# Synthetic Speech Detection

by

Armaan Lalani

*Supervisor: Professor Yuri Lawryshyn*

An Undergraduate Thesis

Submitted to the Faculty of Applied Science and Engineering

University of Toronto

In Fulfillment of the Requirements

For a Bachelor's Degree

March 2023

# Acknowledgements

This thesis was carried out in collaboration with the University of Toronto's Centre for Management of Technology and Entrepreneurship (CMTE) and The Bank. Thank you for providing me with the opportunity to work on this project and guidance throughout the process. I would like to thank Professor Yuri Lawryshyn for his guidance and support throughout the thesis. Finally, I would also like to thank The Bank for providing me with inspiration for this thesis.

# Abstract

The rise of deepfake technology in recent years has underscored the importance of distinguishing between authentic and synthetic content. For established financial institutions such as The Bank, ensuring security and confidentiality for clients and themselves is a top priority. However, voiceprint technology, which relies on vocal features to authenticate clients over the phone, is vulnerable to synthesized speech. The Bank acknowledges the potential risks posed by synthetic speech and recognizes the need to identify effective mitigation techniques to safeguard security. To this end, the present study aims to compare the performance of handcrafted and deep feature-based methods of synthetic speech detection across several key evaluation metrics, including accuracy, equal error rate, processing time, and trainable parameters. Two models were developed and tested, and the results indicate that the deep features model outperforms the handcrafted features model on three of the four metrics, suggesting that it should be prioritized for implementation.

# Executive Summary

The Bank is a leading Canadian financial institution that places high value on security and confidentiality. As digital technology advances, the potential for fraudulent activity, including adversarial deepfakes such as synthetic speech, grows. It is vital to distinguish between synthetic speech and genuine content to prevent harm to The Bank and its clients. One particular use case is for voiceprint, where clients are authenticated on the phone simply based on their vocal features. As synthetic speech detection methods become more robust, the necessary infrastructure should be implemented in order to distinguish between which voice samples are genuine versus synthesized. There are two primary detection methods of synthetic speech: handcrafted features and deep features (Masood et al., 2023). While both methods have shown success, more research is needed to compare their effectiveness. The Bank is committed to ongoing research to mitigate the risks of deepfakes and ensure the safety and security of its clients. The objective of this thesis is to implement state of the art handcrafted and deep feature detection methods and compare their performance in terms of both accuracy and time, both of which are vital metrics for The Bank. By utilizing open-source data, the models will be compared on an even playing field to accurately gauge their overall relative performance.

Prior to beginning the development of detection models, an emphasis was placed on understanding the existing methods of generating synthesized speech to guide the research conducted on detection methods. Existing methods of synthetic speech development fall into two primary categories: text-to-speech synthesis and voice conversion. TTS is the conversion of text into phonemic representation and then into waveforms (Wikipedia). VC is the digital cloning of an individual's voice appeared to be spoken by another (Voice Conversion Challenge). As a result of these two primary generation methods, an open-source dataset, ASVSpoof 2019, was chosen as the primary dataset for analysis as it contains samples that utilize numerous methods belonging to both TTS and VC. The final research step involved identifying two methods of synthetic speech generation, one belonging to handcrafted features and the second belonging to deep features. Several potential detection methods were identified, however, the two pursued for development were a Bicoherence + STLT model for the handcrafted features model and a TSSDNet for the deep features model. The Bicoherence + STLT model combines bicoherence, which is based on high-frequency spectral magnitudes and phases, (AlBadawy et al.) with STLT,

which calculates both short and long-term prediction error at various points in the signal (Borrelli et al., 2021). The TSSDNet uses two forms of CNNs with traditional ResNet skip connections and inception-style parallel convolutions (Hua et al., 2023). Both models were chosen due to their proven success in their respective research papers.

In order to generate results for the two models, the Bicoherence + STLT model was built from scratch while a pretrained version of the TSSDNet was used. The Bicoherence + STLT model was simplified relative to the paper due to time constraints, processing constraints, and a couple of complications during the development process. Therefore, the results generated by the model were combined with the paper results to compare its performance more accurately with the deep features model. Both models were compared using four primary metrics: evaluation accuracy, equal error rate, processing time per sample, and the number of trainable parameters. The deep features model outperforms the handcrafted features model in terms of evaluation accuracy – 96% versus 73% (86% in the paper), equal error rate – 0.03 versus 0.18, and processing time per sample – 0.2 seconds versus 6 seconds. The handcrafted features model outperforms the deep features model in terms of number of trainable parameters – 72 versus 92658. Based on the results summarized above, it can be concluded that the deep features model outperforms the handcrafted features model. Even when extending the results of the handcrafted features model based on the paper, it falls short in achieving the same level of success of the deep features model. The deep features model is able to more accurately classify speech samples and do so significantly faster than the handcrafted features model, which are both essential for The Bank. Therefore, the results of this thesis signify that The Bank should prioritize the implementation of a deep features model over a handcrafted features model to ensure a reliable analysis.

There are two potential directions for future research given the results discussed. The first involves extending the handcrafted features model by including all speech samples in the ASVSpoof 2019 Dataset and increasing the number of features for each sample. However, additional research is needed to determine how to handle an unbalanced dataset when using the *C* parameter for SVM. The second direction involves generating personalized audio samples to test the effectiveness of both the handcrafted and deep features models, using new synthetic speech generation methods such as Microsoft Vall-e. Resource limitations prevented further progress in this area during the research process.

# Table of Contents

## List of Figures

# List of Tables

# Glossary

**Constant-Q Cepstral Coefficients (CQCC):** a feature extracted from the Constant-Q transform (CQT) that captures long range information of a signal (Yang et al., 2018)

**Equal Error Rate (EER):** location on an ROC curve where the false acceptance rate and false rejection rate are equal (Innovatrics)

**Generative Adversarial Network (GAN):** two neural networks that contest with each other where one agent's gain is another agent's loss (Wikipedia (a), 2023)

**Linear-Frequency Cepstral Coefficients (LFCC):** utilization of a frequency scale to generate coefficients that mimics how human ears process sounds which covers all speech frequency ranges (Zhou et al., 2011)

**Long Short Term Memory (LSTM):** a type of recurrent neural network, see below, capable of learning order dependence in sequence prediction (Brownlee, 2017)

**Mel-Frequency Cepstral Coefficients (MFCC):** utilization of a frequency scale to generate coefficients that mimics how human ears process sounds (Zhou et al., 2011)

**Recurrent Neural Network (RNN):** a type of neural network that uses sequential data (IBM)

**Short-Term Long-Term Prediction Traces (STLT):** a method of analyzing speech signals using short-term prediction error and long-term prediction error – discussed in increasing detail in Subection 2.3.1 (Borrelli et al., 2021)

**Support Vector Machine (SVM):** a machine learning technique that identifies a separator as a hyperplane between different categories of data (IBM)

**Temporal Convolutional Networks (TCN):** a convolutional neural network that employs causal convolutions and dilations to adapt for sequential data – discussed in increasing detail in Subsection 2.3.2 (Khochare et al., 2021)

**Time-Domain Synthetic Speech Detection Nets (TSSDNet):** an advanced CNN which employs skip connections and parallel convolutions – discussed in increasing detail in Subsection 2.3.2 (Hua et al., 2023)

**Text-to-Speech Synthesis (TTS):** conversion of text into phonemic representation and then into waveforms (Wikipedia (b))

**Voice Conversion (VC):** digital cloning of an individual's voice appeared to be spoken by another (Voice Conversion Challenge)

# 1 - Introduction

## 1.1 - The Problem

The Bank[1] is one of Canada's largest banks by market capitalization and provides their clients with several financial services (The Bank). Security and confidentiality are paramount to the operations of any financial institution; especially one with the reputation of The Bank. As the world becomes increasingly digital, so does the potential of fraudulent activity, one such example being adversarial deepfakes. Adversarial deepfakes, including synthetic speech, involve the use of deep learning to generate convincing fake media (The Guardian, 2020). Deepfakes and synthetic speech can cause severe harm to The Bank if not mitigated appropriately, and therefore, methods of quickly and accurately deciphering between synthetic and genuine content is vital to ensure security for both The Bank and their clients.

Synthetic speech is a huge branch of adversarial deepfakes and over the years, there has been an increase in methods used for detection. All forms of detection fall into two primary categories: handcrafted features and deep features (Masood et al., 2023). Handcrafted features combine the extraction of primary coefficients from audio signals with a statistical classification system to decipher between synthesized and genuine audio (Hua et al., 2023). More recently, extracting features from audio signals has been entirely replaced by a deep learning pipeline due to their increased prevalence (Hua et al., 2023). Both methods have been researched extensively and have displayed success in their classification ability, however, there is limited literature on the comparison between the two.

## 1.2 - Objective and Goals

The objective of this thesis is to implement state of the art handcrafted and deep feature detection methods and compare their performance in terms of both accuracy and processing metrics, both of which are vital for The Bank. By utilizing open-source data, the models will be compared on an even playing field to accurately gauge their overall relative performance. To

[1] In accordance with the signed nondisclosure agreement the client cannot be named

accomplish the primary objective of this thesis, there are a number of intermediary steps that were completed in order to ensure project success.

The following is a list of goals and their corresponding deliverables that need to be achieved to successfully analyze the outlined scope and objective of this thesis. The Gantt Chart in Appendix 1 provides a more detailed timeline on the major project goals and deliverables. These goals are outlined as follows:

1. Understand the most common methods of adversarial deepfake generation and detection (both audio and visual) to inform decisions about model architecture
    a. Key Deliverable: Interim Report (January 30, 2023) and weekly progress updates to CMTE and The Bank
2. Develop a set of metrics relating to both accuracy and processing that will be used to validate the performance of the designed models
    a. Key Deliverable: Interim Report (January 30, 2023)
3. Locate open-source data to train and test the models
    a. Key Deliverable: ASVSpoof 2019 was utilized to train and evaluate the performance of both models
    b. Metrics: class balance (similar proportion of genuine versus synthetic), sufficient data samples to train the model
4. Utilize methods from various speech detection research to generate appropriate models used for synthetic speech detection
    a. Key Deliverable: two workable models, one handcrafted features model and one deep features model, that can be used to effectively classify audio samples by their respective class
    b. Metrics: model techniques are proven to be successful based on research and testing (accuracy, prediction error, etc.)
5. Final thesis report containing of a detailed description of the above goals and deliverables with project results
    a. Key Deliverable: final project and presentation (end of March 2023 to beginning of April 2023), code written in Python that develops and utilizes both models in order to report their performance with respect to the key metrics identified

The primary scope of the work for this thesis focuses on the research and implementation of synthetic speech detection models while also completing some initial research on the creation of synthetic speech to inform model decisions.

## 1.3 Document Structure

The remainder of the document is structured into five primary chapters, each serving a distinct purpose. Chapter 2 offers a comprehensive background on synthetic speech generation and detection, presenting the two dominant detection methods currently in use. In addition, this chapter introduces the open-source dataset utilized to evaluate the success of the models presented in this study. Chapter 3 delves into the two primary models utilized in this study, one based on handcrafted features and the other on deep features. This chapter builds upon the foundation laid in Chapter 2, drawing upon the latest research to implement state-of-the-art models. Chapter 4 provides a detailed visual and statistical analysis of the models' performance, presenting the results in an easily comprehensible manner. Chapter 5 offers a more detailed commentary on the findings of the two models, while also identifying the limitations of the comparison between them. Finally, Chapter 6 presents potential areas of research to be considered in order to further validate the credibility of the study.

# 2 - Background

The Background is organized as follows. Section 2.1 discusses a few of the most common methods of generating synthetic speech samples in order to inform possible detection methods. Section 2.2 introduces the open-source dataset, ASVSpoof 2019, that was used to train and evaluate models. The dataset makes use of synthetic speech generation methods discussed in Section 2.1. Finally, Section 2.3 introduces the two primary methods of synthetic speech detection while commenting on a few examples from each class. These methods are extended in Chapter 3.

Deepfakes are synthetic media where characteristics of a person are replaced with those of others (Wikipedia (d)). While the term was coined to primarily encompass videos, deepfake technology can also involve photos and audio (Wikipedia (d)); the former will be the primary research focus. As AI technology becomes more advanced, the threat deepfakes pose to privacy is an increasing concern (The Guardian, 2020). In the case of The Bank and the broader banking industry, synthetic speech is a threat to voiceprint architecture. Voiceprint is a method of verifying a user's identity over the phone by extracting important audio features, which can be in the form of mel-frequency cepstral coefficients, or MFCCs (Boles et al., 2017). MFCCs are one of the many representations of sound in the frequency domain consisting of numerous transformations, convolutions, and filtering which are briefly outlined in Figure 1.

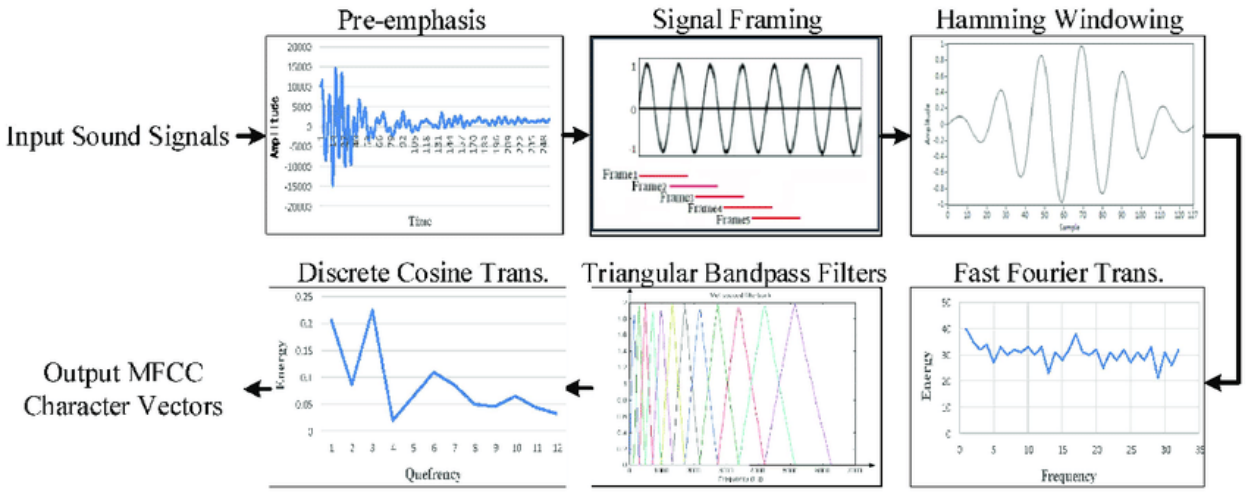


*Figure 1- a brief visual representation of MFCCs which is a method used to recognize an individual's voiceprint signature (Tsai et al., 2019)*

## 2.1 - Synthetic Speech Development

Prior to developing an understanding of the existing synthetic speech detection methods, it is equally important to delve into how quality synthetic speech is developed. The research article 'Deepfakes generation and detection: state-of-the-art, open challenges, countermeasures, and way forward' (Masood et al., 2023) provides a comprehensive overview on the different methods of developing synthetic speech as well as considerations for the future as deepfake detection continues to grow. Additionally, the paper identifies an initial research gap, stating that there has been less attention paid to the research on audio deepfake detection in comparison to image and video (Masood et al., 2023). Current infrastructure primarily focuses on detecting video manipulations, however, due to the recent rise of audio manipulation techniques, audio forgery detection should be studied in more detail (Masood et al., 2023). The paper highlights two primary methods of developing state-of-the-art synthetic speech: text-to-speech synthesis or TTS and voice conversion or VC, both of which are discussed in detail in the following subsections.

### 2.1.1 - Text-to-Speech Synthesis

TTS generates natural sounding waveforms based on an inputted text, which aims to mimic a target's speaking style (Masood et al., 2023). Initial TTS systems relied on separating audio into smaller fragments and concatenating them into new speech based on certain parameters, however, these methods have become outdated with the development of deep learning (Masood et al., 2023). More recently, TTS systems utilize parametric models that map text to salient speech parameters and use vocoders to convert them into audio signals (Masood et al., 2023). As deep learning continues to dominate methods of generation, various methods stemming from TTS have emerged, some of which are defined below:

- Neural vocoder: a vocoder is a signal processing device that synthesizes audio using a variety of speech factors such as fundamental frequency, spectral envelope, etc. Neural vocoders aim to generate waveforms using low-dimensional audio representations such as MFCCs (Mehta, 2021).

- Autoregressive vocoder: uses maximum likelihood probabilistic models that aim to forecast waveforms based on prior knowledge.
    - Wavenet and WaveRNN: models that use speech patterns to predict which sounds are most likely to follow each other (Mehta, 2021).
- Generative Adversarial Network, GAN, vocoder: utilizes a generator to 'generate' waveforms based on inputted text and a discriminator to continuously improve the audio quality (Mehta, 2021).

There are countless TTC systems which have proven to be successful and which utilize various methods highlighted above. Some of the most prominent include WaveNet (developed by DeepMind), Tacotron (developed by Google), and DeepVoice3 (Masood et al., 2023). A general architecture of TTC systems is displayed in Figure 2. These methods and various others have displayed the proven ability to generate realistic synthetic speech; more of which are highlighted in Appendix 2.

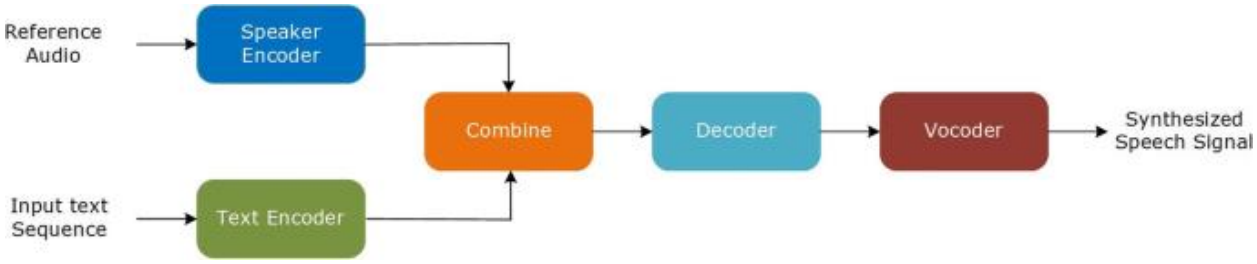


*Figure 2- a general representation of TTC systems (Masood et al., 2023)*

### 2.1.2 - Voice Conversion

VC, as the name suggests, manipulates a particular input voice to sounds like a target voice while ensuring the linguistic properties of the input (Masood et al., 2023). Although VC is applicable to various real-life activities, such as voice-controlled assistants, the realism of new methods poses security risks when utilized unethically (Masood et al., 2023). At a high level, VC utilizes two primary characteristics of speech:

- Voice timbre: tonal quality of the vocal tract during speech (Masood et al., 2023)

- Prosody characteristics: properties of syllables such as intonation, stress, and rhythm (Wikipedia (c), 2023).

Early VC techniques relied on parallel data – where speech samples from both the input and target contain the same linguistic content (Masood et al., 2023) – and started with simple statistical methods such as Gaussian Mixture Models and least squares regression. Through the development of deep learning, more sophisticated methods came to rise such as Recurrent Neural Networks, RNNs, and Long-Short Term Memory, LSTMs, which have the proven ability to learn time sequential data (Masood et al., 2023). Through additional research and testing, it became apparent that parallel methods of VC were not feasible as readily available data of both the input and target speaker was limited (Masood et al., 2023).

As a result, methods for performing VC for non-parallel data were developed, which could be utilized across various languages and speakers (Masood et al., 2023). VC techniques for non-parallel data are similar to TTS, however, with varying model optimization processes since the objectives of these models differ (Masood et al., 2023). Non-parallel VC methods use models such as neural network vocoders, GANs, and auto-encoders. The use of these methods has proven success in their ability to deliver natural sounding speech, however, results are often speaker-dependent and have lower generalization ability (loss of performance for unseen speakers) (Masood et al., 2023). Additionally, new techniques are also continually being developed, such as one-shot and zero-shot VC. One-shot VC does not require data samples of the input and target speaker during training, "[the] speaker embedding is extracted from the target speech, which can control the speaker identity of the converted speech independently" (Masood et al., 2023). Zero-shot VC is similar, but also does not need to retrain the model using encoder approaches. While both methods display promise, they do not solve the problem of non-parallel VC methods having poor generalization ability (Masood et al., 2023). Additional methods of VC techniques are highlighted in Appendix 3.

This brief introduction to synthetic speech generation methods provides important information on locating a viable dataset. TTS and VC dominate the various generation techniques, and therefore, a dataset that includes several implementations of TTS and VC methods should be prioritized to simulate real world scenarios most accurately.

## 2.2 - ASVSpoof 2019 Dataset

The ASVSpoof 2019 Dataset was used for the Third Automatic Speaker Verification Spoofing and Countermeasures Challenge (Yamagishi et al., 2019). It contains a collection of both bona fide[2] speech samples as well as numerous synthetic speech samples generated using numerous methods belonging to both TTS and VC. The dataset contains two primary directories which will be utilized to conduct the majority of the analysis. The *Protocols* directory contains text (txt) files which list filenames, the speaker ID, the method of spoofing if applicable, and the associated label (bona fide or synthetic). Secondly, the *train*, *dev*, and *eval* directories contain the actual speech samples which are referenced by the txt files. The entire ReadMe file included with the dataset is included in Appendix 4. The overview of the train, dev, and eval datasets is summarized in Table 1.

*Table 1- Summary of audio samples in each ASVSpoof 2019 Dataset (Borrelli et al., 2021)*

| | | $\mathcal{D}_{tr}$ | $\mathcal{D}_{dev}$ | $\mathcal{D}_{eval}$ | Category |
|---|---|---|---|---|---|
| **Samples** | **Bona fide** | 2580 | 2548 | 7355 | |
| | **Synthetic** | 22800 | 22296 | 63882 | |
| **Speakers** | **Bona fide** | 20 | 10 | 48 | |
| **Synthetic** | **A01** | ✓ | ✓ | | NN |
| **Methods** | **A02** | ✓ | ✓ | | VC |
| | **A03** | ✓ | ✓ | | VC |
| | **A04 = A16** | ✓ | ✓ | ✓ | WC |
| | **A05** | ✓ | ✓ | | VC |
| | **A06 = A19** | ✓ | ✓ | ✓ | VC |
| | **A07** | | | ✓ | NN |
| | **A08** | | | ✓ | NN |
| | **A09** | | | ✓ | VC |
| | **A10** | | | ✓ | NN |
| | **A11** | | | ✓ | NN |
| | **A12** | | | ✓ | NN |
| | **A13** | | | ✓ | NN |
| | **A14** | | | ✓ | VC |
| | **A15** | | | ✓ | VC |
| | **A17** | | | ✓ | VC |
| | **A18** | | | ✓ | VC |

In the above table, tr, dev, and eval represent the respective datasets, A01-A18 represent the different spoofing algorithms with NN representing TTS and VC representing VC.

[2] 'Bona fide' is a term defined in ASVSpoof and is synonymous with 'genuine'

## 2.3 - Synthetic Speech Detection

Due to the advances in TTS and VC techniques, significant research has been conducted on optimizing detection techniques to limit privacy concerns. The paper (Masood et al., 2023) highlights the two primary methods of detection: utilizing handcrafted features and utilizing deep features.

### 2.3.1 - Handcrafted Features

While there are countless implementations that involve handcrafted features, the overarching concept remains constant. Handcrafted feature methods of detection rely on extracting primary coefficients from audio samples, conducting some form of data manipulation, followed by passing the data through a classification model, such as an SVM, logistic regression, etc. (Masood et al., 2023). The paper outlines numerous methods of extracting primary coefficients, which include Constant Q Cepstral Coefficients (CQCCs), Linear-Frequency Cepstral Coefficients (LFCCs), MFCCs, etc. all of which involve transformations of audio signals into the frequency domain. Many of the primary cepstral coefficients are quite similar with minor variations, which pose both pros and cons in auditory analysis. Speaker identification is a difficult task and deciding the best features to extract from audio signals is often ambiguous, which is why research has been conducted on combining features to improve performance (Mohammadi et al., 2017). The paper ‘Robust Feature Fusion for Text Independent Speaker Verification Enhancement in Noisy Environments’ discusses the fusion of various cepstral coefficients for speaker verification. Based on the results of the experiment, fusions of various coefficients perform better in speaker verification tasks, which highlights the difficulty of selecting the optimal features to use depending on the task at hand (Mohammadi et al., 2017). As a result, there are countless studies that have been conducted on the performance of synthetic speech detection utilizing different coefficient extraction methods, a few of which are discussed below.

The research and experimentation conducted in the paper ‘Synthetic speech detection through short-term and long-term prediction traces’ provides one method of handcrafted feature implementation which has displayed success when tested on the ASVSpoof 2019 Dataset. The

ASVSpoof 2019 Dataset was used for the Third Automatic Speaker Verification Spoofing Countermeasures Challenge and contains audio samples of bona fide audio in addition to synthetic samples of diverse algorithms, both belonging to TTS and VC (Yamagishi et al., 2019). ASVSpoof 2019 will also be the primary dataset utilized to test the models required to achieve the research goal. The paper discusses the importance of long-term features and sub-band analysis features, such as LFCCs and MFCCs, since the artifacts of synthetic speech generation are not distributed evenly across the frequency bands (Borrelli et al., 2021). The baseline model developed in the paper relies on the bicoherence of an audio signal and was chosen due to its proven success in the paper 'Detecting AI-Synthesized Speech Using Bispectral Analysis'. Bicoherence is based on high-frequency spectral magnitudes and phases which are the most effective in distinguishing between bona fide and synthesized speech (AlBadawy et al.). The bicoherence spectrum, or bispectrum, is used to determine third-order correlations which can be a primary differentiating factor between bona fide and synthesized speech (AlBadawy et al.). The bicoherence of a signal is defined as follows, given a signal $s(n)$ that is split into $W$ windows with signal $s_w(n)$ with respective Fourier transform of $S_w(\omega)$ and the complex conjugate operator denoted as *, the bicoherence is calculated as a function of harmonics $\omega_1, \omega_2$ as:

$$B(\omega_1, \omega_2) = \frac{\sum_{w=0}^{W-1} S_w(\omega_1) S_w(\omega_2) S_w^*(\omega_1 + \omega_2)}{\sqrt{\sum_{w=0}^{W-1} |S_w(\omega_1) S_w(\omega_2)|^2 \sum_{w=0}^{W-1} |S_w^*(\omega_1 + \omega_2)|^2}}. \quad (1)$$

The size of windows to calculate bicoherence are usually powers of 2 with a step size half of the length of the window size to allow for overlapping. Once the bicoherence is calculated, the paper utilizes four primary moments of bicoherence to classify the speech samples accordingly: mean, variance, skew, and kurtosis, defined as:

$$\mu_B = E_B[B] \quad (2)$$

$$\sigma_B = E_B[(B - \mu_B)^2] \quad (3)$$

$$\gamma_B = E_B[(\frac{B - {\mu_B}^2}{\sigma_B})]^3 \quad (4)$$

$$\kappa_B = E_B[(\frac{B - {\mu_B}^2}{\sigma_B})]^4 \quad (5)$$

By calculating the four primary moments over both the mean and phase of the calculated bicoherence, each audio sample is reduced to an 8D vector (AlBadawy et al.). These vectors are then used as feature vectors for a classification model (i.e., SVM, random forrest, regression, etc.) to decipher between bona fide and synthesized speech. The visualization of the mean magnitude and phase (2 of the elements of the 8D vector) are displayed in Figure 3 alongside the classification results in a confusion matrix.

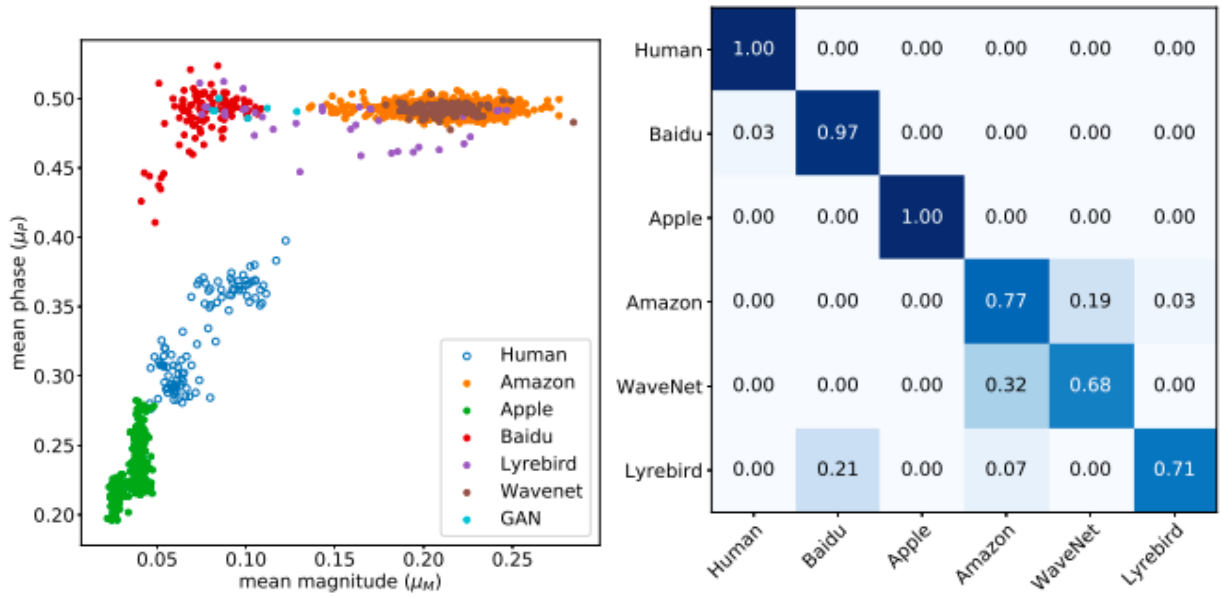


*Figure 3- the results of the Bispectral Analysis with the bicoherence vector representation shown on the left and the classification confusion matrix shown on the right (AlBadawy et al.)*

As shown in the figure, the bicoherence approach was able to achieve a considerable amount of success in classifying bona fide versus synthetic speech, while also displaying the ability to classify which synthesis method was utilized. Returning to the research conducted in (Borrelli et al., 2021), bicoherence was used as a baseline model, however, was not able to achieve similar levels of success when tested on the ASVSpoof 2019 Dataset as outlined in Figure 4.

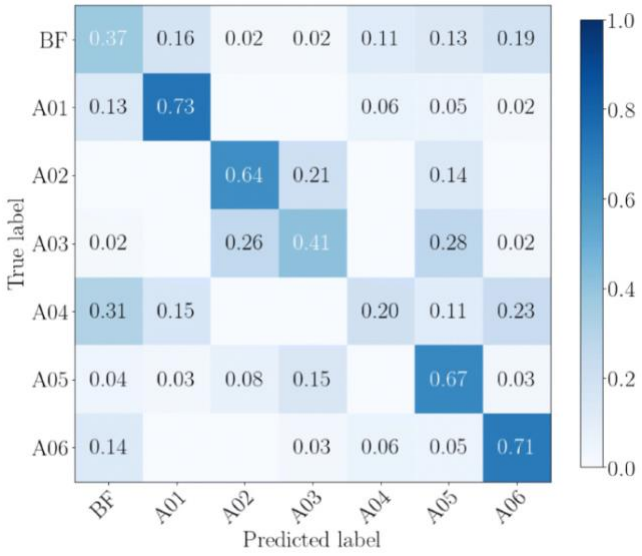


*Figure 4- the results of bispectral analysis on the ASVSpoof 2019 Dataset in a classification confusion matrix (Borrelli et al., 2021)*

Due to this reduced performance, the paper proposes additional features to improve the classification ability of the model being used. The additional features are motivated by the source-filter model which expresses speech as a stimulation signal caused by the vocal folds, followed by a filter to approximate the effects of the vocal tract (Borrelli et al., 2021). The general process is displayed below in Figure 5.

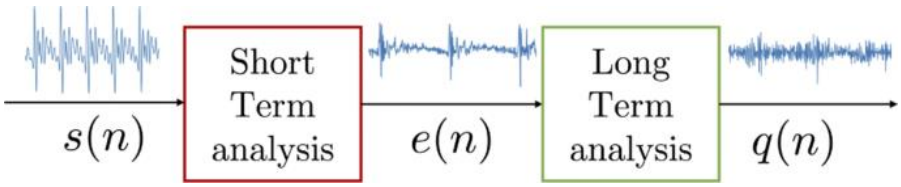


*Figure 5- a visual representation of the analysis method used in (Borrelli et al., 2021)*

In the above, *s(n)* denotes the input signal, *e(n)* denotes the source excitation signal or short-term prediction error, and *q(n)* denotes the wide-band noise signal or long-term prediction error. The short-term analysis involves determining coefficients, denoted by *a*, that minimize the energy *e(n)*, or the prediction error between the actual signal and the predicted signal. The coefficients, defined as *a*, are solved by the following equation:

$$\begin{matrix} a_1 \\ \vdots \\ a_L \end{matrix} = \begin{bmatrix} r(0) & \cdots & r(L-1) \\ \vdots & \ddots & \vdots \\ r(L) & \cdots & r(0) \end{bmatrix}^{-1} \begin{matrix} r(1) \\ \vdots \\ r(L) \end{matrix} \quad (5)$$

where the vector *r* represents the correlation, and *r(i)* represents the i[th] element in the vector. The matrix, defined as the correlation matrix *R* is constructed based on the elements of the correlation vector, *r*. Once the values of *a* are estimated, the short-term error can be estimated as follows:

$$e(n) = s(n) - \sum_{i=1}^{L} a_i s(n-i) \quad (6)$$

where the value of *L* is varied from 1 to 50 in order to generate short-term predictions with varying levels of information. In Equation 6, the short-term prediction error behaves similarly to an autoregressive model which is primarily utilized in time series analysis. The primary goal is to predict the signal value at *n* by utilizing a certain number of samples directly before *n*, which is determined by the value of *L*.

The long-term analysis has a primary goal of understanding long-term correlations in the signal, and involves estimating two parameters, *k* (signal delay) and a coefficient $\beta_k$ through minimizing the long-term prediction error based on the following equation:

$$J_{LT}(k) = E[q^2(n)] = E[\big(e(n) - \beta_k e(n-k)\big)^2] \quad (7)$$

where $\beta_k$ is approximated as *r(k)/r(0)* and the long-term prediction error can be determined as *q(n)* using the definition given in Equation 7. By calculating the short-term and long-term prediction error over the same windows to calculate the bicoherence features, the error energy and error gain over the short-term and long-term can be calculated for each window as follows, where *N* is the size of the window:

$$E_{ST} = \frac{1}{N}\sum_{i=0}^{N-1} e(i)^2 \, , E_{LT} = \frac{1}{N}\sum_{i=0}^{N-1} q(i)^2 \, , G_{ST} = \frac{\frac{1}{N}\sum_{i=0}^{N-1} s(i)^2}{\frac{1}{N}\sum_{i=0}^{N-1} e(i)^2} , G_{LT} = \frac{\frac{1}{N}\sum_{i=0}^{N-1} e(i)^2}{\frac{1}{N}\sum_{i=0}^{N-1} q(i)^2} \quad (8)$$

As a result of the above, the 8D feature vector obtained through the bicoherence calculations can be extended to also include the mean, standard deviation, maximum, and minimum values of the short-term and long-term estimations to achieve better classification results over the ASVSpoof 2019 dataset, as highlighted by Table 2 and 3. The paper conducts the analysis with values of $L$ from 1 to 50 as discussed above, which leads to a final vector of size 8 + 16 * 50 = 808. These additional elements lead to far superior classification results in comparison to solely using the 8D bicoherence vector which are highlighted in the Tables 2 and 3.

*Table 2- Binary Classification Accuracy over the development dataset in ASVSpoof 2019 with varying model types and window sizes (Borrelli et al., 2021)*

| | Bicoherence | | | STLT | STLT + Bicoherence | | |
|---|---|---|---|---|---|---|---|
| | 512 | 256 | 128 | | 512 | 256 | 128 |
| **A01** | 0.615 | 0.526 | 0.570 | 0.929 | 0.917 | 0.919 | **0.941** |
| **A02** | 0.881 | 0.873 | 0.863 | 0.940 | 0.940 | 0.939 | **0.946** |
| **A03** | 0.859 | 0.846 | 0.847 | 0.952 | 0.948 | 0.950 | **0.962** |
| **A04** | 0.546 | 0.505 | 0.499 | 0.886 | 0.827 | 0.879 | **0.915** |
| **A05** | 0.805 | 0.801 | 0.778 | 0.946 | 0.943 | 0.945 | **0.955** |
| **A06** | 0.655 | 0.628 | 0.609 | 0.898 | 0.868 | 0.898 | **0.932** |
| **All** | 0.726 | 0.695 | 0.687 | 0.926 | 0.907 | 0.921 | **0.942** |

*Table 3- Binary Classification Accuracy over the evaluation dataset in ASVSpoof 2019 with varying model types and window sizes (Borrelli et al., 2021)*

| | Bicoherence | | | STLT | STLT + Bicoherence | | |
|---|---|---|---|---|---|---|---|
| | 512 | 256 | 128 | | 512 | 256 | 128 |
| **A07** | 0.541 | 0.505 | 0.501 | 0.865 | 0.813 | 0.864 | **0.905** |
| **A08** | 0.693 | 0.627 | 0.591 | 0.951 | **0.955** | 0.955 | 0.954 |
| **A09** | 0.543 | 0.508 | 0.508 | 0.835 | **0.882** | 0.865 | 0.835 |
| **A10** | **0.534** | 0.516 | 0.504 | 0.511 | 0.492 | 0.487 | 0.493 |
| **A11** | 0.617 | 0.685 | **0.762** | 0.629 | 0.489 | 0.481 | 0.474 |
| **A12** | **0.547** | 0.524 | 0.511 | 0.509 | 0.504 | 0.498 | 0.487 |
| **A13** | 0.768 | 0.779 | 0.767 | 0.948 | **0.955** | **0.955** | 0.945 |
| **A14** | 0.718 | 0.708 | 0.726 | 0.882 | **0.916** | 0.906 | 0.880 |
| **A15** | **0.567** | 0.514 | 0.507 | 0.466 | 0.479 | 0.473 | 0.465 |
| **A16** | 0.544 | 0.516 | 0.509 | 0.872 | 0.833 | 0.871 | **0.908** |
| **A17** | 0.510 | 0.532 | 0.578 | 0.656 | 0.649 | **0.660** | 0.653 |
| **A18** | 0.515 | 0.534 | 0.537 | **0.869** | 0.849 | 0.843 | 0.849 |
| **A19** | 0.611 | 0.586 | 0.575 | 0.882 | 0.863 | 0.885 | **0.906** |
| **All** | 0.592 | 0.578 | 0.578 | 0.739 | **0.741** | 0.737 | 0.735 |

As shown in the above tables, the introduction of STLT leads to a significant improvement in classification accuracy in bona fide versus synthetic speech samples. In Table 2, it is apparent that all the highest classification on the development dataset occurs when using a Bicoherence + STLT model, and well as the majority of cases on the evaluation dataset signified by the bold entries. As a result of this performance, the above paper is utilized as a guide in generating a similar model to test on the ASVSpoof 2019 dataset. By utilizing Python and the package stingray (used to calculate bicoherence), a baseline model similar to the one discussed in (AlBadawy et al.) was initially developed and tested. Based on the performance, additional features are introduced using STLT to improve model performance.

### 2.3.2 - Deep Features

Synthetic speech detection with deep features is far less intuitive than handcrafted features. There are numerous models, and countless variations of each, deployed with varying loss functions, optimizers, etc. that have been used for detection, each displaying certain strengths with drawbacks likewise.

The paper 'One-Class Learning Towards Synthetic Voice Spoofing Detection' proposes a detection method based on a one-class feature (bona fide speech) in order to improve generalization ability (Zhang et al., 2021). Due to class imbalance, the authors highlight the poor generalization ability of various models and as a result, introduce a one-class feature with a loss function called one-class softmax, which is a variation of the traditional softmax function. By implementing angular margin to limit the embedding distributions, the one-class softmax function is expressed as follows:

$$L_{OCS} = \frac{1}{N}\sum_{i=1}^{N} \log\left(1 + e^{\alpha\left(m_{y_i} - \hat{w}_0 \hat{x}_i\right)(-1)^{y_i}}\right) \quad (9)$$

where $N$ is the number of samples, $\alpha$ is a chosen parameter, $m$ is the margin for cosine similarity, $w$ is the weight vector, $x$ is the feature vector, and $y$ is the associated label. The introduction of this loss function with ResNet-18 leads to roughly a 50% decrease in the equal error rate (EER) (Innovatrics).

The paper ‘A Deep Learning Framework for Audio Deepfake Detection’ proposes a detection method based on Temporal Convolutional Networks, or TCNs. TCNs have two distinctive features relative to CNNs (Khochare et al., 2021):

1. Output and input have the same length – a 1D full CNN where all hidden layers are kept identical to the input with 1 sized kernels
2. Causal convolutions are used – outputs from a certain time are convolved with components from that time.

   A general depiction of the TCN architecture is displayed in Figure 6.

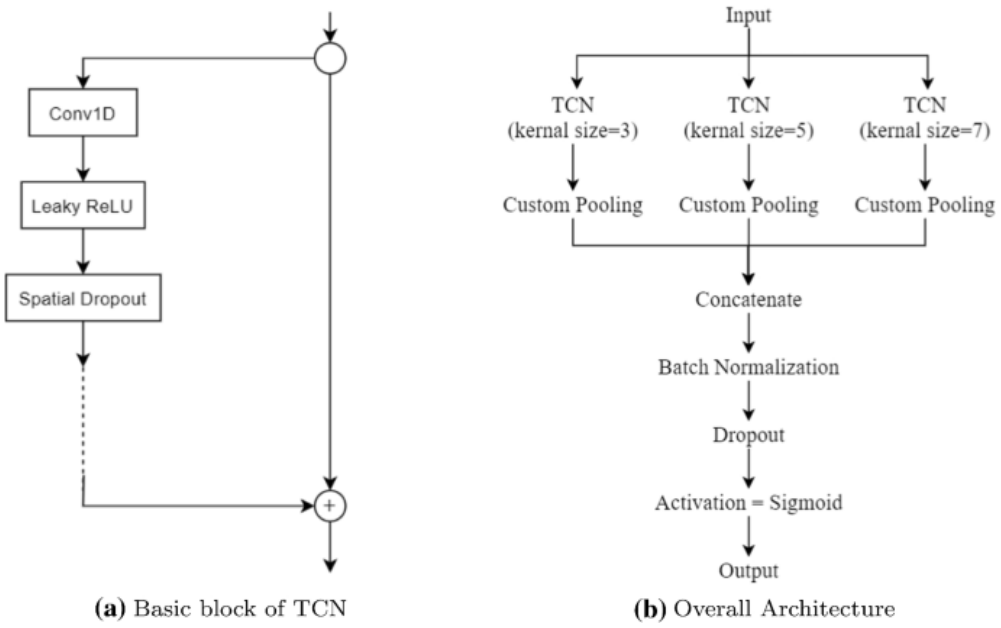


*Figure 6- general architecture of a TCN (Khochare et al., 2021)*

The TCN is able to achieve a validation and test accuracy of 98% and 92% relative to a feature-based implementation with maximum values of 85% and 67% respectively using the Fake or Real (FoR) Dataset (Khochare et al., 2021).

Finally, the paper ‘Towards End-to-End Synthetic Speech Detection’ proposes a model based on ResNet architecture called Time-Domain Synthetic Speech Nets, or TSSDNets (Hua et al., 2023). The paper is based on the development of an end-to-end system that replaces pre-transforms and handcrafted features, and utilizes only the standard components of audio. The TSSDNet uses two forms of CNNs with traditional ResNet skip connections and inception-style parallel convolutions (Hua et al., 2023). An inception module allows for the use of varying filter

sizes which are then concatenated to the following layer (Papers With Code). The paper hypothesizes that the use of a shallower network will outperform a deeper one since deeper ones may fail to represent the subtle data forgeries that are essential to synthetic speech detection (Hua et al., 2023). The two primary models explored in the paper are outlined in the below figure. The paper makes note of three primary techniques to improve model performance, the first is generating the CQT features, the second is the use of weighted cross-entropy loss due to the data imbalance, and finally the use of mixup regularization which aims to improve generalization ability (Hua et al., 2023), which are further elaborated in on in Section 3.3. The model was also trained and tested using the ASVSpoof 2019 dataset, and based on the results discussed, the proposed models lead to a decrease in the EER compared to state-of-the-art solutions designed for the challenge. As a result of this performance, the TSSDNet is utilized as a baseline model for the deep features model. The architecture of the models used in the paper's analysis are displayed in Figure 7.

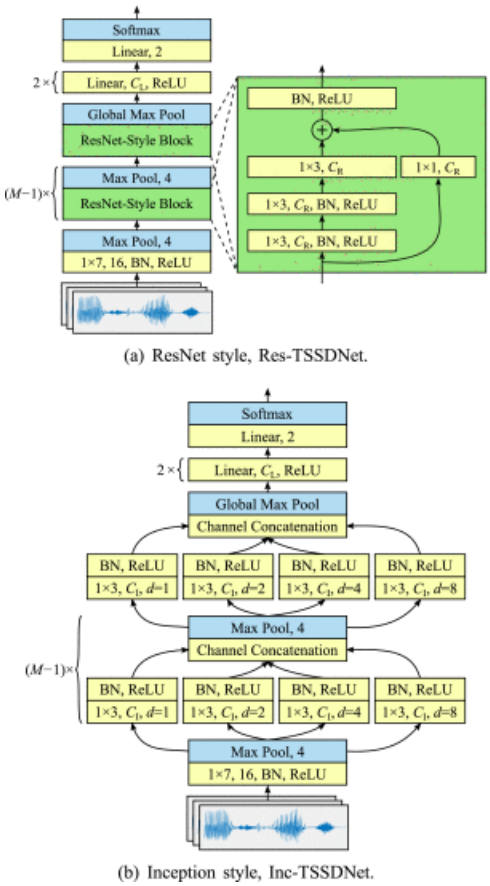


*Figure 7- end-to-end synthetic speech detection model architecture (Hua et al., 2023)*

Based on the research conducted from the listed papers and through exploring the goals of additional research articles, there tends to be a gap in the comparison of utilizing handcrafted features versus deep features. In more recent research, where deep features are highlighted, they are often described to have outperformed handcrafted features based on metrics such as EER, but the analysis does not develop any further. As a result, the goal of this thesis is to develop a handcrafted and deep feature model and compare their performance not only in terms of classification performance (accuracy and EER), but also in terms of computational power (number of trainable parameters and processing time per sample) in order to understand the trade-offs between the two. The following chapter will highlight the process of developing the two models based on the research conducted.

# 3 - Methodology

This chapter is organized as follows. Section 3.1 discusses the initial handcrafted features model that was built, using strictly bicoherence to attempt to categorize audio samples as bona fide or synthetic. Section 3.2 builds on the method introduced in 3.1 by adding STLT (short-term long-term prediction traces) as additional features to categorize the audio samples. Section 3.3 discusses the deep features model that will be used based on the Github repository. Finally, Section 3.4 discusses the two primary metrics that will be used to evaluate the performance of the two models relative to each other.

Figure 8 provides a high-level overview of the methodology which are referenced in the following two sections. The orange boxes represent the process for Section 3.1 and the blue boxes represent the additional preprocessing required for Section 3.2.

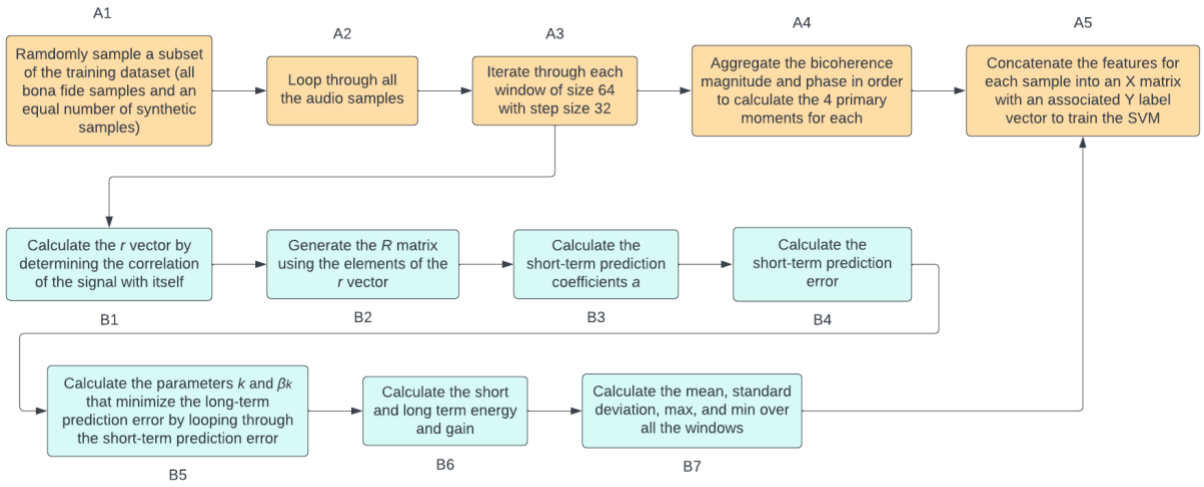


*Figure 8- flowchart that provides a high-level overview of the methodology*

## 3.1 - Handcrafted Features Model – Bicoherence Baseline

As discussed in Chapter 2 utilizing the bicoherence of audio samples is the baseline model for handcrafted features. The bicoherence is calculated using Equation 1 over multiple windows, followed by using Equations 2, 3, 4, and 5 to reduce the bicoherence into an 8D

feature vector for classification. The Python package *stringray* provides support for bicoherence calculations and the Github repository 'DetectingDeepFakes_BlackHat2019' contains a function to generate the bicoherence which was utilized in the developed code (Camacho-Rodriguez, 2019). The initial code pipeline is summarized in the following subsections.

### 3.1.1 – Data Preparation

For the training pipeline, the training txt file is loaded into Python which contains the training file names, the speaker ID, spoofing method, and associated label. In order to maintain class balance between the bona fide and synthetic samples, all the bona fide samples are utilized and a random subset of 2580 synthetic samples (with an equal representation of all spoofing methods) are selected. These 5160 samples are used as the training dataset as shown in **A1**.

### 3.1.2 – Data Processing

The bicoherence is calculated for each of the 5160 training samples. Within each sample, a window size of 64 and a step size of 32 is implemented as shown in **A2** and **A3**. Within each window, the bicoherence is calculated followed by the magnitude and phase over all the windows as shown in **A4**. The function windowed_bicoherence_features from DetectingDeepFakes_BlackHat2019 (Camacho-Rodriguez, 2019) is used to calculate the bicoherence and is extended to also include the analysis described in Equations 2, 3, 4, and 5. The mean, variance, skew, and kurtosis for both the magnitude and phase are calculated to generate an 8D vector for each audio sample. The 8D vectors for each audio sample are concatenated into an X matrix (of size 5160 by 8) and an associated Y vector with the respective labels (0 for bona fide and 1 for synthetic) is generated as shown in **A5**. Finally, a standard scaler is used to normalize the data in the X matrix.

### 3.1.3 – Model Training

An SVM is trained using the X and Y matrices developed in Subsection 3.1.2. The *SVC* object from *sklearn.svm* using the default parameters with X and Y as inputs. A classification report is generated to gather precision, recall, and accuracy metrics for the training loop.

### 3.1.4 – Evaluation

The steps outlined in Subsections 3.1.1, 3.1.2, and 3.1.3 are repeated for evaluation except with 650 bona fide and 650 synthetic samples from the evaluation dataset. Instead of training an SVM model, the model described in Subsection 3.1.3 is used to generate predictions on the evaluation dataset. The classification report is printed out to get the evaluation accuracy and the EER is also calculated.

### 3.1.5 – Class Imbalance

Due to the class imbalance over the entire ASVSpoof 2019 Dataset, an important consideration is how to adjust the SVM to account for the imbalance. SVMs contain a parameter *C*, which determines the penalty for misclassifying an observation. In the case of class imbalance, *C* can be adjusted specifically for each class depending on ratio of classes to each other. The equation for adjusting *C* can be expressed as follows:

$$C_k = C * w_j \text{ where } w_j = \frac{n}{k * n_j} \quad (10)$$

where $n$ is the number of observations, $n_j$ is the observations in class $j$, and $k$ is the total classes.

## 3.2 - Handcrafted Features Model – Bicoherence with STLT

Based on the results of the baseline model discussed in the previous section, the data processing in Subsection 3.1.2 is updated to additionally include short-term and long-term prediction traces as discussed in Chapter 2. The results discussed in (Borrelli et al., 2021) show

that a model that only uses bicoherence performs poorly on the ASVSpoof dataset, therefore, it is expected that STLT will also need to be included in the model.

The implementation of STLT is built into the already existing bicoherence analysis. Within each individual chunk of audio signal, the steps outlined by the paper are followed: (these steps are repeated for various $L$ values)

1. Based on the window size being used for bicoherence calculations, the same window is used to conduct STLT analysis
2. The short-term prediction coefficients are calculated first
    a. The autocorrelation of the signal is determined which is calculated by determining the correlation between the signal and itself as shown in **B1**
    b. The autocorrelation is segmented to range from the $N$-1 index to the $2*N-L$-1 index where $N$ is the length window
    c. The autocorrelation matrix, $R$, is set to be of size $L$ by $L$ (if $L$ is 5, want the matrix to be 5 by 5)
    d. The autocorrelation matrix is populated based on the autocorrelation vector according to Equation 5 as shown in **B2**
    e. The system of the inverse autocorrelation matrix and the autocorrelation vector is solved to generate the short-term prediction coefficients as shown in **B3**
3. The short-term prediction error is calculated based on Equation 6 as shown in **B4**
4. The long-term parameters $k$ and $\beta_k$, are estimated by using Equation 7 as shown in **B5**
    a. The short-term prediction error calculated in Step 3 is looped over
    b. The value of $\beta_i$ is calculated at each step in the short-term prediction error by dividing the error at that index by the error at index 0
    c. Using the value of $\beta_i$, the value of $q(n)^2$ is calculated using Equation 7
    d. The minimum $q(n)^2$ is stored which informs the choices of the two parameters and the values of $k$ and $\beta_k$ are set to the values of $i$ and $\beta_i$ where the minimum is achieved
5. The long-term prediction error using the parameters $k$ and $\beta_k$ using Equation 7 is calculated
6. For each window, the short/long term energy/gain is calculated to reduce the feature space to a vector according to Equation 8 as shown in **B6**

7. Over all windows, the mean, standard deviation, minimum, and maximum of the respective short/long term energy/gain are calculated to generate a 16D vector as shown in **B7**

The paper repeats the above steps for *L* values ranging from 1 to 50. Due to time constraints, the values of *L* are limited to 5, 10, 15, and 20. Based on this analysis in addition to the bicoherence process from Section 3.1, each 8D vector is expanded to 72D with the additional 4 * 16 elements added through the STLT analysis.

### 3.2.1 – Model Considerations

The code developed for the STLT + Bicoherence attempts to follow the process of 'Synthetic speech detection through short-term and long-term prediction traces' as closely as possible. Due to the complexity of the process, the initial iteration of the model lead to a training accuracy of 70% (the model developed uses only a partial subset of the ASVSpoof 2019 Dataset in order to maintain class balance and due to time and resource constraints, in comparison to the paper which uses the entire dataset). The paper is able to achieve significantly better results (85%+ accuracy for each class) and while the expectation of the model developed should be lower than the paper results, that significant of a difference highlighted a potential disconnect between what the paper was aiming to communicate compared to the code that was developed.

Due to the potential discrepancy, several authors of the paper were contacted in order to clarify the process of their analysis, however, there was a lack of response. As a result, the code was revisited after looking more closely at the behaviour of autoregressive models. After refactoring the code and rerunning the analysis, the model was able to achieve much better training results.

Even with these developments in performance, there were still a few challenges to overcome which made generating reliable results a challenge. As shown in Equation 5, the autocorrelation matrix needs to be inverted to solve for the short-term prediction coefficients. In some cases, the autocorrelation matrix that was generated was singular, and therefore could not be inverted. This was not an issue that was communicated in the paper; therefore, it was difficult to determine where the problem was stemming from. This was a primary reason for choosing to use *L* values of 5, 10, 15, and 20, since varying the *L* value could lead to singular matrices

generated for any sample. A significant number of samples were analyzed in order to guarantee there were enough samples that could be analyzed on the above $L$ values without the issue of singular matrices.

## 3.3 - Deep Features Model – Baseline

The baseline deep features model, which is also the final model used, is based on the GitHub repository 'end-to-end-synthetic-speech detection' (Hua, 2021) based off the paper 'Towards End-to-End Synthetic Speech Detection' discussed in Subsection 2.3.2. The name of the model is an Inc TSSDNet, which can be found pretrained within the repository.

In order to accurately develop model predictions, audio samples need to be preprocessed correctly, based on a time frame transformation. To ensure the training data is aligned, the audio features are truncated to maintain a six second sample across the two classes. The samples are then directly fed into the network at the same length to ensure the model parameters can be kept consistent.

Once the data has been preprocessed according to the outlined steps above, they are passed through the Inception TSSDNet. The model contains an initial softmax layer, followed by three fully connected linear layers, and a global max pooling layer. The inception style blocks are repeated 4 times as shown in Figure 9, followed by a max pool, followed again by 4 inception style blocks. A final max pool is applied to generate the eventual output of the neural network.

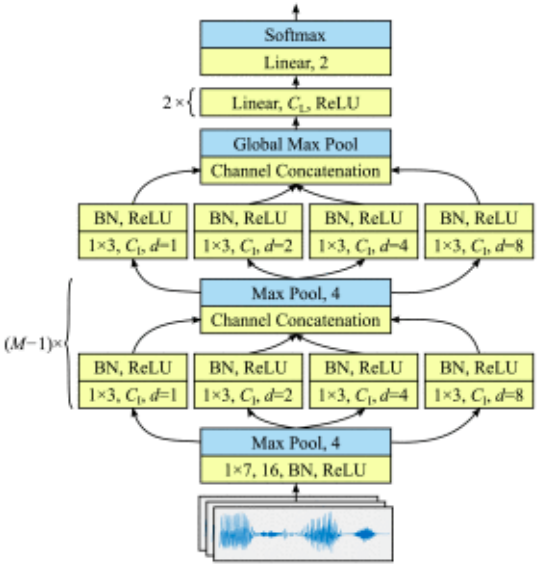


*Figure 9- Inc Time-Domain Synthetic Speech Net*

As mentioned in Subsection 2.3.2, the model utilizes weighted cross-entropy loss to learn the necessary parameters. The purpose of utilizing weighted cross-entropy loss is to account for the class imbalance between bona fide audio samples and synthetic audio samples in the ASVSpoof 2019 Dataset. The loss function is defined as

$$WCE(z, y_i) = -w_{y_i} \log\left(z_{y_i}\right) \quad (11)$$

where *z* contains the softmax probabilities of the two classes, and *w* is the inverse ratio of the label *y* in the training set. The model also utilizes the Adam Optimizer and an exponential learning rate decay.

Finally, the model also makes use of a technique called mixup regularization in order to generalize more effectively to unseen methods of attack. This is essential for the ASVSpoof 2019 Dataset since the evaluation dataset utilizes techniques that are never seen in the training dataset. The equation used for this form of regularization is

$$\tilde{x_i} = \lambda x_i + (1-\lambda)x_j, \qquad \tilde{y_i} = \lambda y_i + (1-\lambda)y_j \quad (12)$$

where $x_i$ and $x_j$ are two randomly selected samples and $\lambda \sim \text{Beta}(\alpha, \alpha)$ where alpha is a chosen parameter. The same equation is carried over to the loss function outlined in Equation 11.

## 3.4 – Evaluation Metrics

### 3.4.1 – Accuracy Metrics

The two primary accuracy metrics that are used to evaluate the handcrafted feature and deep feature models relative to each other are accuracy and EER. Accuracy is a primary metric utilized in classification models and is a very strong indicator of model performance. A model that is learning the data accurately displays a very high level of training accuracy, since there are only two labels, with a slight drop-off in evaluation accuracy in order to ensure the model is not overfitting to the training dataset.

EER is used as the second metric as a primary quantifier of biometric system performance. EER is the point on the ROC curve where the false acceptance rate equals the false rejection rate (of the null hypothesis – 'synthetic' in this case). This is a strong quantifier of biometric system performance such as in audio, visual, fingerprint scans, etc.

### 3.4.2 – Processing Metrics

The two primary processing metrics used to evaluate the handcrafted feature and deep feature models are processing time per sample and number of trainable parameters. Processing time per sample measures the average time required to conduct the necessary preprocessing, data manipulation, followed by passing the sample through the respective model. This provides a quantifiable metric on the speed of determining whether a speech sample is bona fide or synthetic.

Secondly, the number of trainable parameters is utilized to measure the processing ability of each respective model. The more trainable parameters, the longer the model takes to train and the higher the possibility of the model overfitting to the data it is exposed to during the training process.

Based on the methodology described above, there were two final models that were created: a Bicoherence + STLT Model and a pretrained TSSDNet Model. Each of these models were evaluated with respect to the evaluation metrics described in Section 3.4. The following chapter communicates the results of the two models.

# 4 - Results

This chapter is organized in a similar structure as Chapter 3. Section 4.1 begins by displaying the results of the baseline handcrafted features model which only utilizes bicoherence to generate feature vectors for each audio sample. Section 4.2 displays the results of the model that utilizes both bicoherence and STLT which generates more complex feature vectors compared to the baseline model. Finally, Section 4.3 displays the results of the deep features model. In each section, the results of the models are split into two tables: one that shows the accuracy metrics of each model and a second that shows the performance metrics.

## 4.1 - Handcrafted Features Model – Bicoherence Baseline

The initial bicoherence model developed used a window size of 64 with a step size of 32 based on the research conducted in (AlBadawy et al.). Two different iterations of the model were conducted: one with 600 samples of both bona fide and synthetic speech and another with 100 bona fide samples and 600 synthetic samples. A model with class imbalance was generated because the entire dataset contains more synthetic samples than bona fide samples, therefore, some initial results of class imbalance can be explored. The visualization similar to Figure 4 is displayed below in Figure 10. Each plot is generated based on a subset of the training dataset, where the green represents bona fide samples and each other colour represents one of six different spoofing methods contained in the dataset. As displayed, there is far less separation in bona fide samples compared to Figure 4, which is not too surprising based on the classification matrix of using bicoherence for classification on the ASVSpoof dataset as displayed in Figure 5.

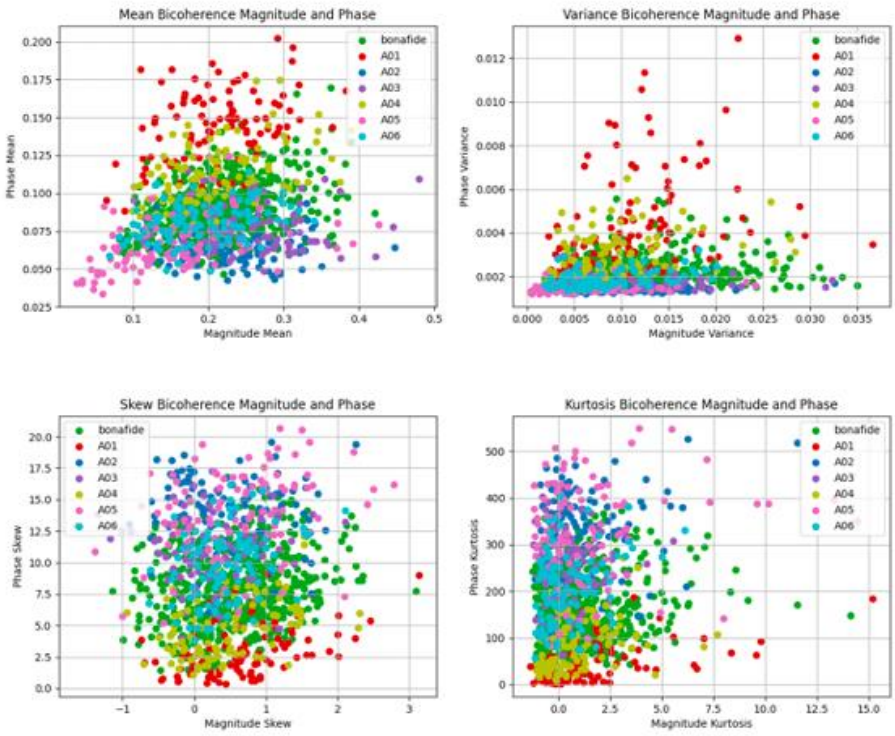


*Figure 10- visualization of 8D features generated from ASVSpoof Dataset*

Based on the process outlined previously and the considerations, the model led to the results summarized in Tables 4 and 5.

*Table 4- Model Performance when using 600 bona fide and synthetic samples with an evaluation set containing 200 bona fide and 600 synthetic samples*

| TRAINING RESULTS | | | | |
|---|---|---|---|---|
| | Precision | Recall | F1-Score | Support |
| 0 | 0.74 | 0.82 | 0.78 | 600 |
| 1 | 0.8 | 0.7 | 0.75 | 600 |
| | | | | |
| Accuracy | | | 0.76 | 1200 |
| Macro Average | 0.77 | 0.76 | 0.76 | 1200 |
| Weighted Average | 0.77 | 0.76 | 0.76 | 1200 |

| EVALUATION RESULTS | | | | |
|---|---|---|---|---|
| | Precision | Recall | F1-Score | Support |
| 0 | 0.38 | 0.84 | 0.52 | 200 |
| 1 | 0.91 | 0.54 | 0.68 | 600 |
| | | | | |
| Accuracy | | | 0.62 | 800 |
| Macro Average | 0.65 | 0.69 | 0.6 | 800 |
| Weighted Average | 0.78 | 0.62 | 0.64 | 800 |

*Table 5- Model Performance when using 100 bona fide and 600 synthetic samples with an evaluation set containing 100 bona fide and 600 synthetic samples*

| TRAINING RESULTS | | | | |
|---|---|---|---|---|
| | Precision | Recall | F1-Score | Support |
| 0 | 0.32 | 0.9 | 0.48 | 100 |
| 1 | 0.98 | 0.69 | 0.81 | 600 |
| | | | | |
| Accuracy | | | 0.72 | 700 |
| Macro Average | 0.65 | 0.79 | 0.64 | 700 |
| Weighted Average | 0.88 | 0.72 | 0.76 | 700 |

| EVALUATION RESULTS | | | | |
|---|---|---|---|---|
| | Precision | Recall | F1-Score | Support |
| 0 | 0.22 | 0.62 | 0.32 | 100 |
| 1 | 0.91 | 0.63 | 0.74 | 600 |
| | | | | |
| Accuracy | | | 0.63 | 700 |
| Macro Average | 0.56 | 0.62 | 0.53 | 700 |
| Weighted Average | 0.81 | 0.63 | 0.68 | 700 |

As displayed in the tables, the model performs relatively similarly overall when using a balanced versus unbalanced training set, albeit a very small training sample. The training statistics from the balanced iteration look relatively promising as the precision, recall, and f1 score are consistent over both classes, however, there is a greater disparity when tested on the evaluation dataset. This indicates that there are not enough samples in the training dataset necessary to provide consistent generalization ability on unseen data. In the case of the unbalanced iteration, both the training and evaluation statistics are quite sparse, even though the overall accuracy metrics is relatively consistent with the balanced iteration. This iteration has even fewer data samples than the balanced iteration, further suggesting the lack of data present in the training dataset. It is also difficult to comment on how modifying the *C* penalty parameter impacts the model results due to the limited data, however, its importance will be highlighted in a larger dataset. As a result of the above observations, another model iteration was developed with all the bona fide samples (2580) and an equal number of synthetic samples. The results of the balanced 2580-2580 model are displayed in Table 6.

*Table 6- Bicoherence Model Performance when using 2580 bona fide and synthetic samples with an evaluation set of 650 bona fide and synthetic samples*

| **TRAINING RESULTS** | | | | |
|---|---|---|---|---|
| | Precision | Recall | F1-Score | Support |
| 0 | 0.67 | 0.79 | 0.72 | 2580 |
| 1 | 0.74 | 0.61 | 0.67 | 2580 |
| | | | | |
| Accuracy | | | 0.7 | 5160 |
| Macro Average | 0.7 | 0.7 | 0.7 | 5160 |
| Weighted Average | 0.7 | 0.7 | 0.7 | 5160 |
| **EVALUATION RESULTS** | | | | |
| | Precision | Recall | F1-Score | Support |
| 0 | 0.62 | 0.63 | 0.62 | 650 |
| 1 | 0.62 | 0.61 | 0.61 | 650 |
| | | | | |
| Accuracy | | | 0.62 | 1300 |
| Macro Average | 0.62 | 0.62 | 0.62 | 1300 |
| Weighted Average | 0.62 | 0.62 | 0.62 | 1300 |
| EER | 0.26 | | | |

As shown in the table, the utilization of a larger training dataset leads to results that are far more consistent, however, still not highly accurate which is expected based on the research papers introduced in Section 2. This particular iteration indicates that the use of 5160 total samples in the training set has the potential to generalize well to unseen data. While there is a dropoff in evaluation performance, which is expected, the consistent metrics in the evaluation results indicate the model has a strong generalization ability if provided with additional features, which are be provided by STLT.

Table 7 summarizes the performance results of the baseline handcrafted features model. The necessary preprocessing steps outlined in Subsection 3.1.2 take roughly 6 seconds per

sample and the total number of trainable parameters is eight. Each training sample contains eight dimensions and the model utilized to perform the classification is an SVM which optimizes for a coefficient along each of the eight dimensions in order to separate the two classes with the greatest margin.

*Table 7- Performance results of Bicoherence Model*

| PERFORMANCE RESULTS | | |
|---|---|---|
| | Processing Time per Sample | Number of Trainable Parameters |
| Value | ~3 seconds | 8 |

## 4.2 - Handcrafted Features Model – Bicoherence with STLT

Table 8 displays the results based on the methodology discussed in Section 3.2. The difference in class representation is due to issues discussed in Subsection 3.2.1. The results communicated in Table 8 below show a significant increase in classification accuracy after introducing STLT to the feature vectors. The model is not only able to achieve a greater overall accuracy, but stronger consistency between the two classes in terms of precision, recall, and f1-score.

*Table 8- Bicoherence Model + STLT Performance when using 2520 bona fide and 2280 synthetic samples with an evaluation set of 650 bona fide and synthetic samples*

| **TRAINING RESULTS** | | | | |
|---|---|---|---|---|
| | Precision | Recall | F1-Score | Support |
| 0 | 0.83 | 0.86 | 0.84 | 2520 |
| 1 | 0.84 | 0.81 | 0.82 | 2280 |
| | | | | |
| Accuracy | | | 0.83 | 4800 |
| Macro Average | 0.83 | 0.83 | 0.83 | 4800 |
| Weighted Average | 0.83 | 0.83 | 0.83 | 4800 |
| **EVALUATION RESULTS** | | | | |
| | Precision | Recall | F1-Score | Support |
| 0 | 0.69 | 0.82 | 0.75 | 650 |
| 1 | 0.78 | 0.64 | 0.70 | 650 |
| | | | | |
| Accuracy | | | 0.73 | 1300 |
| Macro Average | 0.74 | 0.73 | 0.73 | 1300 |
| Weighted Average | 0.74 | 0.73 | 0.73 | 1300 |
| EER | 0.18 | | | |

The evaluation results display a decrease in the classification accuracy of the model which is to be expected. The precision and recall metrics become increasingly sparse and the model on average can correctly classify approximately three out of every four samples. The EER produced by the bicoherence + STLT model is lower than when using only bicoherence, which indicates stronger predictive power in the model.

Table 9 summarizes the performance results of the baseline handcrafted features model with STLT added. The necessary preprocessing steps outlined in Section 3.2 take roughly 6 seconds per sample and the total number of trainable parameters is 72. The bicoherence accounts for 8 dimensions, with an additional 16 * 4 dimensions added due to the STLT calculations.

*Table 9- Performance results of Bispectrum Model + STLT*

| PERFORMANCE RESULTS | | |
|---|---|---|
| | Processing Time per Sample | Number of Trainable Parameters |
| Value | ~6 seconds | 72 |

## 4.3 - Deep Features Model – Baseline

Table 10 summarizes the accuracy results for the deep features model. The training results are limited since the model is pretrained, however, the evaluation results include the same metrics discussed for the handcrafted features model in Sections 4.1 and 4.2. As shown by the accuracy, precision, recall, and f1-score, the deep feature model performs exceptionally well in its ability to decipher between bona fide and synthetic samples in comparison to the handcrafted features model. The accuracy metrics are not only almost 100%, but the consistency in the metrics across both classes is very consistent which signifies strong predictive performance of the model towards data it has not seen.

*Table 10- Deep Features Model Performance when using the entire ASVSpoof training dataset with an evaluation set of 650 bona fide and synthetic samples*

| TRAINING RESULTS | | | | |
|---|---|---|---|---|
| Accuracy | 0.99 | | Support | 25380 |
| **EVALUATION RESULTS** | | | | |
| | Precision | Recall | F1-Score | Support |
| 0 | 0.95 | 0.97 | 0.96 | 650 |
| 1 | 0.97 | 0.95 | 0.96 | 650 |
| | | | | |
| Accuracy | | | 0.96 | 1300 |
| Macro Average | 0.96 | 0.96 | 0.96 | 1300 |
| Weighted Average | 0.96 | 0.96 | 0.96 | 1300 |
| EER | 0.03 | | | |

Table 11 summarizes the performance results of the deep features model. As shown, the processing time per sample is significantly quicker in comparison to the handcrafted features model because there is relatively little preprocessing on the audio samples that has to be conducted. On the other hand, since the deep features model contains several highly complex layers, the number of trainable parameters is exponentially greater than the handcrafted features model.

*Table 11- Performance results of the deep features model*

| **PERFORMANCE RESULTS** | | |
|---|---|---|
| | Processing Time per Sample | Number of Trainable Parameters |
| Value | ~0.2 seconds | 92658 |

The results of the two generated models are summarized in the above tables. Tables 8, 9, 10, and 11 communicate the final results of both the handcrafted features and the deep features model. The following chapter will discuss the implications of the results in further detail and make a final recommendation.

# 5 – Discussion, Conclusion, and Future Work

This chapter is organized as follows. Section 5.1 discusses the results of both the handcrafted features model and deep features model before comparing the performance of both relative to each other in order to determine which model is superior. Section 5.2 concludes the findings of the thesis and makes a final recommendation to The Bank based on the results discussed. Finally, Section 5.3 discusses future paths to consider to expand the research conducted in this thesis.

## 5.1 – Discussion

### 5.1.1 – Handcrafted Features Model

The development of results was an iterative process; by interpreting results at various stages of research, model performance improved with more promising results. At the onset of the project, the methodology discussed in Section 3.1 based on (AlBadawy et al.) was implemented using both concepts discussed in the paper as well as starter code provided in (Camacho-Rodriguez, 2019). This process was quite challenging for a couple of reasons: there were a few dependency issues when first using the stingray package and the results of using bicoherence on the ASVSpoof 2019 Dataset did not display significant signs of promise. While the results discussed in (AlBadawy et al.) are not based on the ASVSpoof 2019 Dataset, the predictive power of the model discussed in the paper compared to the results that are generated on a subset of the ASVSpoof 2019 Dataset significantly diverge. This is highlighted by the left image in Figure 3 and Figure 10; the paper is able to achieve a high degree of separation between bona fide and synthetic speech across only two dimensions while the results on the ASVSpoof 2019 Dataset are not able to achieve any significant separation across all eight dimensions of bicoherence. Even when expanding the dataset to 2580 samples of each bona fide and synthetic speech as displayed in Table 6, the model performs relatively poorly. The evaluation results show that the model is only approximately 60% accurate on out-of-sample data, which is a minor

improvement from 50% which can be achieved by guessing, since this is a binary classification problem.

Subsection 3.1.5 discusses the introduction of the *C* parameter to deal with class imbalance in SVM models and some initial results are displayed in Table 4 and 5. There was very limited consistency in the results of using a larger unbalanced dataset, and therefore, only training on a balanced dataset was pursued in future model iterations.

Due to the model's limited performance compared to the paper's results, one consideration made was to manually calculate the bicoherence values of audio samples, in the case that the stingray package differs from the method implemented in the paper. However, even with manual calculations, the performance of the model was still relatively poor. Additional research was conducted to identify potential methods of improving the performance of the model specifically on the ASVSpoof 2019 Dataset, and the STLT method discussed in (Borrelli et al., 2021) was considered a method of improving model performance. Similar to the baseline bicoherence model, implementing STLT into the data pipeline was a complex process which required several iterations before being able to generate any significant results as discussed in Subsection 3.2.1. However, the final iteration of the STLT model developed shows significant signs of promise. The results discussed in (Borrelli et al., 2021) show that the SVM model was able to achieve 93% training accuracy while not only classifying based on bicoherence versus synthetic, but also determining which spoofing method was used in each synthetic sample. In the final model iteration discussed in Section 4.2, the training accuracy achieved was 83%, which is lower than the paper while also utilizing a simpler binary classification methodology. The discrepancy in training results can be attributed to a couple of reasons:

1. *L* values: the final model iteration developed uses *L* values of 5, 10, 15, and 20 compared to the paper which uses *L* values starting at 1 and ending at 50. Generating STLT features for one value of *L* takes approximately 8 hours of processing for only 4800 samples, while the entire ASVSpoof Dataset contains 20000+ samples. With a longer timeline as well as faster processor, the model can be expanded to consider all the *L* values the paper uses.
2. Number of samples: due to timing constraints and issues that arose from using an unbalanced dataset, only a subset of the ASVSpoof Dataset is used in the final model.

Similar to the problem discussed above, a longer timeline and faster processor will help to mitigate this issue.

While the training metrics are important to discuss, the out-of-sample results are the most important in analyzing the predictive power of the model. Specifically in the case of The Bank, if a similar model was to be utilized in voiceprint, the model needs to show significant generalization ability to unseen spoofing attacks that the model was trained on. The results in Table 8 show that the handcrafted model is able to correctly classify approximately three out of every four samples (73%) which is a strong improvement compared to the baseline model while only including four values of $L$. The paper is able to achieve an out-of-sample accuracy of ~86%, and the discrepancy in the model generated compared to the paper can be attributed to the same two reasons discussed above. The final model is also able to achieve a 0.18 EER, which signifies a fairly strong performance in terms of a biometric system. For a simpler model with limited training data, the EER value is not surprising and can be expected to increase with a larger training dataset and increased STLT features.

Overall, the results of the handcrafted model display a strong progression in performance as additional considerations were made. The initial model iteration had an evaluation accuracy of ~61% and after adding additional features based on STLT, the final model iteration has an evaluation accuracy of ~73%. This final model iteration should be considered a proof of concept. The results of the model indicate that STLT in addition to bicoherence serve as appropriate features to classify audio samples, and given a longer timeline with additional processing power, the model can be extended even further to achieve results closer in line with the research conducted. These points are important to consider especially when comparing results with the deep features model.

### 5.1.2 – Deep Features Model

The results of deep features model are much easier to interpret than the handcrafted features model. The deep features model is pretrained, and therefore, the results that are generated are primarily to confirm the performance discussed in the paper, as well as calculate the training and evaluation accuracy, EER, processing time per sample, and the number of trainable parameters. The deep features model performed extremely well, with a training

accuracy of ~99% and an evaluation accuracy of ~96%. These metrics alone indicate the level of predictive power the deep features model has, as well as its generalization ability to data it has not seen indicated by small 3% drop-off in evaluation accuracy relative to training accuracy. In addition to accuracy, the deep features model is also able to achieve an EER of 0.03 which indicates very strong performance of a biometric system.

### 5.1.3 – Comparison of Handcrafted and Deep Features Model

The primary goal of the thesis is to compare the performance of both handcrafted and deep features in terms of accuracy and processing time. This subsection compares the performance of models based on their evaluation accuracy, EER, processing time per sample, and number of trainable parameters. One important consideration is the handcrafted features model is a modified version compared to the ideal model implementation if this project were to continue. Therefore, the statements made to indicate model performance take into account how the model performed based on the results discussed and how the model in (Borrelli et al., 2021) performed on the entire ASVSpoof Dataset with an increased feature set.

The evaluation accuracy for the handcrafted features model is 73% compared to the deep features model with a value of 96%. In comparing this metric, it is clear that the deep features model significantly outperforms the handcrafted features model. Even with training the model on the entire ASVSpoof Dataset and by including features for all 50 L values, the paper is only able to achieve an evaluation accuracy of 86%, which is still a significant difference from the 96% achieved by the deep features model. By developing a model similar to the paper that also simply classifies audio samples as bona fide or synthetic, there will unarguably be an increase from the 86% achieved in classifying audio samples by their spoofing method. It is difficult to comment on the increase that can be expected; something that is only possible to confirm practically. The second accuracy metric to rate the performance of the models is EER. The handcrafted features model is able to achieve an EER of 0.18 compared to the 0.03 of the deep features model. Similar to evaluation accuracy, the deep features model significantly outperforms the handcrafted features model. By further developing the model, it is hard to see the EER of the handcrafted features model coming close to the deep features model. Since the evaluation accuracy will be significantly less, and due to the fact that the accuracy of a binary classification

model is highly correlated to the EER, it can be concluded that the deep features model also outperforms the handcrafted features model in terms of EER.

The two performance metrics to rank the handcrafted and deep features model are processing time per sample and the number of trainable parameters. The processing time for the handcrafted features model is approximately six seconds in comparison to the handcrafted features model which is approximately 0.2 seconds. These numbers indicate that given an audio sample, the handcrafted features model takes six seconds to determine the class of the speech sample compared to only 0.2 seconds for the deep features model. In voiceprint architecture for The Bank, being able to decipher between bona fide and synthetic audio within a relatively small timeframe is essential to minimize call wait times and avoid the discussion of private information should the analysis take too long. Based on this metric, it can be concluded that the deep features model outperforms the handcrafted features model. The final metric utilized is the number of trainable parameters, and it is clear the handcrafted features model outperforms the deep features model by a significant margin, 72 compared to 90000+. This metric indicates that the training process of the deep features model will take significantly longer than the handcrafted features model. If new spoofing methods are developed over time and The Bank implements a model to their voiceprint architecture, retraining the deep features model to account for the new spoofing methods would take longer in comparison to a handcrafted features model. Therefore, it can be concluded that the handcrafted features model outperforms the deep features model in terms of number of trainable parameters.

Based on the four primary metrics to rank the two models' performance, the deep features model outperforms the handcrafted features model in three of the four. It provides a significant advantage in both accuracy metrics which are arguably the two most important metrics to consider when implementing synthetic speech detection models.

## 5.2 – Conclusion

The results displayed in Chapter 4 and the ensuing discussion in Chapter 5 highlight the superior performance of the deep features model compared to the handcrafted model both in terms of time and accuracy. Comparing the actual metric values of the two models should be done with a grain of salt considering the handcrafted features model is a modified version of

what can be possible with additional time and processing ability. However, even with keeping this in mind and given the results discussed in research papers, it is hard to argue against the superior performance of the deep features model. The results of this research indicates that should The Bank look to implement synthetic speech detection methods in their voiceprint architecture, a deep features model should be the preferred choice over a handcrafted features model. The strong generalization ability paired with the fast processing time of an audio sample highlight the robustness of the model and practicality of implementation in The Bank's existing architecture.

## 5.3 – Future Work

There are a couple of future directions the research in this paper can be extended to verify the conclusions made. The first is to extend the handcrafted features model to include all the speech samples in the ASVSpoof 2019 Dataset and extend the number of features for each sample by including additional values of $L$. In order to include all the samples in the dataset, additional research would have to be conducted involving the $C$ parameter when using an unbalanced dataset for SVM. Using the $C$ parameter with an unbalanced dataset led to inconsistent accuracy results which would need to be investigated further. Increasing the number of features for each sample involved conducting analysis using more values of $L$. The paper (Borrelli et al., 2021) uses $L$ values ranging from 1 to 50 and it would be interesting to gauge model performance by extending this number. Additionally, given the limitations discussed in Subsection 3.2.1 on the inversion of the autocorrelation matrix, the Moore-Penrose Inverse can be implemented into the data pipeline to approximate the inverse of non-invertible autocorrelation matrices to ensure all audio samples can be utilized to train the model.

The second direction to extend the research in the paper is to generate personalized audio samples to test the model's behaviour. This was considered towards the end of the research process, however, there were resource limitations which prevented any further progress. Utilizing new synthetic speech generation methods, such as Microsoft Vall-e, can provide high quality synthesized speech based on personal samples in order to test the effectiveness of both the handcrafted and deep features model.

# 6 - Appendix

## Appendix 1

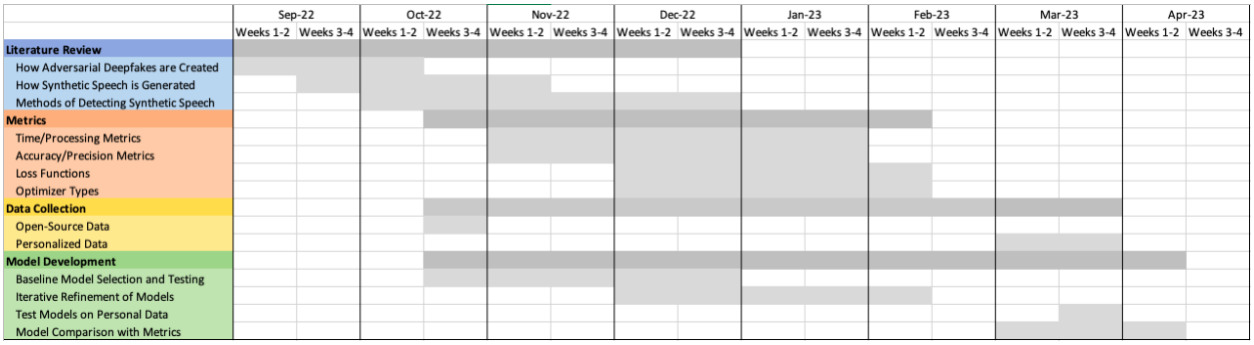


*Figure 11- thesis project plan*

# Appendix 2

*Table 12- TTS synthesis methods*

| Methods | Technique | Features | Dataset | Limitations |
|---|---|---|---|---|
| WaveNet [55] | Deep neural network | ▪ linguistic features<br>▪ fundamental frequency (log F0) | ▪ VCTK (44 hrs.) | ▪ Computationally complex |
| Tacotron [56] | Encoder-Decoder with RNN | ▪ Deep features | Private (24.6 hrs.) | ▪ Costly to train the model |
| Deep Voice 1[57] | Deep neural networks | ▪ linguistic features | Private (20 hrs.) | ▪ Independent training of each module leads to a cumulative error in synthesized speech |
| Deep Voice 2 [237] | RNN | ▪ Deep features | VCTK (44 hrs.) | ▪ Costly to train the model |
| DeepVoice3 [224] | Encoder-decoder | ▪ Deep features | ▪ Private (20 hrs.)<br>▪ VCTK (44 hrs.)<br>▪ LibriSpeech ASR (820 hrs.) | ▪ Does not generalize well for unseen samples. |
| Parallel WaveNet [231] | Feed-forward neural network with dilated causal convolutions | ▪ linguistic features | Private | ▪ Requires a large amount of the target's speech training data. |
| VoiceLoop [230] | Fully-connected neural network | ▪ 63-dimensional audio features | ▪ VCTK (44 hrs.)<br>▪ Private | ▪ Low ecological validity |
| Tacotron2[238] | ▪ Encoder-decoder | ▪ linguistic features | ▪ Japanese speech corpus from the ATR Ximera dataset (46.9 hrs.) | ▪ Lack of real-time speech synthesis |
| Arik et al. [59] | Encoder- decoder | ▪ Mel spectrograms | ▪ LibriSpeech (820 hrs.)<br>▪ VCTK (44 hrs.) | ▪ Low performance for multi-speaker speech generation in the case of low-quality audio |
| Jia et al. [233] | Encoder-decoder | ▪ Mel spectrograms | ▪ LibriSpeech (436 hrs.)<br>▪ VCTK (44 hrs.) | ▪ Fails to attain human-level naturalness<br>▪ Lacks in transferring the target accent, prosody to synthesized speech |
| Luong et al. [228] | Encoder-decoder | ▪ Mel spectrograms | ▪ LibriSpeech (245 hrs.)<br>▪ VCTK (44 hrs.) | ▪ Low performance in the case of noisy audio samples |
| Chen et al. [235] | Encoder + deep neural network | ▪ Mel spectrograms | ▪ LibriSpeech (820 hrs.)<br>▪ private | ▪ Low performance in the case of a low-quality audio sample |
| Cong et al. [236] | Encoder-decoder | ▪ Mel spectrograms | ▪ MULTI-SPK<br>▪ CHiME-4 | ▪ Lacks in synthesizing utterances of a target speaker |

# Appendix 3

*Table 13- VC synthesis methods*

| Methods | Technique | Features | Dataset | Limitations |
|---|---|---|---|---|
| Ming et al. [248] | DBLSTM | F0 and energy contour | ▪ CMU-ARCTIC [273] | ▪ Requires parallel training data |
| Nakashika et al. [247] | Recurrent temporal restricted Boltzmann machines (RTRBMs) | MCC, F0, and aperiodicity | ▪ ATR Japanese speech database [274] | ▪ Lacks temporal dependencies of speech sequences |
| Sun et al. [249] | DBLSTM-RNN | MCC, F0 and Aperiodicity | ▪ CMU-ARCTIC [273] | ▪ Requires parallel training data |
| Wu et al. [250] | DBLSTM- i-vectors | 19D-MCCs, Delta and Delta-Delta, F0, 400-D i-vector | ▪ VCTK corpus | ▪ Computationally complex |
| Liu et al. [251] | WaveNet vocoder | MCC and F0 | ▪ VCC 2018 | ▪ Performance degrades on inter-gender conversions |
| Kaneko et al. [255] | Encoder-decoder with GAN | 34D-MCC, F0, and aperiodicity | ▪ VCC 2018 | ▪ Computationally complex<br>▪ Domain-specific voice |
| Kameoka et al. [258] | Encoder-decoder with GAN | 36D-MCC, F0, and aperiodicity | ▪ VCC 2018 | ▪ Performance degrades on cross-gender conversion<br>▪ Low performance for unseen speakers |
| Zhang et al. [259] | WAW-GAN | STRAIGHT spectra [275], F0 and aperiodicity | ▪ VCC2016 | ▪ Lacks target speaker similarity |
| Huang et al. [260] | Encoder-decoder | STRAIGHT spectra [275] MCCs | ▪ VCC 2018 | ▪ Lacks multi-target VC<br>▪ Introduces abnormal fluctuations in generated speech |
| Chorowski et al. [262] | VQ-VAE, WaveNet decoder | 13D-MFCC | ▪ LibriSpeech<br>▪ ZeroSpeech 2017 | ▪ Over smooth and low naturalness in generated speech<br>▪ Increased training complexity |
| Tanaka et al. [263] | BiLSTM encoder-LSTM decoder | Acoustic features | ▪ CMU Arctic database | ▪ Requires extensive training data |
| Park et al. et al. [264] | Encoder-decoder | Mel-spectrogram | ▪ LibriTTS<br>▪ VCTK dataset | ▪ Requires transcribed data<br>▪ Lacks target speaker similarity |
| Huang et al. [265] | VAE-vocoder | MCCs, log F0, and aperiodicity | ▪ CMU ARCTIC<br>▪ VCTK corpus | ▪ Requires parallel training data |
| Lu et al. [266] | Attention mechanism in encoder-decoder | 13D-MFCCs, PPGs and log F0 | ▪ VCTK corpus | ▪ Low target similarity and naturalness in generated speech |
| Liu et al. [267] | Encoder and DBLSTM | 19 MFCCs, log F0 and PPG | ▪ VCTK corpus | ▪ Low target similarity and naturalness in generated speech |
| Chou et al. [270] | Attention mechanism in encoder-decoder | 19 MFCCs, log F0 and PPG | ▪ VCTK Corpus | ▪ Low quality of converted voices in case of noisy samples |
| Qian et al. [271] | Encoder-decoder | speech spectrogram | ▪ VCTK corpus | ▪ Prosody flipping between the source and the target.<br>▪ Not well-generalized to unseen data |

## Appendix 4

==========

ASVspoof 2019: The 3rd Automatic Speaker Verification Spoofing and Countermeasures Challenge database

Logical access (LA)

==========

1. Directory Structure

---

--> LA

- --> ASVspoof2019_LA_asv_protocols
- --> ASVspoof2019_LA_asv_scores
- --> ASVspoof2019_LA_cm_protocols
- --> ASVspoof2019_LA_dev
- --> ASVspoof2019_LA_eval
- --> ASVspoof2019_LA_train
- --> README.LA.txt

2. Description of the audio files

---

ASVspoof2019_LA_train, ASVspoof2019_LA_dev, and ASVspoof2019_LA_eval contain audio files for training, development, and evaluation (LA_T_*.flac, LA_D_*.flac, and LA_E_*.flac, respectively). ASVspoof2019_PA_dev, and ASVspoof2019_PA_eval contain audio files to enroll ASV system. The audio files in the directories are in the flac format.

The sampling rate is 16 kHz, and stored in 16-bit.

3. Description of the protocols

---

CM protocols:

ASVspoof2019_LA_cm_protocols contains protocol files in ASCII format for ASVspoof countermeasures:

ASVspoof2019.LA.cm.train.trn.txt: training file list
ASVspoof2019.LA.cm.dev.trl.txt: development trials
ASVspoof2019.LA.cm.eval.trl.txt: evaluation trials

Each column of the protocol is formatted as:

SPEAKER_ID AUDIO_FILE_NAME - SYSTEM_ID KEY

1) SPEAKER_ID: LA_****, a 4-digit speaker ID
2) AUDIO_FILE_NAME: LA_****, name of the audio file
3) SYSTEM_ID: ID of the speech spoofing system (A01 - A19), or, for bona fide speech SYSTEM-ID is left blank ('-')
4) -: This column is NOT used for LA.
5) KEY: 'bona fide' for genuine speech, or, 'spoof' for spoofing speech

Note that:

1) the third column is left blank (-) to make the structure coherent with physical access file list;
2) Brief description on LA spoofing systems, where TTS and VC denote text-to-speech and voice-conversion systems:

| | | |
|---|---|---|
| A01 | TTS | neural waveform model |
| A02 | TTS | vocoder |
| A03 | TTS | vocoder |
| A04 | TTS | waveform concatenation |
| A05 | VC | vocoder |
| A06 | VC | spectral filtering |
| A07 | TTS | vocoder+GAN |
| A08 | TTS | neural waveform |
| A09 | TTS | vocoder |
| A10 | TTS | neural waveform |
| A11 | TTS | griffin lim |
| A12 | TTS | neural waveform |
| A13 | TTS_VC | waveform concatenation+waveform filtering |
| A14 | TTS_VC | vocoder |

| | | |
|---|---|---|
| A15 | TTS_VC | neural waveform |
| A16 | TTS | waveform concatenation |
| A17 | VC | waveform filtering |
| A18 | VC | vocoder |
| A19 | VC | spectral filtering |

ASV protocols:

ASVspoof2019_LA_asv_protocols contains the protocol files for ASV system

ASVspoof2019.LA.asv.<1>.<2>.<3>.txt
where
<1> is either 'dev' or 'eval' based on whether the files describe the development or evaluation protocol,
<2> ie either 'male (m)' or 'female (f)' separating the genders from each other or 'gender independent (gi)'
contains trials for both genders (male trials followed by female trials),
<3> is either 'trl' or 'trn' (trl = trial list, trn = speaker enrollment list).

Trial (trl) file format for LA scenario:
1st column: claimed speaker ID
2nd column: test file ID
3rd column: spoof attack ID (or 'bona fide' if the speech is not synthetic)
4th column: key (target = target trial, nontarget = impostor trial, spoof = spoofing attack)

Enrollment (trn) file format:
1st column: ID of enrolled speaker
2nd column: IDs of files used in the enrollment separated by commas

4. Baseline ASV scores

ASVspoof2019_LA_asv_scores contains the scores calculated by a baseline ASV system for t-DCF evaluation

ASVspoof2019.LA.asv.dev.gi.trl.scores.txt: scores given by the ASV system for development set data
ASVspoof2019.LA.asv.eval.gi.trl.scores.txt: scores given by the ASV system for evaluation set data

Each column is formatted as:

CM_KEY ASV_KEY SCORES

1) CM_KEY: 'bona fide' for genuine speech, or, the ID of the spoofing attack (A01 - A19)
2) ASV_KEY: 'target' for claimed speaker, or, 'nontarget' for impostor speaker, or, 'spoof' for spoofing speech
3) SCORES: similarity score value

---

# 7 - References

AlBadawy, E. A., Lyu, S., & Farid, H. (n.d.). Detecting AI-Synthesized Speech Using Bispectral Analysis.

Boles, A., & Rad, P. (2017). Voice biometrics: Deep Learning-based Voiceprint Authentication System. 2017 12th System of Systems Engineering Conference (SoSE). https://doi.org/10.1109/sysose.2017.7994971

Borrelli, C., Bestagini, P., Antonacci, F., Sarti, A., & Tubaro, S. (2021). Synthetic speech detection through short-term and long-term prediction traces. EURASIP Journal on Information Security, 2021(1). https://doi.org/10.1186/s13635-021-00116-3

Brownlee, J. (2017, May 31). A Gentle Introduction to Long Short-Term Memory Networks for Machine Learning. Machine Learning Mastery. https://machinelearningmastery.com/gentle-introduction-long-short-term-memory-networks-experts/

Camacho-Rodriguez, C. (2019). DetectingDeepFakes_BlackHat2019. GitHub repository. https://github.com/cmrfrd/DetectingDeepFakes_BlackHat2019

Hua, G. (2021). End-to-End Synthetic Speech Detection. GitHub repository. https://github.com/ghua-ac/end-to-end-synthetic-speech-detection

Hua, G., Teoh, A. B., & Zhang, H. (2021). Towards end-to-end synthetic speech detection. IEEE Signal Processing Letters, 28, 1265–1269. https://doi.org/10.1109/lsp.2021.3089437

IBM. (n.d.). How SVM works. Retrieved March 28, 2023, from https://www.ibm.com/docs/en/spss-modeler/saas?topic=models-how-svm-works#:~:text=SVM%20works%20by%20mapping%20data,be%20drawn%20as%20a%20hyperplane.

IBM. (n.d.). Recurrent Neural Networks. Retrieved March 28, 2023, from https://www.ibm.com/topics/recurrent-neural-networks

Innovatrics. (n.d.). Equal Error Rate (EER) definition. Innovatrics Glossary. Retrieved March 28, 2023, from https://www.innovatrics.com/glossary/equal-error-rate-eer/#:~:text=Equal%20Error%20Rate%20(EER)%20definition&text=The%20EER%20is%20the%20location,accuracy%20of%20the%20biometric%20system.

Innovatrics. (n.d.). Equal Error Rate (EER) definition. Retrieved March 28, 2023, from https://www.innovatrics.com/glossary/equal-error-rate-eer/#:~:text=Equal%20Error%20Rate%20(EER)%20definition,false%20rejection%20rate%20are%20equal

Khochare, J., Joshi, C., Yenarkar, B., Suratkar, S., & Kazi, F. (2021). A deep learning framework for audio deepfake detection. Arabian Journal for Science and Engineering, 47(3), 3447–3458. https://doi.org/10.1007/s13369-021-06297-w

Masood, M., Nawaz, M., Malik, K.M. et al. Deepfakes generation and detection: state-of-the-art, open challenges, countermeasures, and way forward. Appl Intell 53, 3974–4026 (2023). https://doi.org/10.1007/s10489-022-03766-z

Mehta, S. (2021, June 22). Neural vocoder and its application in speech recognition. Analytics India Magazine. Retrieved March 28, 2023, from https://analyticsindiamag.com/neural-vocoder-and-its-application-in-speech-recognition/#:~:text=Neural%20vocoders%20are%20a%20frequent,representations%20such%20as%20Mel%2DSpectrograms.

Mohammadi, M., & Sadegh Mohammadi, H. R. (2017). Robust features fusion for text independent speaker verification enhancement in Noisy Environments. 2017 Iranian Conference on Electrical Engineering (ICEE). https://doi.org/10.1109/iraniancee.2017.7985357

Papers With Code. (n.d.). Inception Module. Retrieved March 28, 2023, from https://paperswithcode.com/method/inception-module

The Bank. (n.d.). About The Bank. Retrieved March 28, 2023, from https://www.TheBank.com/about-TheBank.html

The Guardian. (2020, January 13). What are deepfakes and how can you spot them? Retrieved March 28, 2023, from https://www.theguardian.com/technology/2020/jan/13/what-are-deepfakes-and-how-can-you-spot-them

Tsai, W.-C., Shih, Y.-J., & Huang, N.-T. (2019). Hardware-accelerated, short-term processing voice and Nonvoice sound recognitions for Electric Equipment Control. Electronics, 8(9), 924. https://doi.org/10.3390/electronics8090924

Voice Conversion Challenge. (n.d.). Voice Conversion Challenge. Retrieved March 28, 2023, from http://www.vc-

challenge.org/#:~:text=Voice%20conversion%20(VC)%20refers%20to,the%20original%20speaker%20(source).

Wikipedia contributors (a). (2023, March 27). Generative adversarial network. In Wikipedia. Retrieved March 28, 2023, from https://en.wikipedia.org/wiki/Generative_adversarial_network

Wikipedia (b). (2022, December 19). Speech Synthesis. Retrieved March 28, 2023, from https://en.wikipedia.org/wiki/Speech_synthesis#:~:text=Text%2Dto%2Dspeech%20(TTS)%20refers%20to%20the%20ability,can%20be%20output%20as%20sound.

Wikipedia (c). (2023, March 20). Prosody (linguistics). Retrieved March 28, 2023, from https://en.wikipedia.org/wiki/Prosody_(linguistics)

Wikipedia (d). (n.d.). Deepfake. In Wikipedia. Retrieved March 28, 2023, from https://en.wikipedia.org/wiki/Deepfake

Yamagishi, J., Todisco, M., Sahidullah, M., Delgado, H., Wang, X., Evans, N., Kinnunen, T., Lee, K. A., Vestman, V., & Nautsch, A. (n.d.). Edinburgh Mouse Atlas Project. [Data set]. University of Edinburgh. The Centre for Speech Technology Research (CSTR). Retrieved March 28, 2023, from https://datashare.ed.ac.uk/handle/10283/3336

Yang, J., Das, R. K., & Li, H. (2018). Extended constant-Q cepstral coefficients for detection of spoofing attacks. 2018 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC). https://doi.org/10.23919/apsipa.2018.8659537

Zhang, Y., Jiang, F., & Duan, Z. (2021). One-class learning towards synthetic voice spoofing detection. IEEE Signal Processing Letters, 28, 937–941. https://doi.org/10.1109/lsp.2021.3076358

Zhou, X., Garcia-Romero, D., Duraiswami, R., Espy-Wilson, C., & Shamma, S. (2011). Linear versus Mel Frequency Cepstral coefficients for speaker recognition. 2011 IEEE Workshop on Automatic Speech Recognition & Understanding. https://doi.org/10.1109/asru.2011.6163888