WENDELL, TYLER JAMES, M.S. Feature Extraction and Feature Reduction for Spoken Letter Recognition. (2016)
Directed by Dr. Shanmugathasan Suthaharan. 46 pp.

The complexity of finding the relevant features for the classification of spoken letters is due to the phonetic similarities between letters and their high dimensionality. Spoken letter classification in machine learning literature has often led to very convoluted algorithms to achieve successful classification. The success in this work can be found in the high classification rate as well as the relatively small amount of computation required between signal retrieval to feature selection. The relevant features spring from an analysis of the sequential properties between the vectors produced from a Fourier transform. The study mainly focuses on the classification of fricative letters f and s, m and n, and the eset (b,c,d,e,g,p,t,v,z) which are highly indistinguishable, especially when transmitted over the modern VoIP digital devices. Another feature of this research is the dataset produced did not include signal processing that reduces noise which is shown to produce equivalent and sometimes better results. All pops and static noises that appear were kept as part of the sound files. This is in contrast to other research that recorded their dataset with high grade equipment and noise reduction algorithms. To classify the audio files, the machine learning algorithm that was used is called the random forest algorithm. This algorithm was successful because the features produced were largely separable in relatively few dimensions. Classification accuracies were in the 92%-97% depending on the data set.

FEATURE EXTRACTION AND FEATURE REDUCTION FOR SPOKEN LETTER RECOGNITION

by

Tyler James Wendell

A Thesis Submitted to
the Faculty of The Graduate School at
The University of North Carolina at Greensboro
in Partial Fulfillment
of the Requirements for the Degree
Master of Science

Greensboro
2016

Approved by

Committee Chair

APPROVAL PAGE

This thesis written by Tyler James Wendell has been approved by the following committee of the Faculty of The Graduate School at The University of North Carolina at Greensboro.

Committee Chair _____
Shanmugathasan Suthaharan

Committee Members _____
Fereidoon Sadri

_____
Jing Deng

_____
Date of Acceptance by Committee

_____
Date of Final Oral Examination

ACKNOWLEDGMENTS

All praise, honor and glory to my Lord Jesus Christ for His richest grace and mercy for the accomplishment of this thesis. His goodness and sovereignty has provided for all opportunities to continue my education and has surrounded me with the people to succeed.

This work would not have been possible without the direction of my advisor, Dr Shanmugathasan Suthaharan. He has consistently pushed me to be my best and to have a depth in my understanding of the subject material. I am so thankful to have an advisor who is absolutely committed to his students to learn and succeed in their goals. Thank you, Dr. Suthaharan, for all I have learned from you and for all your encouragement.

Thank you to the fellow mathematics graduate assistances who allowed me to think out loud and bounce ideas off of through some of the issues in this thesis work.

Finally, it would not have been possible without my wife, Michelle. I am grateful for her constant support, encouragement and prayers that have fueled my motivation to reach my goals.

TABLE OF CONTENTS

LIST OF TABLES

Page

LIST OF FIGURES

Page

CHAPTER I

INTRODUCTION

Technology that uses spoken word recognition is all around us. Generally, humans naturally have a high sensitivity to auditory information. They are able to pick out the melody of a single instrument among many other playing instruments, detect slight variations in language accents, or even detect emotions based on the sound or timbre of someones voice. In the digital era, it is natural to build computer models to similarly be able to discern, classify, and recognize patterns in audio data for research or automated processes. Thanks to a lot of hard work done in the past, machines now have ways of understanding what you say; people can talk into their phones and ask it to send text messages, find the nearest gas station or even ask for word definitions. However, the systems are not perfect. This is because when we digitize sound, minor variations in the signal become pronounced complications. To compensate, research in computational signal analysis has had to devise complex algorithms with multiple steps to manipulate the data so that patterns become more pronounced and the desired outcome can be achieved.

Word recognition has been used by the industry in various forms throughout the years. If you have ever called a business's customer service line, chances are you have run into technology that uses word recognition. There is a high chance you have also experienced frustration when the electronic response is "I'm sorry I did not understand your response," or "Did you say _" when you didn't actually say anything close to that. Despite some of these personally frustrating instances,

word recognition technology has become more accurate over the years with the rise of distributed computing and advanced machine learning algorithms. For example, Microsoft, Google and Apple has made it a standard to allow you to make commands to their devices with very accurate recognition [HBR14], but why are these advanced techniques required? What makes word recognition so difficult?

The nature of audio data is what makes word recognition hard. When examining audio data we realize that it is, considered big data when we examine it using the three C's of big data: cardinality, complexity, and continuity [Sut14, GH15]. Consider the example of asking questions or giving commands to your smartphone. Cardinality represents the number of observations that can grow dynamically. This can be considered a massive problem when considering the large number of words that are being added to general vocabulary through slang or the naming of new products or businesses. Audio data has problems in continuity through the demand of being recognized in dynamic quantities from any number of people talking into their smartphone requiring near real time answers at any moment. The complexity of audio data is understood when considering the vastness of sounds used to make up words in any language and in the vast variety of minor alterations of the phonetic makeup given someone's region. To add to the complexity every person has a different timbre to their voice and some individuals have incomplete knowledge of the language they are speaking due to inexperience. These all add to the mass amounts of variation [BDMD+07]. All of these variations, however slight they may seem to humans, are pronounced in digital representations. Audio data is also high dimensional data. An audio clip that is milliseconds long can return tens of thousands of signal points based on the sampling rate. There are a variety of au-

dio properties to reduce the dimensionality of the data but any number of these properties is often not sufficient to accurately classify spoken words [O'S08]. To get successful classification rates advanced machine learning and statistical modeling techniques have been necessary.

The research in this paper uses a small vocabulary environment that focuses on confusable sets of words, that are commonly studied in other literature in aims to find new methods of dimensionality reduction for successful word recognition. The motivation comes from discovering possible patterns present in the imaginary vectors of the signals mapped in Fourier space. Finding relevant features that reduce the dimensionality in Fourier space could lead to quicker recognition rates because the step of feature extraction in real time scenarios would take fewer algorithmic steps. The dataset produced for this research is presented in chapter III. It was produced without noise reduction, allowing the noise to present relevant information for classification and skipping the extra steps of cleaning the audio similar to what other audio databases have been subject to. Using noise in analysis is a method proposed by Dalessandro [Dal13] who shows that in many big data problems, including noise, helps to increase predictive performance. This work presents patterns in the Fourier space derived from basic linear algebra and reduces the dimensionality using principal components for successful classification. For simplification, we experiment with three sets of confusable letters, f and s, m and n, and the eset (b,c,d,e,g,p,t,v,z). The machine learning technique used for classification was the random forest algorithm. A literature review of past techniques is given in chapter II. The complete methodology is presented in chapter IV. Results from this process is presented in chapter V and a discussion of the results in chapter VI.

The final chapter concludes with research that can be done to continue this work to realize the full potential of this methodology.

CHAPTER II

BACKGROUND

There are a lot of common techniques for analyzing audio data. Representing the signal in the power frequency domain using a discrete Fourier transform is one of the most common methods . The discrete Fourier transform takes a signal that is represented by amplitude over time and converts it to discrete values that estimate the power of the frequencies of the different sine waves that make up the signal [Wel67]. Reducing the data to Mel-Frequency Cepstral Coefficients (MFCC) has been shown to provide many advantages in various audio analysis problems [DM80]. The success of using MFCC's comes from its estimation of human perceived frequency bands in signals. Another feature extraction technique is to use perceptual linear predictive (PLP) analysis to reduce the signals to a scale based on the psychophysics of hearing [Her90]. A signal feature called zero crossing rate measures the rate at which a signal changes from positive to negative. Other spectral characteristics such as spectral roll-off and spectral centroid allow us to gain insight into the texture of the sound [Gia15]. Some researchers have also added features such as lip movement analysis to help in speech recognition. This adds information derived from videos to the dataset, which greatly increases the dimensionality but it is shown to improve accuracy [BK94, BHMW93].

Typical categories of audio analysis include music analysis and speech analysis. Work done in music analysis include music genre classification and classification of instruments among bands being played [MEKR11]. There are also a variety of

speech analysis problems including detecting emotions, classifying speakers, language detection, and many different types of word recognition. Many of the problems in the mentioned categories use the basic features mentioned above or build from them to detect patterns in audio data. In the paper "Towards Detecting Emotions in Spoken Dialogues," [LN05] Lee presents using acoustical information paired with the transcribed language to detect the emotions of individuals to recognize negative and non-negative emotions. The types of acoustical information used was the fundamental frequency, energy, duration, formant frequencies, pitch, and the ratio of voiced to unvoiced portions of the audio. They found that combining acoustical information with language information gives you the best accuracies. Li and Narayanan propose several new computational models for speaker verification and language detection using MFCC with shifted delta cepstral, regular MFCC, and gammatone frequency cepstral coefficients for features that allow for the better results [LN14].

Word recognition can be defined as building computer systems that take data derived from an audio input and discerning the word(s) spoken. The vocabulary the computer system is expected to understand can be defined as small, large or extended. In extended vocabularies the model should be able to handle new words not available at the time of training. Word recognition is in itself, an umbrella of problems including: isolated word recognition, connected word recognition, continuous speech recognition, and spontaneous speech recognition[GGY10]. There are also more complicated forms of word recognition that require understanding the context of what is actually being said. An example of this is creating a computing system

to understand which spoken words should be capitalized. This is useful when the computer transcribes audio data so that it can be easily read by humans [BS13].

There are many different machine learning and statistical modeling methods that have been used to deal with the complexity of word recognition. The most popular and successful methods are Deep Neural Networks (DNN) and Hidden Markov Models (HMM) [HDY$^+$12, CL16, HBR14]. Both methods require dimensionality reduction and large distributed systems for successful classification. In 2010, Google's voice assistance for android devices use 39 dimensional perceptual linear prediction coefficients with linear discriminant analysis to build models that simulate human auditory characteristics. The complex classifier Google used at that time involved decision trees tied to HMM [Sch10]. Although HMM showed impressive results in speech recognition, the word error rate reached impressive reductions with the introduction of DNN. Later in 2014 a deep learning architecture was proposed for Google's speech recognition system [BH14]. Microsoft researchers present the advantages of using DNN as feature extractors while using sequence based classifiers for recognition [DC14]. Some studies have shown that the best results comes from a combination of DNN and HMM [HBR14, HDY$^+$12]. These are not the only learning and modeling techniques used. In "Spoken language understanding for natural interaction: The siri experience," Bellegarda suggests that Apple's Siri is able to do spoken request recognition through a reinforcement learning model powered by Bellman principles called partially observable markov decision processes [Bel14]. These methods have been addressed in large or extended vocabulary situations but the complexities still persist in small vocabulary environments.

Small vocabulary word recognition problems such as spoken letter recognition still fit into the context of big data word classification. This is due to how many letters have similar phonetic information. For example, a common set of confusable letters used as a benchmark for classification accuracy is the eset. Most people can relate to having to spell their name over the phone and the receiver on the other end, whether it is a human or a computer, mishearing your f as s, b as v or m as n. That is why phonetic alphabets exist. We say: "s as in Sam" and "f as in frog" because even humans need additional information to help us distinguish between the different sets of confusable letters.

One of the more popular research papers on spoken letter recognition is titled "Spoken Letter Recognition" [CF90] and was published in 1991. The paper is split up into three main parts. The first part is a full description of the development of the ISOLET database, which is a very common database in this research area and it was first developed for this paper. Next, the methodology and systems developed for isolated spoken letter classification is discussed. Finally, the paper talks about letter classification when letters are spoken in sequence.

The ISOLET database is a very popular database and it is now available for free at the UCI data repositories [Lic13]. It is comprised of 150 different speakers uttering the alphabet twice. The signals are recorded with noise canceling microphones and processed using professional audio equipment so that all the analysis is done on the clean utterances alone. The signals were then processed into 4 categories of features they named: contour, sonorant, pre-sonorant, and post-sonorant totaling 617 features. In each category, there are a variety of signal properties captured such as zero crossing rate, peak to peak amplitude, estimated pitch, duration,

and spectral analysis. The majority of features were spectral features, but all the features were developed to represent specific information to discriminate between vowel and consonant sounds in the beginning, middle, and end of the spoken letter [Col90].

After the ISOLET database was generated classification was done using feed forward neural networks trained using back-propagation. In their research, the classification performance is influenced immensely by the segmentation algorithms that breaks apart the audio signal for feature extraction. They also found if they broke up the neural network into three parts isolating easily confusable sets they can achieve better performance. The final classification system included 3 networks: one for the eset, one for m and n, and the final for all the remaining letters. With this system, they were able to achieve an average classification accuracy of 96%.

The paper "Spoken English Alphabet Recognition with Mel Frequency Cepstral Coefficients and Back Propagation Neural Networks" [AS+12] builds a system that uses the MFCC's for letter recognition. The research presents the classification accuracy when the dimensionality is reduced by using only the twelve MFCC's. The dataset used for this study is called the TI ALPHA set and consists of 16 speakers and 26 utterances of each letter. The signals were preprocessed similar to the ISOLET dataset mentioned above. The classification tests were run only on the eset because of its particularly high confusability. Using 12 MFCC's and back propagation neural networks the highest accuracy they were able to achieve was 79.17%.

Speech recognition models have also been built from successful image analysis techniques. In the paper "Rapid Speaker Adaptation in Eigenvoice Space" [KJNN00] the methods for feature reduction and accurate classification was mod-

eled after methods used in face recognition in images [TP91]. The goal of the paper was to find models of letter recognition built from a speaker dependent system that could adapt to a new speaker with a relatively small amounts of adaptation modeling. The work used the ISOLET database for all experimentation. The strength of their system comes from the development of eigenvoices which are the eigenvectors obtained from a principal component analysis of certain characteristics built from the dataset. The training set contained 120 speakers and for each speaker they built what they called a "supervector" of parameters. The supervectors contained for each letter a HMM with six Gaussian outputs and eighteen perceptual linear predictive cepstral coefficients for each Gaussian output. Principal component analysis was performed on the 120 supervectors creating the speaker dependent eigenvoices. Using this model they were able to obtain classification rates for the 30 new speakers at 86.3% with minimal adaptation weighting.

It is noticeable that HMM are a common statistical technique for letter recognition. The two papers: "On The Use of High Order Derivatives for High Performance Alphabet Recognition"[Mar02] and "Signal Modeling for High-Performance Robust Isolated Word Recognition"[KZ01] also use HMMs for classification. Both papers also use the ISOLET database for their experiments. They do however, differ in feature extraction. The first paper takes sequential derivatives of 11 Mel-cepstrum vectors for each utterance totaling 72 features. The second paper computes Discrete Cosine Transform Coefficients over variable windows for each utterance totaling 50 features. The average success rates for each are comparable, 97.54% and 97.6% respectively.

The work in this thesis takes a step back from asking what is a good classification algorithm for accurate prediction. It looks at the data and asks the question how do we make relevant features easier to obtain? Having relevant features that reduce the dimensionality and the computational complexity allow the possibilities for faster real time analysis. This is because every new observation that comes into your trained model needs to have the same properties extracted from it that was extracted from the training data to build the classification model. We have seen plenty of successful models but the possibility of easily acquiring the information the models use easier to get is a very exciting research area. This would allow speech recognition systems to become more versatile. This is important because spoken communication is a very common way to pass on information. Think about the vast amount of data that is present in spoken communication; it is all out there ready for machines to use, the job of the scientist is to design the proper ways to extract the information and analyze it.

CHAPTER III

DATASET

The dataset used for this work was generated especially for this work. This is because all available data sets that are readily available have already been through some degree of signal processing and cleaning. This is not aligned with the goal of this work. The goal is to find methods that require far fewer steps compared to other works. This includes the steps of cleaning and processing. Approaching the research from this angle allows for effective methods to be more scalable and more practical for real world applications.

Generating the observations for the dataset consisted of recording a single speaker uttering the alphabet 50 times. Each utterance was recorded separately in 2 second intervals. The subject sat in front of the microphone and was prompted to speak the letter presented. Each letter was played back to be either accepted or rejected by the speaker. This was done to ensure that the full letter was captured in the recording. The recording software then clipped the silence in the beginning and end of the audio clip. The software did this by starting at both ends of the clip and moving inward, removing silence until the first instance of sound was found. The software was not advanced enough to remove clicks or pops in the recording and if they occurred somewhere in the silenced sections at the beginning or end of the clip those sections were removed only until the click or the pop. This caused remaining sections of silence to exist in between the noise and the speaker's voice. This can be seen in an example sound wave shown in Figure 1. This wasn't dealt with because

noise of all kinds will be present in real world recordings. The microphone used to record the audio was a standard USB logic computer microphone. The recording software was written using the python libraries Pyaudio and Wave. Every signal was saved using the *.wav format.

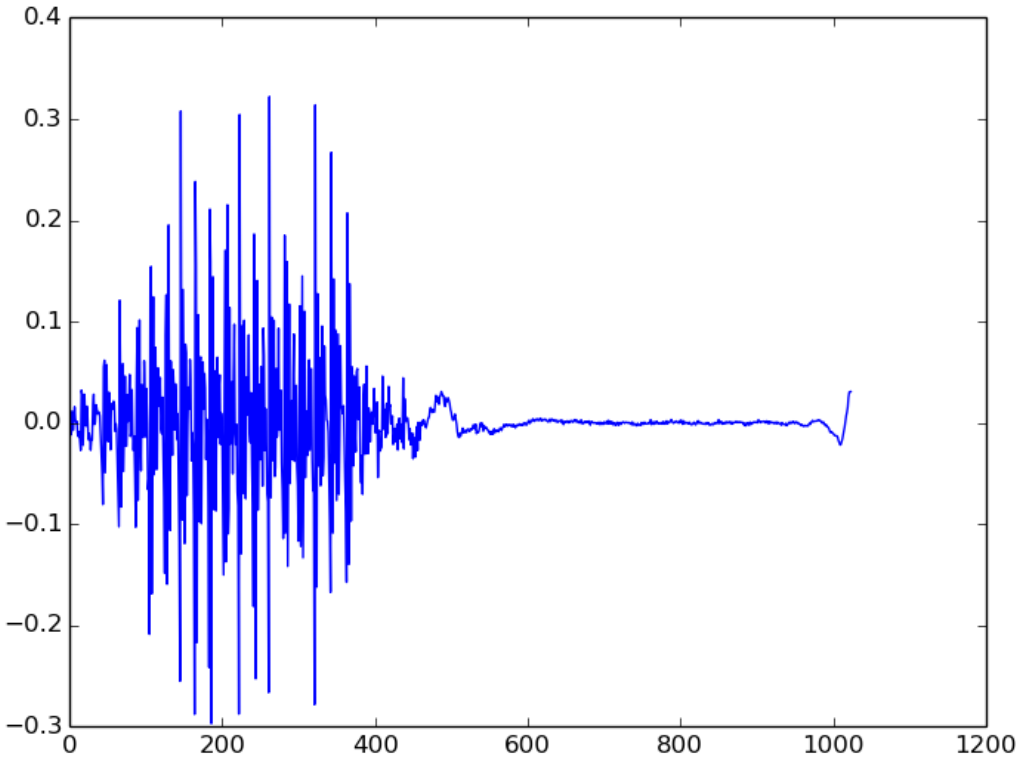


Figure 1. Example Sound Wave with Noise.

CHAPTER IV

METHODOLOGY

The entirety of feature selection performed in this study is a spatial analysis of the signals. There are two parts to our spatial analysis. First we use the Discrete Fourier Transformation (DFT) algorithm to look at the signals in Fourier space, then we look into something that hasn't been proposed elsewhere. We look at the rotational information between sequential Fourier data points. In doing so we no longer look at the Fourier data but the sine and cosine components between each point.

Before any spatial analysis is performed on the signals, each signal was binned so that every signal had uniform length. This was done according to the following algorithm:

$$sig_k = s_0, s_1, ..., s_{z-1} \textit{ data points based on the sampling } frequency.$$
$$binned\ sig_k = \frac{\sum_{j=0}^{b-1} s_j}{b}, \frac{\sum_{j=b}^{2\cdot b-1} s_j}{b}, ..., \frac{\sum_{j=i\cdot b}^{(i+1)\cdot b-1} s_j}{b}, ..., \frac{\sum_{j=(m-1)\cdot b}^{(m)\cdot b-1} s_j}{b}. \tag{4.1}$$

In Equation 4.1 m is the number of bins, $b = \frac{z}{m}$ and z is the length of the original clip. All this represents is a binning that took the mean of a uniform number of points based on the size the signal. This was done for every signal so they would all be compressed to length m. Different lengths of m were experimented with to find

the bin sizes that yield the best classification performance. In Chapter V we show the results of different bin sizes.

An audio signal can be decomposed into a sum of sine waves, this is called a composite signal. A Fourier transform takes the signal and decomposes it into the powers of the frequencies of the different sine waves [Blo04]. The specific algorithm used was the Fast Fourier Transformation (FFT) which is computationally more efficient and produces the same results [Wel67]. To see the effects of the FFT algorithm we walk through an example.

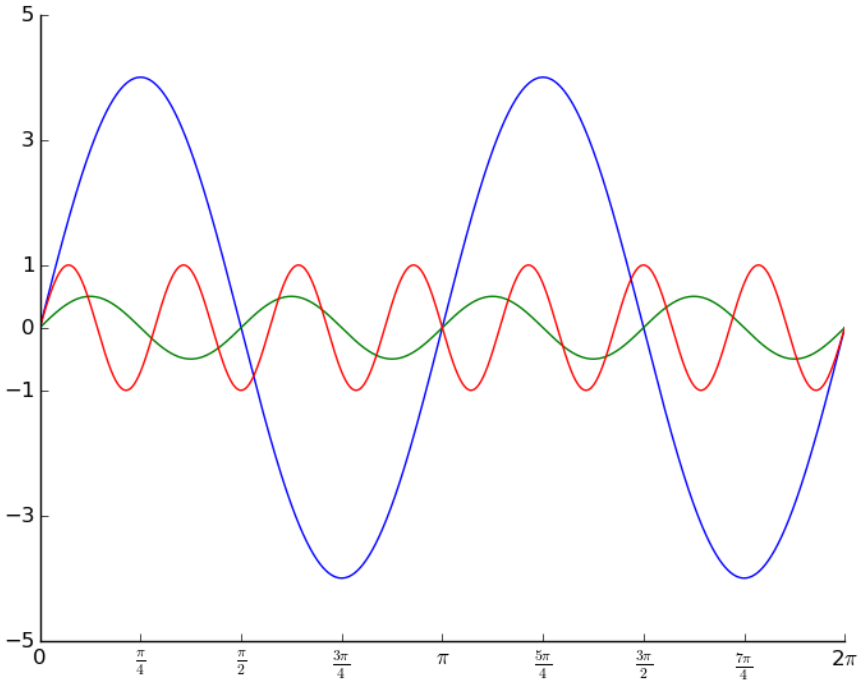


Figure 2. Sine Waves $f(x), g(x)$, and $h(x)$.

In Figure 2 we can see three different sine waves with different frequencies and different amplitudes. The three sine waves are:

$$
\begin{aligned}
f(x) &= 4sin(2x), \\
g(x) &= \frac{1}{2}sin(4x), \\
h(x) &= sin(7x). \\
C(x) &= f(x) + g(x) + h(x).
\end{aligned}
\tag{4.2}
$$

We can see the sum of the three signals in Figure 3 as a single wave. Audio signals produce similar composite waves. Practically, audio signals are the sum of a multitudinous number of sine waves and recorded audio will include deformations due to noise. If you look at Figure 4 we can see a small sample of an audio recording of the spoken letter 'a.' Looking at both Figure 3 and Figure 4 we can easily conceive that audio signals are the sum of sine waves.

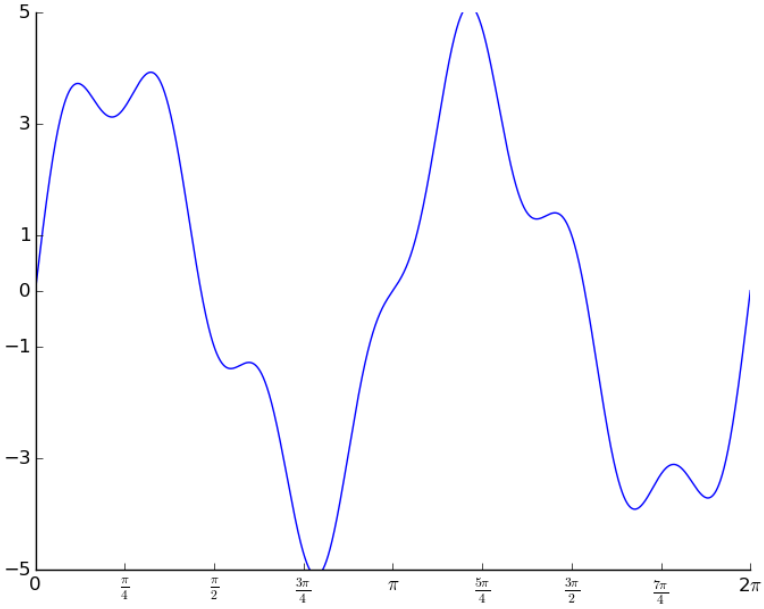


Figure 3. Composite Signal $C(x)$.

The FFT algorithm outputs the data in the complex frequency domain, that is data in the form of $a + bi$. The real part of all the data points, or the transformation of the imaginary point to a real point, gives us the power of the frequencies. This can be seen in the simple example of performing an FFT on our example composite wave $C(x)$. Figure 5 shows the results of FFT($C(x)$). The domain represents the frequency of the composite signal and the range represents the power at those frequencies. The question asked that directed this study was: Is there useful information embedded in the real AND imaginary parts of the FFT output that could help us to classify audio signals?

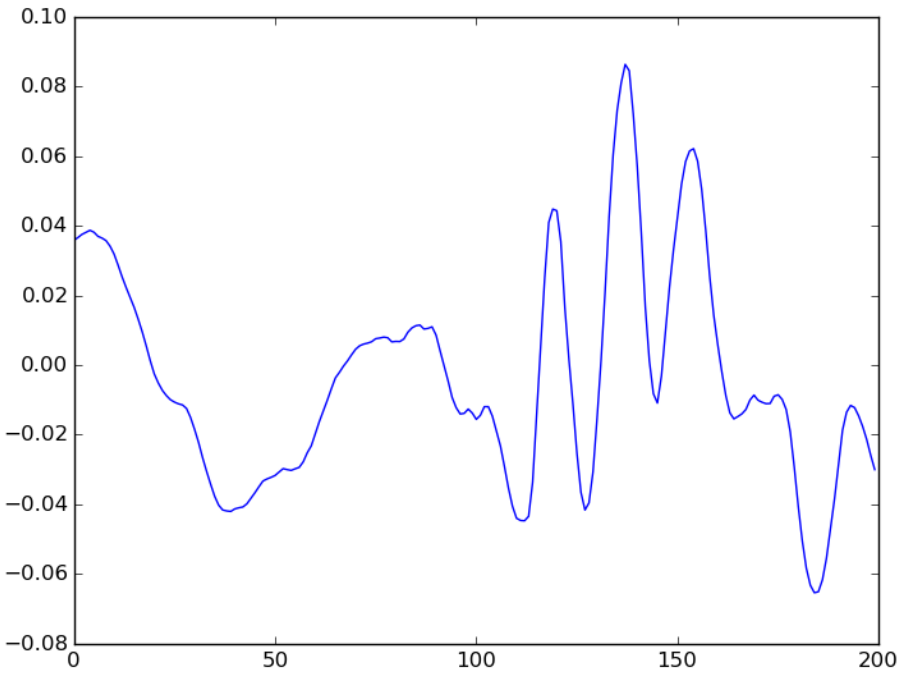


Figure 4. Sample Signal of Spoken 'a.'

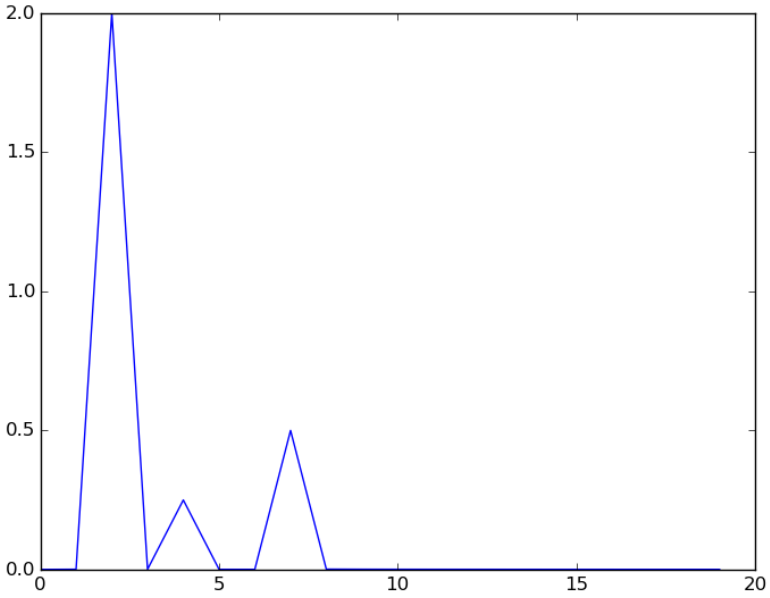


Figure 5. FFT Real Output.

4.1 Feature Selection

The frequency domain on an audio signal looks tremendously different than the simple example above. This is due to noise and when we speak we create much more complicated waves than just the sum of a few sine waves. An example of the frequency domain of the audio signal of someone speaking the letter 'f' is shown in Figure 6. As mentioned before the output of an FFT is in the complex domain. Figure 7 shows the FFT output on the complex plane, where the domain are real values and the range are imaginary values. This picture is very important to the research. We started to notice visual patterns between letters based on the geometric shapes of their plots similar to Figure 7. The belief that there are simple patterns in the raw complex data that can be described computationally in such a way that allows machines to distinguish between spoken letters, is what was explored.

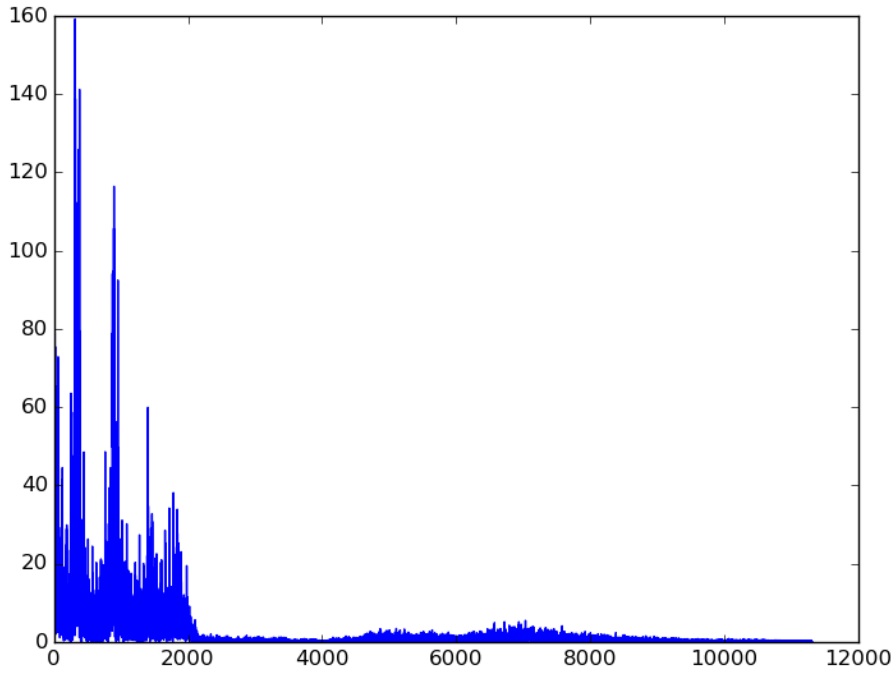


Figure 6. Frequency Domain of Spoken Letter 'f.'

Consider a complex point $a + bi$ as a vector in the complex plane:

$$v_1 = \begin{bmatrix} a \\ b \end{bmatrix}. \tag{4.3}$$

We can rotate the vector v_1 to a new vector v_2 in the complex plane by multiplying v_1 by the matrix A as shown in equation 4.4.

$$A = \begin{bmatrix} cos(\alpha) & -sin(\alpha) \\ sin(\alpha) & cos(\alpha) \end{bmatrix}. \tag{4.4}$$

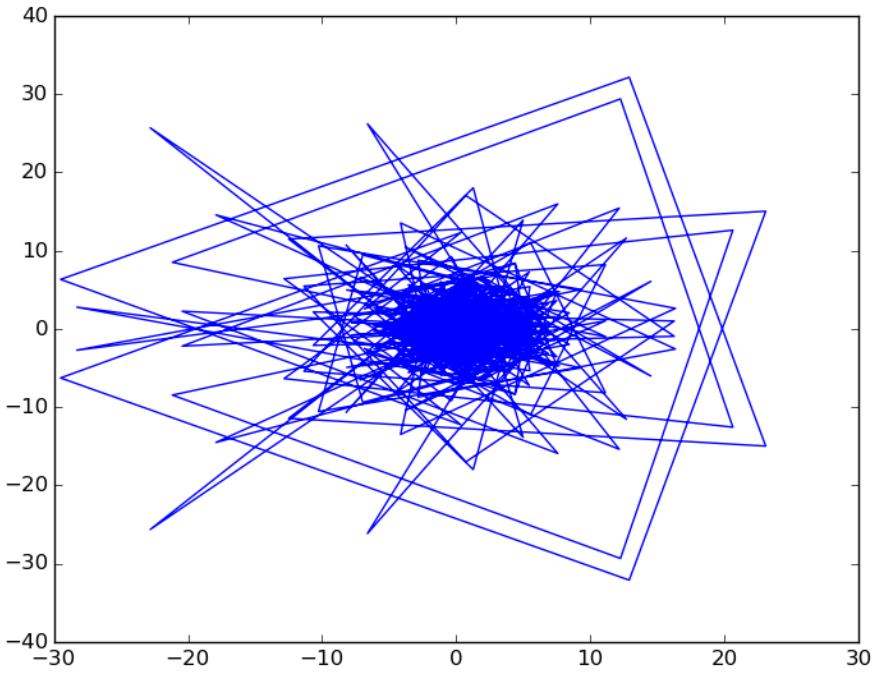


Figure 7. FFT Output in the Complex Plane.

If we know vector v_1 and v_2 we can find the rotational transformation matrix A by computing the following:

$$A = \begin{bmatrix} \frac{v_1 \cdot v_2}{||v_1||^2} & -\frac{v_1 \times v_2}{||v_1||^2} \\ \frac{v_1 \times v_2}{||v_1||^2} & \frac{v_1 \cdot v_2}{||v_1||^2} \end{bmatrix}. \tag{4.5}$$

The two operators in the matrix 4.5, $\cdot$ and $\times$, are the traditional dot product and cross product, respectively.

To understand how this works an example was formed to show the steps practically. If we look at Figure 8 we can see two vectors, $v_1 = -1 + 4i$ and $v_2 = 7 + 2i$. We can calculate the rotational transformation that would send v_1 to v_2 using the

matrix A defined above. By calculating: $v_1 \cdot v_2 = 1$, $v_1 \cdot v_1 = 17$ and $v_1 \times v_2 = -30$, we can build A such that:

$$\begin{bmatrix} \frac{1}{17} & -\frac{-30}{17} \\ \frac{-30}{17} & \frac{1}{17} \end{bmatrix} \cdot \begin{bmatrix} -1 \\ 4 \end{bmatrix} = \begin{bmatrix} 7 \\ 2 \end{bmatrix}. \tag{4.6}$$

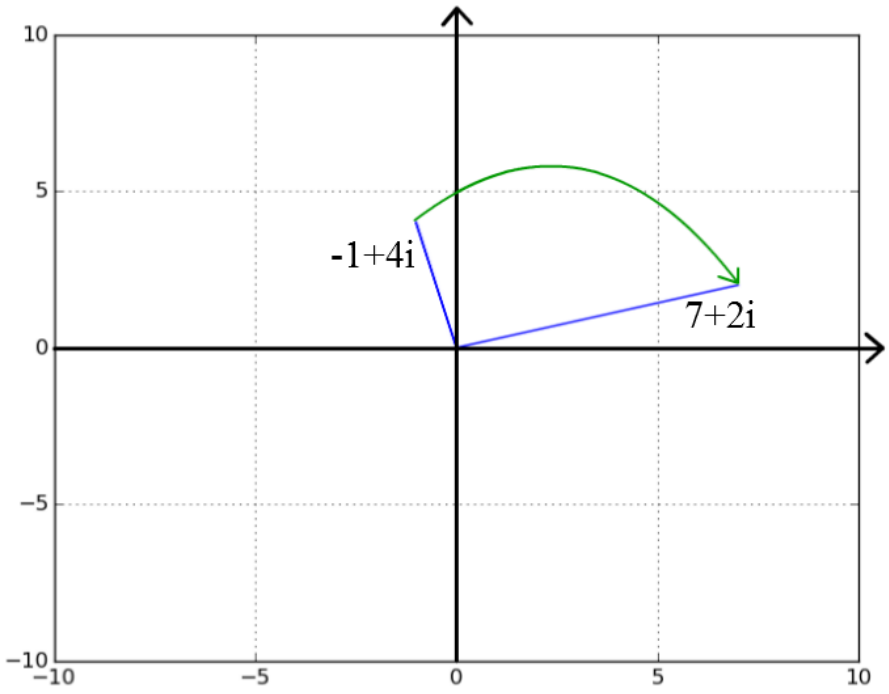


Figure 8. Two Imaginary Vectors in the Complex Plane.

This process has been explained in such detail because it is an integral part of the feature selection for this research. We can reduce the transformation matrix to two distinct parts: $cos(\alpha)$ and $sin(\alpha)$. The features generated for this research are the $cos(\alpha)$ and $sin(\alpha)$ values between each sequential point generated from the

FFT. The mathematical representation of this process is shown in the Equation 4.7 with the final data table shown in Equation 4.8.

$$\begin{bmatrix} FFT(sig_0) \\ FFT(sig_1) \\ \dots \\ FFT(sig_n) \end{bmatrix} = \begin{array}{|c|c|c|c|c|} \hline f_1 & f_2 & \dots & f_m & sig_0 \\ \hline f_1 & f_2 & \dots & f_m & sig_1 \\ \hline \multicolumn{5}{|c|}{\dots} \\ \hline f_1 & f_2 & \dots & f_m & sig_n \\ \hline \end{array} \tag{4.7}$$

$cos_1(\alpha), sin_1(\alpha)$	...	$cos_i(\alpha), sin_i(\alpha)$	...	$cos_{m-1}(\alpha), sin_{m-1}(\alpha)$	labels
$\frac{f_1 \cdot f_2}{f_1 \cdot f_1}, \frac{f_1 \times f_2}{f_1 \cdot f_1}$	...	$\frac{f_i \cdot f_{i+1}}{f_i \cdot f_i}, \frac{f_i \times f_{i+1}}{f_i \cdot f_i}$	...	$\frac{f_{m-1} \cdot f_m}{f_{m-1} \cdot f_{m-1}}, \frac{f_{m-1} \times f_m}{f_{m-1} \cdot f_{m-1}}$	sig_0
$\frac{f_1 \cdot f_2}{f_1 \cdot f_1}, \frac{f_1 \times f_2}{f_1 \cdot f_1}$	...	$\frac{f_i \cdot f_{i+1}}{f_i \cdot f_i}, \frac{f_i \times f_{i+1}}{f_i \cdot f_i}$	...	$\frac{f_{m-1} \cdot f_m}{f_{m-1} \cdot f_{m-1}}, \frac{f_{m-1} \times f_m}{f_{m-1} \cdot f_{m-1}}$	sig_1
...					
$\frac{f_1 \cdot f_2}{f_1 \cdot f_1}, \frac{f_1 \times f_2}{f_1 \cdot f_1}$	...	$\frac{f_i \cdot f_{i+1}}{f_i \cdot f_i}, \frac{f_i \times f_{i+1}}{f_i \cdot f_i}$	...	$\frac{f_{m-1} \cdot f_m}{f_{m-1} \cdot f_{m-1}}, \frac{f_{m-1} \times f_m}{f_{m-1} \cdot f_{m-1}}$	sig_n

(4.8)

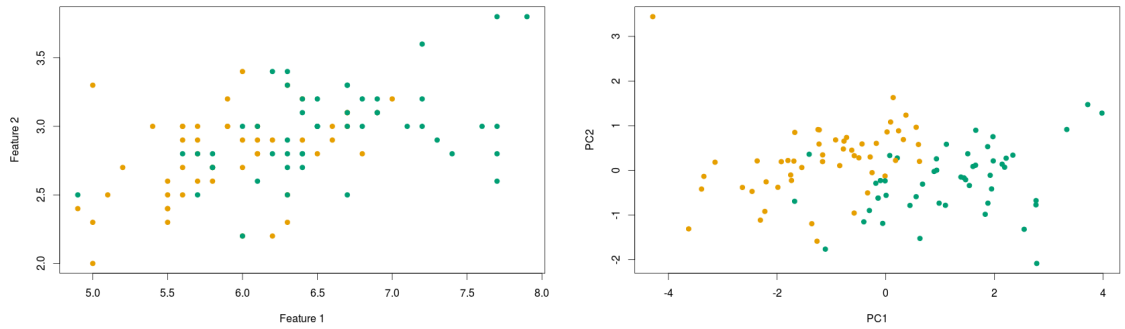
4.2 Feature Reduction

The data table presented in Equation 4.8 almost doubles the amount of information present after the signals have been binned. It seems as though we have taken a step back from the goal of simplifying the data. A step for feature reduction is necessary to achieve the goal of simplified classification. The algorithm used to reduce the features is Principal Component Analysis (PCA).

When reducing the dimensionality of the data there are a number of ways to do so. It would be possible to select a random subset of dimensions, or to construct new features based on an arithmetic combination of multiple features but PCA looks to reduce the dimensionality by finding the directions of greatest vari-

ation of your dataset. Every observation in your dataset can be constructed as a linear combination of its features where $sig_i(t) = b_0t_0 + b_1t_1 + ... + b_mt_m$ such that the b_i variables are constants to be manipulated. These linear functions can be changed by altering the constants to find the maximum variance between the line and the dataset. Using matrix algebra helps us to find the directions of greatest variance. Given our dataset R, we can compute the correlation matrix CM, of R. The eigenvalues of the matrix CM are the orthogonal directions of greatest variance also called principal components. The first eigenvectors is the direction of maximum variance, second greatest variance is the second eigenvalue and it descends respectively with each eigenvalue [Jol90]. There are associated eigenvectors for each eigenvalue. If we multiply our data by the matrix made up of these eigenvectors as columns we get a new dataset mapped to its principal component space where each axis is a principal component.

To show the affect this process can have on data we give an example. Looking at Figure 9 we see two images, figure 9a is a plot of some test data in two of its four dimensions. Notice that the data in these two dimensions is not very separable, meaning we cant really draw a line or plot a function that separates the data in a way that we can say data on this side is mainly of type 1 while data on the other is of type 2. However, if we find the principal components of the data and scale all the dimensions based on the principal components we can see that in only two dimensions, the first and second principal components, the data becomes vastly more separable. This seperation can be seen in Figure 9b.



(a) Two Dimensions of Raw Data Before the Principal Components are Found.

(b) The Same Data Scaled by its Two Best Principal Components.

Figure 9. Principal Component Analysis

The process of scaling the data like this in relation to its principal components is the essential piece to feature reduction in this work. Given the dataset R we split the data by their classes. Each of those datasets are split into two other datasets. The first with only the specified class's cosine features and the second with the class's sine features. Each new dataset has the principal components calculated and is then scaled by its associated eigenvectors, as discussed above. This is useful because we can choose how many dimensions we want to use for classification. Instead of using all $2 * (m - 1)$ cosine and sine values we can choose a subset of them based on the dimensions of greatest variance. For example, we can choose the first three sine and cosine columns for each class. This information is the data scaled by the first three sine and the first three cosine principal components for each class. The dimensionality has been greatly reduced from tens of thousands of cosine and sine values to just a few dimensions. This data is then ready for classification. In Chapter V, the results for varying the number of principal component dimensions that give the best classification accuracy is shown.

4.3 Classification

The machine learning algorithm used to classify the data for this work is the Random Forest Algorithm first proposed by Leo Brieman [Bre01]. The name suggests it is an algorithm that uses many decision trees. The random forest is an algorithm that divides the data into many subspaces and builds decision trees for those subspaces. The subspaces are built randomly and the observations produced for those subspaces are selected randomly as well, using a statistical method called bootstrapping. The bootstrapping method is random selection with replacement which allows for observations to be selected many times. This is an important feature in the random forest algorithm because it allows for outliers to be less prominent and the data centralized to the classes to be more isolated [Sut15]. It also builds in the validation step of the 3 step train-validation-test model building process for machine learning.

The data was split into two sets: train and test. The set sizes were varied to find optimal train sizes for the best classification accuracies and to avoid over-fitting. Over-fitting happens when models are over-trained for the specific dataset and lose the ability to detect generalized patterns to correctly classify unseen data. Experimentation varied the training sizes from 5% to 75% of the total dataset and was incremented by 5. The test set was the remaining amount of the dataset. This gives us information into how much of the data is necessary for a successful model to be built.

The entire process has been mapped out visually in Figure 10. The experimentation has been centered in 3 locations and are highlighted in blue: the number of bins, number of principal component dimensions, and the size of the training/testing

sets. These three parameters are what directly affect the feasibility of this method, thus we vary these parameters in isolation to find an optimal solution. There is also a comparison to this method with other methods shown below. The comparison shows the different algorithmic steps for the popular feature selection methods mentioned in Chapter II, MFCC and PLP, in comparison to the above method. The steps in Figure 11 do not include as much detail as shown in Figure 10 because it leaves out the steps of data management and classification. It is easy to see in the side-by-side comparison that this new method is much shorter algorithmically and has the potential for much quicker real time results.

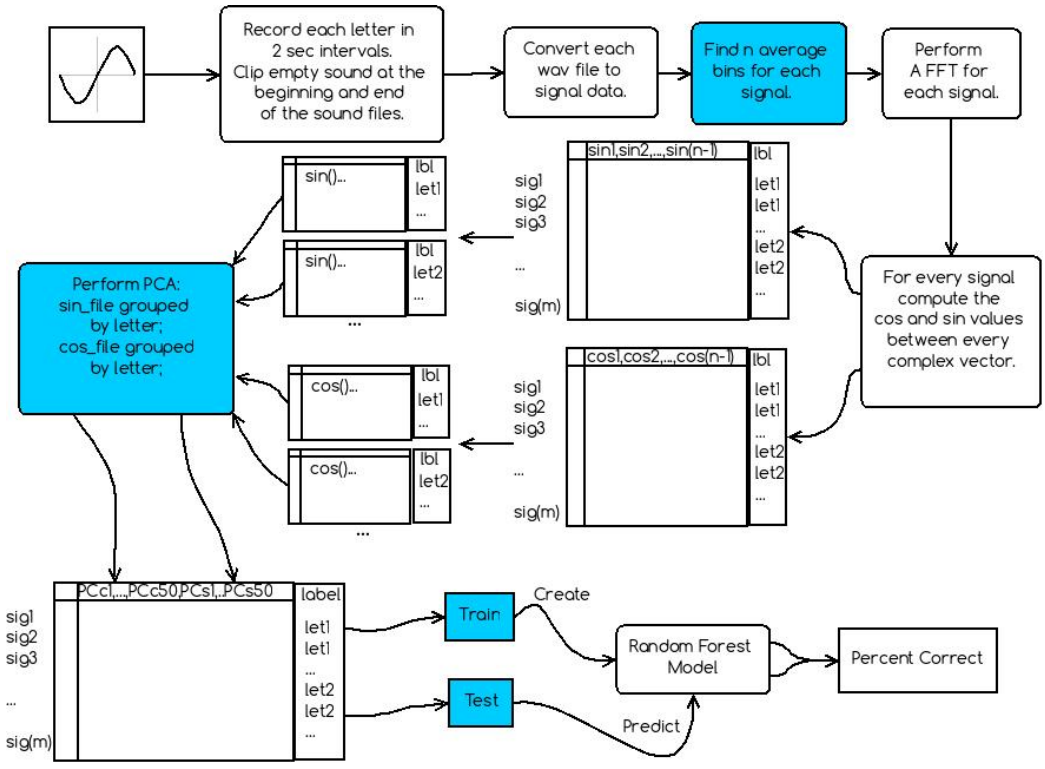


Figure 10. Visual Representation of the Letter Classification Methodology.

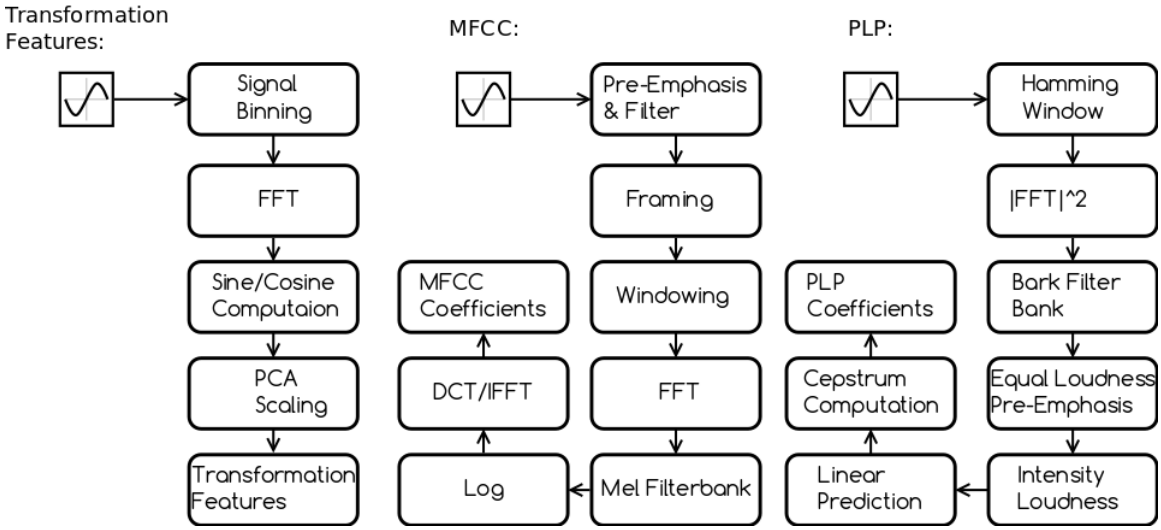


Figure 11. Comparison of Feature Extraction Methods [Dav13, AS+12].

CHAPTER V

RESULTS

In this section the results of experimentation are shown. For every experiment 100 simulations are run per variable to best understand the overall quality of the classification. This is done because training and test sets are generated randomly. Generating 100 different training sets per simulation is plenty to have a complete understanding of the realistic classification accuracy. As stated in the last section, the three parameters that are tested are the number of principal component dimensions, the size of the training/testing sets, and the number of bins. There is an additional section that shows the results of the cleaned values. This is to see if removing noise will help classification. The subsections that display specific results are respectively in this order. Discussion of the results follows in the next chapter.

The next six figures, that is Figures 12,13,14,15,16,17, provide the results for each data subset m and n, f and s, and the eset for the different bin sizes. The dots represent the average accuracy from 100 trials of randomly selected training samples. There are three graphs that display the accuracies based on the number of principal component dimensions and three graphs that display the accuracies based on the size of the training set in percentages of the full dataset. Each of the six graphs also provide a comparison of the different bin sizes. Each section thereafter will reference these graphs and provides a table of the best scores focused on the particular component of that section.

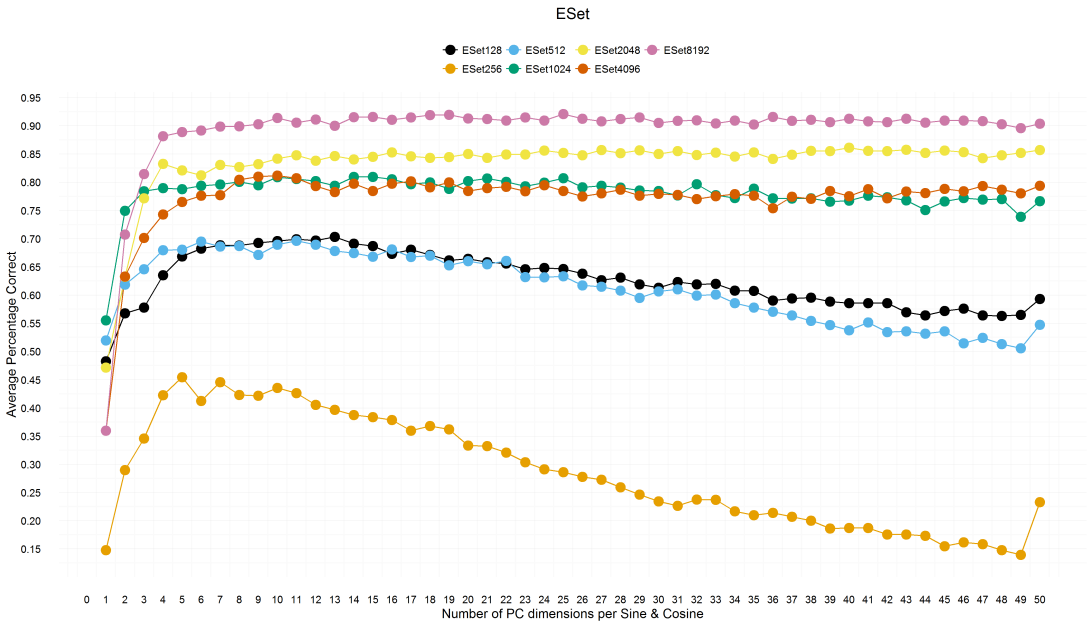


Figure 12. Eset Results for Different Principal Component Dimensions. The Different Colors Represent the Different Bin Sizes.

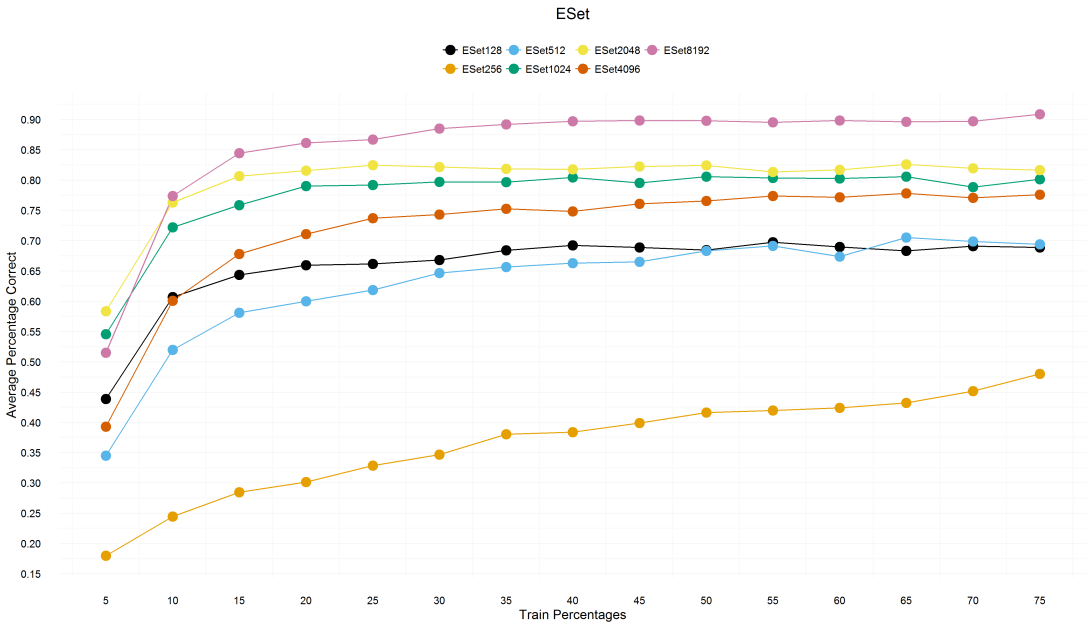


Figure 13. Eset Results for Different Training Sizes Based on Percentages of the Original Dataset. The Different Colors Represent the Different Bin Sizes.

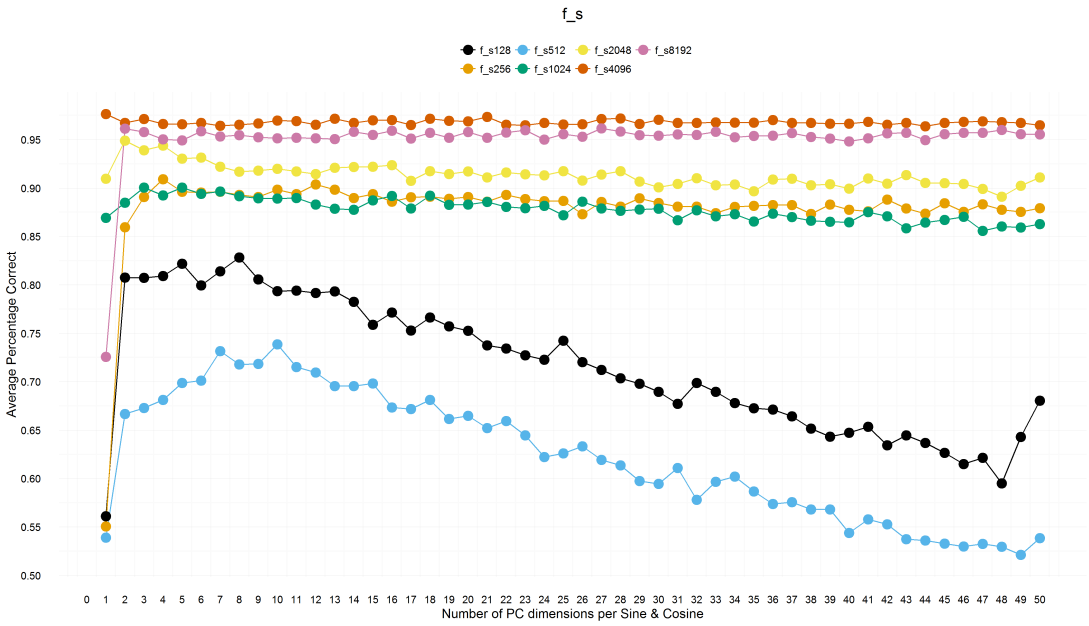


Figure 14. F,S Results for Different Principal Component Dimensions. The Different Colors Represent the Different Bin Sizes.

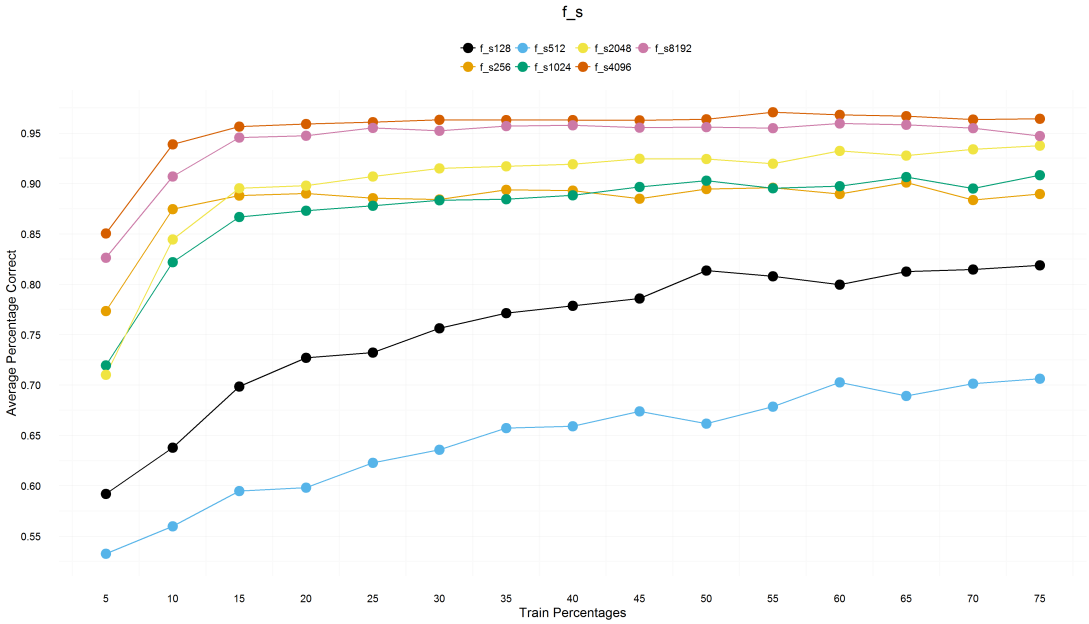


Figure 15. F,S Results for Different Training Sizes Based on Percentages of the Original Dataset. The Different Colors Represent the Different Bin Sizes.

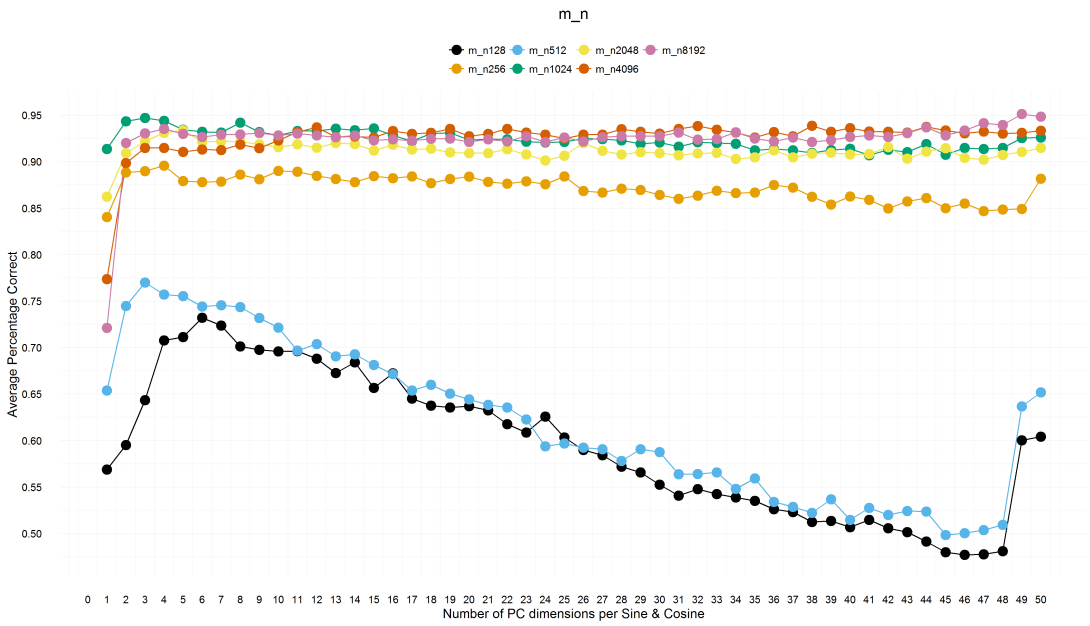


Figure 16. M,N Results for Different Principal Component Dimensions. The Different Colors Represent the Different Bin Sizes.

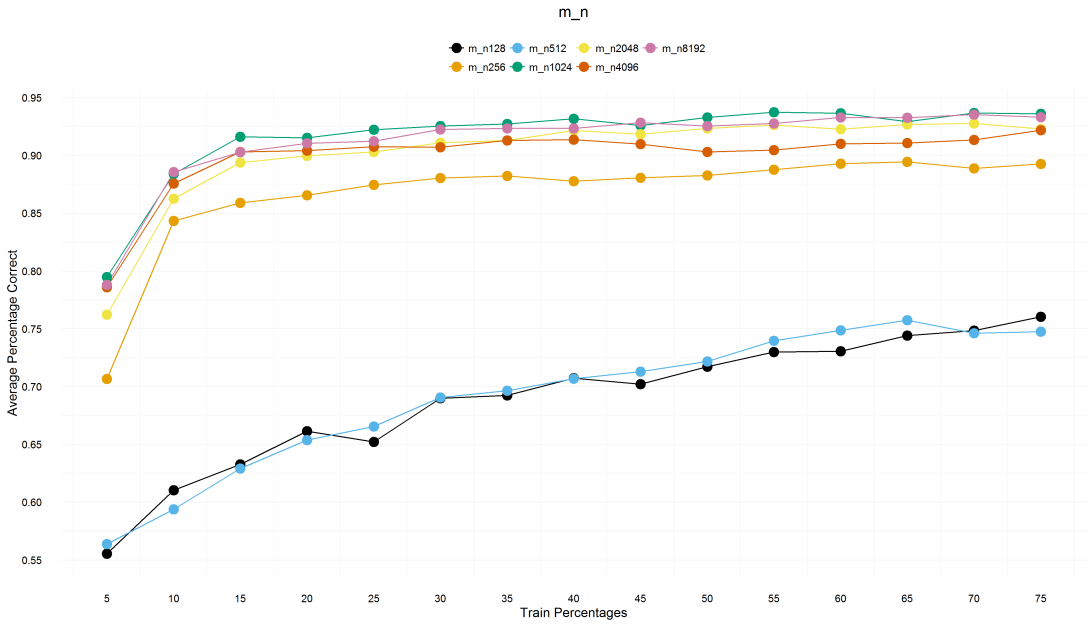


Figure 17. M,N Results For Different Training Sizes Based on Percentages of The Original Dataset. The Different Colors Represent the Different Bin Sizes.

5.1 Principal Component Analysis Experiment

In this experiment we varied the the number of principal component dimensions that we viewed the data in. There was a total of 50 principal components found for both sine and cosine. The data was scaled by these principal components so there is a total of 100 dimensions, or features, for the full dataset. For each principal component dimension 100 models were built for 100 randomly selected training sets. Figures 12, 14, and 16 show the average accuracies for the tests. Each test used a training set size of 65% of the original data. The purpose of this experiment is to find out how many dimensions are necessary to get accurate recognition. The nature of the principal components tells us the direction of greatest variance is the first principal component. It is a fair hypothesis that the majority of the important information necessary for classification will be captured in the first few dimensions of each sine and cosine.

Table 1. Show the Best Accuracies for the Corresponding Principal Component Dimensions.

Data Subset	Best Accuracy Percentage	PCA per sine and cosine
F,S	97.63	1
M,N	95.11	49
Eset	92.08	25

5.2 Train/Test Size Experiment

This experiment tests how much information is needed to develop an accurate model. For each training set size, 100 models were built for 100 randomly selected training sets. Figures 13, 15, and 17 show the average accuracies for the tests. Each test used 8 principal component dimensions. General intuition says that the more

data you train your model on the better it will perform. The problem of over-fitting the data is one that needs to be considered very seriously when building predictive and classifying models. If the model can perform well on a smaller percentage of training data the features used to build the model are highly scalable. It is expected that a small percentage of training data is necessary because the spoken letters come from only a single person. It should be expected that more training data should be needed if the dataset is generated from a variety of speakers.

Table 2. Show the Best Accuracies and the Corresponding Training Set Sizes.

Data Subset	Best Accuracy Percentage	Training Size
F,S	97.06	55
M,N	93.53	70
Eset	90.87	75

5.3 Initial Signal Binning Experiment

The experiment involving varying the number of bins affects an initial prepossessing step. After the signals were recorded each signal was compressed into m bins by averaging. This was partly done to create signals of equal length to make processing easier in the methodology section. Although, a variety of windowing and binning functions are available[Har78, HRS02] before performing an FFT, this is one of the simplest type of binning but it is all that is needed. It is likely in these tests that the larger number of bins will yield better results. Binning allows for a certain level of smoothing of the data so it is possible that some averaging is necessary. Every training and principal component test was ruFn on the different datasets created from the different bin sizes. The results come from these tests. We can see clearly the difference in the bins in the Figures 12-17.

Table 3. Show the Best Accuracies and the Corresponding Bin Size.

Data Subset	Best Bin Size
F,S	4096
M,N	8192
Eset	8192

5.4 Noise and Without Noise

There are various ways to clean the values. The cleaning process in this research is to remove the less prominent frequencies from the signal data. After the FFT is performed a certain percentage of higher frequencies are removed from the data. These are where a lot of the noise values are found. The higher frequency values have a much smaller power and potentially a smaller effect on the overall signal. In the context of audio data, removing them intuitively seems like it will help the overall accuracies. The test is o see if not removing them is consistent with other studies that show that keeping noise in big data environments improves performance. The tests were run the same way as above with separate training and principal component tests with the same binning sizes. This table only displays the information for the dimensional analysis because it is enough to draw conclusions.

Table 4. Show the Best Accuracies For Noise Removed Data and the Corresponding Principal Component Dimensions.

Data Subset	Best Accuracy Percentage	PCA
F,S	91.55	14
M,N	95.82	4
Eset	91.68	12

CHAPTER VI

DISCUSSION

The first discussion point should be on the issue of noise in the data since the rest of the discussion will be on the results from the noisy data. The results from the noiseless signals had similar or worse results for each dataset. This show that we do not gain any additional information and possibly lose some predictive power by cleaning the signals. Since the goal of this work is to make the process of audio classification as simple and quick as possible we can say confidently that we can remove the steps of cleaning the audio because they are nonessential. This is not entirely the same result as was found in Dalessandros paper [Dal13], which stated that adding noise helped predictability. This is the case for the f and s classification but for the other two, we get practically the same result. It is not intuitive why this is the case for audio because noise often inhibits humans in everyday conversations from understanding what is being said. We can see that at the very least cleaning the audio with this methodology becomes irrelevant for the machines performance.

Focusing on the principal component dimensions, Table 1 showed us that for m and n and the eset the best results needed a lot more dimensions that f and s. While looking at Figures 12, 14, 16 we can see that the highest score is not much higher than the rest of the results. In fact, for m and n, most of the results higher than 3 principal component dimensions were above 93.5%. For the eset all of them above 7 dimensions were above 90%. These differences in accuracies are negligible when remembering the limitation of our dataset in this training and testing envi-

ronment. Looking at Figure 18 we can see why it is negligable . It shows the results of every 100 tests per dimension in the form of box-plots and bubbles. The bubbles represent the density of the values at that point and the red dots are the averages that were first represented in Figure 12. This particular set of accuracy scores shows that the scores are not spread out between a bunch of different values. This is because a similar set of letters are failing every time they are left to be in the test set. Therefore these small variations in average scores are the result of random chance selection of the training set. These outlier values can also be seen in Figure 19 and 20.

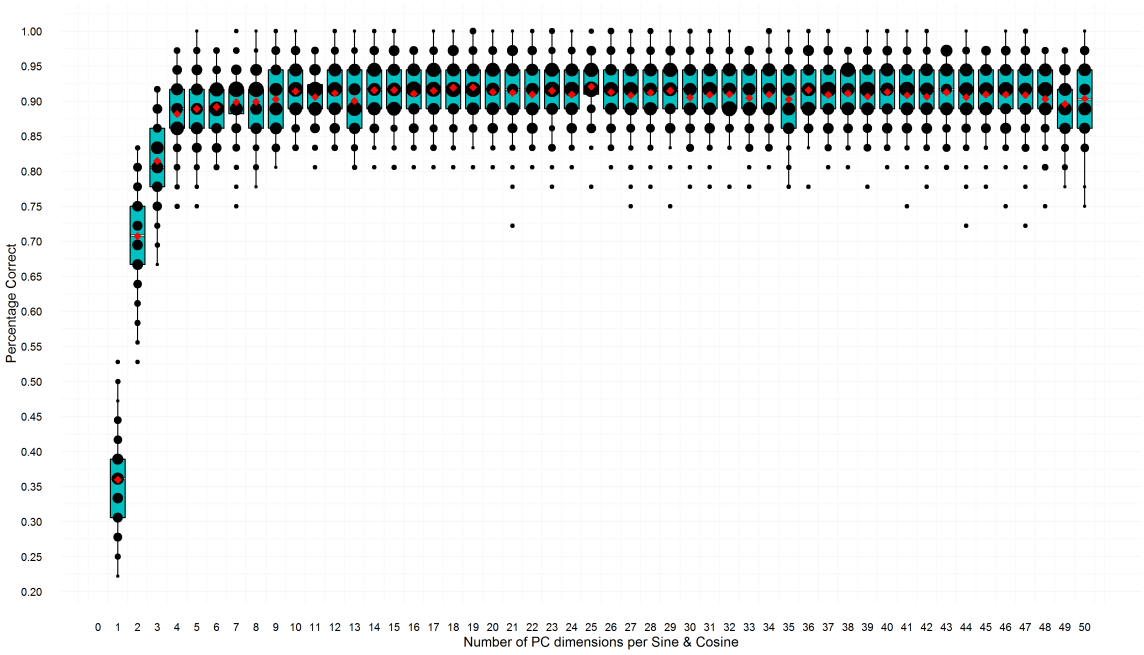
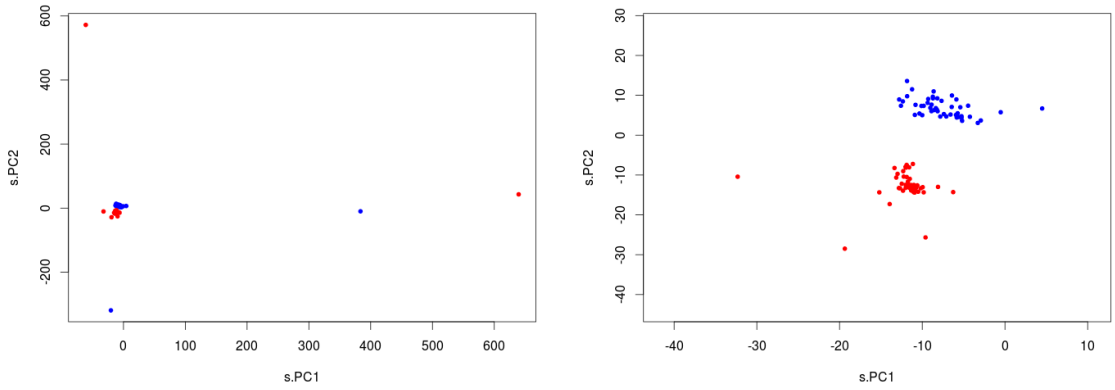


Figure 18. Specific Eset Results for Principal Component Dimensions with Bin Size 8192. The black bubbles represent the individual scores and the size of the bubbles represent the density of scores at that point. The red dots are the averages which are the same points found in Figure 12.

The hypothesis for the principal component dimensions was that there would not be a need for many dimensions because of the nature of principal component analysis. This can be shown best when we look at the plots in Figures 19 and 20. We see f points in blue and s in red. These plots show that the data is almost entirely separable in only two dimensions. This displays the true strength of this methodology. In comparison to other research in this area, to my knowledge, there has not been a feature reduction that can show spoken letters almost entirely separable in 2 dimensions. There are a few outliers which support the discussion in the previous paragraph and shows us the need for a slightly higher dimensional analysis.



(a) First Two Principal Component Dimensions for Sine F,S Data.

(b) Same Data Zoomed in to See the Separability of F,S.

Figure 19. The First Two Principal Component Dimensions for the Sine Values.

The training experiment shows that the higher amount of training data used to build a model the better the average accuracy. The hypothesis is still supported when we dig into the values found in the figures similar to our principal component dimension discussion. If we look at Figures 13, 15, 17 it can be seen that the high

average accuracy isnt much different from other training set sizes. In Figure 13 we can see that at training sizes at 40% and above we get accuracies of at least 90%. In Figure 15 we only need training sizes of 20% to receive accuracies above 95% and in Figure 17 it shows to receive an accuracy of 92% or higher we only need a training set of 15% of the original dataset. It is important to test whether or not these results occur because the methodology is highly scalable or simply because the dataset was generated from a single speaker. It needs to be confirmed but because the results are consistently high in different letter test sets it gives the impression of a highly scalable methodology.

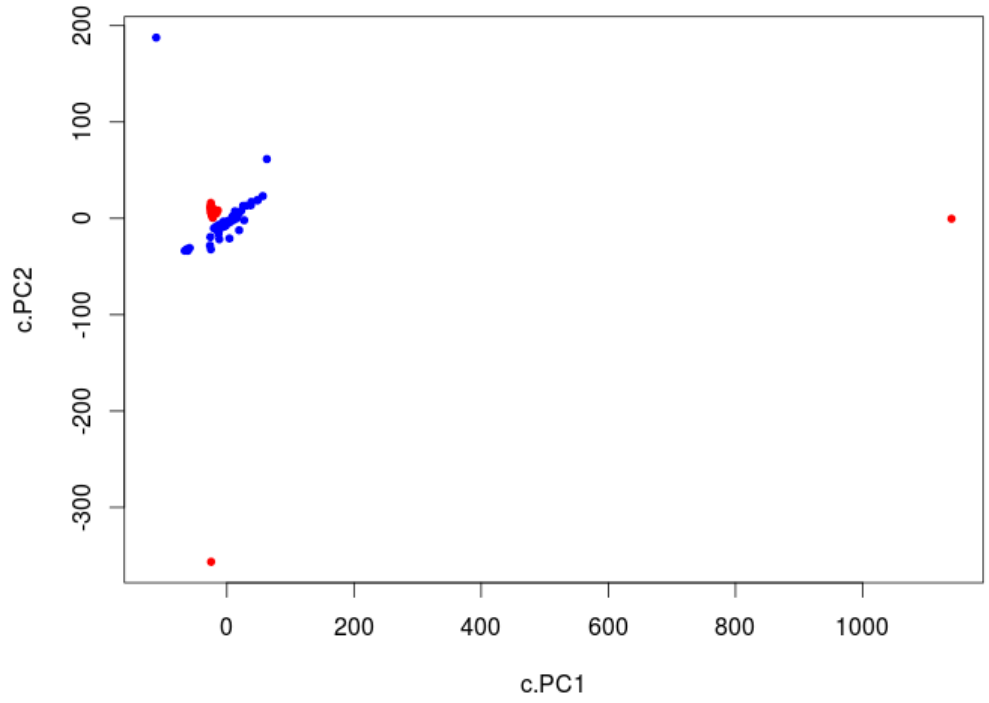


Figure 20. First Two Principal Component Dimensions For Cosine F,S Data.

The most inconsistent results have to be with picking the best bin sizes. Looking at Figures 12-17 different bin sizes seem to peak above others in an inconsistent way. Looking at Figure 16 the line for 1024 bins does really well in the beginning and then becomes about equivalent to 8192 and 4096. For the f and s test the two highest bin amounts, 4096 and 8192, do much better than the other bins, with 4096 doing slightly better than 8192. However, for the eset tests, 8192 does much better than the rest. One thing is clear, it is not necessarily that the lower number of bins, the worse the result, and the higher the better the result. Between all the tests 8192 does consistently better even if it isn't the best in every circumstance. These results are the most mysterious. The inconsistency is probably due to the phonetic makeup of the words. The average binning may blur important information based on the make up of the word. Further tests would need to be done to find out exactly what is the cause of this inconsistency.

CHAPTER VII

CONCLUSION

An analysis of spoken letter recognition has been shown in the letter sets: f,s, m,n, and b,c,d,e,g,p,t,v,z. These sets propose a large level of difficulty and has been the focus of other letter recognition research. The methods described in this paper offer a simplistic method for generating the features necessary for quick and accurate classification. The features are generated from a spatial analysis initially in Fourier Space and then in transformations of the data based on the cosine and sine values between the imaginary vectors returned from the Fast Fourier Transformation algorithm. The methods involve feature reduction via principal component analysis which allow our classification to be done in a low dimensional space. The random forest algorithm is a sufficient algorithm for classification due to the fact that the data is largely separable in low dimensions seen in Figures 19 and 20. This has allowed for a highly accurate classification rate for even the most confusable letters.

There is a variety of work that can still be done in this area. Some of the questions raised while reflecting on the results are why is there an inconsistency in the best bins and whether or not the low training sets necessary for good accuracies are due to the scalability of the methodology. To test the robustness of these methods a larger data set needs to be generated with a variety of speakers including male, female, native and non-native. Having a dataset with this large amount of variability would show how scalable the methods actually are. It would also give greater

variability with the same letter. The variability within the same letter is a good place to start researching why different bin sizes are necessary for different letters.

This work is a great proof of concept and seeks plenty of more research to be done to test the power of this simple feature selection process. This work could also be tested in many other problems such as speaker gender detection, accent detection, and speaker verification. These methods are algorithmically simple enough to possibly allow themselves to be integrated in native web or phone applications that wont need a cloud infrastructure for computation.

The biggest work that needs to be done is to see if these methods can be used in larger vocabulary problems. If word recognition can be done with an algorithmically simple process it would allow for great improvements to devices that use voice recognition. Large quantities of data and work are still needed to achieve this. The further we can improve speech recognition in any of the many sub-problems the further launched into the digital era we will be. To have machines that can analyze and recognize human spoken communication, the greater capacity we will have to allow machines to help us manage everyday situations.

REFERENCES

[AS+12] TB Adam, Md Salam, et al., *Spoken english alphabet recognition with mel frequency cepstral coefficients and back propagation neural networks*, International Journal of Computer Applications (0975–8887) Vol (2012).

[BDMD+07] Mohamed Benzeghiba, Renato De Mori, Olivier Deroo, Stephane Dupont, Teodora Erbes, Denis Jouvet, Luciano Fissore, Pietro Laface, Alfred Mertins, Christophe Ris, et al., *Automatic speech recognition and speech variability: A review*, Speech Communication **49** (2007), no. 10, 763–786.

[Bel14] Jerome R Bellegarda, *Spoken language understanding for natural interaction: The siri experience*, Natural Interaction with Robots, Knowbots and Smartphones, Springer, 2014, pp. 3–14.

[BH14] Samy Bengio and Georg Heigold, *Word embeddings for speech recognition.*, INTERSPEECH, 2014, pp. 1053–1057.

[BHMW93] Christoph Bregler, Hermunn Hild, Stefan Manke, and Alex Waibel, *Improving connected letter recognition by lipreading*, Acoustics, Speech, and Signal Processing, 1993. ICASSP-93., 1993 IEEE International Conference on, vol. 1, IEEE, 1993, pp. 557–560.

[BK94] Christoph Bregler and Yochai Konig, *"eigenlips" for robust speech recognition*, Acoustics, Speech, and Signal Processing, 1994. ICASSP-94., 1994 IEEE International Conference on, vol. 2, IEEE, 1994, pp. II–669.

[Blo04] Peter Bloomfield, *Fourier analysis of time series: an introduction*, John Wiley & Sons, 2004.

[Bre01] Leo Breiman, *Random forests*, Machine learning **45** (2001), no. 1, 5–32.

[BS13] Françoise Beaufays and Brian Strope, *Language model capitalization*, Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on, IEEE, 2013, pp. 6749–6752.

[CF90] Ronald Cole and Mark Fanty, *Spoken letter recognition*, Proc. Third DARPA Speech and Natural Language Workshop, 1990, pp. 385–390.

[CL16] Meng Cai and Jia Liu, *Maxout neurons for deep convolutional and lstm neural networks in speech recognition*, Speech Communication **77** (2016), 53–64.

[Col90] Ron Cole, *The isolet spoken letter database.*

[Dal13] Brian Dalessandro, *Bring the noise: Embracing randomness is the key to scaling up machine learning algorithms*, Big Data **1** (2013), no. 2, 110–112.

[Dav13] Namrata Dave, *Feature extraction methods lpc, plp and mfcc in speech recognition*, International Journal for Advance Research in Engineering and Technology **1** (2013), no. 6, 1–4.

[DC14] Li Deng and Jianshu Chen, *Sequence classification using the high-level features extracted from deep neural networks*, Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on, IEEE, 2014, pp. 6844–6848.

[DM80] Steven B Davis and Paul Mermelstein, *Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences*, Acoustics, Speech and Signal Processing, IEEE Transactions on **28** (1980), no. 4, 357–366.

[GGY10] Santosh K Gaikwad, Bharti W Gawali, and Pravin Yannawar, *A review on speech recognition technique*, International Journal of Computer Applications **10** (2010), no. 3, 16–24.

[GH15] Amir Gandomi and Murtaza Haider, *Beyond the hype: Big data concepts, methods, and analytics*, International Journal of Information Management **35** (2015), no. 2, 137–144.

[Gia15] Theodoros Giannakopoulos, *pyaudioanalysis: An open-source python library for audio signal analysis*, PLoS ONE **10** (2015), no. 12, e0144610.

[Har78] Fredric J Harris, *On the use of windows for harmonic analysis with the discrete fourier transform*, Proceedings of the IEEE **66** (1978), no. 1, 51–83.

[HBR14] Xuedong Huang, James Baker, and Raj Reddy, *A historical perspective of speech recognition*, Communications of the ACM **57** (2014), no. 1, 94–103.

[HDY+12] Geoffrey Hinton, Li Deng, Dong Yu, George E Dahl, Abdel-rahman Mohamed, Navdeep Jaitly, Andrew Senior, Vincent Vanhoucke, Patrick Nguyen, Tara N Sainath, et al., *Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups*, Signal Processing Magazine, IEEE **29** (2012), no. 6, 82–97.

[Her90] Hynek Hermansky, *Perceptual linear predictive (plp) analysis of speech*, the Journal of the Acoustical Society of America **87** (1990), no. 4, 1738–1752.

[HRS02] Gerhard Heinzel, Albrecht Rüdiger, and Roland Schilling, *Spectrum and spectral density estimation by the discrete fourier transform (dft), including a comprehensive list of window functions and some new at-top windows*.

[Jol90] Ian T Jolliffe, *Principal component analysis: a beginner's guide. introduction and application*, Weather **45** (1990), no. 10, 375–382.

[KJNN00] Roland Kuhn, Jean-Claude Junqua, Patrick Nguyen, and Nancy Niedzielski, *Rapid speaker adaptation in eigenvoice space*, Speech and Audio Processing, IEEE Transactions on **8** (2000), no. 6, 695–707.

[KZ01] Montri Karnjanadecha and Stephen A Zahorian, *Signal modeling for high-performance robust isolated word recognition*, Speech and Audio Processing, IEEE Transactions on **9** (2001), no. 6, 647–654.

[Lic13] M. Lichman, *UCI machine learning repository*, 2013.

[LN05] Chul Min Lee and Shrikanth S Narayanan, *Toward detecting emotions in spoken dialogs*, Speech and Audio Processing, IEEE Transactions on **13** (2005), no. 2, 293–303.

[LN14] Ming Li and Shrikanth Narayanan, *Simplified supervised i-vector modeling with application to robust and efficient language identification and speaker verification*, Computer Speech & Language **28** (2014), no. 4, 940–958.

[Mar02] Joseph Di Martino, *On the use of high order derivatives for high performance alphabet recognition*, Acoustics, Speech, and Signal Processing (ICASSP), 2002 IEEE International Conference on, vol. 1, IEEE, 2002, pp. I–953.

[MEKR11] Mathias Muller, Daniel PW Ellis, Anssi Klapuri, and Guilhem Richard, *Signal processing for music analysis*, Selected Topics in Signal Processing, IEEE Journal of **5** (2011), no. 6, 1088–1110.

[O'S08] Douglas O'Shaughnessy, *Invited paper: Automatic speech recognition: History, methods and challenges*, Pattern Recognition **41** (2008), no. 10, 2965–2979.

[Sch10] Mike Schuster, *Speech recognition for mobile devices at google*, PRICAI 2010: Trends in Artificial Intelligence, Springer, 2010, pp. 8–10.

[Sut14] Shan Suthaharan, *Big data classification: Problems and challenges in network intrusion prediction with machine learning*, ACM SIGMETRICS Performance Evaluation Review **41** (2014), no. 4, 70–73.

[Sut15] Shanmugathasan Suthaharan, *Machine learning models and algorithms for big data classification: Thinking with examples for effective learning*, vol. 36, Springer, 2015.

[TP91] Matthew Turk and Alex Pentland, *Eigenfaces for recognition*, Journal of cognitive neuroscience **3** (1991), no. 1, 71–86.

[Wel67] Peter D Welch, *The use of fast fourier transform for the estimation of power spectra: A method based on time averaging over short, modified periodograms*, IEEE Transactions on audio and electroacoustics **15** (1967), no. 2, 70–73.

APPENDIX A

TRANSFORMATION MATRIX

Given,

$$v_1 = \begin{bmatrix} a \\ b \end{bmatrix}, v_2 = \begin{bmatrix} c \\ d \end{bmatrix}, A = \begin{bmatrix} \frac{v_1 \cdot v_2}{||v_1||^2} & -\frac{v_1 \times v_2}{||v_1||^2} \\ \frac{v_1 \times v_2}{||v_1||^2} & \frac{v_1 \cdot v_2}{||v_1||^2} \end{bmatrix}.$$

We will show that:

$$A \cdot v_1 = v_2. \tag{1.1}$$

$$\begin{bmatrix} \frac{\begin{bmatrix} a \\ b \end{bmatrix} \cdot \begin{bmatrix} c \\ d \end{bmatrix}}{||\begin{bmatrix} a \\ b \end{bmatrix}||^2} & -\frac{\begin{bmatrix} a \\ b \end{bmatrix} \times \begin{bmatrix} c \\ d \end{bmatrix}}{||\begin{bmatrix} a \\ b \end{bmatrix}||^2} \\ \frac{\begin{bmatrix} a \\ b \end{bmatrix} \times \begin{bmatrix} c \\ d \end{bmatrix}}{||\begin{bmatrix} a \\ b \end{bmatrix}||^2} & \frac{\begin{bmatrix} a \\ b \end{bmatrix} \cdot \begin{bmatrix} c \\ d \end{bmatrix}}{||\begin{bmatrix} a \\ b \end{bmatrix}||^2} \end{bmatrix} \cdot \begin{bmatrix} a \\ b \end{bmatrix} \implies \begin{bmatrix} \frac{ac+bd}{\sqrt{a^2+b^2}^2} & -\frac{ad-bc}{\sqrt{a^2+b^2}^2} \\ \frac{ad-bc}{\sqrt{a^2+b^2}^2} & \frac{ac+bd}{\sqrt{a^2+b^2}^2} \end{bmatrix} \cdot \begin{bmatrix} a \\ b \end{bmatrix}$$

$$= \begin{bmatrix} a\frac{ac+bd}{\sqrt{a^2+b^2}^2} + -b\frac{ad-bc}{\sqrt{a^2+b^2}^2} \\ a\frac{ad-bc}{\sqrt{a^2+b^2}^2} + b\frac{ac+bd}{\sqrt{a^2+b^2}^2} \end{bmatrix} \implies \begin{bmatrix} \frac{a^2c+abd}{a^2+b^2} + \frac{-bad+b^2c}{a^2+b^2} \\ \frac{a^2d-abc}{a^2+b^2} + \frac{bac+b^2d}{a^2+b^2} \end{bmatrix}$$

$$\implies \begin{bmatrix} \frac{a^2c+abd+-bad+b^2c}{a^2+b^2} \\ \frac{a^2d-abc+bac+b^2d}{a^2+b^2} \end{bmatrix} \implies \begin{bmatrix} \frac{a^2c+b^2c}{a^2+b^2} \\ \frac{a^2d+b^2d}{a^2+b^2} \end{bmatrix} \implies \begin{bmatrix} \frac{c(a^2+b^2)}{a^2+b^2} \\ \frac{d(a^2+b^2)}{a^2+b^2} \end{bmatrix} = \begin{bmatrix} c \\ d \end{bmatrix}$$