Fall 2022

Speaker Encoding for Zero-Shot Speech Synthesis

Tristin W. Cory
Missouri State University, Tristin70@live.missouristate.edu

SPEAKER ENCODING FOR ZERO-SHOT SPEECH SYNTHESIS

A Master's Thesis

Presented to

The Graduate College of

Missouri State University

In Partial Fulfillment

Of the Requirements for the Degree

Master of Science, Computer Science

By

Tristin Cory

December 2022

SPEAKER ENCODING FOR ZERO-SHOT SPEECH SYNTHESIS

Computer Science

Missouri State University, December 2022

Master of Science

Tristin Cory

ABSTRACT

Spoken communication, for many, is an essential part of everyday life. Some individuals can lose or not be born with the ability to speak. To function on a day-to-day basis, these individuals find other ways of communication. Adaptive speech synthesis is one of those ways. It recreates a user's previous voice or creates a voice that blends with their regional dialect. Current adaptive speech synthesis techniques that achieve human-like speech require thirty minutes, to a few hours of high-quality audio recordings of a target speaker. This amount of recorded audio is not commonly possessed by people in need of a speech synthesis system. One adaptive speech synthesis technique that requires only ten to thirty seconds of data is called zero-shot. However, there are currently no zero-shot speech synthesis methods able to produce human-like speech or replicate a speaker with a high degree of similarity. In this thesis, I propose a novel speaker encoder model to make zero-shot speech synthesis more human-like. The proposed model results in a speaker embedding vector called Multi-Scale Speaker (MSS) vectors. MSS-vectors aim to improve current state-of-the-art speaker embeddings for more natural and similar-sounding synthesized speech for unseen speakers in a zero-shot speech synthesis model. The proposed architecture relies on encoder layers, which are coupled with a multi-scale approach to learning both local and global mel-spectra features of a reference speaker. To evaluate the proposed approach, I compare the MSS-vectors model against a modified generalized end-to-end (GE2E) speaker encoder, as well as the s-vector speaker encoder. My comparison includes quantitative measures, such as mel-cepstrum distortion and cosine similarity measures, as well as subject mean opinion scores from human listening surveys. The experimental results from these evaluations indicate improvements over current state-of-the-art speaker encoder models, and thus a shift towards more human-like speech.

KEYWORDS: speaker adaptation, speaker embedding, speaker encoder, text to speech, speech synthesis, zero-shot

SPEAKER ENCODING FOR ZERO-SHOT SPEECH SYNTHESIS

By

Tristin Cory

A Master's Thesis
Submitted to the Graduate College
Of Missouri State University
In Partial Fulfillment of the Requirements
For the Degree of Master of Science, Computer Science

December 2022

Approved:

Razib Iqbal, Ph.D., Thesis Committee Chair

Yassine Belkhouche, Ph.D., Committee Member

Lloyd Smith, Ph.D. Committee Member

Julie Masterson, Ph.D., Dean of the Graduate College

ACKNOWLEDGEMENTS

Firstly, I thank my wonderful wife for her never-ending support of my endeavors in all things. The inspiration for this research was found by her sharing her studies in communication sciences and disorders with me. I also have a great amount of gratitude for my parents and the confidence they have always had in me. In addition, I thank Dr. Iqbal for all the mentoring and feedback he has given me over the course of my graduate studies. Finally, I am very appreciative of the Missouri State University CODERS grant for the unique experiences and opportunities they have given me to help teach computer science curriculum to rural Missouri students and teachers.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ALGORITHMS

1 INTRODUCTION

1.1 Overview

Creating artificial speech can trace its origins back to 1780 with a machine that produced sustained speech sounds designed by Christian Gottlieb Kratzenstein [1]. Synthesizing full sentences came later, when Joseph Faber debuted a talking machine in 1845. Faber's machine, shown in Figure 1, was made up of a small organ connected via mechanical levers to a wooden head with a jaw, ivory tongue, and rubber glottis, which together were able to produce convincing speech.

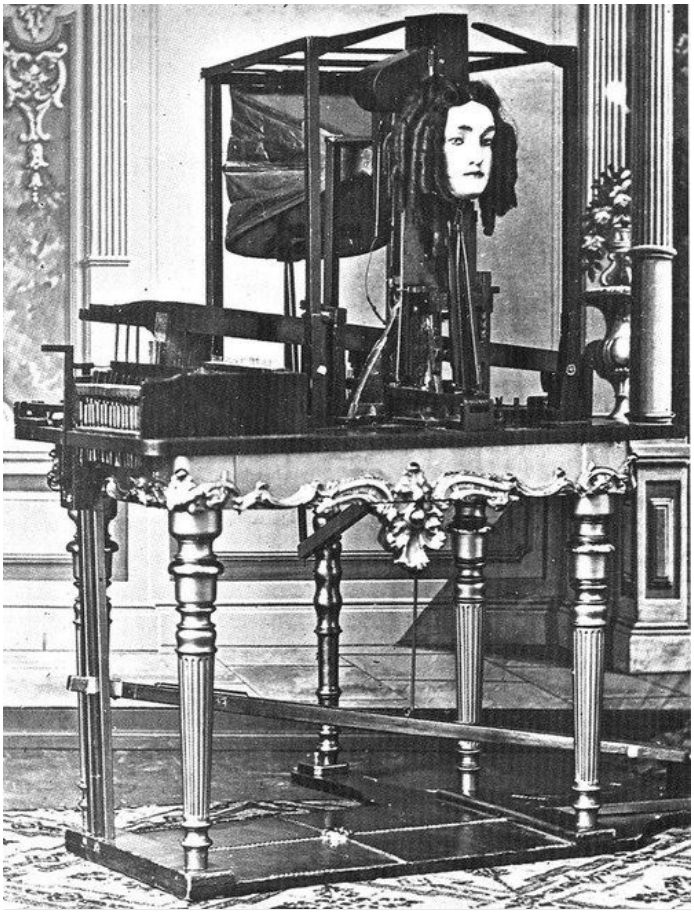

Figure 1. Joseph Faber's talking machine, Euphonia.

The first fully electronic speech synthesis system came in 1922 from John Q. Stewart [1]. Stewart's algorithm aimed to define a set of rules based on a set of variables related to speech production. The goal of specifying a set of rules for speech synthesis has been the focus of most speech synthesis algorithms since. This led to several types of speech synthesis systems, e.g. formant synthesis, unit selection, statistical parametric [1], all utilizing machine learning techniques, before eventually the use of deep learning networks for learning relationships between speaker features was adopted.

Today, speech synthesis systems are tools that can improve the quality of communication in various situations. Synthesized speech is utilized in user interfaces to provide users with opportunities to communicate in a more natural and intuitive manner. This type of human interaction with speech synthesis systems has increased immensely since the introduction of applications such as Amazon's Alexa, Apple's Siri and Google Assistant. Another scenario where speech synthesis is used is in the medical field. Individuals who are vocally handicapped via trauma, disease, or complications at birth, naturally find alternative ways to express themselves. This expression can take the form of vibrant facial expressions, pointing to objects with body parts, eye contact, or gestures. This naturally developed expressive communication is sometimes supplemented with Augmented and Alternative Communications (AAC). AAC can be implemented with techniques as low-tech as picture/symbol boards and sign language, or with high-tech methods such as voice output devices. The quality of the output voice in these devices can range from choppy and robotic-sounding speech to near-perfect imitations of human speech. The user interfaces of these devices can mimic the low-tech picture and symbols boards mentioned previously. Some devices take typed text as input and output synthesized speech of the text input. This type of system is known as a text-to-speech or TTS system. The voice output

by AAC devices aims to provide a user with a convenient and relatable method of communication.

Therefore, my focus in this thesis is the investigation of voice output component of AAC systems. Specifically, the improvements that can be made to speech synthesis systems to make them more accessible and natural sounding for use in AAC applications.

1.2 Research Motivation

Voice output AAC devices, as previously mentioned, use speech synthesis to create an artificial voice. A speech synthesis technique that is of interest for use in AAC devices is that of adapting the voice output to match or more closely resemble the end user of said device. Adapting voice output in speech synthesis would have direct applications in AACs. Primarily, creating a voice that matches the user's own prior voice or more closely resembles the local vernacular where the user lives. This type of speech synthesis is known as adaptive speech synthesis. The authors in [2] presented two primary ways of achieving adaptive speech synthesis. The first method is a general approach to improve the adaptability of the system as a whole, for synthesizing new speakers. This general approach most commonly involves increasing the diversity present when training the system. Increasing diversity when training the system would require several speakers that speak the same local dialect as the end user, with an hour or more of speech from each speaker. This data requirement can be expensive and time-consuming to gather. The second approach attempts to accomplish adaptivity more efficiently by reducing the amount of data needed to adapt to a new speaker. This less data intensive method for adaptive speech synthesis is often achieved by focusing on differentiating between speakers. When adapting to a new speaker with this method, a sample of 4 to 20 words is provided to the system.

This approach significantly reduces the amount of data required to adapt to a new user. However, this method results in artificial speech that is less natural and less similar to the target speaker than the general approach previously mentioned.

1.3 Research Contributions

My research contributions as a result of this thesis are as follows. I present a novel speaker encoder architecture called Multi-Scale Speaker Vectors (MSS-vectors). MSS-vectors uses a multi-scale concept for learning speaker features, which has not been utilized by any speaker encoders to date. The multi-scale concept is made up of two encoders. One of which learns features globally across an audio sample, and the other learns features from sub-ranges of the audio sample. Additionally, I compare the MSS-vectors model against the Generalized End-to-End (GE2E) and s-vector models on the following metrics: naturalness mean opinion scores, similarity mean opinion scores, speaker encoder cosine similarity, and mel-cepstral distortion. This research has led to the publication of two conference papers.

- T. Cory and R. Iqbal, "Multi-Scale Speaker Vectors for Zero-Shot Speech Synthesis," *IEEE 46th Annual Computers, Software, and Applications Conference (COMPSAC)*, 2022.
- T. Cory and R. Iqbal, "Comparison of Multi-Scale Speaker Vectors and S-Vectors for Zero-Shot Speech Synthesis," *IEEE 24th IEEE International Symposium on Multimedia (ISM)*, 2022.

1.4 Organization of Thesis

The rest of this thesis is organized as follows: in Chapter 2, a history is given on techniques in speech synthesis. Chapter 3 introduces adaptive speech synthesis and covers related speech synthesis works. Chapter 4 formally introduces speaker encoders and covers the

related speaker encoder works. Chapter 5 introduces the proposed novel MSS-vector speaker encoder. Implementation details for Chapter 5 and all other models and evaluations are contained in Chapter 6. Chapter 7 covers the setup of experiments, as well as the results of the experiments. Lastly, Chapter 8 contains the final statements of this research and potential directions for future work.

2 HISTORY OF SPEECH SYNTHESIS

In this chapter, I will give a brief history of speech synthesis, starting with early efforts and progressing into modern-day initiatives. It should be noted that this chapter will not exhaustively cover all approaches to speech synthesis. Rather, it will cover the major milestone methodologies that most approaches can be classified under. For each of these milestone techniques, I will give a simplified overview of how each technique creates artificial speech.

2.1 Early Initiatives in Speech Synthesis (approx. 1780 - 1950)

2.1.1 Articulatory Speech Synthesis. As mentioned previously in Chapter 1.1, synthetic speech can trace its roots back to about 1780 with Kratzenstein's entirely mechanical machine [1]. Kratzenstein's design only produced sustained vowel sounds. Full words came later in 1791 by a Hungarian engineer, von Kempelen [1]. Kempelen's device included a bellow to produce air pressure, reeds served as a voice source, and a rubber funnel acted as a vocal tract. To use the device, the user would depress the bellows while manipulating the rubber funnel to produce speech sounds. Inspired by Kempelen's written account of his experiments and work in speech production titled "On the Mechanism of Human Speech", Joseph Faber made his own speech device, which was detailed in Chapter 1.1. Faber's machine, named Euphonia, made significant design changes to Kempelen's device, namely in the manipulation of the artificial vocal tract which being connected to 16 organ keys [1]. All the previously mentioned mechanical talking machines are examples of a method of synthesizing synthetic speech called articulatory synthesis. Articulatory synthesis is a biological approach to producing synthetic speech, as it aims to replicate how humans biologically produce speech. This is done by mimicking the

human articulators, which include the tongue, glottis, lips, and vocal tract. Theoretically, articulatory synthesis should produce the most natural results available in speech synthesis. However, researchers early on discovered that modeling the biological way humans make speech with articulators was quite difficult [2].

2.1.2 Formant Speech Synthesis. Towards the end of the 1800s, it had become mostly accepted that a core principle of speech production was that resonances produced by a given air space highlighted certain components of a sound spectrum, later known as formants [1]. During this time, there was also a change in speech devices, from using a mechanical or electro-mechanical means of operation to being primarily electrically based. In 1922, John Q. Stewart presented the first entirely electrical circuit-based speech synthesizer in his paper "An Electrical Analogue of the Vocal Organs" [1]. Stewart's circuit used a buzzer and two parallel resonant branches with variable resistors. This circuit was not necessarily an electrical analogue of the human speech production system, as the paper states, but rather an analogue of the resonant frequencies that human speech produces, and thus was the first electrical, formant speech synthesizer. Stewart also observed that the production of speech sounds was not of great difficulty, but instead the problem was the manipulation of those sounds to result in natural and intelligible speech. This observation led many researchers, including Stewart, to focus on defining rules for synthesis systems that guided how sounds were manipulated and transitioned between, this later was called synthesis-by-rule. As per [3], when synthesis-by-rule is performed, the rules which govern the synthesis process can be viewed as a theory of which speech cues are most important in human speech. For example, Stewart's theory was based on formants and their importance in speech production, and as such his system focused almost entirely on the creation and manipulation of specific formants.

Early articulatory synthesis had success due to the nature of replicating human articulators. Thus, most potential cues of human speech were present simply as a byproduct of modeling those articulators. In contrast, when synthesizing speech by rules, not all cues are present in the resulting speech. In the early 1900s, a large portion of research was directed at synthesis-by-rule and determining which vocal cues were most vital to our perception of human speech [2].

2.2 Modern Initiatives in Speech Synthesis (approx. 1953 - Current)

2.2.1 Text-to-Speech. Naturally, the history of TTS directly coincides with the progression of speech synthesis. Although the reverse is not true, as not all speech synthesis systems require text or linguistic features to synthesize speech. For example, Kratzenstein's mechanical talking machine was played like an instrument and was an example of articulatory synthesis, since it attempted to replicate how humans produce speech with articulators [1]. Many early speech synthesis devices like Kemepelen's and Faber's were similar to Kratzenstein's machine, in that they required a method of human interaction, such as moving keys, bars, or buttons [2]. In this thesis, I define TTS as a system that takes input text and performs operations on said text, resulting in an audible waveform. The first time this definition of TTS was accomplished was in 1968 by Teranishi and Umeda [2]. Teranishi and Umeda's system used a computer-based articulatory model of speech synthesis and a parser for text analysis and determining pauses in speech (pause assignments). Text analysis was performed based on a set of predetermined rules. This set of rules is similar to the synthesis-by-rule concept discussed in Chapter 2.1.2.

As per [3], in current-day TTS systems, three primary components are responsible for

their functionality - a text analysis module, an acoustic module, and a vocoder, as shown in Figure 2. The text analysis module is responsible for taking text and producing linguistic features for the provided text. These linguistic features can be as basic as characters or phonemes, to more complex features such as prosody prediction or part-of-speech tagging. The acoustic module is provided with the linguistic features from the text analysis module. The acoustic module then commonly produces an intermediate representation, such as a mel-spectrogram. The intermediate representation is finally fed to the vocoder, which converts the intermediate representation into an audio waveform. With these 3 components, the influence of past areas of speech research can be seen. As mentioned in the previous paragraph, Teranishi and Umeda implemented the rule-based text parser into a speech synthesis system in 1968. The acoustic model is similar to the acoustic parameter model seen in the upcoming Chapter 2.2.3, and the formant synthesis model discussed in Chapter 2.1.2. The vocoder is the inverse to early attempts to visualize sound waves, such as Koenig and Scott's phonoautograph which received sound via a cone and diaphragm and etched a waveform onto paper rotating on a cylinder [1].

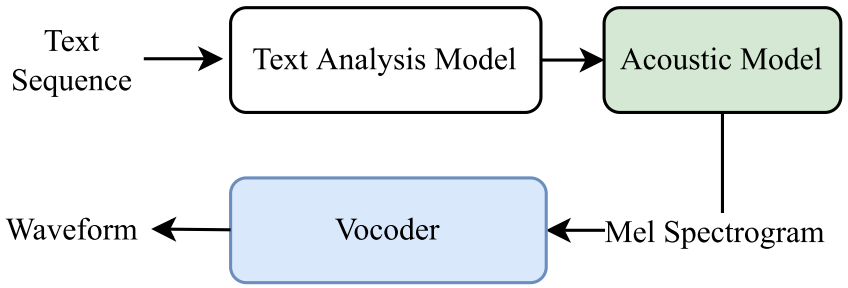

Figure 2. Generic single speaker TTS model.

2.2.2 Concatenative Speech Synthesis. In 1953, Harris introduced another type of synthesis-by-rule technique, later referred to as concatenative or unit selection synthesis [1]. In this theory, the idea is to take recorded pieces of natural speech, and as the name implies,

concatenate them together to form words. Initial concatenative synthesis models, such as Harris's, utilized phonemes, which are the smallest unit of sound in a language. However, these early attempts sounded unnatural due to the effects that a phoneme has on the sound of its adjacent phonemes. To address this problem, the diphone was introduced. The diphone is made up of the sound from the middle of one phoneme to the middle of another phoneme [2]. Since there are 40 phonemes in English, that results in 1,600 possible diphones. However, each diphone can have several variations depending on whether the syllable the diphone is part of is stressed or unstressed. The first diphone unit selection system came from Dixon and Maxey in 1968 [4]. In Dixon and Maxey's system, a limited inventory of diphones was utilized to achieve this, and the effort took many years of trial-and-error to optimize the provided diphone inventory. While the inventory of diphones can be limited, it is estimated that 8000 diphones are necessary for optimal performance [5].

2.2.3 Statistical Parametric Speech Synthesis. The next approach was first introduced in 2009 by Zen, Tokuda, and Black, who presented a technique called statistical parametric speech synthesis (SPSS). In SPSS, the goal is nearly the opposite of unit selection. In unit selection, the focus is to concatenate small, unmodified units of recorded speech. Whereas with SPSS, the goal is to produce an average of similarly sounding speech segments. This average is produced by extracting parametric representations from a speech database, and then modeling these representations with a generative model, such as a Hidden Markov Model (HMM). This generative model can be used to find the most probable set of acoustic parameters for a given sequence of words or linguistic representation. The acoustic parameters are then used to construct a speech waveform for playback [6]. The use of an average model gives SPSS some notable advantages over unit selection. Perhaps the most important one is adaptability. With

SPSS, by transforming the acoustic parameters obtained from the generative model, the output speech can be manipulated to easily adapt to another person's voice, and the interpolation of voices to produce new speaking styles and emotions not represented in the original inventory of speech.

2.2.4 Neural Speech Synthesis. More recently in the 2010s, researchers have been using neural networks, an effective machine learning technique, to enhance SPSS or concatenative techniques to produce synthetic speech [7], [8]. Neural speech synthesis models typically take one of two approaches [3]. The first approach that some neural network-based systems take is to model the desired text directly to a waveform and is sometimes referred to as end-to-end TTS. While there are many variations of the end-to-end TTS approach, potentially the two most common architectures are a combination of all 3 TTS components (text analysis module, acoustic module, and vocoder) into a single model or a two-part model, made up of a text analysis module paired with a combined acoustic and vocoder model. The second approach follows the same methodology as SPSS by modeling the acoustical parameters of speech with a neural network. This approach makes use of all 3 TTS components discussed in Chapter 3. The benefit of neural speech synthesis, in general, over SPSS, is the reduced amount of human data processing required for extracting acoustic parameters, as well as increases in intelligibility and naturalness.

In this chapter, I have given a general overview of articulatory, formant, concatenative, parametric, and neural speech synthesis techniques. These speech synthesis techniques, as previously mentioned, are not exhaustive, and there are more techniques and methods in which artificial speech can be produced. However, most speech synthesis techniques can be classified as one of those five types. In the following chapter, I will cover specific type of speech synthesis

model called adaptive TTS.

3 ADAPTIVE SPEECH SYNTHESIS

With the introduction of SPSS, a specific feature of some models allow the addition of a new speaker's voice without a large database of speaker recordings typically needed to add another speaker. Models with this feature are called adaptive TTS or adaptive speech synthesis models [2]. Adaptive TTS is also sometimes referred to as voice cloning or voice adaptation. In this chapter, starting with the TTS system shown in Figure 2, I will build up to and detail how adaptive TTS is often accomplished, as well as discuss adaptive speech synthesis research works related to my thesis.

A TTS system made up of just the three components discussed in Chapter 2.2.1 is representative of a single-speaker model, which produces only a single synthetic voice, and is commonly trained with data from only a single speaker. For a TTS model to produce multiple synthetic voices, another component is typically used in training the TTS model and is called a speaker encoder. The speaker encoder component produces a unique embedding vector for a given speaker utterance. The speaker encoder is usually trained separately from the rest of the TTS model as a speaker recognition/classification task. Multi-speaker TTS models often use a speaker id which is associated with a large array of speaker embeddings derived in training for a given speaker. However, if a new voice is needed, then the training process must be continued or restarted from the beginning, both of which can be time-consuming. To solve this problem, a speaker encoder is added to the TTS model and is used by the TTS model to differentiate between speakers by directly generating a unique speaker embedding. The acoustic model uses the unique speaker embedding to tune the intermediate representation to different speakers.

Where the speaker encoder fits into the previously discussed three-component TTS model can be seen in Figure 3.

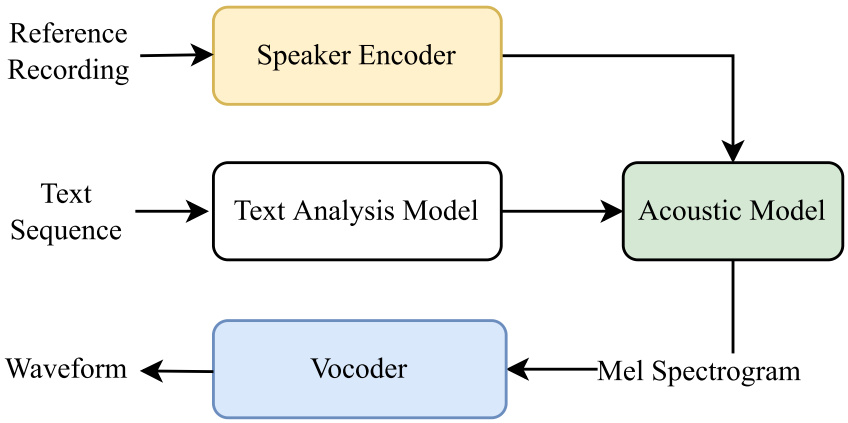

Figure 3. Generic zero-shot TTS model.

Adding a new, unseen speaker at synthesis time often results in intelligible results, but not as high quality of results achieved with speakers seen in training. The authors in [3] presented two primary ways to improve the adaptive feature in a TTS model. The first method is a general approach to improve the adaptability of the TTS model, as a whole, for synthesizing new speakers. However, the general approach most commonly involves either modeling the acoustic conditions of the speakers (such as prosody or speaker timbre) or increasing the diversity present in the training data. An implication for both methods is a high data collection cost, which renders them unusable if the amount of new speaker data is as low as a few sentences. The second approach attempts to accomplish speaker adaptivity more efficiently by reducing the amount of data needed to adapt to a new speaker. An efficient method to adaptive TTS is often achieved with either one/few-shot adaptation or zero-shot adaptation. Few-shot adaptation consists of fine-tuning the TTS model on a few audio and text pairs, typically less than 20 sentences (one-shot uses just one audio and text pair). Zero-shot adaptation only uses the

speaker encoder, which provides the TTS model with a speaker embedding computed from reference audio, which contains 4 to 20 words from that user, and does not update or further train the model. While the data required for one-shot and zero-shot is essentially the same, the amount of data needed is less than few-shot. The benefits of a zero-shot TTS system over one-shot models are primarily in the hardware need. For example, if a user of a TTS system wanted to add their own or someone else's voice to the system, a zero-shot model would allow the user to do so without access to a graphics processing unit capable of performing additional training. The less intensive hardware requirements of zero-shot expand the types of devices that can be used for adaptive TTS. As a result, this could reduce the financial impact on users if a user already owns a device that can run an adaptive TTS model. Zero-shot synthesis models also allow the customization of speech from user interface devices such as Amazon Alexa or Google Home, without offloading training to the cloud and instead, handling the change at the edge device. However, the downside of zero-shot TTS is the difference in quality between it and one/few-shot adaptation when judged on the similarity to the target speaker and overall naturalness. While one/few-shot adaptation produces more similar and natural-sounding speech, the quality is still not to the level of non-adaptive models.

The paradigm for zero-shot learning was first introduced by two separate papers in 2008 [9], [10]. In [9], the authors introduced the paradigm as dateless classification and applied the technique to a semantic classification task. In [10], the authors called the technique zero-data learning and utilized it in both a character recognition task and a multi-task problem, observing molecular compound activity in the process of drug discovery. Zero-shot learning can be seen as a next step from one-shot or few-shot learning. Zero-shot learning aims to learn as much information as possible from labeled training data, and transfer that information to the task of

classifying unlabeled and unseen classes in testing. Thus, zero-shot learning can be viewed as a type of transfer learning. By comparison, in one-shot or few-shot learning, the models can be provided a single or a few labeled instances of a class in testing to update the model with that new class [11]. The key difference between the zero-shot technique and one/few-shot learning is that in zero-shot, the training and testing instances are disjoint. In other words, a zero-shot model is not updated when it classifies an unseen class in a testing instance. I chose to pursue the use of a zero-shot TTS system due to the benefits implied by that key difference between zero-shot learning and few-shot learning.

As discussed in Chapter 2.2.4, not all TTS systems are based on the 3 components - text analysis model, acoustic model, and vocoder. Some systems take an end-to-end approach, including when expanding a system to include multi-speaker or zero-shot capabilities. Therefore, it can be more difficult to evaluate the influence of architectural changes when comparing the performance of these systems, because of the direct effect a change can have on the entire model. So, I will limit my related works and potential approach to one which uses a traditional 3 component architecture. One of the notable and benchmark models of a 3 component architecture is the Tacotron 2 model [12]. Tacotron 2 is a fully neural speech synthesis architecture based on the sequence-to-sequence Tacotron model [13] and a modified WaveNet vocoder [14]. The Tacotron 2 model, shown in Figure 4, starts with input text going to a learned character embedding model, which is the text analysis component. The acoustic component is a mel-spectrogram prediction network made up of an encoder and decoder. The encoder takes the character embedding produced by the text analysis component and passes it through 3 convolutional layers, a bidirectional long-short term memory (LSTM) network, and outputs a feature vector. The feature vector is given to an attention network, which uses attention weights

given by previous time steps from the decoder. The decoder predicts one mel-spectrogram for every time step. The mel-spectrogram from the previous time step is given to a pre-net for bottlenecking information for attention learning. The output of the pre-net and attention network is given to a 2 layer LSTM network. The output concatenation of the pre-net and attention vectors from the LSTM are passed through a linear projection to predict a mel-spectrogram. The predicted mel-specprogram is given to a post-net, which predicts a residual, and is then added to the original mel-spectrogram prediction. Parallel to the linear projection for prediction of the mel-spectrogram, another linear projection produces a scalar value on which a sigmoid function is used to determine when to stop generation of new mel-spectrograms. The final mel-spectrogram is passed to the modified WaveNet vocoder. The vocoder mostly follows the architecture shown in [15]. The modifications are made in two areas. The first is in the conditioning stack, to accommodate a frame hop of 12.5ms, only 2 upsampling layers are used. Second is that, rather than utilizing a softmax layer for prediction, a 10 component mixture of logistic distributions (MoL) is used.

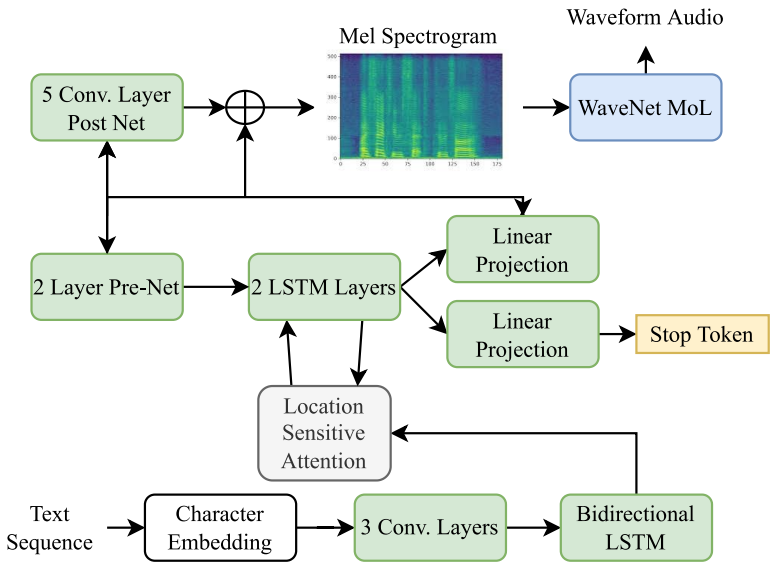

Figure 4. Tacotron 2 architecture.

The original Tacotron 2 model was later modified to include multi-speaker and zero-shot synthesis capabilities [16]. These modifications follow the same approach for modifying a single speaker TTS system into a multi-speaker TTS model discussed previously in this chapter, and the modified architecture is shown in Figure 5. The authors added a LSTM based speaker encoder model to provide speaker embeddings. The speaker embedding is concatenated to the synthesis encoder's output feature vector before being given to the attention network.

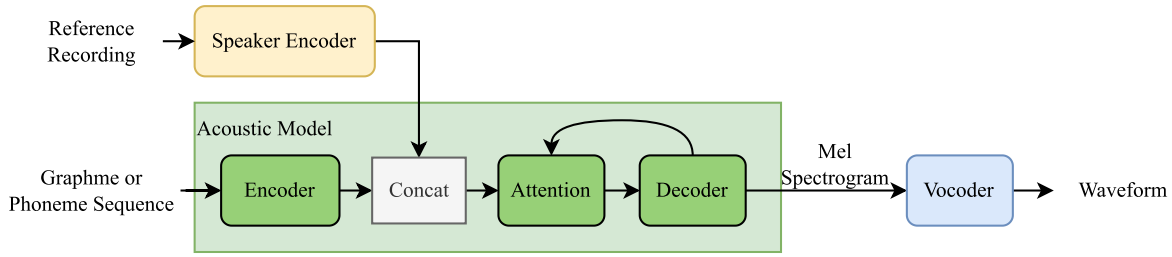

Figure 5. Zero-shot Tacotron 2 architecture.

The authors of [17], presented a model named Speaker Conditional GlowTTS (SC-GlowTTS), the authors focused on zero-shot TTS performance and made improvements to the synthesis model. Their contributions came in the form of modifications to the GlowTTS synthesis model [18], namely the addition of an external speaker encoder, which allows zero-shot TTS synthesis, and testing different encoder networks in place of the original GlowTTS network's encoder. The SC-GlowTTS model outperformed Tacotron 2 [12] and Attentron [19] in mean opinion scores in both similarity and naturalness for unseen speakers. The SC-GlowTTS model, as shown in Figure 6, starts with a phonemizer to convert the input text into a linguistic specification. The acoustic component is similar to the Tacotron 2 model, in that it contains encoder and decoder networks. The SC-GlowTTS model tested three encoder architectures, including a residual dilated convolutional network, a gated convolutional network and the

transformer network from the original GlowTTS model. I will describe the transformer variation only, because it performed the best of the 3 variations and is most relevant to my thesis work. GlowTTS used the encoder from the TransformerTTS [20], in which it was added to the Tacotron 2 architecture. The encoder is built from two encoder blocks. Each block consists of a multi-headed attention network and a feed forward network. The encoder's output is given to a convolutional projection and a duration predictor. The duration predictor is composed of two convolutional layers, with ReLU, normalization, dropout and a projection layer. The convolutional projection is given to a flow-based decoder. The decoder is made up of several blocks, each containing a normalization layer, 1x1 convolution layer that's invertible, and an affine coupling layer.

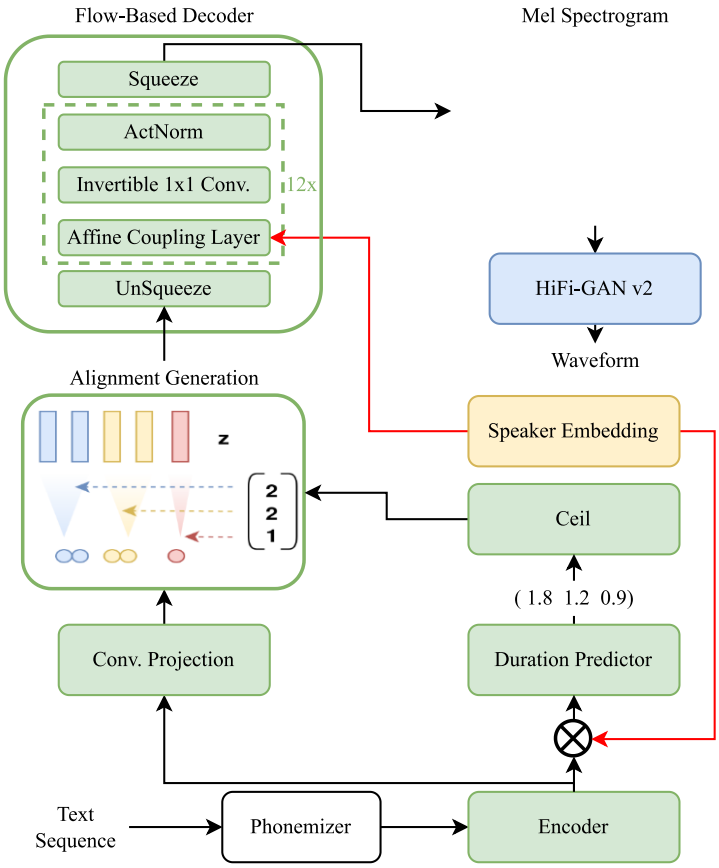

Figure 6. SC-GlowTTS architecture.

The SC-GlowTTS model made GlowTTS zero-shot capable by concatenating speaker embeddings to the encoder's output to the duration predictor, and in the affine coupling layers of the decoder. The speaker encoder used in the SC-GlowTTS model is nearly identical to the LSTM based speaker encoder used in the Tacotron 2 model. The vocoder utilized was the HiFi-GAN v2 model [21], which is a generative adversarial network (GAN) based model. HiFi-GAN was designed from the MelGAN vocoder [22]. The HiFi-GAN model will be detailed in more depth later in my proposed approach.

In this chapter, I have described an architecture commonly used for zero-shot speech synthesis. I then detailed what zero-shot learning is and highlighted the differences between it and one/few-shot learning. Lastly, I covered the notable TTS models which are related to my thesis research and follow the 3 component architecture introduced in Chapter 2.2.1. These TTS models include the original Tacotron 2 mode and its zero-shot variation. As well as the SC-GlowTTS model which is a variant of the GlowTTS model.

4 SPEAKER ENCODERS

In the previous chapter, it was stated that speaker encoders produce a unique embedding vector for a given speaker. This statement is correct, but the use case for speaker encoders was not originally for adaptive speech synthesis, it was for speaker recognition. As per [23], a speaker recognition system has three basic sections: pre-processing, feature extraction, and speaker modeling. The pre-processing section cleans and normalizes the speech signal before feature extraction. Feature extraction is the name given to the speaker encoder component in speaker recognition systems. This section functions the same as in a TTS system, it outputs a unique vector describing the characteristics of the given speech signal. Finally, speaker modeling is where feature matching algorithms are run to determine recognition. In many speaker recognition systems, the line between these 3 sections is blurred. However, the speaker encoder/feature extraction component tends to make up most of a model, as you will see in the following figures.

The goal of speaker recognition is to verify whether a given utterance is spoken by a specific speaker [24]. In [24], the authors presented a survey on speaker recognition categorizes models into two types, stage-wise and end-to-end. A commonality between both types of recognition systems is speaker feature extraction and a similarity calculation. In stage-wise speaker verification systems, feature extraction takes place in the frontend and the similarity calculation is taken care of by a backend. Whereas in end-to-end speaker verification systems, a similarity score is generated directly from two input utterances. Stage-wise speaker verification systems predate end-to-end systems, with one of the first notable proposals [25], presenting the idea of reducing a high dimension vector of speaker features to a lower dimension through factor

analysis, resulting in i-vectors. With the introduction of deep neural networks (DNN), came a type of speaker feature extraction that resulted in what are called deep embeddings [24]. Deep embeddings are divided into two main types - d-vectors and x-vectors. D-vectors (also known as frame-level embeddings) are generated by a DNN, which has been trained as a classification problem over the time frames of a training utterance. X-vectors (also known as segment or utterance level embeddings) produce an utterance level embedding by concatenating the standard deviation and mean from each frame in an utterance. Another embedding technique that is often used and modifies the d-vector/frame-level embeddings is learnable dictionary embeddings (LDE), which applies a clustering algorithm on the frame level embeddings.

The terminology used to describe different types of speaker encoders is often used loosely, with the most common confusion I have found is with the difference between deep embeddings and end-to-end speaker verification systems. This difference is clarified by the authors in [24], where the main differences are in the objective functions. Deep embeddings use a classification-based objective function (e.g. softmax), while end-to-end systems use a verification-based objective function, which keeps the problem an open-set problem, rather than a closed-set one like softmax does.

The GE2E model is an example of a deep embedding technique, which results in an utterance level embedding. The original GE2E architecture, as shown in Figure 7, consists of 3 LSTM layers followed by a linear layer. After the linear layer, a similarity matrix is built that defines the similarity between each embedding vector and the centroid of other vectors. Then the loss function is applied. In the GE2E paper, two different loss functions are utilized, a softmax and contrast loss function. The softmax loss function had the effect of pushing each embedding vector closer to its centroid and away from other centroids. Whereas the contrast lost function

pushes the embedding towards the centroid of the target speaker and away from the most similar but different speaker.

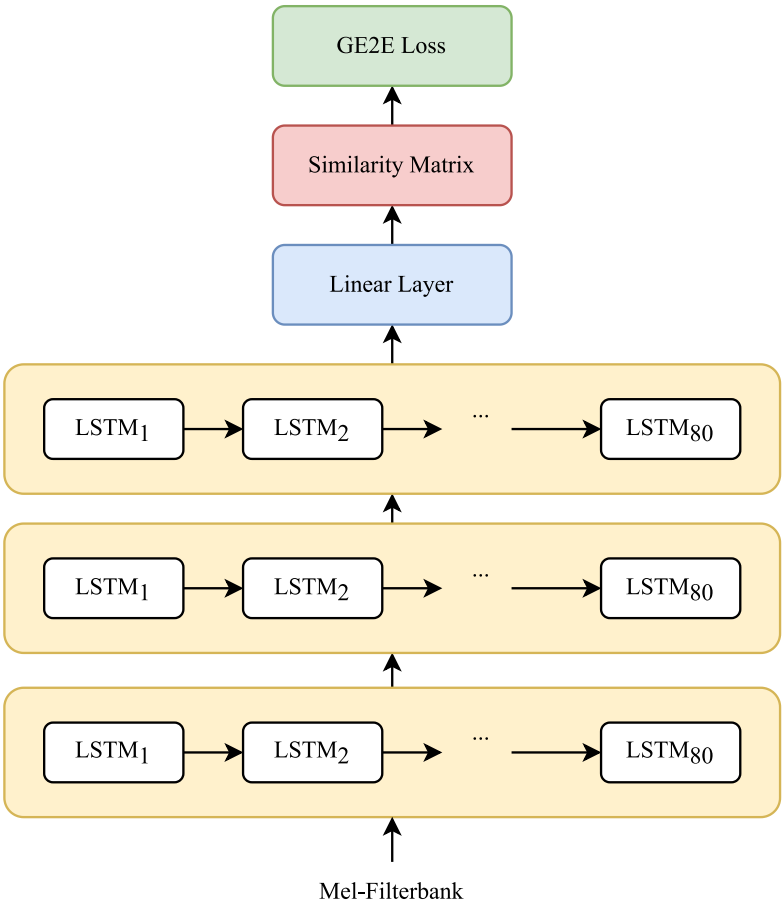

Figure 7. GE2E speaker encoder architecture.

The x-vector model is an example of a deep embedding technique, which produces an utterance-level embedding. The original x-vector model took an input of a 24-dimensional mel-filterbank and input the filterbank through 5 time delayed neural networks (TDNN), as shown in Figure 8 [26]. A statistics pooling layer followed the TDNN layers, where the means and standard deviation of the final TDNN layer are concatenated into one vector. The concatenated statistics vector is provided to a stack of two feed-forward neural networks (FNN). The FNNs are

followed by a linear projection layer. Lastly, the cross-entropy softmax loss function is applied for speaker modeling.

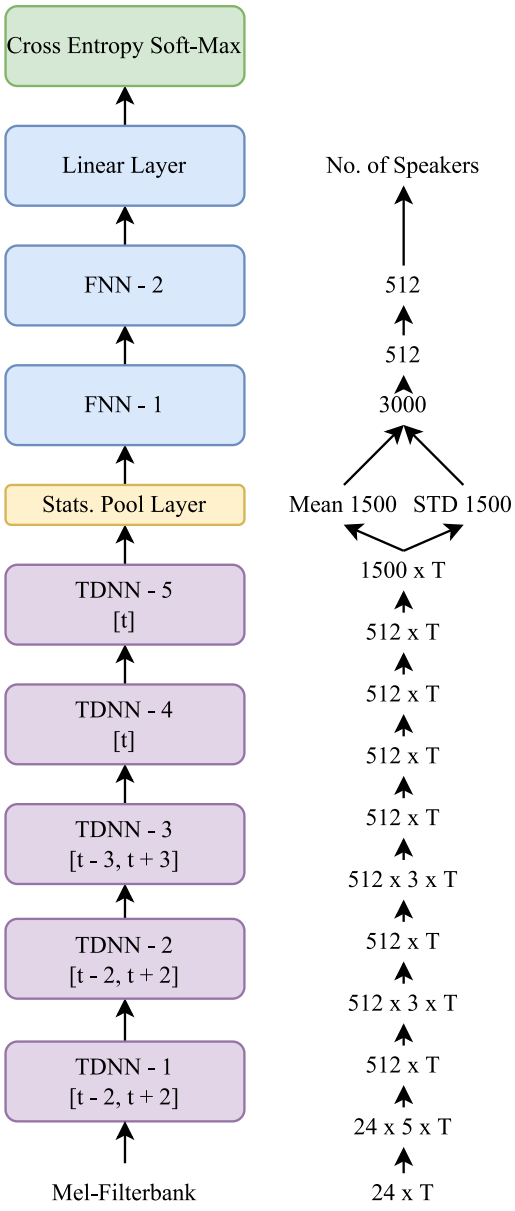

Figure 8. X-vector speaker encoder architecture.

LSTM and TDNNs model seen in the GE2E and x-vector models have been proven to be capable of learning features from large amounts of data. One downside to these types of models

is that they can struggle with recalling features seen very little or early in training [27]. One speaker encoder that addressed this issue was the s-vector model [28] shown in Figure 9.

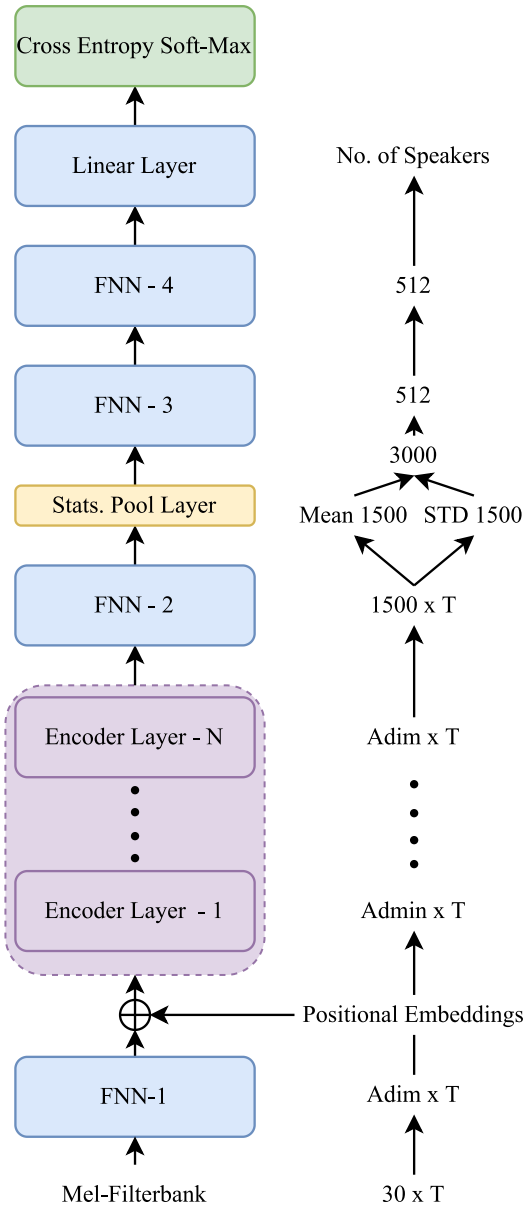

Figure 9. S-vector speaker encoder architecture.

The s-vector model took a similar approach to improving the x-vector model, as the TransformerTTS model did with the Tacotron 2. S-vectors start with a FNN layer, but replace

the TDNN 1-5 layers with encoder layers from the transformer architecture. Then a statistics pooling layer follows another FNN layer, two FNNs, a linear layer, and loss function, just as in the x-vector model.

Unfortunately, there are not many studies which investigate the effect that different speaker encoders have on either zero-shot TTS or TTS systems. However, one study that investigated various speaker embeddings and their impact on a zero-shot TTS system, similar to my thesis work, utilized a zero-shot multi-speaker Tacotron 2 network [29]. And in [29], the authors investigated x-vectors and LDE, and found that LDEs produced a more natural-sounding voice and higher similarity to the unseen speaker than what was produced with x-vectors. However, both the Tacotron 2 network and LDE embeddings are outperformed by more current state-of-the-art TTS systems and speaker encoders [29] [28].

5 PROPOSED APPROACH

As discussed in Chapter 4, current speaker encoders, such as [30], [28], used in conjunction with zero-shot TTS systems, come from the field of speaker recognition [29], [31]. In speaker recognition, the speaker encoder's job is to provide embedding data that uniquely identifies a speaker. The speaker encoder's ability to identify unique vocal cues for a given speaker allows it to perform well in a classification task like speaker recognition. However, in adaptive speech synthesis, the speaker embedding is used as an input to recreate a reference speaker's unique vocal spectrum, not make a classification. What the speaker embedding is being used for in each use case is fundamentally different. Therefore, it would be reasonable to suggest that what is optimal in speaker recognition may not be optimal for adaptive speech synthesis. As such, my proposed approach to improving adaptive speech synthesis aims to develop a speaker embedding that contains features that are more productive in recreating unique vocal cues than identifying vocal cues.

In designing a speaker encoder model to derive an embedding vector specifically for adaptive speech synthesis, I looked at the deep learning techniques used by the speaker encoders in Chapter 4, which revealed the use of LSTM and recurrent neural networks as the basis for many of these models. However, due to the issues mentioned with these types of networks, they could present problems when synthesizing a new unseen speaker, possibly degrading the similarity between the synthesized speech and the target speaker. For example, synthesizing an unseen speaker with an accent with few training samples. One way this general problem was previously addressed was with the introduction of transformers and self-attention. As previously mentioned in Chapter 4, the s-vector speaker encoder was the first speaker encoder to utilize self-

attention and a portion of the transformer architecture. As a result, I chose to use the s-vector speaker encoder as a basis for my proposed approach to help mitigate the loss of underrepresented speech features. While the benefits of self-attention help the s-vector model outperform other state-of-the-art speaker encoders, it still poses the same problems in speech synthesis as other speaker encoders due to its training as a speaker classification problem. Since there is not yet an ideal way to mathematically model how natural or similar to a reference speaker a given audio waveform sounds to humans, training a speaker encoder by using a synthesis model to generate audio waves and adjusting the speaker encoder accordingly is not possible. I had to look elsewhere for ways to improve the s-vector model for speech synthesis.

Operating under the assumption that training speaker encoders as a classification problem focuses the speaker encoder on outputting the most unique vocal features of a given speaker. Viewing this in light of speech synthesis, it would be a desirable behavior to recreate a specific speaker. However, I chose to reduce it in order to help capture longer-term vocal cues that might be beneficial for producing more natural or similar sounding speech, and missed if too much focus is placed on the smaller nuanced vocal cues. To do this, I introduced a multi-scale concept to focus speaker feature learning equally between separate frequency ranges divided across the whole of the target speakers reference audio and the reference audio as a whole. This multi-scale approach is similar to the multi-scale discriminators used in the MelGAN vocoder [22]. To use this multi-scale concept with the s-vector model, I replaced the single encoder with two nearly identical encoder networks. The first of which will be referred to as the local scale encoder (LSE). The purpose of the LSE is to learn speaker characteristics that are local to a subsection of the speaker's total frequency range characteristics, such as phoneme pronunciation. The second encoder network is designated the global scale encoder (GSE). The goal of the GSE is to capture

the general characteristics of the speaker as a whole, such as the average fundamental frequency and prosody. How the input mel-spectrogram is evaluated by the LSE and GSE is shown in Figure 10.

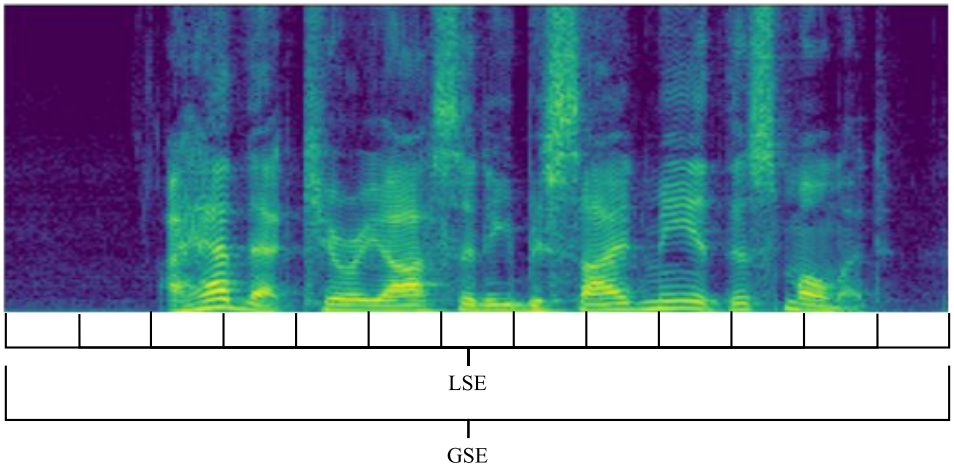

Figure 10. LSE and GSE evaluation of input mel-spectrogram.

The LSE is made up of two identical layers. An encoder layer begins with the batch normalization of the input, before being fed to a self-attention network. The self-attention network is used to learn what is important in the input sequence. In the self-attention network, the input is divided into query (Q), key (K), and value (V) matrices for each head. The LSE's self-attention network uses a multi-headed self-attention network with 8 heads to learn localized speaker features from different mel-spectrum ranges with no overlaps. The features expected to be learned by the LSE are features like unique differences in syllable or phoneme pronunciations. A scaled dot product is calculated from all the Q, K, and V matrices respectively, before the three matrices are concatenated. Lastly, batch normalization is performed on the output of the self-attention network before being given to a final feed-forward neural network (FNN) layer shown in Figure 11.

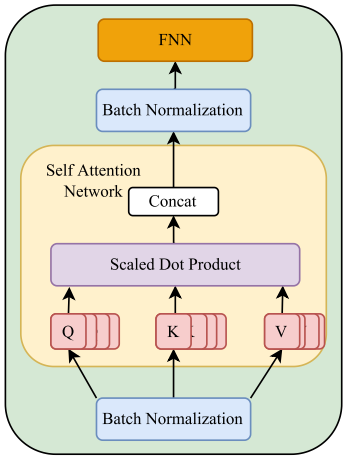

Figure 11. LSE encoder layer.

The GSE is also made up of two identical layers. Similar to the LSE, the GSE also starts with batch normalization of the input, followed by a self-attention network. However, the GSE self-attention network uses single-headed attention, shown in Figure 12, so that it can extract more generalized characteristics of the speaker by taking in the input as a single piece of data. Ideally, this should result in the GSE learning speaker features, such as the prosody of the speaker, vocal timbre, and stress patterns. Just as in the LSE, scaled dot products are calculated from the Q, K, and V matrices, which are then concatenated before going through batch normalization and a final FNN layer.

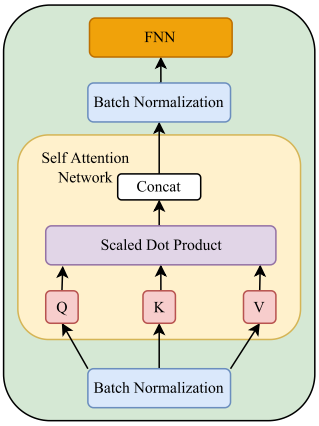

Figure 12. GSE encoder layer.

With the LSE and GSE discussed, I will now cover the proposed model as a whole. The MSS-vector model takes in 80 mel-scaled filter banks, as opposed to the 30 mel-frequency cepstral coefficients (MFCCs) the s-vector model uses as input. This was done because filter banks are more highly correlated than MFCCs. The filter banks are transformed into attention vectors by the first FNN, i.e. FNN1 in Figure 13. The attention vectors have positional embeddings added and are given as input to the previously discussed LSE and GSE. The resulting vectors from the second layer of both the LSE and GSE are given to separate FNNs (i.e., FNN2), which outputs a 1500×T vector. Statistics pooling is performed as per [32], resulting in a 1×3000 vector made up of 1500 means and 1500 standard deviations. The output of the statistic pooling layers from both the LSE and GSE is concatenated, before being fed through two more FNNs (FNN3 and FNN4) to help stabilize the output 1×256 vector. The algorithm for this model can be seen in Algorithm 1.

Algorithm 1. MSS speaker encoder

```
Input: Mel-Spectrogram
Output: Speaker Vector
for each input:
    pass to FNN1 to transform into attention vectors
    for each LSE and GSE:
        for each Encoder Layer:
            batch normalize attention vectors
            pass to self-attention network
            batch normalize self-attention network output
            pass to FNN
        end
        pass output of encoder layers to FNN2 to reshape
        perform statistics pooling on reshaped vector
    end
    concatenate results from both encoders into single vector
    pass to FNN3 and FNN4 to stabilize
    pass to FNN5 to reshape.
end
```

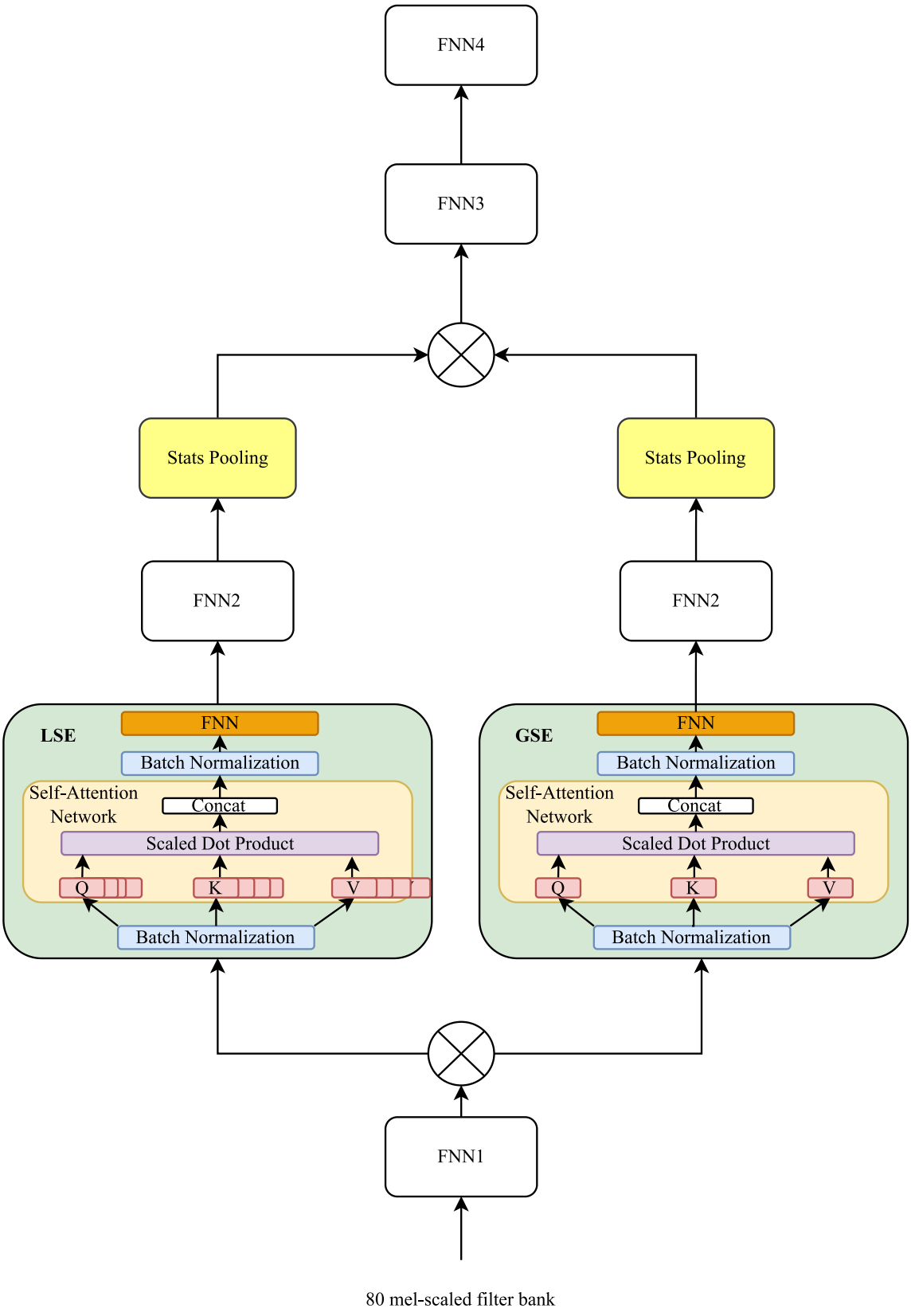

Figure 13. Proposed MSS-vector speaker encoder architecture.

6 IMPLEMENTATION

To perform evaluations to measure the effectiveness of proposed MSS-vector speaker encoder in adaptive speech synthesis, several trained models are needed. In this chapter, I will first discuss the training details of all the models needed for evaluation, and then provide information on the datasets used to train said models, before giving specifics about the evaluation metrics used in the experiments performed. To implement all the models discussed in this chapter I used Coqui TTS [26]. Coqui TTS is a framework for implementing and training TTS systems. Using Coqui TTS allowed the use of pre-implemented models. However, the only already implemented models available on the framework were the Hifi-GAN v2, GE2E, and SC-GlowTTS models. I made modifications to the GE2E model and implemented both S-vectors and MSS-vectors from the ground up. This study was approved by the Institutional Review Board on 12/20/2021 and received Approval #IRB-FY2022-384 (See Appendix).

6.1 Models

6.1.1 Speaker Encoder. For comparison against the proposed speaker encoder model, two other benchmark speaker encoders were chosen. The s-vector speaker encoder, on which the proposed model is based, and a GE2E speaker encoder.

The GE2E speaker encoder I used in my experiments follows the architecture of the GE2E model described in Chapter 4 [19]. However, a few modifications to the model were made to improve performance. Rather than the generalized end-to-end loss function, I utilized an angular prototypical (AP) loss function. The AP loss function has been proven to provide good performance and outperform the GE2E loss function in speaker recognition tasks, as per the

authors in [33]. Also, I chose to utilize the training parameters used by the authors of SC-GlowTTS [28] in their experiments, e.g. an input of 80 mel coefficients, 3 LSTM layers, 768 hidden LSTM units and a projection size of 256. The adapted model was trained for 30K steps using 64 speakers per batch and 3 utterances per speaker.

The s-vectors model, shown in Figure 9, used in the experiments is also described in detail in Chapter 4. The model takes an input of 30 Mel-frequency cepstral coefficients (MFCCs). The input MFCCs are fed through an FNN. Positional embeddings are added to the output of the FNN before going to the encoder layers. Multi-headed attention is performed in every encoder layer. The final encoding layer gives its output to another FNN, i.e., FNN-2, which outputs a 1500xT vector. Then stats pooling produces a 3000x1 vector (1500 standard deviation and 1500 mean). After which, the vector is reduced to a size of 512 twice by two more FNNs, i.e., FNN-3 and FNN-4. The original s-vector model has a final soft-max layer for speaker recognition. For experiments in zero-shot TTS, the final soft-max layer is not required, since there is no classification being done.

Four speaker encoder models were trained, the s-vector and modified GE2E models as detailed above. The other two speaker encoders are variations of the MSS-vector model. The reason for two MSS-vector models is due to the model differences in the GE2E and s-vector models. The output vector size of GE2E and s-vectors is 256 and 512 respectively. As such, one MSS-vector model was trained with its specified parameters found in Chapter 5, which results in a 256 dimension vector. The other MSS-vector model was trained with input and output parameters to match that of the s-vector model.

6.1.2 Synthesis Model. All speaker encoder models would need to be used to train a synthesis/acoustic model to implement adaptive speech synthesis. The model with which all

speaker encoders were assessed is the SC-GlowTTS model, since it implements current state-of-the-art techniques in speech synthesis. Specifically, I used the transformer variation of SC-GlowTTS described in the paper [17]. The transformer variation of SC-GlowTTS uses the transformer network from GlowTTS and adds FastSpeech's [31] duration prediction network. In our implementation, the model was trained to 280K steps with a batch size of 32.

Similar to the speaker encoders above, four speech synthesis models were trained. The SC-GlowTTS model was trained with the s-vector and modified GE2E models, and are referred to as SV-GlowTTS and GE2E-GlowTTS respectively. The other two synthesis models were trained with the proposed MSS-vector speaker encoders, which are referred to as MSS1-GlowTTS and MSS2-GlowTTS. MSS1-GlowTTS being the MSS-vector model with training parameters detailed in chapter 5. And MSS2-GlowTTS using the MSS-vector model with training parameters to match the s-vector model.

6.1.3 Vocoder. The vocoder I chose to pair with the SC-GlowTTS model is the Hifi-GAN v2 model. This decision was made for two reasons. Firstly, the Hifi-GAN model performs well compared to other vocoders [21]. Secondly, to ensure consistent performance relative to the SC-GlowTTS paper, as it is the vocoder used by the original SC-GlowTTS model. HiFi-GAN is made up of two discriminators and one generator.

The generator is a convolutional neural network that takes in mel-spectrograms. The mel-spectrograms are transposed convolutionally and given to a multi-receptive field fusion (MRF). The MRF observes waveform patterns of varying lengths in parallel and returns the sum of multiple residual blocks. The architecture of the generator, as illustrated in [21] is shown in Figure 14.

Each of the two discriminators learn to recognize different patterns. One focuses on

learning separated, periodic patterns and is referred to as the multi-period discriminator. The other learns consecutive patterns, and is called the multi-scale discriminator. The multi-period discriminator takes 5 disjointed, non-overlapping audio waveforms as input to its sub-discriminators, as shown in Figure 15. Whereas the multi-scale discriminator accepts 3 smoothed averaged waveforms as input to 3 sub-discriminators, shown in Figure 16.

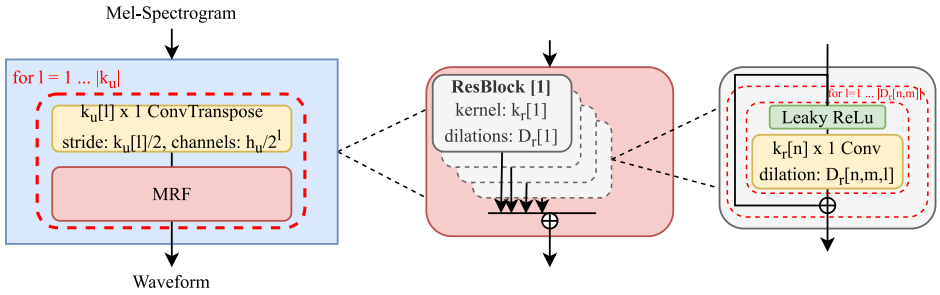

Figure 14. Multi-receptive field fusion architecture.

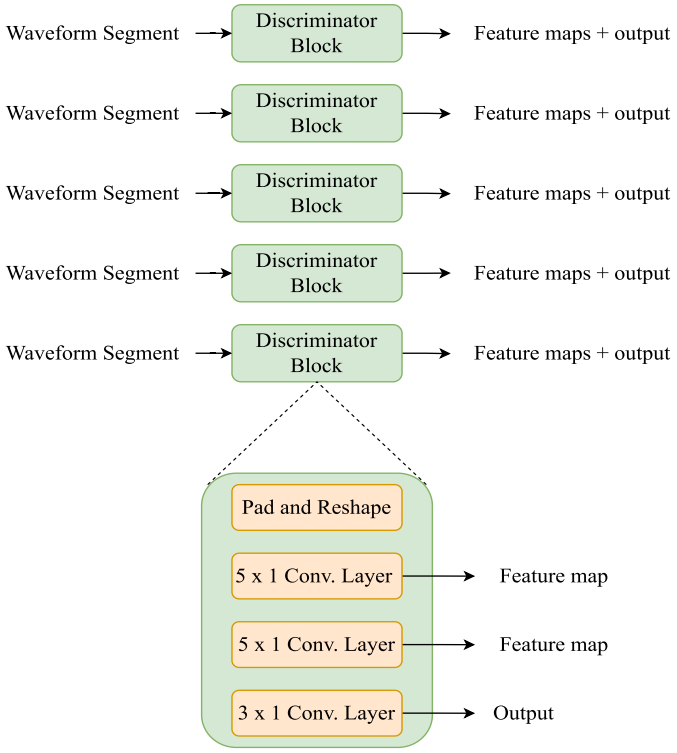

Figure 15. Multi-period discriminator architecture.

The original Hifi-GAN v2 model was trained to 500K steps on the LJ Speech [34] dataset. SC-GlowTTS used this model and trained it with an additional 75K steps on the LibriSpeech [35] dataset, and then another 190K steps on the VCTK [36] dataset. To replicate the model used in the SC-GlowTTS paper, I also used a HiFi-GAN v2 model trained with these same steps.

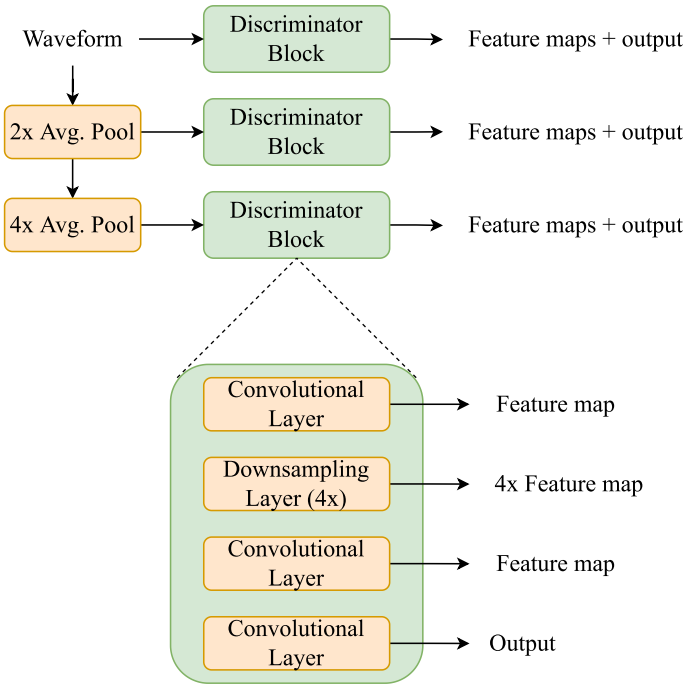

Figure 16. Multi-scale discriminator architecture.

6.2 Datasets

To train the speaker encoder models, MSS-vectors, the modified GE2E and s-vector speaker encoders, I used two partitions of the LibriSpeech corpus [36]. LibriSpeech is a public domain corpus of read speech from public domain audio books. In its entirety, this corpus contains approximately 1,000 hours of audio. The corpus is divided into multiple partitions, which vary in word error rate. The training data is composed of 3 partitions, “train clean 100”,

“train clean 360”, “train other 500”, each contain 100 hours, 360 hours, and 500 hours respectively. The test and development data sets are divided into two partitions, each; “dev-clean”, “dev-other”, “test-clean”, and “test-other”. All four of these partitions are composed of about 5 hours of audio each. The partitions chosen for training the speaker encoders were the "train clean 100” and “train clean 360”. These two partitions were used due to the variety of speakers and recording quality. Both of these partitions combined provide the speaker encoder models with 564 female and 608 male speakers, totaling 464.2 hours of recordings, all with a 24kHz sample rate.

All synthesis models were trained on the voice cloning toolkit (VCTK), which has been used to train many speech synthesis and speaker encoders, such as [29] [15] [19]. The VCTK dataset contains 109 native English speakers. Each speaker recorded approximately 400 utterances. The speakers read texts selected by a greedy algorithm (based on phonetic and contextual coverage) from the Glasgow Herald. As well as The Rainbow Passage and an elicitation paragraph. The audio from all 109 speakers totals approximately 44 hours of recordings. Out of the 109 speakers, I used 85 speakers for training, 8 for validation and 16 for testing. Before testing and training the speech synthesis models, the recordings were resampled from 48kHz to 24kHz to match the data from the LibriSpeech corpus with which the speaker encoders were trained.

The LJ speech dataset is another public domain dataset and was used in the training of the HiFi-GAN vocoder and the original Mel-GAN vocoder it’s based on. The dataset is composed of 13,100 audio clips from a single female speaker. The utterances from selected from 7 public domain, non-fiction books published between 1884-1964. Each audio clip ranges from 1 to 10 seconds. The dataset is approximately 24 hours.

6.3 Evaluation Metrics

As mentioned in the previous subsection, to assess the proposed speaker encoder, I trained four speaker encoder models and four speech synthesis models. All trained models were evaluated from two perspectives - a subjective perspective and an objective perspective. While many studies on speech synthesis systems rely heavily on crowdsourced mean opinion scores (MOS),I decided to supplement the collected MOS with speaker encoder cosine similarity (SECS) and mel-cepstral distortion (MCD) as objective measures.

6.3.1 Speaker Encoder Cosine Similarity. Speaker encoder cosine similarity (SECS) is an objective measure to evaluate the similarity in the embedding space that the models can achieve [19], [17]. SECS measures how disjoint the reference audio and its synthesized output are in the vector space of the speaker encoder. To calculate the SECS, inference is run with a speaker encoder model using each test utterance, which computes a speaker vector. Then the corresponding text of that utterance is synthesized using the test utterance as a reference sample and computes the speaker vector of that output. Lastly, cosine similarity is calculated from these two vectors.

6.3.2 Mel-Cepstral Distortion. Another objective measure used to assess the naturalness of the output speech is the mel-cepstral distortion (MCD). MCD is the difference between the MFCC features of the referenced and synthesized audio, which has been shown to provide a good indication of naturalness [20]. Similar to SECS, MFCC features were calculated for each test utterance, and the corresponding text of that utterance was used to synthesize a sample using the test utterance as reference audio. MFCC features were again computed from the speech sample, and the MCD was derived from the two corresponding MFCC vectors.

6.3.3 Naturalness Mean Opinion Score. Mean opinion scores (MOS) were chosen as a

subje-

ctive measure for the naturalness of the synthesized speech. To collect MOS, volunteer participants were asked to rate the overall quality of the speech sample on a Likert scale, consisting of “bad, poor, fair, good, excellent”. Participants were gathered through a combination of word of mouth and digital communications. Once a potential candidate expressed interest, a prospective agreement was sent to them to be signed, after which the participant would be able to take the online listening survey on their own time.

6.3.4 Similarity Mean Opinion Score. I also report the similarity of synthesized speech to the reference speaker, with similarity mean opinion scores (Sim-MOS). To collect Sim-MOS, volunteer participants were asked to evaluate the similarity of a pair of speakers, regardless of the content, grammar, or audio quality of the sentences, as done in [37], [16]. Participants used a Likert scale consisting of “not at all similar, slightly similar, moderately similar, very similar, and extremely similar”. Recruiting participants followed the same procedure as with MOS.

7 EXPERIMENTAL RESULTS

Two comparison experiments were performed to assess the proposed speaker encoder approach against two state-of-the-art speaker encoders. The first compared the MSS1-GlowTTS model against the GE2E-GlowTTS model. The second compared the MSS2-GlowTTS model to the SV-GlowTTS model. Each experiment was composed of two parts. The first part was a listening survey that was conducted to gather mean opinion scores. The second part was the calculation of SECS and MCD measures. In this results chapter each experiment will have its own subsection. However, I will discuss the naturalness MOS and the MCD together in their own subsubsection. Likewise with Sim-MOS and SECS. This is because these measures are related to each other and produces a clearer discussion of the results.

7.1 Experiment 1

7.1.1 Speaker Naturalness. In assessing the effect the proposed model has on the perceived naturalness of zero-shot TTS output, MCD and MOS were calculated. 33 volunteers English-speaking participants aged 24-65 participated in the listening test to gather MOS, and were asked questions described in Chapter 5. Samples for the survey were chosen as follows: For each model, MSS1-GlowTTS and GE2E-GlowTTS, five speech samples of 14 or more words were randomly chosen from the 16 unseen test speakers. Five more ground truth samples were selected in the same fashion, which resulted in 15 total speech samples from the two models and ground truths.

The collected data is presented in Table I. The ground truth reached a score of 4.11. The SC-GlowTTS model trained with the proposed speaker encoder (MSS1-GlowTTS) scored the

closest to the ground truth with a score of 3.20 ± 0.01, while the modified GE2E speaker encoder (GE2E-GlowTTS) achieved a score of 3.08 ± 0.16. On the other hand, MCD was calculated by using all test utterances from the unseen test speakers, amounting to 6309 utterances. Results found the amount of distortion to be marginally less in the MSS1-GlowTTS model than in the GE2E-GlowTTS model, each scoring 7.622 ± 0.024 and 7.738 ± 0.026, respectively. Therefore, based on the results presented in Table 1, it is evident that the proposed speaker encoder produced a higher level of naturalness than the modified GE2E model.

Table 1. Experiment 1: GE2E and MSS-vectors speaker naturalness

Model	MOS	MCD
GE2E-GlowTTS	3.08 ± 0.16	7.738 ± 0.026
MSS1-GlowTTS	3.20 ± 0.01	7.622 ± 0.024
Ground Truth	4.11 ± 0.17	

7.1.2 Speaker Similarity. Two measures were used to measure speaker similarity, SECS and Sim-MOS, and the results are presented in Table 2. The same 33 participants who participated in the listening test for MOS also rated speech samples for collecting the Sim-MOS. Samples for collecting Sim-MOS were chosen in a similar method to how samples for collecting MOS were chosen. Five ground truths were paired with the corresponding synthesized sample from the MSS1-GlowTTS and GE2E-GlowTTS models, giving a total of 10 pairs of speech samples. The text spoken in the synthesized speech was different from the ground truths.

SECS was calculated from the same 6309 utterances used for MCD calculation. From Ta-

ble 2, it is shown that the best Sim-MOS was achieved by the MSS1-GlowTTS model with a score of 2.717 ± 0.006, whereas GE2E-GlowTTS scored 2.274 ± 0.203. However, for SECS, GE2E-GlowTTS resulted in a higher SECS of 0.494, and MSS1-GlowTTS received a SECS of 0.415. This difference suggests that our proposed model would be less effective in speaker verification tasks than the modified GE2E model.

Table 2. Experiment 1: GE2E and MSS-vectors speaker similarity

Model	SECS	Sim-MOS
GE2E-GlowTTS	0.494	2.274 ± 0.203
MSS1-SGlowTTS	0.415	2.717 ± 0.006

To add more to this observation, visualizations of speaker embeddings for select speakers, shown in unique colors, via principal component analysis (PCA). As can be seen in Figure 17, the modified GE2E speaker encoder results in tighter groupings of a speaker from a given source (synthesized or ground truth) than my proposed MSS speaker encoder, shown in Figure 18.

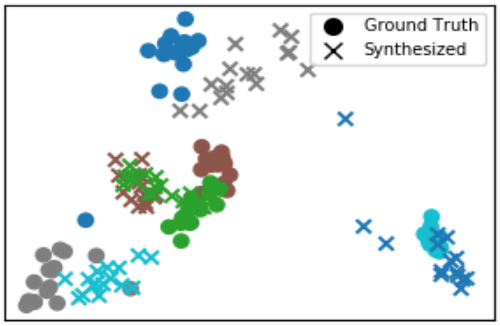

Figure 17. Principal component analysis of GE2E model.

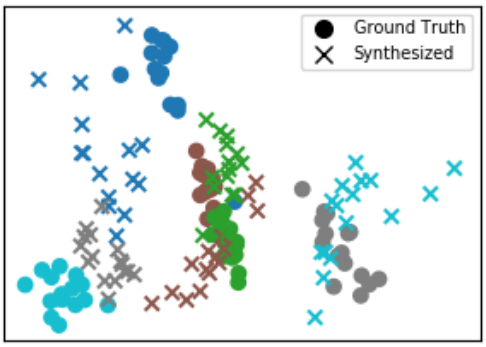

Figure 18. Principal component analysis of MSS-vector model.

This provides additional insight for speaker verification performance not being correlated to an increase in synthesized speaker naturalness or similarity. In summary, MSS1-GlowTTS scored higher than GE2E-GlowTTS in MOS, MCD, and Sim-MOS, but it was outperformed by GE2E-GlowTTS in SECS. MOS and MCD both show that MSS1-GlowTTS produces more natural speech than GE2E-GlowTTS and produced speech significantly more similar to a target speaker in subjective listening tests. The results also support the observation made by the authors of [29] that there is no meaningful correlation between speaker verification scores and synthesis quality.

7.2 Experiment 2

7.2.1 Speaker Naturalness. Much like in experiment one, a listening survey was givento 28 voluntary English-speaking participants aged 21-65. The same procedure in experiment 1 for selecting speech samples for use in both the collection of MOS and Sim-MOS was followed in experiment 2. The ground truth speech samples resulted in a MOS of 4.48 ± 0.327. They were followed by the SV-GlowTTS and MSS2-GlowTTS models, producing scores of 3.31 ± 0.346 and 3.14 ± 0.353, respectively. Evaluating the MCD of the models resulted in scores of 7.626 ±

0.024 for SV-GlowTTS and 7.558 ± 0.024 for MSS2-GlowTTS. SV-GlowTTS scored higher than MSS2-GlowTTS in MOS, but the reverse occurred in the MCD measure. This could be explained by the greater standard deviation present in the MSS2-GlowTTS MOS score, as well as the smaller sample size evaluated. These results can be seen in Table 3.

Table 3. Experiment 2: S-vectors and MSS-vectors speaker naturalness

Model	MOS	MCD
SV-GlowTTS	3.31 ± 0.346	7.626 ± 0.026
MSS2-GlowTTS	3.14 ± 0.353	7.558 ± 0.024
Ground Truth	4.48 ± 0.327	

7.2.2 Speaker Similarity. The same SECS and Sim-MOS measures were used to evaluate the similarity between an input reference audio and the output synthesized speech for experiment 2. The results in Table 4 show that SV-GlowTTS was 2.16 ± 0.414 and 2.49 ± 0.464 for MSS2-GlowTTS. The SECS produced was 0.459 and 0.414 for MSS2-GlowTTS and SV-GlowTTS, respectively. It shows that the MSS2-GlowTTS model averaged closer speaker groupings in vector space. Both SECS and Sim-MOS support the hypothesis that the MSS2-GlowTTS model produced more similar speech to the input reference speech than the SV-GlowTTS model.

To further investigate the differences between the MSS-vectors and s-vectors models, Isomap dimension reduction was used to visualize the ground truth and synthesized embeddings in a Euclidean 3-dimensional space. Isomap reduces the dimensionality of the 512-dimension

embedding vectors to 3 dimensions in a non-linear way, so as to preserve the relationships between neighboring embeddings as much as possible. As shown in Figure 19 and Figure 20, the nature of embedding groupings is fundamentally different between the two models. The s-vectors model in Figure 19 results in different speakers, represented by different colors, tightly grouped in linear formations with a moderate separation between ground truth and synthesized samples. However, the MSS-vectors model in Figure 20 results in most speakers being grouped loosely into sphere-like formations, with less separation between ground truth and synthesized samples.

Table 4. Experiment 2: S-vectors and MSS-vectors speaker similarity

Model	SECS	Sim-MOS
SV-GlowTTS	0.414	2.16 ± 0.414
MSS2-GlowTTS	0.459	2.49 ± 0.464

The difference in separation between the two models' ground truth and synthesized speaker embeddings could suggest two possible conclusions. First, the MSS-vectors model scores higher in speaker similarity because the zero-shot TTS model learned more effectively from the speaker embeddings produced by the MSS-vectors model and produced more similar sounding speech. Secondly, it could indicate that the MSS-vectors model would be less effective in speaker verification tasks. Additionally, the MSS-vectors Isomap shows the blue and green speakers spread out much more in comparison to the other speakers that were plotted. A similar feature can be seen in the s-vectors Isomap, with the blue and green speakers deviating from the linear formation common with the other speakers, resulting in a more compact shape. This could

indicate that these two speakers' voices had significantly different features than the speakers the models were trained on.

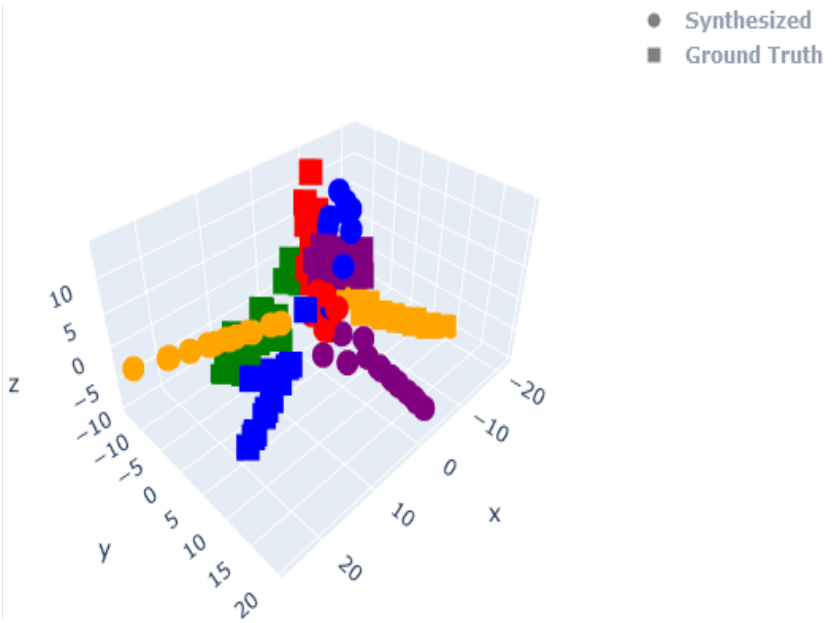

Figure 19. ISOmap plots for S-vector model.

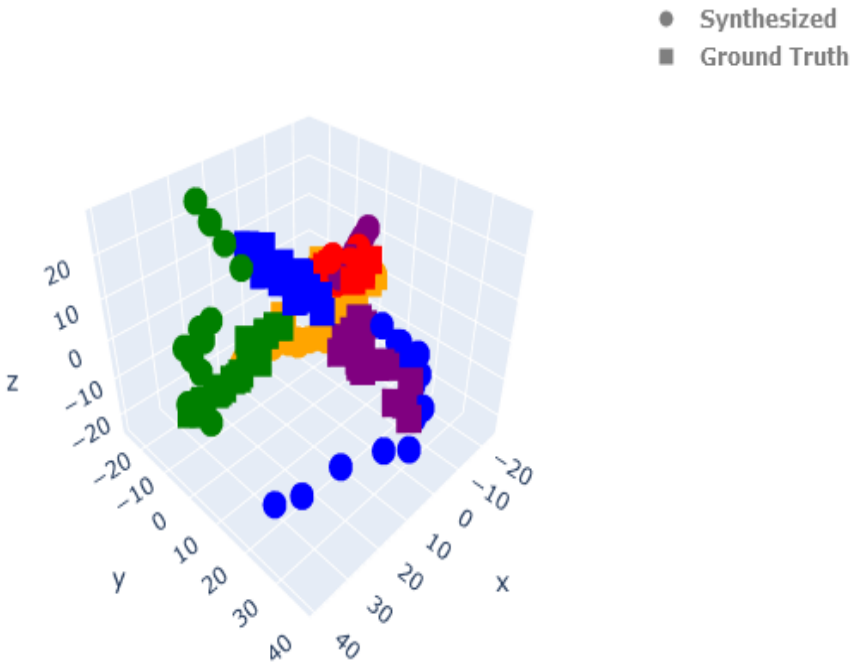

Figure 20. ISOmap plot for MSS-vector model.

Based on the results from the listening survey, as well as the SECS and MCD measures, differences can be observed between the two models. These differences are further seen in the Isomap visualizations of the two models' speaker embeddings. As a result, it can be concluded that the MSS vectors model achieved a small but marked improvement over the s-vectors model in zero-shot TTS systems, particularly in the speaker similarity aspect of zero-shot TTS.

8 CONCLUSION

In this thesis, I presented a novel speaker encoder model designed to optimize zero-shot speech synthesis performance named MSS-vectors. The MSS-vectors model is designed to address a hypothesized shortcoming of the current speaker encoders used in zero-shot speech synthesis systems in order to make this method of adaptive speech synthesis produce more human-like and similar sounding speech. The hypothesized shortcoming is that current speaker encoders extract speaker features that are highly specific to the target speaker. While this is a desirable trait in speaker recognition, it could be a hinderance when synthesizing a specific speaker due to it missing more general vocal cues necessary for base human speech. To address this, the MSS-vector model architecture needed to direct speaker feature extraction equally across a reference speakers' mel-spectrum. This was done by introducing a multi-scale concept for learning speaker features, resulting in two encoders, which I call the global scale encoder (GSE) and local scale encoder (LSE). The GSE extracting general features of the speaker, such as vocal timbre. The LSE extracting nuanced speaker features such as quirks in phoneme pronunciation. Together, these two encoders comprise the primary feature learning of the MSS-vector speaker encoder.

To evaluate the proposed speaker encoder, I implemented and trained two state-of-the-art speaker encoders, a modified GE2E model and a s-vector model for comparison. Then two experimental trials were performed. Each trial consisted of two parts and compared the MSS-vectors model against one of the two state-of-the-art speaker encoders previously mentioned. The first part was a listening survey, which gathered objective mean opinion scores from human subjects based on how natural audio samples sounded and how similar an output sample sounded

to its reference speaker. The second part was a quantitative assessment of both naturalness and similarity to help reinforce findings from the listening survey. The first experimental trial was comparing the GE2E model and my MSS-vector model. The results of experiment 1, part 1, (the listening survey) showed that the MSS-vector model was superior to the GE2E model in both naturalness and replicating a target speaker, with an approximate increase of 3.89% and 19.48% respectively, over the GE2E model. Part 2 of experiment 1 validated the increase in naturalness with the MCD measure. SECS, which measures the cosine similarity of the ground truth and output audio speaker embeddings in vector space, showed that the MSS-vector model resulted in the two speaker embeddings further apart, which traditionally would be interpreted as the two samples being less similar. In the second experimental trial, the MSS-vector model was compared to the s-vector model. The listening survey showed my proposed MSS-vector model to be a 5.41% decrease in naturalness and a 15.27% increase in similarity compared to the s-vector model. Experiment 2, part 2, however showed marginally less distortion in the MSS-vector model, which would suggest it would be more natural sounding than the s-vector model. Additionally, the SECS showed the speaker embeddings from the MSS-vector model to be closer together than the s-vector model, suggesting MSS-vectors produce speech which sounds more like the reference speaker. In both experiment 1 and 2, there was a discrepancy between either MCD or SECS and the mean opinion scores. In the case of MCD, the two models resulting distortions were only marginally different. For SECS, however, there was a much larger difference between the two. This outcome could suggest that SECS does not correlate with output speaker similarity. In summary, the MSS-vector model produces speech that is more similar to the reference speaker than either of the other two state-of-the-art speaker encoder models. This is confirmed by the mean opinion scores, which are the gold standard of speech

quality evaluation, from both listening surveys.

The MSS-vector model has proven to produce more similar speakers in a zero-shot TTS model than other current state-of-the-art speaker encoders. The motivation of this research was to improve naturalness and reduce the gap in quality between adapted unseen speakers and trained seen speakers in zero-shot text-to-speech systems. While the MSS-vector model has made progress towards that goal by replicating speakers more closely, it has not yet achieved human-like naturalness. Based on this thesis's findings, the next step would be to investigate objective measures that correlate with the naturalness and similarity of synthesized speech systems and propose a method of training speaker encoders based on those objective measures.

9 REFERENCES

[1] B. H. Story, "History of Speech Synthesis," *The Routledge Handbook of Phonetics*, Taylor and Francis, pp. 9-33, 2019.

[2] X. Tan, T. Qin, F. K Soong, and T.-Y. Liu, "A Survey on Neural Speech Synthesis," *ArXiv*, vol. abs/2106.15561, 2021.

[3] D. H. Klatt, "Review of text-to-speech conversion for English," *The Journal of the Acoustical Society of America,* vol. 82, no. 3, pp. 737-93, 1987.

[4] N. Dixon and H. Maxey, "Terminal analog synthesis of continuous speech using the diphone method of segment assembly," *IEEE Transactions on Audio and Electroacoustics,* vol. 16, pp. 40-50, 1968.

[5] W. S.-Y. Wang and G. E. Peterson, "Segment Inventory for Speech Synthesis," *The Journal of the Acoustical Society of America,* vol. 30, pp. 683, 1958.

[6] A. W. Black, H. Zen and K. Tokuda, "Statistical Parametric Speech Synthesis," *IEEE International Conference on Acoustics, Speech and Signal Processing – ICASSP*, pp. IV-1229-IV-1232, 2007.

[7] H. Zen, A. Senior and M. Schuster, "Statistical parametric speech synthesis using deep neural networks," *IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 7962-7966, 2013.

[8] T. Capes, P. Coles, A. Conkie, L. Golipour, A. Hadjitarkhani, Q. Hu, N. Huddleston, M. Hunt *et al*., "Siri On-Device Deep Learning-Guided Unit Selection Text-to-Speech System," *Interspeech*, pp. 4011-4015, 2017.

[9] M.-W. Chang, L. Ratinov, D. Roth and V. Srikumar, "Importance of semantic representation: Dataless classification," *Proceedings of the 23rd Nation Conference on Articificial Intelligence*, vol. 2, pp. 830-835, 2008.

[10] H. Larochelle, D. Erhan and Y. Bengio, "Zero-data learning of new tasks," in *Proceedings of the 23rd National Conference on Artificial Intelligence*, vol. 2, pp. 646-651, 2008.

[11] L. Fei-Fei, R. Fergus and P. Perona, "One-shot learning of object categories," in *National Conference on Artificial Intelligence*, vol. 28, pp. 594-611, 2006.

[12] J. Shen, R. Pang, R. Weiss, M. Schuster, N. Jaitly, Z. Yang, Z. Chen, Y. Zhang and Y. Wang, "Natural tts synthesis by conditioning wavenet on mel spectrogram predictions,"

IEEE Intl. Conf. on acoustics, speech and signal processing, pp. 4779-4783, 2018.

[13] W. Y, R. Skerry-Ryan, D. Stanton, Y. Wu, R. Weiss, N. Jaitly, Z. Yang, Y. Xiao, Z. Chen, S. Bengio, Q. Le, Y. Agiomyrgainnakis, R. Clark and R. Saurous, "Tacotron: Towards end-to-end speech synthesis," *Interspeech*, pp. 4006-4010, 2017.

[14] A. Tamamori, T. Hayahsi, K. Kobayashi, K. Takeda and T. Toda, "Speaker-dependent WaveNet vocoder," *Interspeech*, pp.1118-1122, 2017.

[15] A. Oord, S. Dieleman, H. Zen, K. Simonya, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior and K. Kavukcuoglu, "WaveNet: A generative model for raw audio," *CoRR*, 2016.

[16] Y. Jia, Y. Zhang, R. J. Weiss, Q. Wang, J. Shen, F. Ren, Z. Chen, P. Nguyen, R. Pang, I. L. Moreno and Y. Wu, "Transfer Learning from Speaker Verification to Multispeaker Text-to-Speech Synthesis," *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pp. 4485-4495, 2018.

[17] C. Edresson, S. Christopher, G. Eren, M. M. Nicolas, F. S. d. Oliveira, C. J. Arnaldo, S. Silva, M. A. Sandra and A. P. Moacir, "SC-GlowTTS: An Efficient Zero-Shot Multi-Speaker Text-To-To-Speech Model," *Interspeech*, pp. 3645-3649, 2021.

[18] J. Kim, S. Kim, J. Kong and S. Yoon, "Glow-tts: A generative flow for text-to-speech via monotonic alignment search," *Advances in Neural Information Processing Systems*, vol. 33, pp. 8067-8077, 2020.

[19] S. Choi, S. Han, D. Kim and S. Ha, "Attentron: Few-shot text-to-speech utilizing attention-based variable-length embedding," *ArXiv*, vol. abs/2005.08484, 2020.

[20] N. Li, S. Liu, Y. Liu, S. Zhao, M. Liu, "Neural Speech Synthesis with Transformer Network," *AAAI*, vol. 33, pp. 6706-6713, 2019.

[21] J. Kong, J. Kim and J. Bae, "Hifi-gan: Generative adversarial networks for efficient and high fidelity speech synthesis," *Proceedings of the 34th International Conference on Neural Information Processing Systems*, 2020.

[22] K. Kumar, R. Kumar, T. de Boissiere, L. Gestin, Z. W. Teoh, J. Sotelo, A. De Brebisson, Y. Bengio and A. C. Courville, "MelGAN: Generative Adversarial Networks for Conditional Waveform Synthesis," *Advances in Nerual Information Processing Systems*, 2019.

[23] M. Kabir, M. Mridha, J. Shin, I. Jahan and A. Ohi, "A Survey of Speaker Recognition: Fundamental Theories, Recognition Methods and Opportunities," *IEEE Access*, vol. 9, pp 79236-79263. 2021.

[24] Z. Bai and X.-L. Zhang, "Speaker Recognition Based on Deep Learning: An Overview," *Neural Networks,* vol. 140, pp. 65-99, 2021.

[25] N. Dehak, P. Kenny, R. Dehak, P. Dumouchel and P. Ouellet, "Front-end factor analysis for speaker verification," *IEEE Trans. on Audio, Speech, and Language Processing*, vol. 19, pp. 788-798, 2010.

[26] D. Snyder, D. Garcia-Romero, G. Sell, D. Povey and S. Khudanpur, "X-vectors: Robust dnn embeddings for speaker recognition," *IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 5329-5333, 2018.

[27] N. S. A. Vaswani, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser and I. Polosukhin, "Attention is All You Need," *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[28] S. V. Katta, S. Umesh, *et al*., "S-vectors: Speaker embeddings based on transformer's encoder for text-independent speaker verification," *ArXiv*, vol. abs/2008.04659, 2020.

[29] E. Cooper, C.-I. Lai, Y. Yasuda, F. Fang, X. Wang, N. Chen, J. Yamagishi *et al*., "Zero-shot multi-speaker text-to-speech with state-of-the-art neural speaker embeddings," *IEEE Intl. Conf. on Acoustics, Speech and Signal processing*, pp. 6184-6188, 2020.

[30] L. Wan, Q. Wang, A. Papir and I. L. Moreno, "Generalized end-to-end loss for speaker verification," *IEEE Intl. Conf. on Acoustics, Speech and Signal processing*, pp. 4879-4883, 2018.

[31] Y. Ren, Y. Ruan, X. Tan, T. Qin, S. Zhao and T. -Y. Liu, "Fastspeech: Fast, Robust and Controllable Text to Speech," *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[32] coqui ai, "Coqui tts," [Online]. Available: https://github.com/coqui-ai/TTS.

[33] J. S. Chung, J. Huh, S. Mun, M. Lee, H.-S. Heo, S. Choe, C. Ham, S. Jung, B.-J. Lee and I. Han, "In defence of metric learning for speaker recognition," *Interspeech*, pp. 2977-2981, 2020.

[34] K. Ito and L. Johnson, "The lj speech dataset," [Online]. Available: https://keithito.com/LJ-Speech-Dataset/., 2017.

[35] V. Panayotov, G. Chen, D. Povey and S. Khudanpus, "Librispeech: an asr corpus based on public domain audio books," *IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 5206-5210, 2015.

[36] C. Veaux, J. Yamagishi, K. MacDonald *et al*., "Superseded-cstr vctk corpus: English multi-

speaker corpus for cstr voice cloning toolkit," Univ. of Edinburgh. The Centre for Speech Technology Research, 2016.

[37] J. Kominek, T. Schultz and A. W. Black, "Synthesizer voice quality of new languages calibrated with mean mel cepstral distortion," *SLTU*, pp. 63-68, 2008.

APPENDIX: RESEARCH COMPLIANCE

To:
Razib Iqbal
Computer Science

Date: Dec, 20, 2021 10:29:57 AM CST

RE: Notice of IRB Exemption

Study #: IRB-FY2022-384

Study Title: Synthesis of Natural Sounding Speech

This submission has been reviewed by the Missouri State University Institutional Review Board (IRB) and was determined to be exempt from further review. However, any changes to any aspect of this study must be submitted, as a modification to the study, for IRB review as the changes may change this Exempt determination. Should any adverse event or unanticipated problem involving risks to subjects or others occur it must be reported immediately to the IRB.

This study was reviewed in accordance with federal regulations governing human subjects research, including those found at 45 CFR 46 (Common Rule), 45 CFR 164 (HIPAA), 21 CFR 50 & 56 (FDA), and 40 CFR 26 (EPA), where applicable.

Researchers Associated with this Project:

PI: Razib Iqbal

Co-PI:

Primary Contact: Tristin Cory

Other Investigators: