subjects that were not used during model training. For this, the previously explained unseen dataset was used.

It should be noted that it was expected for this experiment to be challenging given the three factors explained in Chapter 1, namely, the limitation on the number of microphones, the fact that the medium for the propagation of sound in this case is through the air, and a lack of noise insulation. Therefore, while it was hoped for the system to perform well with unseen data/subjects, it was expected for this to be a challenge, and this question was a main subject of investigation in this study.

The complete set of results obtained for every subject for every letter is provided in Table A.2 in Appendix A in the form of the number of correctly recognised samples, given that the number of samples per subject-letter entry was 8. The same results have been provided as accuracies in Table A.3 in Appendix A.

The results of the unseen testing for precision, recall and f_1 score were 56.2%, 53.7%, and 54.9% respectively, with an overall accuracy of 53.7%. This accuracy falls slightly below the criterion of 60% set out earlier in the chapter. It should be noted that this was expected, as explained previously, but is still higher than the random-guessing accuracy previously discussed. As such, while it may not meet the self-made requirement of 60% in its current form, it can still be considered to be a promising result, given that the classifier still shows significant effectiveness, statistically speaking.

Figure 5.5 summarises the accuracy of each letter class per unseen test subject, sorted in descending order of accuracy. It can be observed in the graph that there is a wide variation in accuracy across letters in this case, and the reduction in average accuracy can be attributed to specific letters, rather than a deterioration in general. Specifically, it is observed that half of the letters achieved above the criterion accuracy of 60%, with "B", "Z", "G", "K" and "Y" achieving greater than 80%, and "I" achieving just above 90% accuracy. The second half of the letters were at or below 50% accuracy, with the letters with the lowest accuracies being "C", "O" and "V". It is quite clear that, without any pre-training, some letters are more easily recognisable to the classifier than others. It is important to note that the accuracy all of the letters are significantly higher than the statistical accuracy of random guessing, even if only half of the letters achieve an acceptable accuracy. This indicates that the system holds promise.

To analyse the factors behind the lower accuracy of some letters, a confusion matrix was created in a heat map format to show the relationship between the predicted and

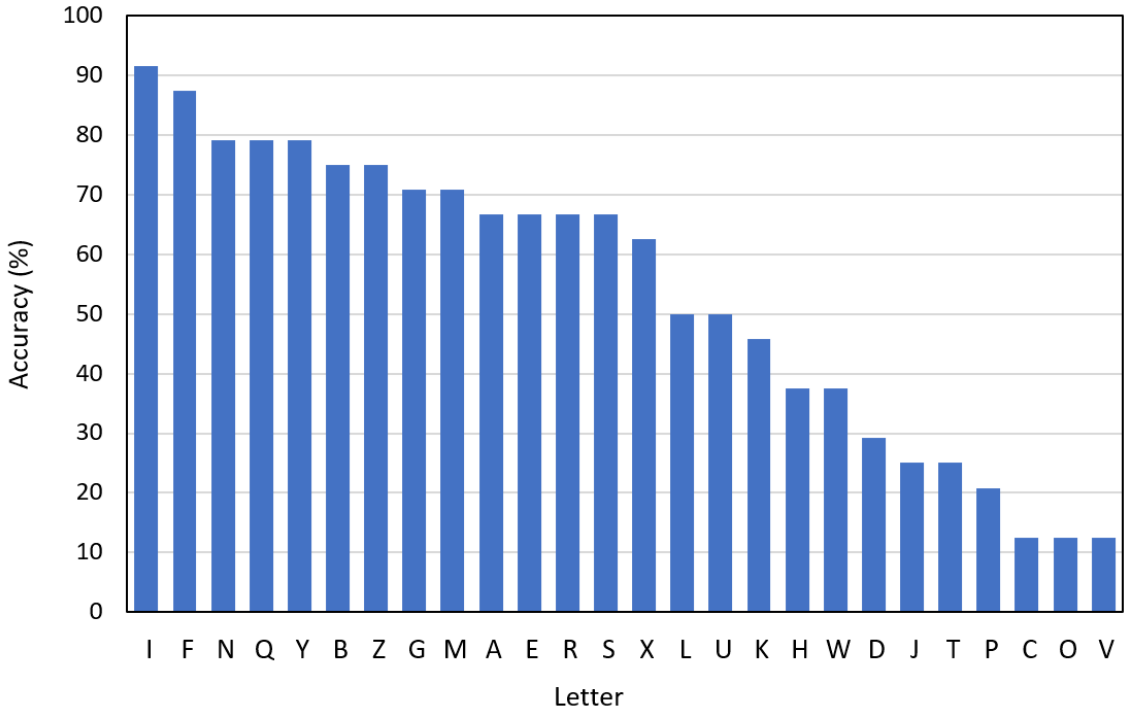


FIGURE 5.5: Results of unseen testing per letter across all test subjects, sorted in descending order of accuracy.

the true class labels. This confusion matrix can be seen in Figure 5.6. Observing the confusion matrix, it can be seen that, outside of the main diagonal, other entries (representing incorrect classifications) are generally either 0, low in count, or with a few entries lighting up brightly, representing cases in which a specific letter has been consistently—as opposed to randomly—confused with another letter. It is therefore clear from this confusion matrix that, rather than generally struggling to correctly classify some letters, the classifier displays consistency in confusing some letters with other letters.

To analyse this further, Table 5.3 summarises the letters which did not meet the acceptable recognition accuracy of 60% ("Letter"), along with letters they were most prominently confused with ("Prominent Confusions"), as well as the the number of strokes in the letter ("Strokes in Letter") and strokes in the corresponding letter(s) it was most prominently confused with ("Strokes in Prominent Confusions"). For all but two cases in the table, it can be seen that the number of strokes in the letter and in the letter(s) with which it was consistently confused with are the same.

This leads to the belief that one main reason for the confusion between these letters can be attributed to the similarity in their stroke counts e.g. "C", "S" and "Z" all share a single stroke pattern, as do "O" and "G" which follow a single stroke pattern along the same trajectories until the last segment with regards to "G". Many other examples can

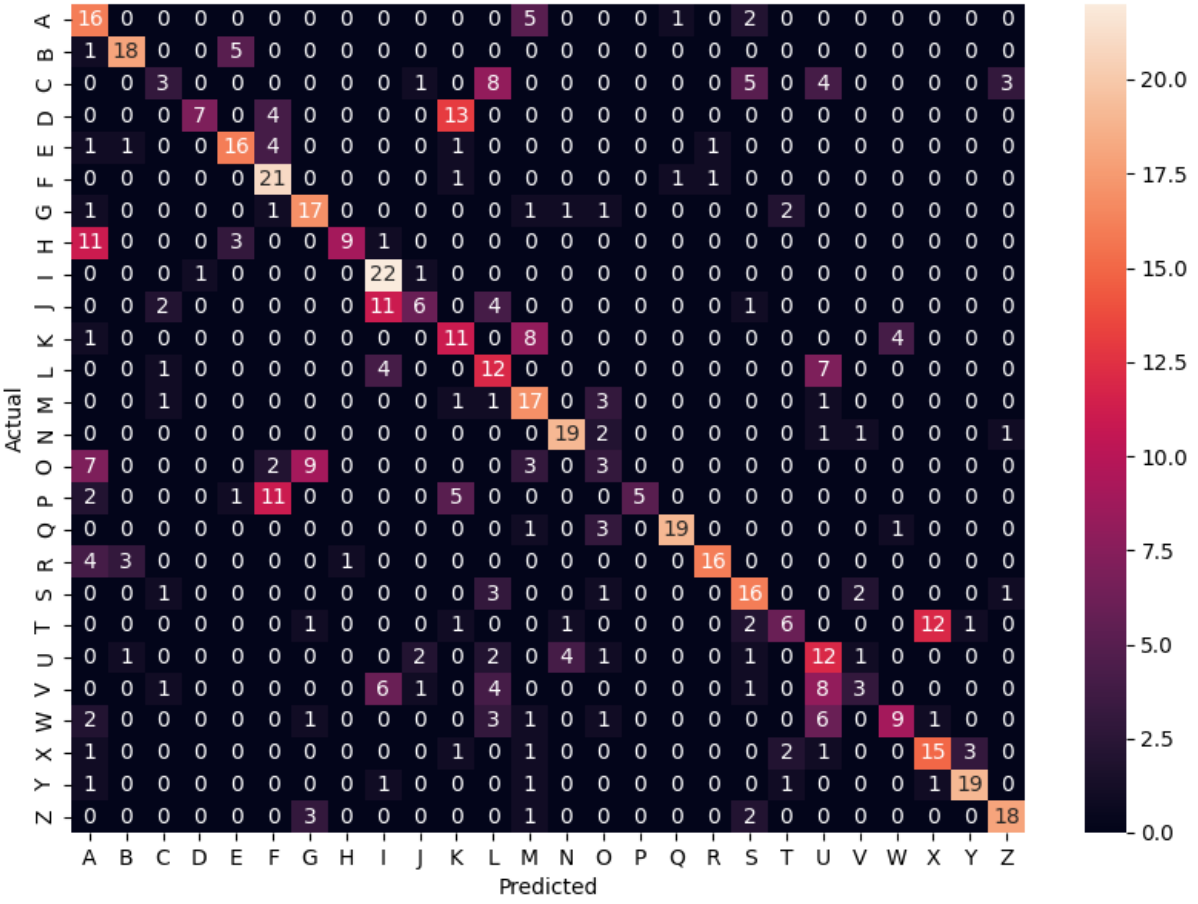


FIGURE 5.6: Confusion matrix across all subjects in unseen testing.

be seen in Table 5.3. It appears that, in the absence of pre-training on the test subjects' unique writing style, some letters with similar stroke counts and stroke patterns become difficult to predict by the system. These findings are similar to the findings of Li and Hammond [15] and Wu *et al.* [27, 28] in which letters with a similar number of strokes were commonly misclassified, especially under unseen conditions.

Another possible factor for the reduction in recognition accuracy for some letters can be attributed to the angle of the writing implement used (shown in Figure 4.6 in Chapter 4; because of the tapered shape, varying the angle of the implement while writing/drawing leads to variations in the pitch and depth of the acoustic signal produced. So, while the experiment on semi-seen data clearly showed that the classifier can learn these variations, it appears that new variations in this respect are not easily identifiable with a dataset of the size used in this research. It is likely that the use of a significantly larger dataset for training will help provide more robustness to such variations on unseen data, and

Letter	Prominent Confusions	Strokes in Letter	Strokes in Prominent Confusions
C	S, Z	1	1, 1
D	K	2	2
H	E	3	4
J	C, I	1	1, 1
L	C	1	1
O	A, G	1	2, 1
S	C	1	1
T	X	2	2
V	U	1	1
W	U	1	1
X	Y	2	2

TABLE 5.3: Letters in the unseen testing experiment which did not meet the acceptable recognition accuracy of 60% (left column), along with letters they were most confused with (right column).

this is a key area of future research in this regard.

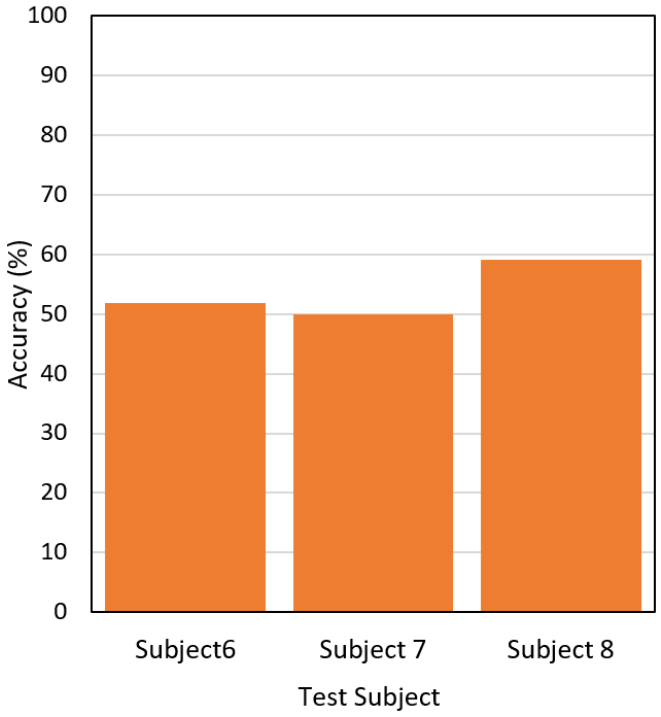


FIGURE 5.7: Results of unseen testing per subject across all classes.

The results of each individual test subject across all letters can be seen in Figure 5.7. The main feature of note in the figure is the fact that the average accuracies across unseen test subjects, while lower than those observed with the semi-seen dataset, were approximately consistent. The accuracies ranged from 59.1% for Subject 8 to 50.0% for Subject 6. It is quite encouraging to note that, despite the circumstances, the range in

accuracy across test subjects—approximately 9%—was low and was almost the same as that in the semi-seen testing. The fact that the range in accuracy in this case is almost identical to the range in the semi-seen experiment indicates robustness in the system. This strengthens the belief that, with a much larger training dataset, it is likely that the classifier will be able to learn subtle variations and better scale to unseen test subjects.

At this point, it can be stated that Research Objective 6 has been fully achieved. Furthermore, completing the previous partial response to Research Sub-question 2, it can be further stated that, if no pre-training / handwriting registration procedure is carried out: the proposed system can provide an accuracy that is above the criterion accuracy of 60% for half of the 26 letters, but lower than the criterion accuracy for the other half of the 26 letters, with a high level of robustness to variations in test subjects. Furthermore, is is likely that the use of a larger training dataset can help increase the accuracy and reduce the variation in accuracy across letters with unseen test subjects.

Furthermore, as a partial response to Research Sub-question 3, it can be stated that, with the feature extraction and classification techniques used, the microphones on a single smartphone may not be enough, where additional microphones perhaps from an additional smartphone may be instrumental in increasing the accuracy of the system on unseen data.

5.3 Comparison of the Proposed Approach to Related Studies

Table 5.4 was created to compare the proposed system to other implementations in the literature. For clarity, the different columns used in the table are defined as is, where 'Overall Accuracy' is defined as the final unseen accuracy results obtained by the study, and 'User Dependency' is the level of variation of the system accuracy across unseen test subjects.

From Table 5.4 that the proposed system did not out-perform any of the related studies in terms of the overall accuracy. As mentioned in the previous section, it is very likely that training on a larger dataset may improve the unseen accuracy of the proposed system, but this can be investigated in future. It can be seen that Yu *et al.*'s system [31] corresponds very closely to the proposed system in every category, and their system obtains an accuracy that is about 10% higher, and can be considered to be approximately in the same range as the proposed system.

Other systems far surpassed the proposed system, as well as Yu *et al.*'s system in accuracy, and this is attributed to three factors: (a) better audio collection with less noise, as in the case of Wu *et al.* [27, 28], Schrapel *et al.* [23] and Li and Hammond [15]; (b) fewer recognised classes, as in the case of Luo *et al.* [18] and Moreira *et al.* [21]; or (c) different and potentially more capable feature extraction and/or classification techniques, as in the case of Chen *et al.* [5] and Zhou *et al.* [30].

This sheds light on Research Sub-question 3, and it is clear from the related studies that the microphones on a single smartphone are sufficient to achieve accurate segmentation and recognition of handwritten letters, but this is provided that at least one of the three factors mentioned previously are incorporated into the system.

Ultimately, given that the intended hardware setup is fixed i.e. smartphone-based, and the number of classes are also fixed, it is evident that an investigation of other types of feature extraction and/or classification techniques, such as various deep learning techniques, is the main important area of investigation in future.

As a final response to Research Sub-question 3, it can be stated that the microphones on a single smartphone are sufficient to achieve accurate segmentation and recognition of handwritten letters: (a) with the feature extraction and classification techniques used in the proposed system, provided that the user carries out a pre-training / handwriting registration procedure with the classifier; or (b) with at least one of the three factors mentioned previously incorporated into the system.

5.4 Summary

This chapter detailed the experiments carried out to assess key components of the proposed system, along with analysing the results obtained to be able to complete the remaining research objective 6, thereby answering the research sub-questions and the main research question set out Chapter 1.

The first experiment carried out aimed to assess the effectiveness of the audio segmentation component of the system. It was found that, within the intended use, the audio segmentation component was able to successfully segment 100% of the audio segments in every recording across all test subjects. This means that data collection and final testing can take place efficiently as each audio sample can be split up without any manual intervention, and without having to manually and frequently carry out stoppages between letters.

Study	Recognition Classes	Audio Hardware	Audio Segmentation	Feature Extraction	Classifier	User Dependency	Overall Accuracy (%)
Proposed	26 letters	Smartphone	✓	MFCC	SVM	Minimal	53.7
Wu *et al.* [27, 28]	26 letters	Piezoelectric microphones	✗	MFCC	SVM	Minimal	80.2 (2 mics)
Yu *et al.* [31]	26 letters	Smartphone	✓	MFCC	SVM	Moderate	64.94 (fixed location)
Luo *et al.* [18]	7 gestures	Smartphone	✓	MFCC	SVM	Minimal	93.2
Chen *et al.* [5]	7 gestures; 26 letters; 10 digits	Smartphone	✓	Spectrograms	CNN	Minimal	91.26
Zhou *et al.* [30]	26 letters	Smartphone	✓	stPSD	Inception LSTM CNN	Minmal	93.2
Moreira *et al.* [21]	Circle, square, triangle	Mini electret microphone attached to stethoscope	✗	Time scaled envelope signal	HMM	Moderate	90
Schrapel *et al.* [23]	26 letters	Custom pen with microphone, pressure and motion sensors	✓	Hilbert envelope	CNN	Minimal	78.3
Li and Hammond [15]	26 letters	Macbook Air (training); iphone and android (testing)	✓	MFCC	Template matching	Moderate	86.8 (pen)

TABLE 5.4: Comparison of proposed system to other implementations in the literature as described in Chapter 2

Accordingly, it was stated that Objective 2 set out in Chapter 1 was successfully achieved, and in response to research sub-question 1, it was stated that the proposed audio segmentation strategy is 100% effective at segmenting audio recordings into constituent letters.

The second experiment carried out aimed to assess the accuracy of the proposed system with respect to recognising the 26 letters on the semi-seen dataset, in order to determine obtain an indication of the ability of the classifier to recognize characters drawn by a user after carrying out a once-off pre-training procedure to allow the classifier to adapt to the user's drawing patterns.

Overall, it was found that the system provided an excellent accuracy of 82.3% across all letters and test subjects. It was also found that the system found it easier to recognise some letters, but was very robust to variations in test subjects. It was therefore stated that Research Objective 6 had been partially achieved. Furthermore, a partial response to Research Sub-question 2 was formulated as: if the user carries out a pre-training / handwriting registration procedure with the classifier, the proposed system can provide very high accuracy recognition that is robust to variations in test subjects and letters.

A third and final experiment aimed to assess the accuracy of the proposed system with respect to recognising the 26 letters on the unseen dataset, to obtain an indication of the system's ability to adapt to previously unseen test subjects.

Overall, it was found that the system provided an accuracy of 53.7% across all letters and test subjects, which was somewhat lower than the target criterion accuracy of 60%. A closer analysis revealed that the reduction in accuracy was mostly attributed to specific letters that were confused with other letters with the same number of strokes. It was very encouraging to note that the classifier remained robust to variations in test subjects in this experiment. It was also determined that training on a much larger training set would most likely improve on the accuracy of the system on unseen test subjects.

It was then stated that Research Objective 6 had been fully achieved, and as a completion to the previous partial response to Research Sub-question 2, it was further stated that: if no pre-training / handwriting registration procedure is carried out, the proposed system can provide an accuracy that is above the criterion accuracy of 60% for half of the 26 letters, but lower than the criterion accuracy for the other half of the 26 letters, with a high level of robustness to variations in test subjects. Furthermore, is is likely that the use of a larger training dataset can help increase the accuracy and reduce the variation in accuracy across letters with unseen test subjects.

Finally, a comparison of the proposed system to systems in the related studies revealed

that the related systems out-performed the proposed system. It was identified that this was as a result of the related systems either making use of more sophisticated and/or effective audio collection and noise insulation techniques; recognising fewer classes; or making use of more effective feature extraction and/or classification techniques. It was resolved to investigate the use of other feature extraction and classification techniques in future.

Finally, a response to Research Sub-question 3 was formulated as follows: the microphones on a single smartphone are sufficient to achieve accurate segmentation and recognition of handwritten letters: (a) with the feature extraction and classification techniques used in the proposed system, provided that the user carries out a pre-training / handwriting registration procedure with the classifier; or (b) with at least one of the three factors mentioned previously incorporated into the system.

The next chapter concludes the thesis.

Chapter 6

Conclusion

This research investigated the viability of utilizing a smartphone to capture the acoustic signals of a user writing uppercase English letters on a hard surface using a wooden utensil, and segmenting and classifying the letters. To this end, an audio segmentation and recognition system was implemented, including the collection of datasets required for the implementation.

The audio segmentation component aimed to allow the user to continuously write letters, and automatically segment the letters to provide a realistic and natural writing experience to the end user of the system. Segmented letters were processed with the MFCC feature descriptor, followed by classification by an SVM classifier to determine the character drawn. An additional feature of this research was the online-based architecture of the system: the system was hosted online to make it accessible from anywhere on any device via the web browser.

The main research question was phrased as "How accurately can handwritten words be segmented and recognized using audio captured by microphones on a smartphone using the proposed techniques i.e. MFCC coupled with SVMs?". Having obtained responses to the Research Sub-questions in the previous chapter, a response to the main research question can now be formulated as follows: it is possible to effectively segment handwritten words in the audio captured from the microphones of a smartphone with a high accuracy; it is also possible to use the proposed techniques to recognise the segmented letters with a high accuracy if the user carries out a pre-training / handwriting registration procedure with the classifier, but the accuracy deteriorates in the absence of such a pre-training / handwriting registration procedure.

It is important to note that it is imperative to investigate the use of a larger training set in future to determine whether this can help improve on the accuracy of the proposed

system on unseen data.

The next section provides directions for future work.

6.1 Future Work

The following section will outline some ideas for future work.

1. **Investigating potentially effective feature extraction and/or classification techniques:** One main area of investigation is the use of deep learning techniques such as LSTMs and CNNs which have shown significant promise in related studies.
2. **Dictionary Lookup:** Explore the performance of using some form of dictionary correction on each individually written word in an attempt to correct incorrectly recognised letters and thereby potentially improve the recognition accuracy. A fuzzy matching algorithm or a Levenstein distance method may be good algorithms to investigate.
3. **Remove stroke restrictions:** Investigate the possibility of removing the required stroke pattern and ordering to possibly enhance the robustness of the system under different conditions.
4. **Larger training dataset:** Investigate the the use of a larger training dataset to potentially help increase the accuracy and reduce the variation in accuracy across letters and unseen test subjects.

6.2 Concluding Comments

This experience has provided the researcher with many ups and downs and shown that anything is possible when some hard work is put in.

Appendix A

Additional Results

Letter	Correctly recognised count (out of 3)				
	Subject 1	Subject 2	Subject 3	Subject 4	Subject 5
A	2	2	2	2	3
B	3	3	3	3	3
C	3	1	3	2	2
D	2	3	2	1	3
E	0	3	2	3	3
F	3	3	3	3	3
G	3	3	3	3	3
H	1	3	2	2	3
I	3	3	3	3	2
J	3	0	3	2	2
K	3	2	1	3	3
L	2	3	3	2	2
M	3	3	3	3	3
N	3	3	3	3	3
O	3	1	2	3	3
P	3	1	2	3	3
Q	2	3	3	2	2
R	3	3	3	3	2
S	2	1	1	3	3
T	3	0	2	3	0
U	2	3	2	3	3
V	3	3	3	3	2
W	2	3	2	2	2
X	3	3	3	2	2
Y	3	2	3	3	0
Z	3	3	3	2	2

TABLE A.1: Number of correctly recognised samples (out of 3) for each subject for each letter class on the semi-seen dataset.

Letter	Correctly recognised count (out of 8)		
	Subject 6	Subject 7	Subject 8
A	4	6	6
B	8	5	5
C	2	0	1
D	0	5	2
E	4	8	4
F	7	8	6
G	4	7	6
H	2	2	5
I	8	6	8
J	1	1	4
K	4	4	3
L	3	2	7
M	5	6	6
N	7	6	6
O	0	0	3
P	4	0	1
Q	3	8	8
R	6	6	4
S	7	3	6
T	3	2	1
U	2	3	7
V	2	0	1
W	6	0	3
X	5	4	6
Y	5	7	7
Z	6	5	7

TABLE A.2: Number of correctly recognised samples (out of 8) for each subject for each letter class on the unseen dataset.

Letter	Correctly recognised (%)		
	Subject 6	Subject 7	Subject 8
A	50	75	75
B	100	62	62
C	25	0	12
D	0	62	25
E	50	100	50
F	88	100	75
G	50	88	75
H	25	25	62
I	100	75	100
J	12	12	50
K	50	50	38
L	38	25	88
M	62	75	75
N	88	75	75
O	0	0	38
P	50	0	12
Q	38	100	100
R	75	75	50
S	88	38	75
T	38	25	12
U	25	38	88
V	25	0	12
W	75	0	38
X	62	50	75
Y	62	88	88
Z	75	62	88
Average	51.9	50.0	59.1

TABLE A.3: Percentage of correctly recognised samples (out of 8) for each subject for each letter class on the unseen dataset.

Bibliography

[1] D. Asonov and R. Agrawal, "Keyboard acoustic emanations," in *IEEE Symposium on Security and Privacy, 2004. Proceedings. 2004.* IEEE, 2004, pp. 3–11.

[2] A. Athanasopoulos, A. Dimou, V. Mezaris, and I. Kompatsiaris, "Gpu acceleration for support vector machines," in *Procs. 12th Inter. Workshop on Image Analysis for Multimedia Interactive Services (WIAMIS 2011), Delft, Netherlands*, vol. 164, 2011.

[3] Y. Berger, A. Wool, and A. Yeredor, "Dictionary attacks using keyboard acoustic emanations," in *Proceedings of the 13th ACM conference on Computer and communications security*, 2006, pp. 245–254.

[4] P. Brandl, C. Forlines, D. Wigdor, M. Haller, and C. Shen, "Combining and measuring the benefits of bimanual pen and direct-touch interaction on horizontal interfaces," in *Proceedings of the working conference on Advanced visual interfaces*, 2008, pp. 154–161.

[5] M. Chen, P. Yang, J. Xiong, M. Zhang, Y. Lee, C. Xiang, and C. Tian, "Your table can be an input panel: Acoustic-based device-free interaction recognition," *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, vol. 3, no. 1, pp. 1–21, 2019.

[6] O. Cheng, W. Abdulla, and Z. Salcic, "Performance evaluation of front-end processing for speech recognition systems," *The University of Auckland*, 2005.

[7] C. Cortes and V. Vapnik, "Support-vector networks," *Machine learning*, vol. 20, no. 3, pp. 273–297, 1995.

[8] Deafblind Information Australia, "Living with deaf blindness," 2022, [Online; accessed 4-October-2022]. [Online]. Available: https://www.deafblindinformation.org.au/living-with-deafblindness/deafblind-communication/block-alphabet/

[9] H. Du, P. Li, H. Zhou, W. Gong, G. Luo, and P. Yang, "Wordrecorder: Accurate acoustic-based handwriting recognition using deep learning," in *IEEE INFOCOM*

2018-IEEE Conference on Computer Communications. IEEE, 2018, pp. 1448–1456.

[10] T. Halevi and N. Saxena, "Keyboard acoustic side channel attacks: exploring realistic and security-sensitive scenarios," *International Journal of Information Security*, vol. 14, no. 5, pp. 443–456, 2015.

[11] M. A. Hearst, S. T. Dumais, E. Osuna, J. Platt, and B. Scholkopf, "Support vector machines," *IEEE Intelligent Systems and their applications*, vol. 13, no. 4, pp. 18–28, 1998.

[12] James Robert, "Pydub," 2022, [Online; accessed 5-October-2022]. [Online]. Available: https://github.com/jiaaro/pydub/releases/tag/v0.25.1

[13] J. Katona, "A review of human–computer interaction and virtual reality research fields in cognitive info communications," *Applied Sciences*, vol. 11, no. 6, p. 2646, 2021.

[14] S. S. Keerthi and C.-J. Lin, "Asymptotic behaviours of support vector machines with gaussian kernel," *Neural computation*, vol. 15, no. 7, pp. 1667–1689, 2003.

[15] W. Li and T. Hammond, "Recognizing text through sound alone," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 25, no. 1, 2011, pp. 1481–1486.

[16] B. Logan, "Mel frequency cepstral coefficients for music modelling," in *In International Symposium on Music Information Retrieval*. Citeseer, 2000.

[17] L. Lu, J. Liu, J. Yu, Y. Chen, Y. Zhu, X. Xu, and M. Li, "Vpad: Virtual writing tablet for laptops leveraging acoustic signals," in *2018 IEEE 24th International Conference on Parallel and Distributed Systems (ICPADS)*. IEEE, 2018, pp. 244–251.

[18] G. Luo, P. Yang, M. Chen, and P. Li, "Hci on the table: robust gesture recognition using acoustic sensing in your hand," *IEEE Access*, vol. 8, pp. 31 481–31 498, 2020.

[19] A. Mason, "Terminology for loudness and level," 2011.

[20] D. Meyer and F. Wien, "Support vector machines," *The Interface to libsvm in package e1071*, vol. 28, p. 20, 2015.

[21] B. S. Moreira, A. Perkusich, and S. O. Luiz, "An acoustic sensing gesture recognition system design based on a hidden markov model," *Sensors*, vol. 20, no. 17, p. 4803, 2020.

[22] A. Rashid, M. A. Zeb, A. Rashid, S. Anwar, F. Joaquim, and Z. Halim, "Conceptualization of smartphone usage and feature preferences among various demographics," *Cluster Computing*, vol. 23, no. 3, pp. 1855–1873, 2020.

[23] M. Schrapel, M.-L. Stadler, and M. Rohs, "Pentelligence: Combining pen tip motion and writing sounds for handwritten digit recognition," in *Proceedings of the 2018 CHI conference on human factors in computing systems*, 2018, pp. 1–11.

[24] I. Syarif, A. Prugel-Bennett, and G. Wills, "Svm parameter optimization using grid search and genetic algorithm to improve classification performance," *TELKOMNIKA (Telecommunication Computing Electronics and Control)*, vol. 14, no. 4, pp. 1502–1509, 2016.

[25] M. Tanveer, T. Rajani, R. Rastogi, Y.-H. Shao, and M. Ganaie, "Comprehensive review on twin support vector machines," *Annals of Operations Research*, pp. 1–46, 2022.

[26] F. M. Wiener and D. N. Keast, "Experimental study of the propagation of sound over ground," *The Journal of the Acoustical Society of America*, vol. 31, no. 6, pp. 724–733, 1959.

[27] Q. Wu, M. Ghaziasgar, J. Connan, and R. Dodds, "Audio recognition on contact surface gestures." 09 2015.

[28] Q. Wu, M. Ghaziasgar, R. Dodds, and J. Connan, "Robust audio-based digit recognition," in *Southern Africa Telecommunication Networks and Applications Conference*, vol. 2018, 2018, pp. 250–255.

[29] H. Yin, A. Zhou, L. Liu, N. Wang, and H. Ma, "Ubiquitous writer: Robust text input for small mobile devices via acoustic sensing," *IEEE Internet of Things Journal*, vol. 6, no. 3, pp. 5285–5296, 2019.

[30] H. Yin, A. Zhou, G. Su, B. Chen, L. Liu, and H. Ma, "Learning to recognize handwriting input with acoustic features," *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, vol. 4, no. 2, pp. 1–26, 2020.

[31] T. Yu, H. Jin, and K. Nahrstedt, "Writinghacker: audio based eavesdropping of handwriting via mobile devices," in *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, 2016, pp. 463–473.

[32] L. Zhuang, F. Zhou, and J. D. Tygar, "Keyboard acoustic emanations revisited," *ACM Transactions on Information and System Security (TISSEC)*, vol. 13, no. 1, pp. 1–26, 2009.