Entropy and Efficiency of Voice Recognition Authentication

Joshua Dow

Abstract— **Is vocal recognition a secure and efficient method for user authentication compared to traditional plaintext methods? Our project explores the different methods of calculating the security of vocal passwords, and compares them with their plaintext counterparts as a baseline. We explore issues related to feasibility, safety and security of vocal-passwords.**

I. INTRODUCTION

Biometric authentication systems have been growing in popularity in the last few years, especially due to the convenience of using these systems as they require no memorization or external knowledge[17]. As technology increasingly becomes bigger parts of our lives, the need for convenient security methods also increases. However, it is important to balance user needs with the physical aspects of any new system, such as its efficiency and security[17]. Some studies suggest that vocal recognition is an excellent authentication system that should be considered moving on into the future. However, voice biometrics remain relatively overlooked and under tested[24].

Currently existing systems are often text-based, requiring users to remember passcodes and enter them every time they use a service[17]. Unfortunately, text based passwords often contain several common flaws. First of all, the human brain remembers patterns easier than a random string of characters, leading many people to create extremely predictable passwords that are far more likely to guess than brute force attacks would suggest[17]. As well, the existing system of text-based passwords have not scaled well. As the internet continues to grow, new websites continue to pop up, and it is usually recommended to use a new password for every website. When combined with certain websites requiring a password change frequently, this is likely to lead to password loss and frustrated users[17]. Apart from usability complaints for text-based passwords, there are also several security concerns. Due to the fact that the average user will not create an extremely long and random passwords, the security of average passwords is not very high. New attacks such as dictionary and pattern matching exploit these weaknesses, showing that some change is necessary[17].

Although there exist several alternate methods for authentication systems, vocal recognition stands out due to its ease of use. In a world where 96% of people own cellphones, there is no shortages of microphones and other infrastructures required for voice authentication[22]. On the user side, vocal authentication is faster, and requires less memorization then text-based passwords[24]. There exist some concerns regarding the information contained in audio files, as it is possible to extract information such as education level and ethnicity from vocal samples, but when compared to other highly implemented biometric authentication techniques such as finger prints, or facial recognition vocal files is less intrusive[18]. As well, the level of information kept in databases depends strongly on the implementation of voice biometrics[23].

Although there exist many different specific techniques for carrying out voice biometrics, they all follow the same rough idea. Vocal samples are taken from the user and crunched through a template. This is used to create a user token which is then compared to any future vocal samples to provide match confirmation[23]. How this system is implemented can greatly impact its efficiency and security. Due to the large quantity of information contained in audio files, it is necessary to cut out any excess, both to adjust for background noise and slight differences in each individuals speech, but also to prevent information overload on the servers[24]. It is important to consider the specific implementation of voice biometrics being used when studying it.

Moving into the future, several studies have predicted the rise of biometric authentication systems[18][24]. However, voice biometrics remain a relatively new field and not many studies have been done comparing it's use or security to existing authentication systems. In this study we were interested in the potential rise of voice biometrics and determining if it is a more secure and efficient method to use then text-based passwords. We explored several different methods of dealing with the complexity of calculating the entropy of vocal passwords and made conclusions by comparing these entropies with those of the plaintext versions of these passwords.

II. Problem

A. *Security of Voice Recognition*

The measuring of entropy of vocal passwords is no easy task but later on we will explore different method of how we came to our results. This part of the problem relies on what is the most reliable way to calculate the entropy of vocal passwords? And compared to the entropy of corresponding text based passwords do vocal passwords exceed or fall short? Which factors in particular allow for a vocal password to be more secure? Tone? Pitch? Speed? All of these sub problems build to the last piece of this problem which is how secure is the average one-word vocal password?

B. *Efficiency*

Given that vocal passwords most certainly take more data to store, analyze and verify we wish to find an algorithm for these things which allows for quick computation time of storing and verifying these vocal passwords. It is not feasible to have users wait 5+ seconds in today's world to verify and access their own data, it must be faster. Although for these password we want to maximize the security of these passwords, but the higher the level of entropy, the larger amount of data is required for this password and it slows down computation heavily. We will touch on finding a balance lastly in this section. However to end here we want to know which algorithm, and length of vocal password allows for efficient computations?

C. *Methods*

We have several different methods for calculating entropy of vocal passwords; Google text-to-speech API, our own python scripts, frequency analysis, arithmetic encoding, etc. These will be outlined in detail during our proposed work and experiment sections, including formal proofs, however we wish to see which of these methods allows us to reliably predict and evaluate the entropy of a vocal password. Which of these methods of producing the entropy of vocal passwords is easiest in terms of computation, but allows for the maximum security of vocal passwords. For example, if you took a range of frequency and generated an optimal prefix code using Arithmetic Coding, you would find that the entropy is quite large however it is extremely susceptible to analysis of the codes and it quite breakable. We will do these comparisons of all of our methods and determine the best.

D. *Balance*

Our last point to touch on is the balance of security and efficiency. Which method(s) of analyzing a vocal password allow for maximum security while still being computationally efficient. For our context we wish to say that verification of a vocal password takes at most two seconds to validate. What is the ideal amount of entropy a vocal password must have such that it is secure and easy to validate/store?

III. Proposed Work

We have two goals in mind with this project. First we would like to be able to calculate the approximate entropy of a one-word passphrase. This passphrase must be spoken by the same user in the same way in order for it to be verified. The second goal is to see how reliable a system built by us (off of the common voice recognition models) will work, when it comes to reliability. We will also attempt to provide an entropy for our rudimentary approximation of what a real-life voice recognition system what look like.

Voice recognition works in generally this way: An audio sample is read in, that audio sample is then broken into it's various components of frequency and spectral feature. It is not very easy however, to directly compare two audio files, so a model is fitted to the data of it's components, and the model is then used for comparisons against other models, generated in the same way. The space between between the two distributions can then be used to quantify the distance or the similarity between the two voice samples.

A. Measuring Audio

The most popular way to transform and represent an arbitrary file for use in voice recognition is the use of the "Mel-Frequency Cepstrum" (MFC)[1]. Mel-Frequency Cepstrum Coefficients (MFCC's) are a number of coefficients that together form a set of data that can accurately describe the spectral features of an audio clip. A cepstrum is described as the inverse Fourier transform of the logarithm of the spectrum (frequency spectrum) of a particular signal. The difference between a regular cepstrum and a Mel-Frequency spectrum is that the representation of the different frequences have equal distance between in them when mapped using the Mel-scale, which is much more similar to how humans perceive audio signals.

MFCC's work by Framing; cutting the signal into short frames. For each frame calculate the periodogram estimate of the power spectrum. Apply the mel filterbank to the power spectra, sum the energy in each filter. Take the logarithm of all filterbank energies. Take the Discrete Cosine Transform of the log filterbank energies. Keep DCT coefficients 2-13, discard the rest.

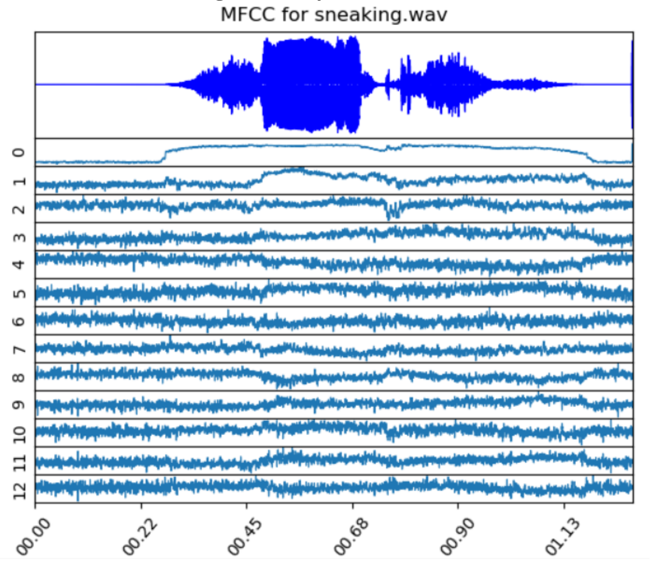
MFCC's are not the only way to represent the spectral features of a sound clip, it is one of many methods of signal processing. It works well because it uses overlapping sound frames to have increased accuracy, as well as because the Mel-scale works favourably when working with human auditory capacities.

After transforming our audio clip into MFCC's, we are left with a matrix C, where Cij is the i'th coefficient at j frames.

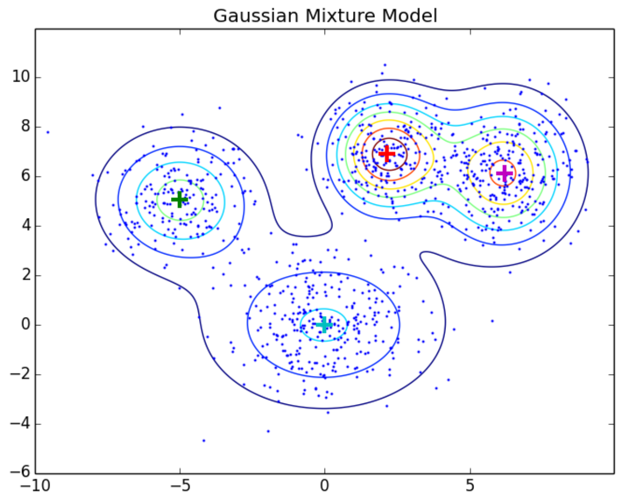
The more frames that are taken for the MFCC, the more accurate the model generated will be. Therefore we will use the smallest frame size allowed by the python library we are using, which is 64 ms per frame.

B. Modelling the Data

An MFCC generally looks like this:



as you can probably tell, this would be extremely difficult to fit a convential statistic model to. Therefor, we have to rely on a mixture model. A a mixture model is a probabilistic model for representing the presence of subpopulations within an overall population, without requiring that an observed data set should identify the sub-population to which an individual observation belongs[1]. For example, see this example graph:



[16]

Each of those clusters is a separate gaussian model. This data would be very difficult to model with just 1 gaussian model, but when split up, we can see that it can be modelled very accurately as a mixture of 4 different gaussian densities. The same idea applies when it comes to modelling our MFCC. Since voice patterns can be unpredictable and wildly varying from word to word, person to person, there are still patterns that exist within subsections of a person's speech. Intuitively, it makes sense that a gaussian mixture would be able to accurately represent an MFCC. We will fit our MFCC data to a gaussian mixture model using a technique called Expectation-Maximization[2]. Firstly we assume randomly centered data points from either a k number of means $(k \in Z)$ or a normal distribution. We then compute a probability of being generated from each component for each point. The parameters are then altered to maximize the likelihood of the data given those probabilities. By repetition of this process we always end of converging to a local optimum

Our data can then be represented as a Gaussian Mixture Model formally:

$$p(x) = \sum_{j=1}^{K} w_j * N(x|\mu_j, \sigma_j)$$

[15]

For the purposes of our experiment, we will be limiting the amount of components in our GMM to 10. The more components there are to a GMM, the more accurate it will be, but the complexity of the system will go up immensely.

C. *Entropy Calculation*

Unfortunately, entropy calculation on a mixture model is notoriously difficult to provide a closed-form solution. However, there have been numerous attempts to estimate the general entropy, and some of these have proven to be quite close.

To calculate the entropy of the Gaussian mixture model of our voice samples, which is continuous function, we calculated differential entropy. And in order to find the differential entropy of our model, we used component-wise entropy estimation method,

$$H(x) = \frac{1}{2}\log|2\pi e\mathbf{C}|$$

where **C** is the covariance matrix of the Gaussian mixture component.

Using this equation, we first approximated the entropy of each component of Gaussian mixture. And then we calculated the expected entropy,

$$H_{exp} = \sum_{i=1}^{N} \frac{w_i}{w_{sum}} \cdot H(x_i)$$

where N, w are the number of components and the weight of the component respectively.

Our process for determining the entropy of a vocal password having been read into the actual word, encoding, and decoding[19]:

1) Read in an audio format file into a byte array.
2) Convert this byte array into an array which contains unsigned integers which are assigned to unique bytes from the original array.
3) Convert the unsigned integer array into ASCII characters
4) Apply Arithmetic Coding as follows[3]:
    a) We first determine l the length of the string, and from here on we will work in base l
    b) We find the frequency of each letter in the string.
    c) Next we calculate the lower bound of number of bits required to represent the message. $L = \sum_{i=1}^{n} n^{n-i} j_i \prod_{a=1}^{i-1} f_a$ where j_i is the cumulative frequency, f_a is the number of occurrences.
    d) Now the upper bound. $U = L + \prod_{a=1}^{n} f_a$
    e) Lastly is the choice of which value in $[L, U)$ to choose. The best case is the number with the greatest number of trailing zeroes. This allows the zeroes to be truncated later.
    f) Note that the final result will have the same number of digits as the highest power of base l that you are working in. So, to decode the integer and get back to the original message we simply take our choice c and divide it by l^p to get r where p is the length of our choice. Then we update the next number to divide, which is calculated by: $\frac{(c - l^p * r)}{r}$ and we repeat this last part until our message is recovered.

5) Our entropy is calculated conservatively by the lower bound L from above.[4]

D. *In Practice*

In addition to the formal model, we will also be using an estimated model to compare two voice files, based off of the concepts we have discussed. However, for the purposes of time and complexity, we have made several subsitutions. Instead of fitting an MFCC to our audio data, we will be sampling the frequency of the data every 64ms, scaling the audio file to 1 minute length so that all of our files will have the same number of data points. In this way, we still end up with a distribution of sampled frequency, which we use in place of an MFCC.

Instead of using a gaussian mixture model to model the distribution of our data, since our data is now simpler, we use a simple n degree polynomial to model the shape of our distribution. We can then compare the polynomial distributions using the Kolmogorov-Smirnov test to test the statistical distance between the distributions.

E. *Complexity*

Given that vocal recognition is substantially more difficult to analyze and compare then text parsing, we have chosen to limit the scope of our project to a one-word passcode, between one to two syllables[1]. This will sufficiently allow us to properly compare the average complexity and entropy of these passwords.

IV. Evaluation

A. *Objectives*

We wish to show whether a vocal password is more or less optimal than a typical plain text password in terms of entropy. We will use our methods described above in order to calculate the estimated entropy for a given voice password. We will also then use our own method to test for the accuracy and safety of a rudimentary voice verification method.

B. *Technology*

To make life easier for ourselves, we utilize a number of python libraries in order to simplify things. We will be using the scipy, numpy, matplotlib, aubio, lmfit, pydub, pyaudio, and the sklearn libraries. These provide us a number of functions, namely the ability to process and clean audio, MFCC tools, and GMM fitting tools.

Finally, We will utilize the SpeechRecognition 3.8.1 library, which is a library for Python that allows easy access to various voice recognition API's, such as Google's and Bing's[4]. We will largely be using Google's Voice Recognition and Bing's Speech recognition APIs, as both of those have clear documentation and provide a level of detail not supplied by the other APIs[4]. These APIs allow us to control for things such as length of the recording, accent of the speaker, as well as the ambient noise in the background. They also allow us to not only see the interpreted statement, but a list of statements that may also have been said, as no voice recognition model is perfect under all scenarios[4].

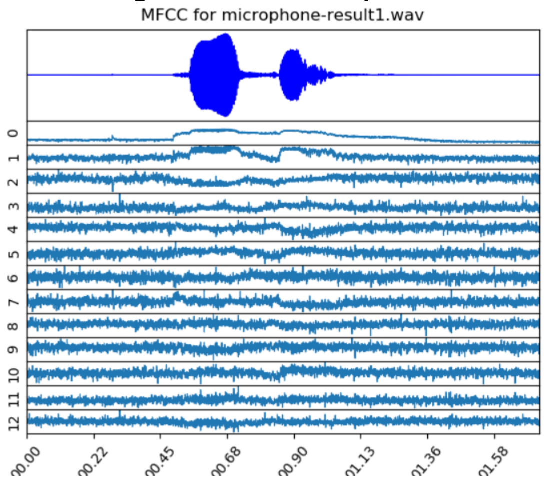
C. *Experiments*

Our experiments are designed to test individual and composite changes to vocal passwords to see the effect on Entropy, efficiency, and ease-of-use.

1) Analyzing a single word
2) Comparing the same word, said the same way
3) Comparing the same word, said differently
4) Comparing two different words

D. *Analyzing a Single Word*

We took a look at the word "coffee", and here is the resulting MFCC matrix, plotted over time

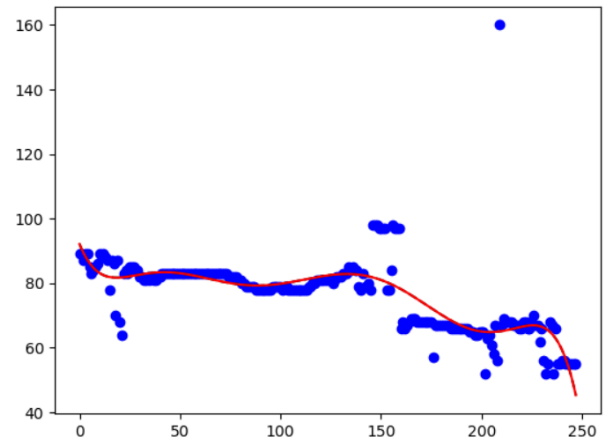


This was a voice file just over 1.58 seconds long ,and the calculation of entropy we get got resulting from our estimated GMM was

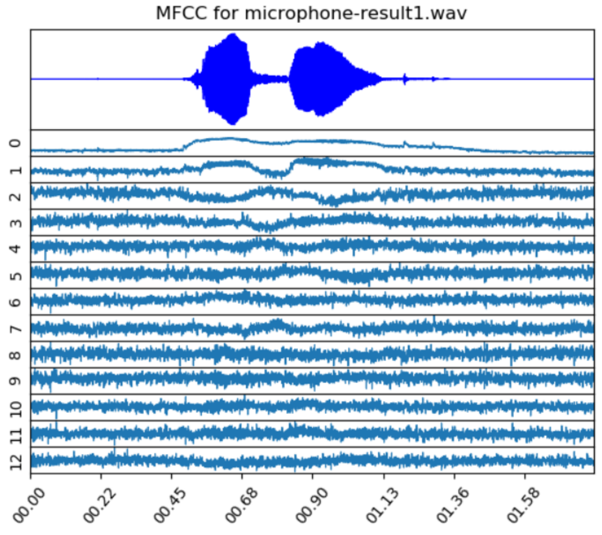
$$H(X) = 16.21$$

Our arithmetic coding entropy gave us 57 bits for the word ”coffee”, which our API successfully detected.

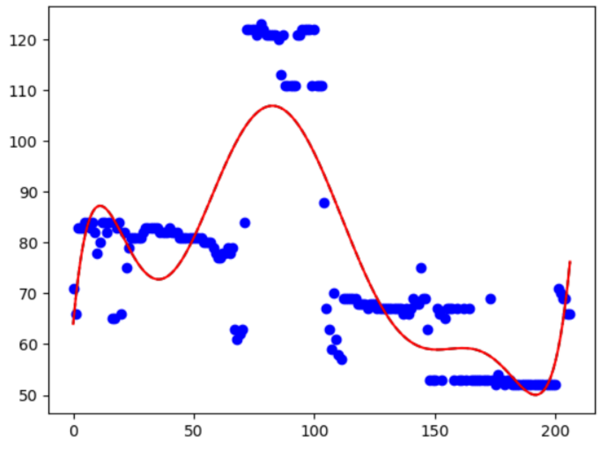
Using our own method, we created a distribution of the sample frequencies, and plotted a polynomial function to it, which looks like this.



We can do this again with the word ”password”, which gives us our resulting MFCC matrix



and our rudimentary sampled frequency model

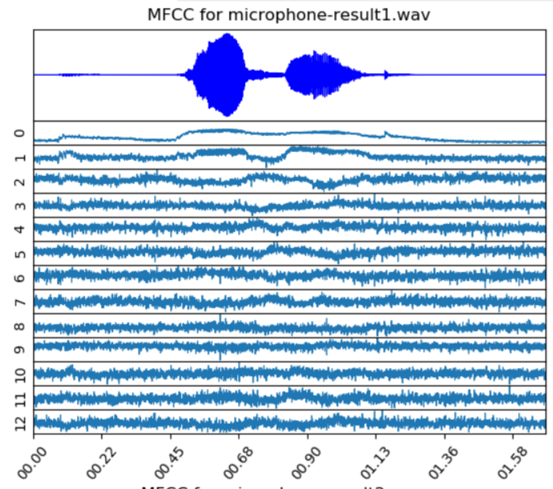


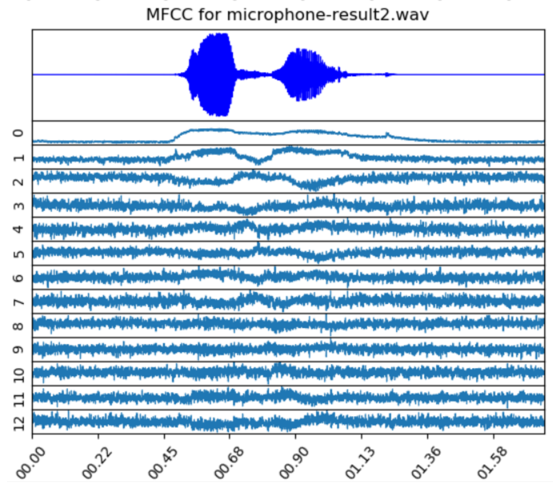
Our GMM gives us an estimated 16.37 bits of entropy, and the arithmetic coding for the word ”password” gives us 74 bits of entropy.

However, just looking at a single word isn’t enough, which brings us to our next experiment.

E. Comparing the same word twice, said in the same way

We again take a look at the word ”password”. We say the word twice and here, both times the word is being said the same way. The MFCC’s are:





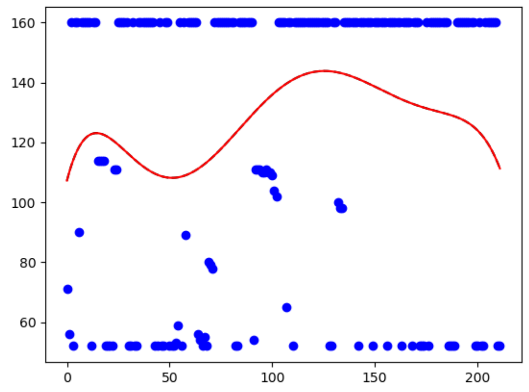
These obviously both have the same text-based entropy of 74 bits. Their estimated GMM entropy is

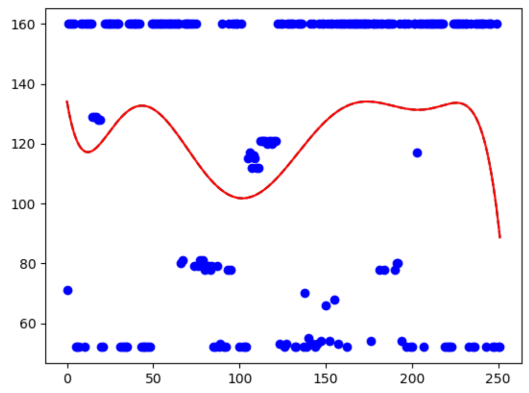
$$H(X1) = 17.11$$

and

$$H(X2) = 16.24$$

Taking a look at our estimation using the sample frequency, these are the graphs of the distributions for the first and second audio files, respectively.





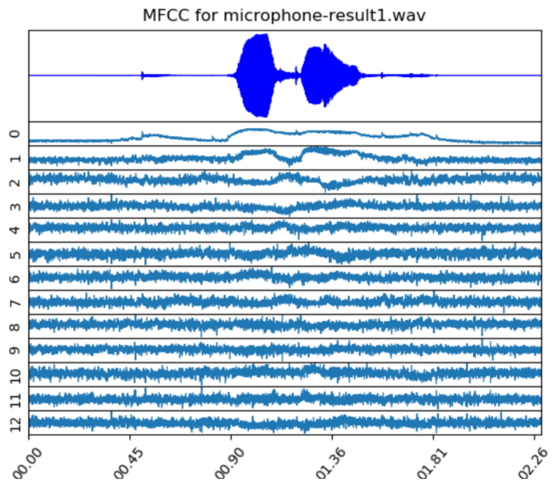
Running the Kolmogorov-Smirnov test on our two distributions we get

$$p = 0.32$$

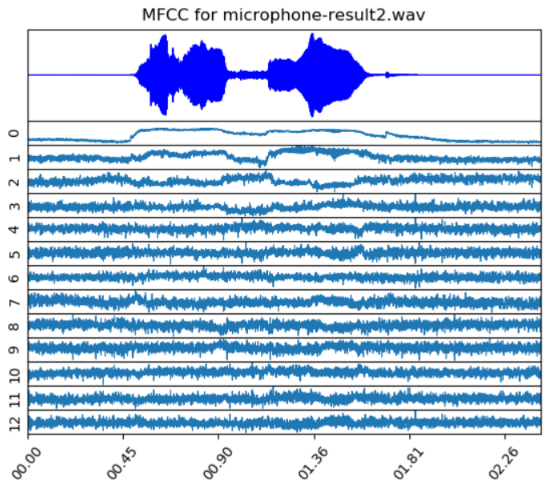
where our null-hypothesis is that the distributions are the same. Therefore, we can say that with 95% confidence that these distributions are equal, and so we can authenticate the first voice file with the second.

F. Comparing the same word twice, said differently

What happens however, if someone knows the word, but doesn't know how to say it? We now look at again the word "password", but said with very different inflection. The MFCC's are:



for the first recording



and the second recording.

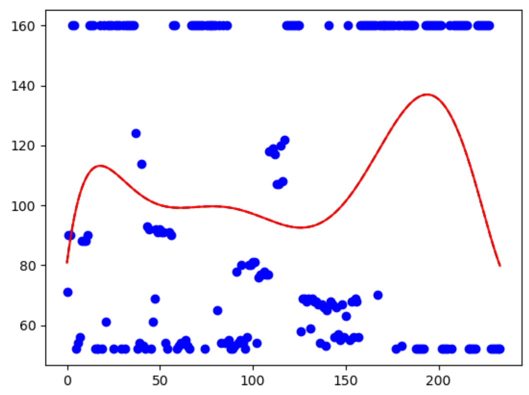
These again have the same text-based entropy of 74 bits. Their estimated GMM entropy is

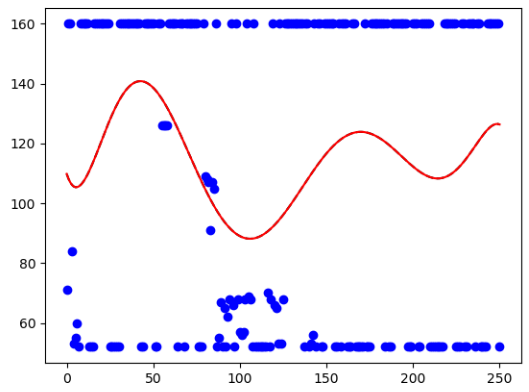
$$H(X1) = 17.52$$

and

$$H(X2) = 16.47$$

Taking a look at our estimation using the sample frequency, these are the graphs of the distributions for the first and second audio files, respectively.





Running the Kolmogorov-Smirnov test on our two distributions we get

$$p = 0.000747$$

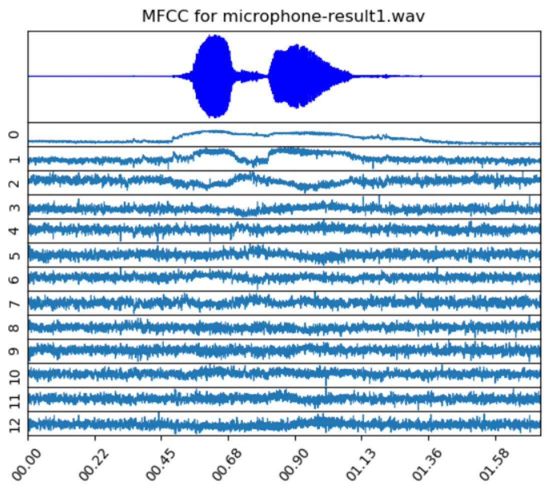
where our null-hypothesis is that the distributions are the same. Therefore, we can say that with 95% confidence that these distributions are NOT equal, and so we cannot authenticate the first voice file with the second.

This makes sense and intuitively it is clear to see from the graphs that there is no way those two sound files could have the same tonal qualities.

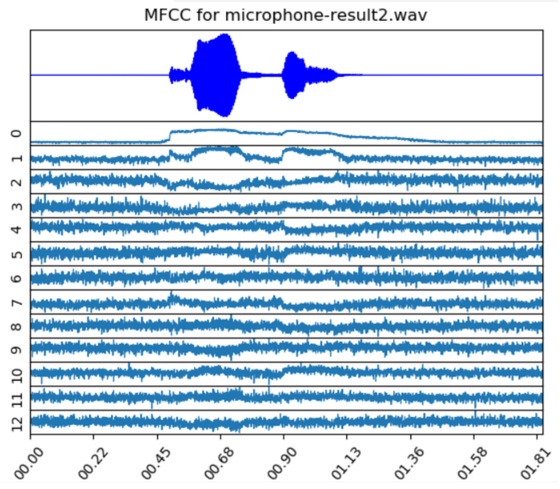
G. *Comparing two different words*

Our final test is the test of saying different words. Since our method is text-independent, there is not a clear difference from our GMM that a specific word is the one being analyzed. With that being said, with different words, there is the potential for different lengths, and naturally, it is hard to say one word and have it sound like a completely different one.

We take a look at the words we've explored before, "coffee" and "password". The MFCC's are:



for the first recording



and the second recording.

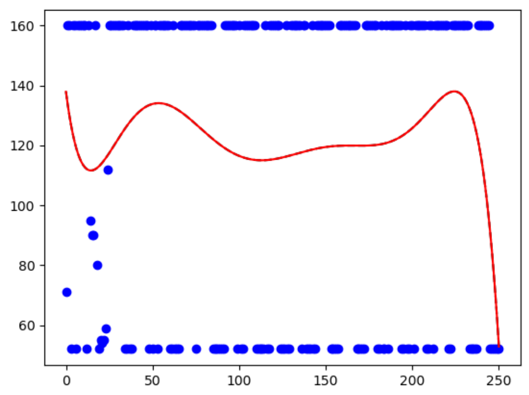
These again have the same text-based entropy of 74 bits. Their estimated GMM entropy is

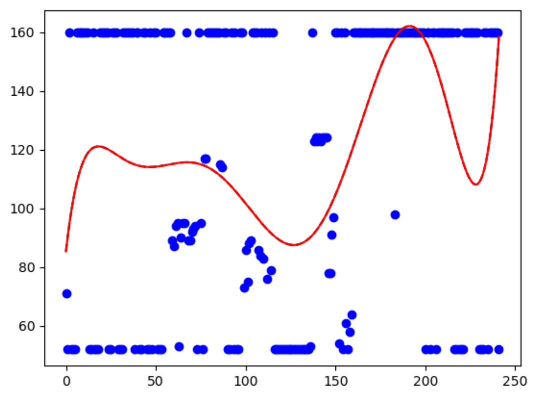
$$H(X1) = 16.10$$

and

$$H(X2) = 16.21$$

Taking a look at our estimation using the sample frequency, these are the graphs of the distributions for the first and second audio files, respectively.





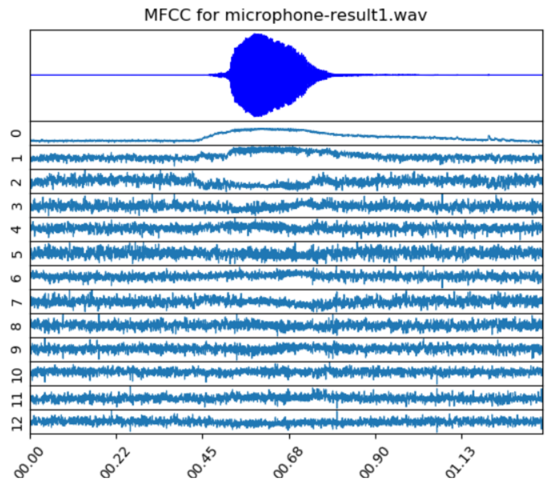
Running the Kolmogorov-Smirnov test on our two distributions we get

$$p = 0.045$$

where our null-hypothesis is that the distributions are the same. Therefore, we can say that with 95% confidence that these distributions are NOT equal, and so we cannot authenticate the first voice file with the second.

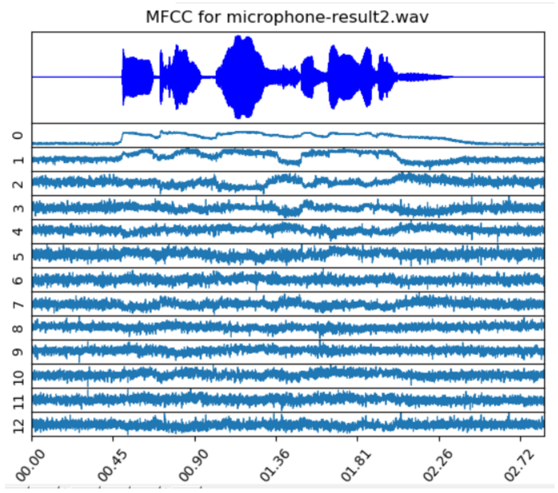
This may not seem as obvious as the results state. Coffee and Password are both two syllable words, with very similar gait of pronounciation. The p-value here is still very close to the threshold of 0.05.

We can try running a test on two even more dissimilar words. Here we try to test the word ”Hi” and the word ”Incomprehensibility”. The MFCC's are:



i

for the first recording



and the second recording.

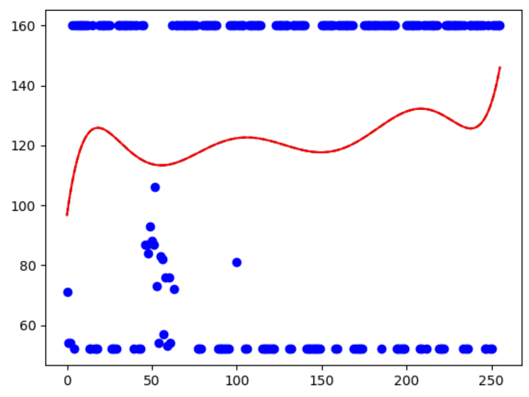
These again have the same text-based entropy of 74 bits. Their estimated GMM entropy is

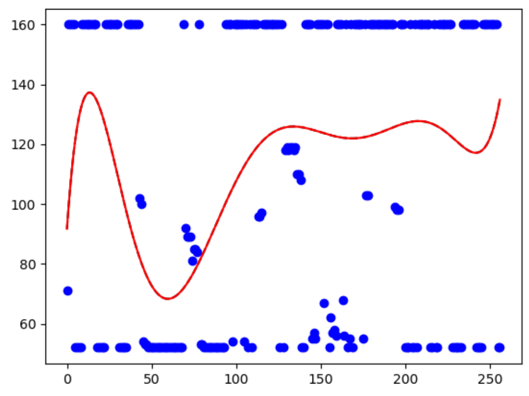
$$H(X1) = 15.94$$

and

$$H(X2) = 16.77$$

Taking a look at our estimation using the sample frequency, these are the graphs of the distributions for the first and second audio files, respectively.





Running the Kolmogorov-Smirnov test on our two distributions we get

$$p = 0.000001$$

where our null-hypothesis is that the distributions are the same. Therefore, we can say that with 95% confidence that these distributions are NOT equal, and so we cannot authenticate the first voice file with the second.

The results of this test make sense. Not only are the lengths of the words drastically different, the two are highly dissimilar words, which is reflected in our p-value of 0.000001.

V. Findings and Conclusion

A. Findings

Given the results of our experiments we found that although vocal passwords are more complex in terms of verification and measurement, they do not provide as much security as a text based passcode encoded with Arithmetic Coding. We can supplement this statement with our experiment resulting in the distributions for ”coffee” and ”password” where our Kolmogorov-Smirnov tests reveal that the distributions were almost deemed the same even though that isnt the case. This occurs because of how similar the words are pronounced. This would never happen in text based passwords, and hence reinforces our conclusion that in fact text based passwords are more secure and efficient. If we compare the Shannon entropy of ”password” we get 37.604 bits which is greater than our GMM entropy of about 16.5 bits. Likewise for ”coffee” we have Shannon entropy of 28.203 bits and GMM of 16.21 bits. However when we compare our Arithmetic Coding results of vocal passwords, they are always larger than the text based Shannon entropy. This is because we create an optimal prefix code to represent each letter in the spoken word, and we would find if we applied Arithmetic Coding to the text it would come out the same. We can consider these results equal but it does raise an interesting point; That the method of encoding, regardless of medium or input method does directly affect the entropy of the password in that system.

B. Conclusion

We conclude that from our results vocal based passwords are not as secure as text-based passwords. This conclusion is reached from our findings that similar sounding and pronounced words can be mistaken for the same. This obviously creates a huge hole in security. This along with the calculated entropy (GMM) being less than the text-equivalent Shannon entropy. We can also assert that vocal passwords are not as efficient as text based passwords. Given the complexity of methods needed to estimate entropy of vocal passwords, and to verify them, the computations required are significantly more expensive than validating text based passwords. In totality, we can say that text based password are superior to vocal passwords in terms of both security and efficiency.

References

[1] http://www.speech.cs.cmu.edu/15-492/slides/03_mfcc.pdf

[2] https://scikit-learn.org/stable/modules/mixture.htmlexpectation-maximization

[3] https://en.wikipedia.org/wiki/Mixture_model

[4] Bhattacharyya, Debnath et al. ”Biometric Authentication: A Review.” International Journal of u- and e- Service, Science and Technology 2.3 (2009): 13-28. Print.

[5] Said, Amir. ”Introduction To Arithmetic Coding - Theory And Practice.” Lossless Compression Handbook (2004): n. pag. Print.

[6] http://www-personal.acfr.usyd.edu.au/tbailey/papers/mfi08_huber.pdf

[7] ”Speechrecognition.” PyPI. N.p., 2018. Web. 6 Oct. 2018.

[8] ”Audacity — Free, Open Source, Cross-Platform Audio Software For Multi-Track Recording And Editing..” Audacityteam.org. N.p., 2018. Web. 6 Oct. 2018.

[9] Prahallad, K. (2018). Speech Technology: A Practical Introduction Topic: Spectrogram, Cepstrum and Mel-Frequency Analysis. Retrieved from: http://www.speech.cs.cmu.edu/15-492/slides/03mfcc.pdf

[10] Yang, C. (2014). Security in Voice Authentication. Retrieved from: https://web.wpi.edu/Pubs/ETD/Available/etd-032714-115410/unrestricted/yang.pdf?fbclid=IwAR3LhEj-t_Bl8WvmyoIdu52xS-akWCV2IVqm_KfiIuTWI3oO6XtrJE5WYLA

[11] Michalowicz, J. et al. (2008). Calculation of Differential Entropy for a Mixed Gaussian Distribution. Retrieved from: http://mdpi.org/entropy/papers/e10030200.pdf?fbclid=IwAR3Nyz-21D1EdfXEnrwC61QTshKp1_NmqECxBq8YNE4yqUB7AfpaIblwXos

[12] Huber, M. et al. (2008). On Entropy Approximation for Gaussian Mixture Random Vectors. Retrieved from: http://www-personal.acfr.usyd.edu.au/tbailey/papers/mfi08_huber.pdf?fbclid=IwAR0_1c5_k4OIeKhA7H1WpCPIfFl8am8RYMLNa2twQEnqVq4K7ZdwRhg7EoI

[13] Stuttle, M. (2003). A Gaussian Mixture Model Spectral Representation for Speech Recognition. Retrieved from: http://mi.eng.cam.ac.uk/ mjfg/thesis_mns25.pdf?fbclid=IwAR1BoDwuKXz2FE9J_xSiro_kHNdOcAuafw5NnxXZU9jsV4j_HnlpbDuIKsQ

[14] J.R. Deller, J.G. Proakis, and J.H.L. Hansen. ”Discrete-time processing of speech signals” Macmillan New York (1993)

[15] http://www.computerrobotvision.org/2010/tutorial_day/GMM_said_crv10_tutorial.pdf

[16] http://yulearning.blogspot.com/2014/11/einsteins-most-famous-equation-is-emc2.html

[17] O’Gorman, L. ”Comparing Passwords, Tokens, And Biometrics For User Authentication.” Proceedings of the IEEE 91.12 (2003): 2019-2020. Web.

[18] Bhattacharyya, Debnath et al. ”Biometric Authentication: A Review.” International Journal of u- and e- Service, Science and Technology 2.3 (2009): 13-28. Print.

[19] Said, Amir. ”Introduction To Arithmetic Coding - Theory And Practice.” Lossless Compression Handbook (2004): n. pag. Print.

[20] ”Speechrecognition.” PyPI. N.p., 2018. Web. 6 Oct. 2018.

[21] ”Audacity — Free, Open Source, Cross-Platform Audio Software For Multi-Track Recording And Editing..” Audacityteam.org. N.p., 2018. Web. 6 Oct. 2018.

[22] ”ICT STATISTICS Home Page.” Itu.int. N.p., 2018. Web. 11 Dec. 2018.

[23] Pato, Joseph N, and Lynette I Millett. Biometric Recognition. Washington, D.C.: National Academies Press, 2010. Print.

[24] ”Voice Biometrics - Sabio.” Sabio. N.p., 2018. Web. 4 Dec. 2018.