POLITECNICO DI TORINO

Master's Degree in Data Science & Engineering

Master's Degree Thesis

Data driven: AI Voice Cloning

Supervisors

Prof. Luca CAGLIERO

Doc. Moreno LA QUATRA

Doc. Lorenzo VAIANI

Candidate

Alessandro Emmanuel PECORA

28/07/2023

Summary

This thesis focuses on speech processing, specifically Speaker Recognition (SR) and Text-To-Speech synthesis (TTS). It explores various speaker embedding techniques for SR, such as i-vectors, d-vectors, and x-vectors, with a focus on x-vectors extracted using the SV objective.

Two deep learning architectures, WAVLM+ and ECAPA-TDNN, are enhanced using the newer Generalized End-to-end (GE2E) loss function, resulting in improved performance on the VoxCeleb 1 dataset.

In the TTS domain, the Tacotron and FastSpeech series of architectures are analyzed. Tacotron 2 utilizes LSTM encoders and decoders with attention mechanisms, while FastSpeech2 employs transformer-based models to convert phoneme embeddings into mel-spectrograms.

Two limitations are faced: the need for a high amount of data for effective synthesis, and the necessity for more controllability on the generated synthesis. To address the first limitation, a case study demonstrating the benefits of incorporating pangram utterances in training data was conducted, highlighting the importance of data quality. To implement a better solution that resolves both limitations, the thesis extends TTS models to develop a Voice Cloning System using a zero-shot learning approach. This is achieved by integrating speaker embeddings from ECAPA-TDNN into the FastSpeech2 text encoder output. The resulting system is able to generate natural sounding speech synthesis in the voice of a target speaker starting from about 5 seconds of speech audio, while permitting modulation of the pitch, energy and speed of the synthesized utterance. The Zero Shot Voice Cloner was evaluated using Voice Clone Error Rate (VC-ER) and Word Error Rate (WER) metrics, demonstrating the successful synthesis of speech in previously unheard voices.

Overall, the thesis enhances speaker verification models through Generalized End-to-End loss fine-tuning, emphasizes the significance of well-constructed datasets for TTS, and successfully utilizes speaker embeddings to condition the TTS model. A demo application is released to showcase the capabilities of the Voice Cloner and provide an implementation for future exploration.

Acknowledgements

English Version:

I am grateful to Prof. Luca Cagliero, Doc. Vaiani Lorenzo, and especially Doc. Moreno La Quatra for their guidance and support in this research. Their invaluable assistance has been crucial in successfully completing this work. Additionally, I appreciate the HPC@Polito lab for providing the necessary hardware resources for this research.

I would like to express my gratitude to my family: my brothers, and especially my parents, Maria and Raffaele, for allowing me to explore and for their unwavering support, even when my choices did not align with their world. With the warmth of the family, they have supported and encouraged me throughout my journey. Despite our differences, they have always had faith in me and have given me the opportunity to become the person I am today.

I would like to thank my friends, those with whom I have shared a constant friendship over time, and also the new friends with whom I have explored Torino. I want to mention also my colleagues, who have become friends during our university journey. I am grateful to all for the shared experiences, the support during difficult times, and the celebrations during moments of joy.

Finally, a special thanks goes to Giorgia, who has been by my side during the past years and throughout the entire process of writing this thesis. Together, we ventured into the world of work, and as a companion, she stood by me during the most stressful moments, always listening to me with the patience and kindness that characterize her.

"I don't know Morty, sometimes I feel like we're living in a simulation. Like, nothing can be real. But then I look at your face and I remember how much you mean to me and I know that, in this reality, we're real and we're together."
Rick Sanchez "Rick & Morty"

Versione italiana:

Sono grato al Prof. Luca Cagliero, al Dott. Vaiani Lorenzo e, in particolar modo, al Dott. Moreno La Quatra per la loro guida e supporto in questa ricerca. Il loro invalutabile aiuto è stato cruciale per completare con successo questo lavoro. Inoltre, ringrazio il laboratorio HPC@Polito per aver fornito le risorse hardware necessarie per questa ricerca.

Desidero esprimere la mia gratitudine alla mia famiglia: ai miei fratelli e, soprattutto, ai miei genitori, Maria e Raffaele, per avermi permesso di esplorare e per il loro sostegno incondizionato, anche quando le mie scelte non si allineavano con il loro mondo. Grazie al calore della famiglia, mi hanno sostenuto e incoraggiato lungo tutto il mio percorso. Nonostante le nostre differenze, hanno sempre avuto fede in me e mi hanno dato l'opportunità di diventare la persona che sono oggi.

Desidero ringraziare i miei amici, coloro con cui ho condiviso un'amicizia costante nel tempo e anche i nuovi amici con cui ho esplorato Torino. Voglio menzionare anche i colleghi, che sono diventati amici durante il nostro percorso universitario. Vi sono grato a tutti per le esperienze condivise, il supporto durante i momenti difficili e le celebrazioni durante i momenti di gioia.

Infine, un ringraziamento speciale va a Giorgia, che è stata al mio fianco durante gli ultimi anni e durante l'intero processo di scrittura di questa tesi. Insieme, ci siamo avventurati nel mondo del lavoro e come compagna, lei è stata al mio fianco durante i momenti più stressanti, ascoltandomi sempre con la pazienza e la gentilezza che la contraddistinguono.

"Non lo so, Morty, a volte mi sembra di vivere in una simulazione. Come se niente potesse essere reale. Ma poi guardo il tuo volto e mi ricordo quanto tu significhi per me e so che, in questa realtà, siamo reali e siamo insieme."
Rick Sanchez "Rick & Morty"

Table of Contents

List of Tables

List of Figures

Acronyms

AI

Artificial Intelligence

SR

Speaker Recognition

SV

Speaker Verification

SI

Speaker Identification

SE

Speaker Embedding

TTS

Text To Speech

CSS

Concatenative Speech Synthesis

EER

Equal Error Rate

VC-ER

Voice Clone Error Rate

WER

Word Error Rate

MSE

Mean Squared Error

GE2E

Generalized end-to-end

Chapter 1

Introduction

"In the end you should only measure and look at the numbers that drive action, meaning that the data tells you what you should do next."
Alexander Peiniger

In the current era, science, computer science, and technology are driving significant societal changes. The digital era demands efficient data management solutions, and we can now harness the power of machine learning theories developed in the 1950s, which had long awaited practical application. In recent years, the undeniable emergence of artificial intelligence as a disruptive force has led to diverse applications ranging from medical field to weather forecasting to image and code generation, as well as voice cloning and more, in this context AI has become an indispensable tool, poised to wield even greater influence in the future.

As humans, we transmit a significant amount of information through speech, and evolution has developed an entire organ with the function of modulating audio signals for communication purposes. Speech is the most commonly used communication channel among humans. With the advent of technology, voice calls and audio messaging have become increasingly prevalent. These data represent a valuable resource if the appropriate technologies are used to process them. There are several transformations that can be applied to achieve tasks that effectively extract value from the data, including automatic speech recognition, speaker identification, speaker verification, voice activation detection, text-to-speech synthesis, speech separation, and more.

This thesis explores the field of **speech processing**, a longstanding area of research that focuses on understanding, generating, and encoding speech. The goal is to develop systems that improve communication between humans and machines, enhancing effectiveness and efficiency. With the increasing demand for advanced communication technologies, there is a growing need for innovative approaches that can provide personalized and expressive speech capabilities. Motivated by this need, the thesis concentrates on enhancing two crucial tasks in speech processing: **speaker recognition (SR)** and **text-to-speech synthesis (TTS)**. By integrating these advancements, the thesis aims to develop a **Voice Cloning System** capable of synthesizing speech in previously unheard voices. This system has the potential to revolutionize various applications, including personalized voice assistants, multimedia localization, and interactive entertainment experiences.

Speaker recognition is designed to determine an individual's identity through their voice. Applications for speaker recognition span from secure authentication on devices to personalized customer service and forensic voice analysis. Works in literature have developed different statistical methods to achieve this goal [1] [2] [3], but the most promising are those based on deep learning solutions [4] [5]. Recently, studies on SR have made separate contributions to advancements in network architectures and loss functions. As natural consequence, this thesis combines these discoveries by utilizing newer architectures, namely **ECAPA-TDNN** [6] and **WAVLM** [7], trained with the **Generalized End-to-End Loss (GE2E)** [8]. The performances of these models are evaluated on the VoxCeleb 1 dataset [9], where they outperform the current ones.

About **Text-to-Speech synthesis**, it is one of the oldest problems in Natural Language and Speech Processing [10]. Such systems entail creating natural-sounding human speech waveforms from a provided input text. TTS applications have a broad range of applications. One particularly valuable and potentially disruptive example is an application capable of automatically dubbing media in various languages using customizable voices. This innovation has the potential to greatly impact industries like film and gaming, which currently allocate significant resources to localize their products through multilingual dubbing. However, it is important to acknowledge the risks associated with such technologies. Malicious applications could exploit voice cloning techniques to impersonate individuals and employ social engineering tactics for illegal purposes. Therefore, responsible development entails a research focus on effectively containing and mitigating the risks posed by these malicious applications.

In recent years, deep learning-based approaches, particularly end-to-end models like **Tacotron** [11], have simplified the text-to-speech pipeline and led to significant improvements. The adoption of transformer architectures has further advanced controllability and training speed, as demonstrated by models such as **FasteSpeech2** [12]. It is important to note that high-quality data is essential for successful speech synthesis due to the extensive variability in speech. In this thesis, a side study is conducted to identify "high-quality" input samples for training. A privately collected dataset called "**61PangramITA**" comprises pangram utterances from 7 Italian speakers, maximizing character diversity. This dataset can help reduce the amount of training data required for effective synthesis.

To enhance the usability and control of text-to-speech (TTS) systems, recent advancements have introduced **Multi-Speaker TTS**, enabling synthesis in multiple known voices [13]. More recently, works on **Voice Cloning Systems** [14] [15], demonstrate the possibility to synthesize unknown voices. In this work, a Voice Cloning system is developed by leveraging state-of-the-art models and loss, following latest trend. The system allows for cloning unseen voices using a zero-shot learning approach, while retaining control over post-synthesis parameters like duration, pitch, and energy. A **demo application** has been released to showcase the system's capabilities and facilitate further exploration.

The thesis is structured as follows:

- **Chapter 2 Backround:** serves as a literature overview, providing a comprehensive exploration of the main tasks under study. It includes a discussion on their recent evolution, limitations, and research directions.
- **Chapter 3 Models Architecures:** examines the key model architectures that have been analyzed and employed to address these tasks, exploring their evolution over time.
- **Chapter 4 Experiments:** different experiments are carried out for Speaker Recognition (SR), Text-to-Speech (TTS), and their integration in the Voice Cloner. This chapter outlines the data and methods used for model training, as well as reports the results compared with the available ones.
- **Chapter 5 Conclusion:** In the last chapter, the comments on the experiments are summarized, highlighting potential areas for improvement. Moreover, it discusses possible future research directions, particularly in the context of containing malicious applications.

The code for this project is openly available on GitHub at the following link: `https://github.com/alessandropec/data_driven_ai_voice_cloning`

Chapter 2

Background

"We live in a society exquisitely dependent on science and technology, in which hardly anyone knows anything about science and technology."
Carl Sagan

2.1 Speaker Recognition

Speaker recognition is a speech processing task, it involves the automatic identification of a person by analyzing the speaker-specific characteristics present in speech waves. The goal of speaker recognition is to understand the identities of individuals starting from speeches. Natural speech is not simply a concatenation of sounds, instead, it is a blending of different sounds, often with no distinct boundaries between transitions, figure 2.1, shows examples of how vocal-tract configurations produce different spectra for two vowel sounds. Different works have developed technologies and techniques for the speaker recognition problem. In [2] various algorithms, to extract features useful for the task, are described. In this work the focus is on artificial intelligent system so machine learning and in particular deep learning methods with their principal implementation are proposed, however at high level the task can be modeled in a method independent way.

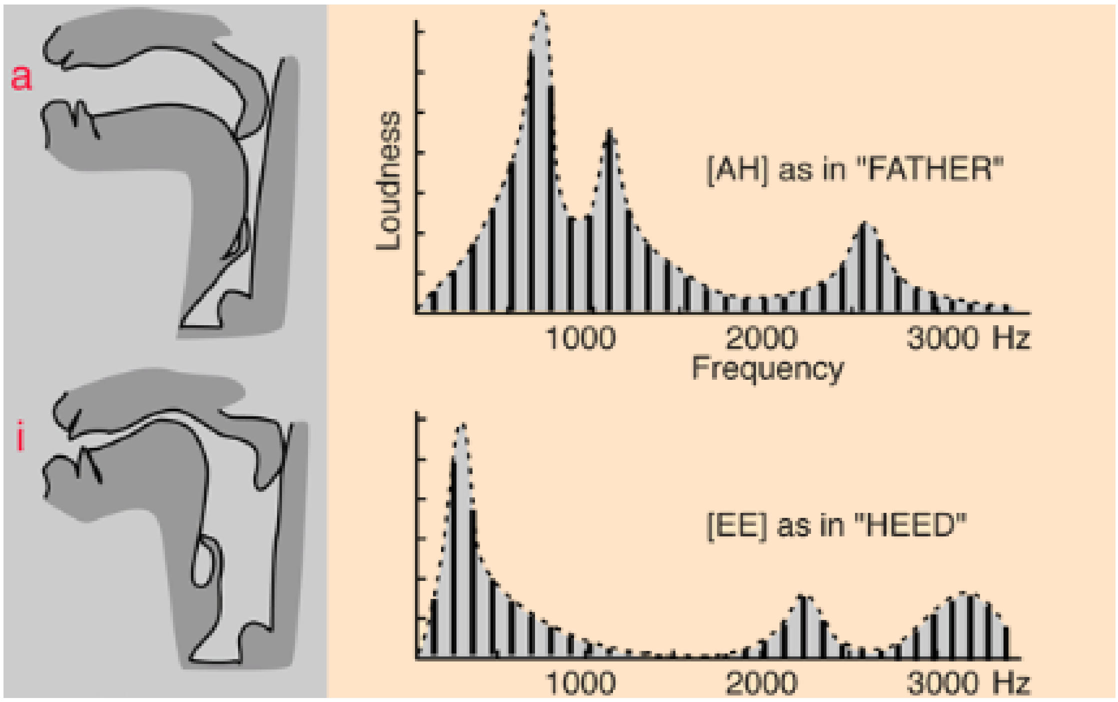


Figure 2.1: Examples of vocal-tract configurations and the corresponding frequency spectra from two steady-state vowels. The peaks, or formants, in the spectra are resonances produced by the particular vocal-tract configuration. Image from [16]

2.1.1 Problem definition: speaker identification & speaker verification

Speaker recognition, as represented in 2.10, can be modeled using two fundamental tasks described in [1]:

Speaker identification: Who is speaking?

The goal is to determine which voice in a predefined group of voices best matches the speaker. This implies a classification problem with N alternatives, forced-choice. Since increasing the set of speakers increase the difficult of the problem and in machine learning application frequently also implies a retraining, speaker identification in real world often not find application, however, an example can be speaker labeling in a video call. ML and DL implementation for speaker identification have the advantages that not needs to store a speaker reference speech audio to perform the task, as is done in speaker verification, in this case the model predict directly the id of the speakers that best match the input. In this setting an **error** is called **missclassification**

Speaker verification: I'am Alessandro?

The goal is to verify the identity of a **claimed** speaker. The input is a speech audio and a claimed speaker identity that is known to the system. In most of applications, in particular in those proposed here, speakers are known to the system in the sense that a set of reference speech audios is stored for each speaker, potentially a new speaker could be enrolled in the system by simply adding a set of it's audio speech. Speaker verification is designed for authentication system, based on voice, in fact with this task, the goal is to exploit biometric features of the voice to uniquely identify a particular speaker. The problem can be modeled as a binary classification problem, the output is a boolean variable that express if the identity is accepted or not, in this setting an **error** can be of two type **false acceptance** and **false rejections**.

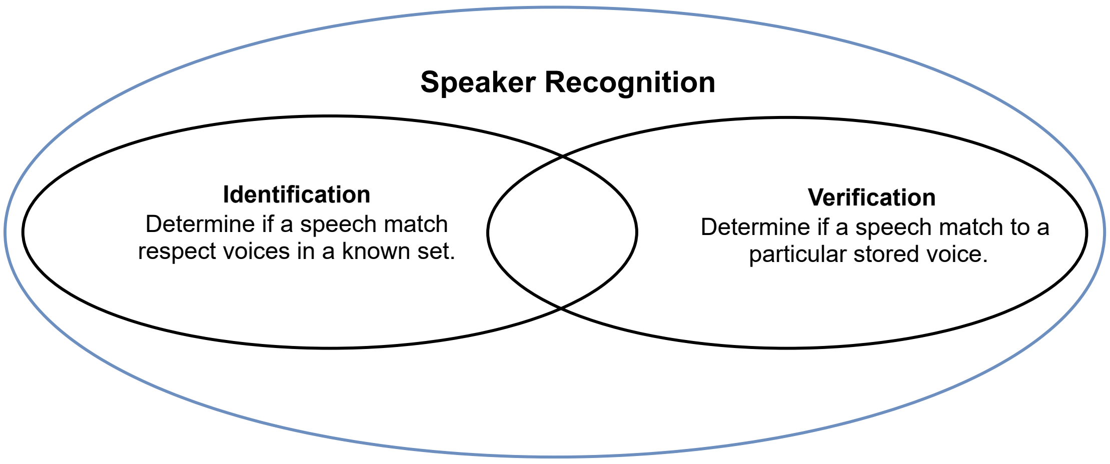

Speaker Recognition
Identification
Determine if a speech match respect voices in a known set.
Verification
Determine if a speech match to a particular stored voice.

Figure 2.2: Speaker Recognition and its two sub-tasks: identification and verification. Identification involves determining the speaker from a set of known speakers using incoming speech. Verification, on the other hand, compares incoming speech from a claimed speaker with a reference speech to confirm the claimed identity of the speaker.

Speaker recognition tasks are further distinguished by the constraints placed on the text of the speech used in the system, we have:

1. **Text-dependent system:** the spoken text used to train and test the system is constrained to be the same word or phrase. For example, in an authentication application a claimant can always use the same personalized code, or for a voice assistant the it can be triggered by a combination of a specific voice and

arbitrary words. The text dependent task take advantage of knowing the text to be spoken and in general the task is more easy.

2. **Text-independent system:** This type of system is the most flexible, applicable and useful system since it solve the uncostrained problem, as we will se in the next section perfectly fits with the goal of exploit speaker recognition task to generate a speaker-voice numerical representation known as speaker embedding.

3. **Vocabulary-dependent system:** Between the extremes of text dependence and text independence falls the vocabulary-dependent system, which constrains the speech to come from a limited vocabulary, such as the digits (e.g., "zero," "one") from which test words or phrases (e.g., "zero-oneeight") are selected.

2.1.2 From speaker recognition to speaker embedding

One of the key techniques used in speaker recognition is the extraction of speaker embeddings. These are low-dimensional representations of a speaker's unique characteristics that can be used to identify them, even in the presence of variability in their speech. Speaker embeddings can be extracted using a variety of techniques, in the following the main three recent techniques are presented:

1. **I-vectors** [3]: low dimensional speaker- and channel-dependent space is defined using a simple factor analysis also known as i-vectors.

2. **D-vectors** [4]: the conventional d-vector model train DNN architecture as a speaker feature extractor operating at the frame level. After sending the frames through the DNN network, d-vector is obtained by element-wise averaging the frame-level outputs from the last hidden layer of DNN network.

3. **X-vectors** [5]: A time-delay neural network (TDNN) with sub-sampling is used as the encoder network. An attentive statistics pooling (ASP) layer aggregates all frame-level outputs from the last encoder layer and computes its mean and standard deviation. After sending through segment level layer (fully-connected layer), the x-vector speaker embedding is obtained.

To generate the D-vectors and X-vectors the speaker recognition task can be exploited: a neural network is trained to solve speaker identification or verification objective. Then the last layer (the output layer) of the net is removed and the resulting output is used as speaker embedding. Often the feature extraction layers are called backbone and resolve the upstream task (i.e. speaker embeddings generation), while the lasts layer are the head and implement the downstream task (i.e. the speaker recognition) for which we have well defined loss function. As we will see in 2.3 partitioning the architecture of a net in different part related with different correlated task is powerful and is a major trend in recent years, in this way a full-stack-task model, often referred as *backbone*, can be trained on large dataset, then different task-specific models, called *head*, can be derived.

Based on work [17], and on avalaible resources, this work delves into speaker verification problem, exploring possible losses and state-of-art avalaible models and datasets. Since speaker verification is performed in order to extract speaker embedding like X-vectors and then, those are used to condition TTS system on a specific voice, text-independent system will be proposed and losses will be proposed.

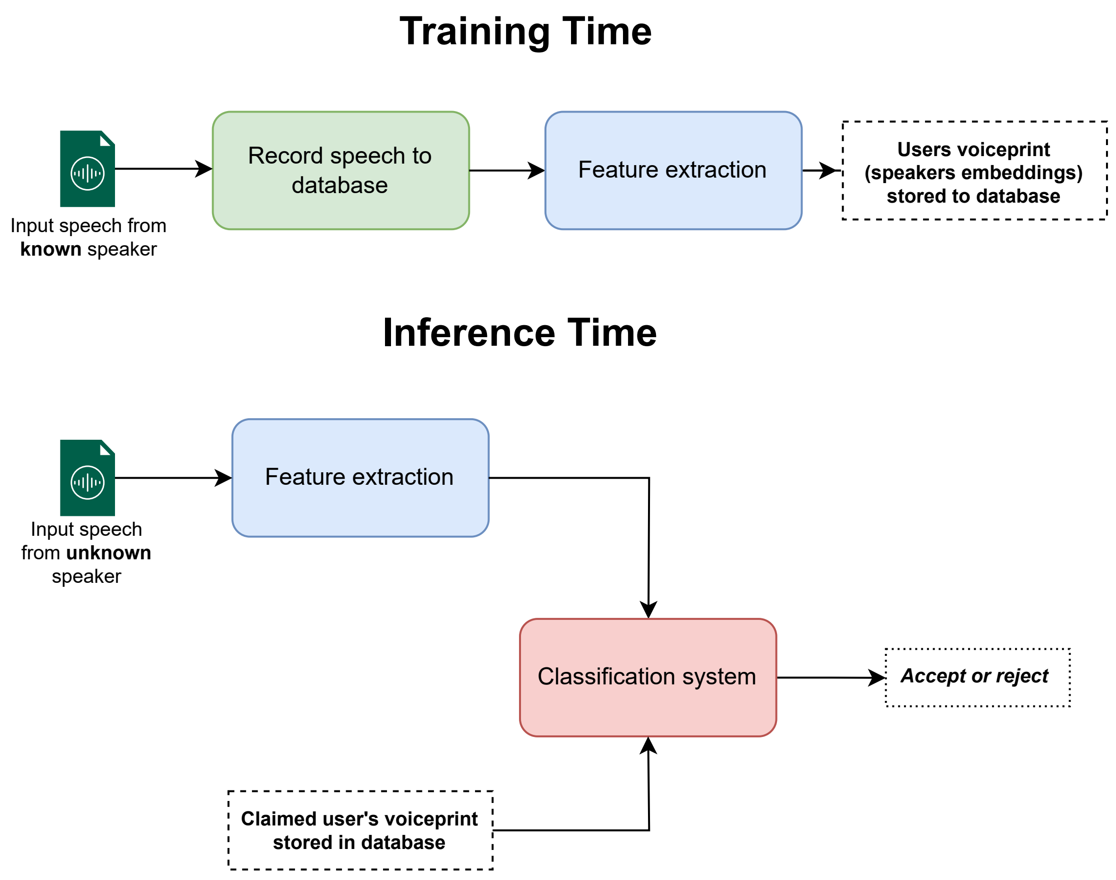

Training Time
Input speech from **known** speaker
Record speech to database
Feature extraction
Users voiceprint (speakers embeddings) stored to database
Inference Time
Input speech from **unknown** speaker
Feature extraction
Classification system
Accept or reject
Claimed user's voiceprint stored in database

Figure 2.3: General speaker verification system: training and test phases, in our case the feature extraction blocks is a DNN network and the classification system is implemented with Generalized end-to-end loss.

2.1.3 Speaker verification losses: triplet-loss, tuple end-to-end-loss, generalized-end-to-end-loss

As shown in 2.3 in order, to perform speaker verification we have different possible choices for feature extractor and classification system. While for the feature extractor part X-vectors are the most refined and latest technique. For classification system the choices are less obviously, for example, we can simply model an FC layer to emit the probability distribution over the two possible outcomes or we can use more fine technique like contrastive learning [18] to learn from positive and negative samples, here three possible losses exploiting contrastive learning are described:

1. **Triplet loss** [19]: During train inputs must be provided in couple. A list of couple is needed in advance. The loss minimize the distance between couples of same speaker (Positive pair) and maximize the distance between couple of different speaker (Negative Pair)

$$L = D(e_i, e_i^p) - D(e_i, e_i^n)$$

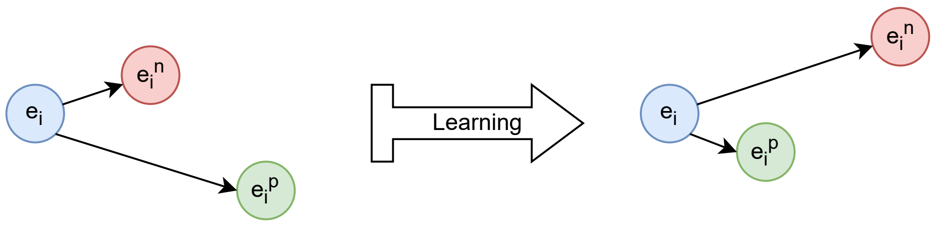


Figure 2.4: Triplet loss, formula and learning process representation.

   (a) *Pros:* simple and correctly model the embedding space.

   (b) *Cons:* since in speaker verification setting, often, more then a single reference speech for each speaker is stored, we can take advantage from having more samples by averaging them. However this loss do not simulate this behaviour. Not model the averaging process in which a user collect different utterance to generate different embedding and average them to get the final embedding.

2. **Tuple end-to-end loss (TE2E)** [17]: Generalize the triplet loss by computing an averaged speaker representation spk, and exploiting this by computing cosine similarity with the input $S(X, spk)$

$$l_{e2e} = -logP(target)$$

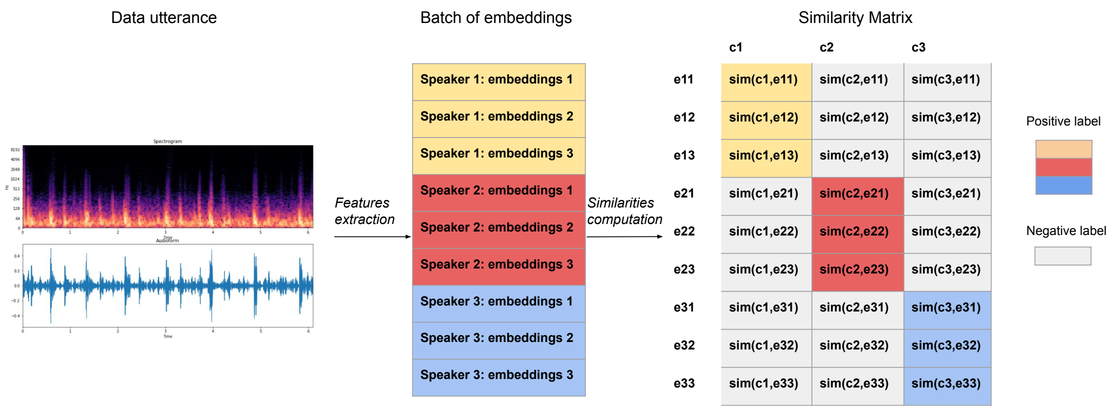
$$target \in \{accept, reject\}$$

$$p(accept) = (1 + e^{wS(X,spk)b})^1,\ p(reject) = 1p(accept)$$

(a) *Pros:* the loss is designed to simulate the averaging behaviour. In each optimization step we have all utterances involved in a verification decision (unlike triplet).

(b) *Cons:* most inputs are easy since couple between averaged representation and positive or negative sample are constructed from the training set randomly.

3. **Generalized end-to-end loss (GE2E)** [8]: The major difference between TE2E and GE2E is as follows: TE2E's similarity is a scalar value that defines the similarity between embedding vector e_j and a single tuple centroid c_k. GE2E builds a similarity matrix that defines the similarities between each e_{ji} and all centroids c_k.



	c1	c2	c3
e11	sim(c1,e11)	sim(c2,e11)	sim(c3,e11)
e12	sim(c1,e12)	sim(c2,e12)	sim(c3,e12)
e13	sim(c1,e13)	sim(c2,e13)	sim(c3,e13)
e21	sim(c1,e21)	sim(c2,e21)	sim(c3,e21)
e22	sim(c1,e22)	sim(c2,e22)	sim(c3,e22)
e23	sim(c1,e23)	sim(c2,e23)	sim(c3,e23)
e31	sim(c1,e31)	sim(c2,e31)	sim(c3,e31)
e32	sim(c1,e32)	sim(c2,e32)	sim(c3,e32)
e33	sim(c1,e33)	sim(c2,e33)	sim(c3,e33)

Figure 2.5: Generalized end to end loss: similarity matrix construction, different colors indicate utterances/embeddings from different speaker.

During training select from similarities matrix distances beetween e_{ji} and true speaker centroid c_j and the closest false positive centroid c_k. Two possibile loss can be used:

(a) *Softmax, text-independent:*

$$L(e_{ji}) = -S_{ji,j} + log \sum_{k=1}^{N} e^{S_{ji,k}}$$

This loss function means that we push each embedding vector close to its centroid and pull it away from all other centroids.

(b) *Contrast, text-dependent:*

$$L(e_{ji}) = 1 - \sigma(S_{ji}) + \max_{1 \leq k \leq N} {}_{k \neq j} \sigma(S_{ji,k})$$

$$where\, \sigma(x) = 1/(1 + e^{x})\, is\, the\, sigmoid\, function.$$

The contrast loss is defined on positive pairs and most aggressive negative pairs.

One aspect to consider when calculating distances between utterances belonging to the same speaker is as follows: the centroid of a given speaker is calculated by removing the point for which the distance is being calculated from the set of utterances. In other words, if a speaker has, for example, three utterances (u1,u2,u3), we will calculate the distances between each point and each centroid calculated by removing that specific point.

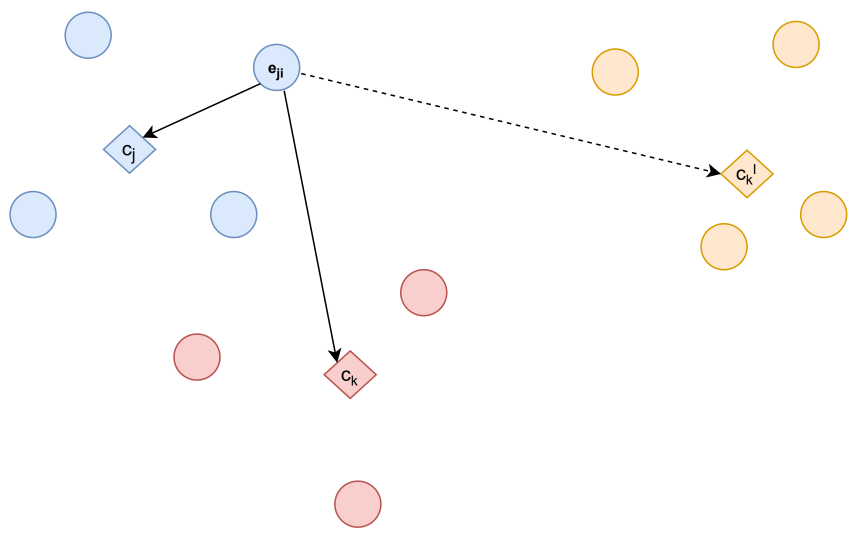


Figure 2.6: Generalized end to end loss: minimize same-speaker embeddings distances and maximize different-speaker embeddings distances. It is important to note that the centroid C_j is computed without considering the point e_j for which we are calculating distances.

2.2 Text To Speech Synthesis (TTS)

Text-to-speech synthesis, the process of converting written text into spoken words, has been a subject of fascination and exploration for many years. It has captivated the imagination of people through its portrayal in science fiction films, often featuring talking computers and advanced speech synthesis systems. One notable example is the computer HAL in the film 2001: A Space Odyssey depicted in 2.7, whose calm and nearly human-like voice conveyed a sense of genuine intelligence. While science fiction movies have sparked our imagination and influenced our perception of the capabilities of speech synthesis, the actual development of perfect synthetic speech has proven to be a complex and challenging task in reality.

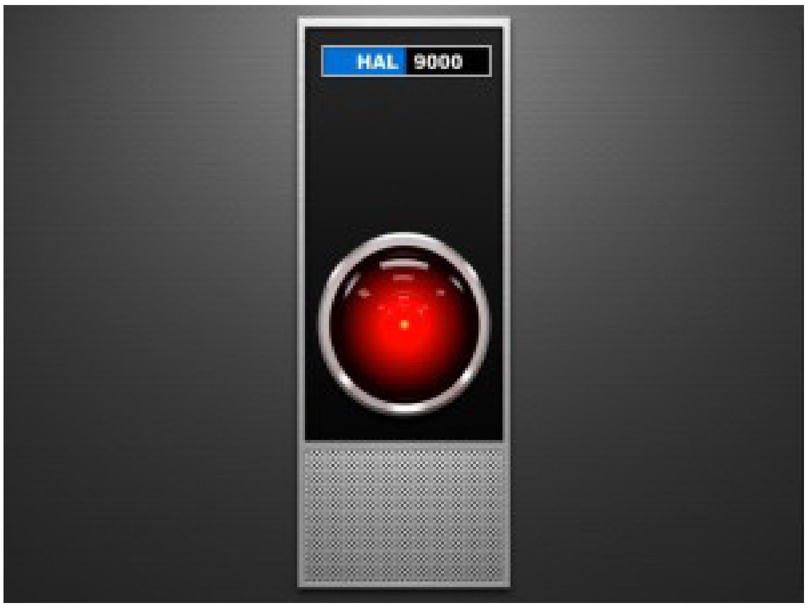


Figure 2.7: The computer HAL in the film *2001: A Space Odyssey*. Despite the influence of science fiction, genuine efforts have been made to predict the future sound of synthetic voices. HAL aimed to provide a realistic portrayal of how computer-generated voices might sound.

Throughout the years, extensive research and investigation have been conducted in the field of text-to-speech synthesis [20], leading to the development of different techniques. These techniques encompass a range of approaches aimed at achieving high-quality and natural-sounding artificial speech.

2.2.1 Concatenative speech synthesis (CSS)

Concatenative speech synthesis [21] is a technique that generates speech by combining pre-recorded speech segments. Unlike recording individual words, CSS uses smaller sub-word units, such as phones, to allow for greater flexibility and scalability. The process involves converting input text into a target specification, selecting the appropriate units from a database based on cost functions that measure the match between the target and candidate units, and concatenating the selected units to form the desired speech. Unit selection is achieved through minimization of two cost functions:

- **Target cost** $C^t(u_i, t_i)$: It describes the mismatch between the target speech unit specification t_i and a candidate unit u_i from the database.

- **Concatenation cost** $C^c(u_{i-1}, u_i)$: This cost measures the mismatch (e.g., acoustic or perceptual) in the join between the candidate unit u_i and the preceding unit u_{i-1}.

An ideal solution aims to find the target units according to the specification without introducing acoustic mismatches at the edges of concatenated units. The target specification includes characteristics such as the identity of target and context phones, pitch, duration, and power. Both the target cost and the concatenation cost can be further divided into T and C subcosts, respectively denoted as $C^t_j(u_i, t_i)$ and $C^c_j(u_{i-1}, u_i)$. The total cost for each unit i can then be calculated as:

$$C^t(u_i, t_i) + C^j(u_{i-1}, u_i) = \sum_{j=1}^{T} w^t_j C^t_j(u_i, t_i) + \sum_{j=1}^{C} w^t_j C^c_j(u_{i-1}, u_i)$$

Here, w^t_j and w^p_j represent the relative weights of each subcost for target and concatenation costs, respectively.

CSS training involves determining the weights for the cost functions, which can be achieved through methods like grid search or regression models.

Post-processing techniques can be applied to smoothen transitions between units and improve the overall quality of the synthesized speech. These techniques help address issues related to unnatural or abrupt changes in the audio output.

Nevertheless, the speech generated by these systems often lacks the clarity and naturalness typically found in human speech, resulting in a muffled and unnatural audio output.

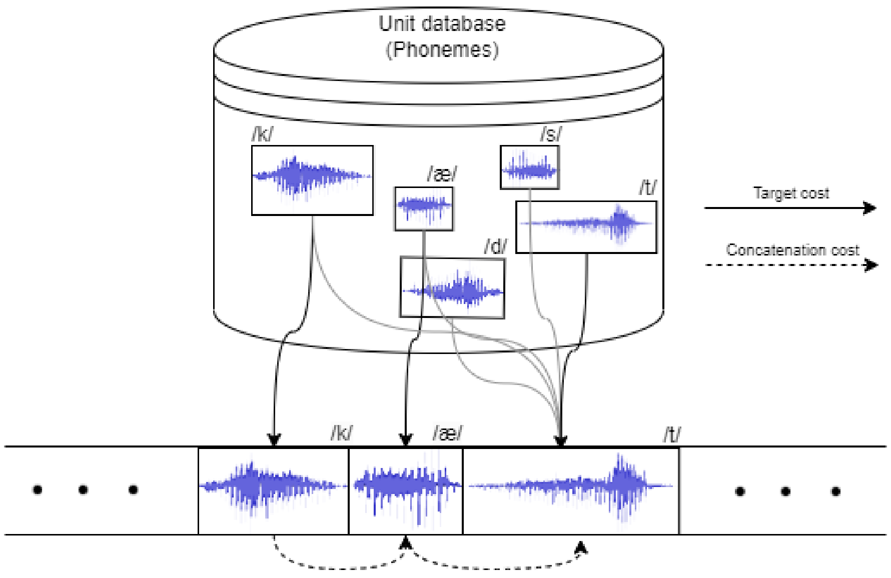


Figure 2.8: Overview of general unit-selection scheme, by using the target cost and the concatenation cost, speech units are selected from the whole speech database, and concatenated in run-time to generate the word "cat" /k/ - /æ/ - /t/.

2.2.2 Gaussian Mixture Models (GMM)

Gaussian Mixture Models (GMMs) have emerged as a powerful technique in speech synthesis [21]. They provide an effective means of capturing the intricate structures inherent in speech signals, enabling more precise modeling and synthesis. In speech, both voiced and unvoiced signals possess distinct statistical properties, reflecting the vibration of vocal cords and the turbulent airflow, respectively. Moreover, within each type of signal, various groups of utterances exhibit their own unique characteristics. GMMs tackle this diversity by assuming that the speech signal comprises multiple classes, each represented by its individual statistical model.

Typically, a GMM represents each class using a multivariate Gaussian distribution. Given a vector $\mathbf{x}$, the distribution is defined as:

$$f(\mathbf{x};\mu,\Sigma) = \frac{1}{\sqrt{(2\pi)^N||\Sigma||}} \exp\left[-\frac{1}{2}(\mathbf{x}-\mu)^T\Sigma^{-1}(\mathbf{x}-\mu)\right]$$

Here, μ denotes the mean and Σ represents the covariance of the process, both with dimensions N.

Let's consider a scenario where the speech signal consists of K classes, each having its specific mean μ_k and covariance Σ_k.

The Gaussian mixture model is then defined as:

$$f(\mathbf{x}) = \sum_{k=1}^{K} \alpha_k f(\mathbf{x}; \mu_k, \Sigma_k)$$

The weights α_k are assigned based on the occurrence frequency of each class, and their summation is unity ($\sum_{k=1}^{K} \alpha_k = 1$).
The utilization of GMMs in speech synthesis offers several advantages. By accurately modeling the diverse structures present in speech signals, GMM-based synthesis methods can generate speech output that is more natural and realistic. The training process involves estimating the parameters of the GMM using speech data. Common techniques, such as the Expectation-Maximization algorithm, are employed for this purpose. However, it is important to note that GMM-based synthesis methods also have limitations. They can be computationally complex, demanding substantial processing power. Furthermore, a considerable amount of training data is typically required to achieve satisfactory results.

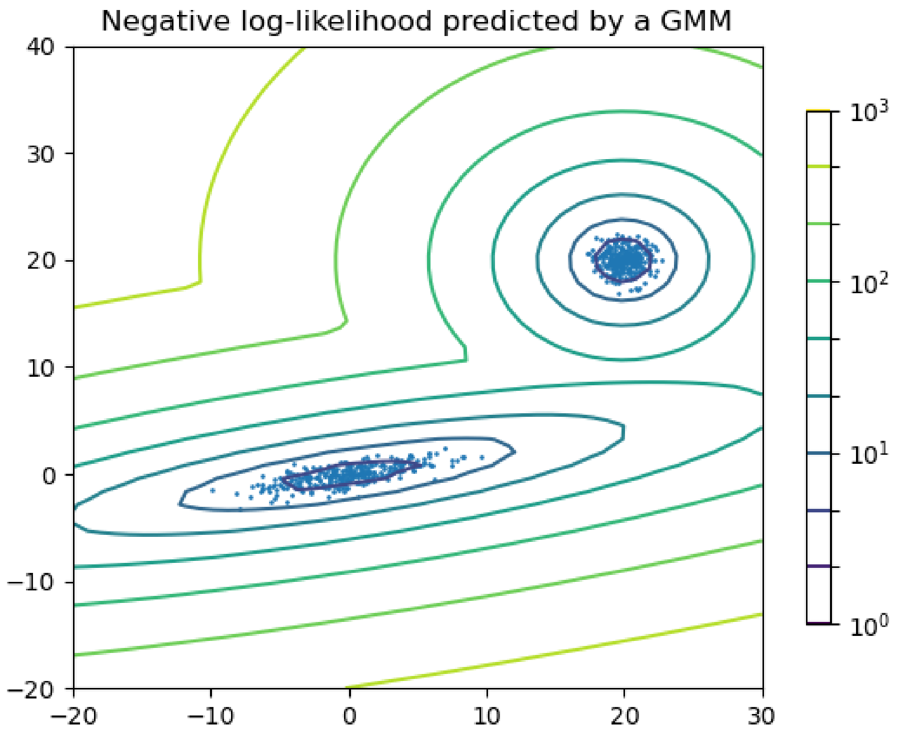


Figure 2.9: A visualization of a two-component Gaussian mixture model, showcasing the data points and equi-probability surfaces. Image source: [22]

2.2.3 Deep Learning methods

The field of text-to-speech synthesis has undergone a revolutionary transformation through the application of deep learning methods. By harnessing the capabilities of neural networks, these methods have enabled the generation of speech with exceptional quality. Among the various approaches within deep learning-based speech synthesis, two notable techniques stand out: Statistical Parametric Speech Synthesis (SPSS) with Deep Neural Networks (DNNs) and the latest advancements in fully deep learning methods.

- **Statistical parametric speech synthesis (SPSS) with DNNs** [23]: In SPSS, DNN-based acoustic models are used to capture the relationship between input features and acoustic features. DNNs have gained popularity due to their ability to accurately capture this connection compared to traditional models like Gaussian mixture models. Various training algorithms, such as the minimum generation error (MGE) criterion, are employed to train these models. However, despite extensive research on training techniques, the speech parameters generated by these models often exhibit excessive smoothing, resulting in synthesized speech of inferior quality compared to natural speech. This limitation of excessive smoothing needs to be addressed to improve the quality of the synthesized speech.

- **Fully deep learning methods** [11][12]: Fully deep learning methods, such as the sequence-to-sequence architecture used in Tacotron or FastSpeech, simplify the conventional speech synthesis pipeline by training a single neural network solely on data to generate magnitude spectrograms from a sequence of characters. This eliminates the need for separate production of linguistic and acoustic features. However, it is important to note that fully deep learning methods have their own limitations. Firstly, these methods often require a large amount of data for training the models effectively. Acquiring such a substantial amount of data can be challenging and may limit the accessibility of these techniques. Secondly, the controllability of the synthesized speech is often limited, making it difficult to fine-tune specific aspects of the generated speech according to user preferences. These limitations of data requirements and controllability should be considered when implementing fully deep learning methods in text-to-speech synthesis.

  In basic implementations, the Griffin-Lim algorithm [24] is commonly used as a vocoder, which converts the spectrogram into the time domain. However, more advanced applications utilize generative networks like HiFiGAN [25]. In this thesis, the pretrained Universal HiFiGAN is employed as the vocoder to enhance the quality of the synthesized speech.

By addressing the limitation of excessive smoothing in SPSS with DNNs and discussing the requirements of a large amount of data and limited controllability in fully deep learning methods, the subsection provides a more comprehensive understanding of the advantages and challenges associated with these techniques in text-to-speech synthesis.

2.3 Zero Shot Voice Cloning System

Voice cloning is a remarkable technology that has gained significant attention in recent years. It enables the synthesis of speech in a target voice, allowing for the creation of highly realistic and personalized speech. Voice cloning leverages the advancements made in the field of Text-to-Speech (TTS) to achieve this capability and is built on top of existing TTS architectures.

There are two main approaches to voice cloning, both of which rely on TTS models to generate speech in the target voice.

2.3.1 Multi-Speaker TTS and Speaker Adaptation

One approach to achieving voice cloning involves training a TTS model on a single-speaker dataset, specifically targeting the voice to be cloned. However, this approach typically requires a significant amount of data, and a separate model must be trained for each speaker, which can be time-consuming and resource-intensive.

To address this limitation, researchers have explored the capabilities of diverse TTS architectures to scale upon multi-speaker datasets [26] [12]. By training a single model on a multi-speaker dataset, it becomes possible to replicate different known voices by specifying a unique speaker identifier as input that is embedded in a naife way from embedding layers of the net. This approach, known as multi-speaker TTS, allows the model to learn to generate speech in different voices. However, one challenge remains: integrating new voices into the system.

To overcome this challenge, speaker adaptation approaches have been developed for voice cloning. These approaches involve fine-tuning a pre-trained multi-speaker model with a small number of samples from an unseen speaker [27]. The finetuning can be done in different ways based by finetuning only the embedding layers or the whole net.

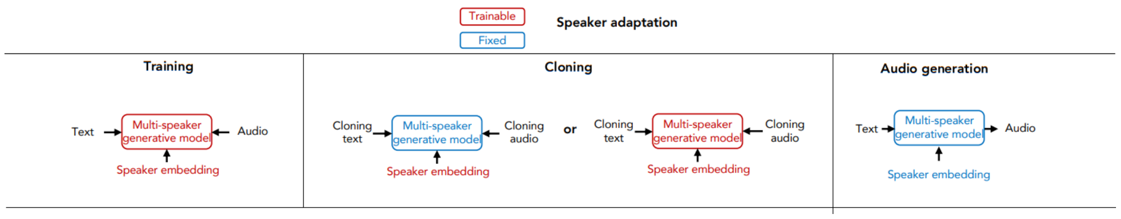


Figure 2.10: Illustration of speaker adaptation training, cloning and generation. The cloning phase, i.e. a fine-tuning, can be done by "freezing" the generative model and train the speaker embedding layers or by training the whole model. [27]

By leveraging a pre-existing model's knowledge of various voices, speaker adaptation enables the cloning of new voices with a limited amount of data. This significantly reduces the training burden and allows for more efficient voice cloning.

The combination of multi-speaker TTS and speaker adaptation techniques presents a powerful solution for voice cloning. It enables the replication of known voices using a single model and facilitates the integration of new voices through adaptation but for each newwer voices a model must be finetuned.

To effectively utilize the fine-tuning approach, it is crucial to leverage small, well-constructed datasets. In Section 4, a simple case study that focuses on this aspect is provided, demonstrating the efficacy of using limited data for speaker adaptation in voice cloning.

2.3.2 Zero-shot learning approach

Zero-shot learning is a machine learning paradigm that addresses the challenge of learning to recognize and classify objects or concepts that were not present during the training phase [28]. Traditional machine learning approaches require labeled data for each class during training, making it difficult to generalize to unseen classes. In contrast, zero-shot learning aims to bridge this gap by leveraging auxiliary information, such as semantic embeddings or attribute descriptions, to enable recognition and classification of novel classes.

Zero-shot learning has gained significant attention in various domains. In the field of computer vision, zero-shot learning (ZSL) is primarily employed in image classification tasks. It involves utilizing information about a novel class, typically provided as a textual description or a set of attributes, to establish the connection between the class description and its visual representation. For instance, as shown in 2.12, if someone is familiar with horses but has never seen a zebra, they can still recognize a zebra by understanding that it shares similarities with a horse, such as having a similar body structure but with black and white stripes. The underlying assumption in zero-shot learning is the presence of a semantic relationship between the seen and unseen classes.

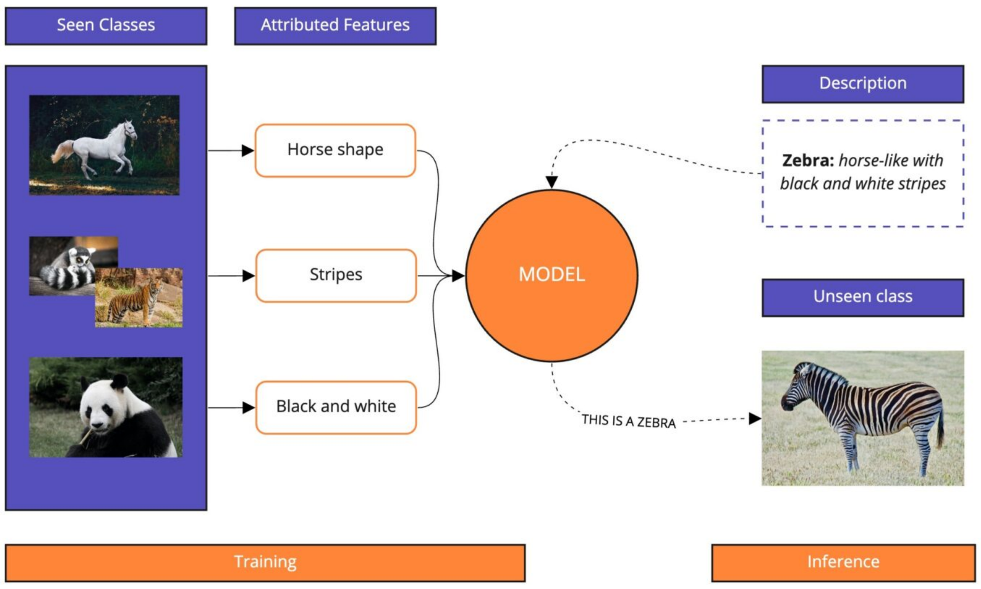


Figure 2.11: Example of Zero Shot Learning in Computer Vision. On the left, training data consisting of images, classes, and attribute features for each. On the right, during inference, the model extrapolates similarities from known attribute features and a textual description to classify the image. Source [29].

In speech processing and Voice Cloning in particular, this approach allows for the synthesis of speech in a target voice without requiring any specific training data from that particular speaker. Instead, as shown in 2.12 as auxiliary information it employ the learned representations of the speaker embedding space from a Speaker Recognition model, i.e. the speaker embeddings. These can be extracted from any also unseen voice and are used in input to generate speech that closely matches the desired voice.

By leveraging pre-existing knowledge about speakers and their unique characteristics, zero-shot learning enables voice cloning for new speakers without the need for extensive data collection and training. This flexibility and adaptability make it an appealing approach for applications that require rapid integration of new voices or personalized speech synthesis.

In the context of voice cloning, zero-shot learning empowers the system to generate speech in novel and unseen voices by leveraging speaker embeddings.

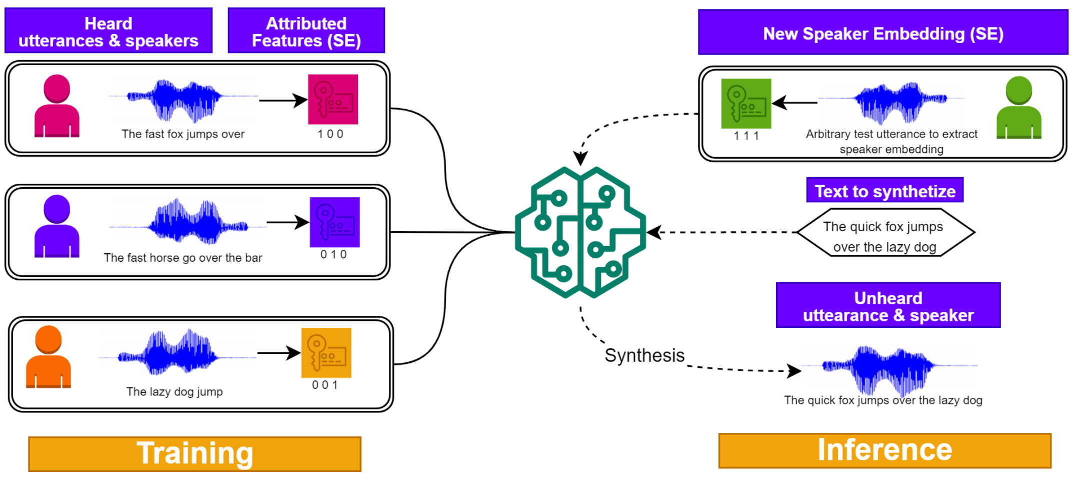

Heard utterances & speakers
Attributed Features (SE)
The fast fox jumps over
1 0 0
The fast horse go over the bar
0 1 0
The lazy dog jump
0 0 1
Training
New Speaker Embedding (SE)
1 1 1
Arbitrary test utterance to extract speaker embedding
Text to synthetize
The quick fox jumps over the lazy dog
Unheard uttearance & speaker
Synthesis
The quick fox jumps over the lazy dog
Inference

Figure 2.12: Example of Zero Shot Learning in Voice Cloning. On the left, training data consisting of speech audio, text of the speech audio and speaker embedding of the speech audio for each. On the right, during inference, the model extrapolates similarities from known speakers embedding and a the new extracted to synthesize the input text in an speech audio within the desired voice.

Chapter 3

Models architectures

"Deep learning model architectures are the majestic tapestries woven from intricate layers of neurons, unraveling the enigmas of data and painting a portrait of intelligence."
ChatGPT

Advancements in model architectures have revolutionized the field of speech processing, driving progress in speaker verification, text-to-speech, and voice cloning. This chapter provides an overview of several prominent architectures that have made significant contributions in these areas.

For speaker verification, two noteworthy architectures are discussed. WAVLM, developed by Microsoft, utilizes a pretrained backbone based on transformers, allowing it to capture speech content and adapt to various tasks. ECAPA-TDNN, developed by SpeechBrain, incorporates enhancements from face verification and computer vision, improving its performance in speaker-related tasks.

In text-to-speech, the Tacotron series from Google, based on recurrent networks, generates mel spectrograms from text using an encoder-decoder structure with attention mechanisms. FastSpeech, from Microsoft, employs transformers for parallel processing and efficient training, offering enhanced controllability and voice quality.

Additionally, the chapter explores the Zero Shot Voice Cloning System, which combines three independently trained models: the Speaker Embedder, the Synthesizer, and the Vocoder. This system enables high-quality speech generation in different voices by conditioning the synthesis process on speaker characteristics.

By understanding these model architectures, we gain valuable insights into their unique components and design principles, contributing to advancements in speech processing.

3.1 Speaker verification

3.1.1 WAVLM

The WAVLM architecture, proposed in [7] by Microsoft, addresses the challenge of learning universal representations for various speech tasks. The model includes a pretrained backbone designed to handle full-stack downstream speech tasks by jointly learning masked speech prediction and denoising. This approach enables WAVLM to capture speech content through masked speech prediction. WAVLM leverages a large amount of unlabeled speech data to learn universal speech representations, allowing it to effectively adapt to different speech processing tasks.

The architecture utilizes a framework that combines masked speech prediction and denoising during the pre-training stage. By predicting pseudo-labels of the original speech on masked regions, WAVLM learns not only Automatic Speech Recognition (ASR) information but also knowledge of non-ASR tasks. For instance, predicting pseudo-labels on overlapped speech enhances the model's capability for diarization and separation tasks. Additionally, the model captures speaker identity information and speech enhancement capabilities through predicting pseudo-labels on simulated noisy speech.

To customize the architecture for specific tasks, a task-specific head can be added on top of the pretrained backbone. For the speaker verification task, the x-vector head is employed, enabling fine-tuning and optimization of the model's performance in speaker verification.

Backbone

The backbone architecture plays a critical role in model architectures by providing the foundational structure for processing and extracting meaningful features from the input data. In the context of speech processing, the backbone architecture is responsible for capturing the relevant information and sequence ordering present in the input speech.

The backbone architecture depicted in Figure 3.5 combines a convolutional feature encoder with a transformer structure that incorporates gated relative position bias. This combination enhances the model's ability to capture the sequence ordering of the input speech, which is crucial for tasks such as speaker verification and text-to-speech synthesis.

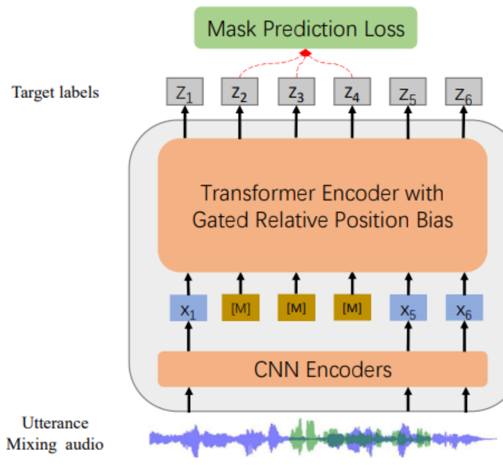

Mask Prediction Loss
Target labels
Z_1 Z_2 Z_3 Z_4 Z_5 Z_6
Transformer Encoder with Gated Relative Position Bias
X_1 [M] [M] [M] X_5 X_6
CNN Encoders
Utterance Mixing audio

Figure 3.1: WAVLM backbone architecture. Source from [7]

- **Convolutional Encoder:** the convolutional encoder comprises seven blocks of temporal convolution, followed by layer normalization and a GELU activation layer. The temporal convolutions have 512 channels with strides (5,2,2,2,2,2,2) and kernel widths (10,3,3,3,3,2,2), resulting in each output representing about 25ms of audio strode by 20ms. The convolutional output representation, denoted as x, is masked and used as the input for the Transformer.

- **Transformer encoder with Gated Relative Position Bias:** the Transformer incorporates a convolution-based relative position embedding layer. It encodes the bias based on the offset between the "key" and "query" in the Transformer self-attention mechanism. This enhancement improves the model's performance in ASR tasks while maintaining a similar number of parameters and training speed. Unlike the convolutional relative position embedding, the gates in grep allow for adaptive adjustment of the relative position bias based on the current speech content.

The WAVLM backbone was developed in different versions, with difference in the scale in terms of the number of parameters. The base version, WAVLM+, has 94.70M parameters, while WAVLM-Large+ is the up-scaled version with 316.62M parameters. Figure 3.2 visualizes the weights of the two versions after training on different benchmarks.

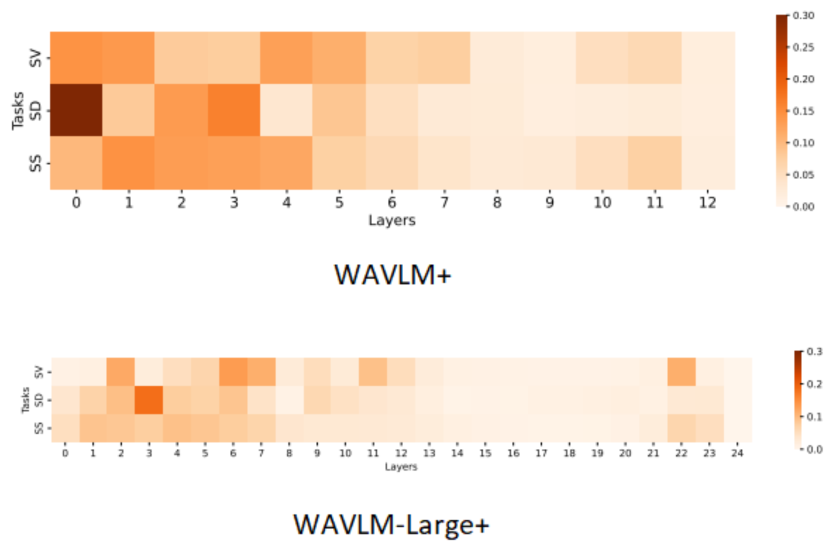


Figure 3.2: The weights of the model on various tasks, including Speaker Verification (SV), Speech Diarization (SD), and Speech Separation (SS). Layer 0 corresponding to the input of the first Transformer layer. The x-axis represents different tasks while the y-axis the weight on different layers.

Speaker Verification Head

in this work, the backbone of WAVLM is used with an X-vector head to achieve the goal of speaker verification tasks. It comprises two main blocks, the frame-level and the segment level netwroks integrated with a statistics pooling layer. The architecture depicted in 3.3 effectively partitions the network to handle specific tasks at different levels of speech processing.

- **Frame-level network:** the frame-level network aims to transform the acoustic features $x_t^{(k)}$ into speaker-discriminant features $F_t^{(k)}$. Given the input sequence $X^{(k)} = [x_1^{(k)}, ..., x_{T_k}^{(k)}]$ from utterance k with T_k frames, a variant of the convolutional neural network (CNN) called the time-delay neural network (TDNN) is utilized. The TDNN architecture incorporates time-delayed connections, allowing the model to capture temporal dependencies over multiple frames. Each layer of the TDNN takes the sliced outputs of the previous layer as input.

- **Statistics Pooling layer:** following the frame-level network, a statistics pooling layer is employed to aggregate the speaker features F_t. This layer calculates the mean and standard deviation of the features across time frames and concatenates them, resulting in a segment-level representation denoted as L. The statistics pooling layer captures the global statistics of the speaker features, enabling the model to capture speaker characteristics over longer time spans.

- **Segment-level network:** After the statistics pooling layer, the segment-level network consists of fully-connected layers with parameters θ_l. These layers process the aggregated features L and learn higher-level representations for speaker verification.

  The output of the segment-level network is then passed through a softmax layer to obtain the posterior probabilities $P(i|k)$, representing the probability of speaker i given the utterance k. The softmax activation function normalizes the outputs, ensuring they sum up to 1. The segment-level network with parameters θ_l is denoted as $F(l|\theta_l)$.

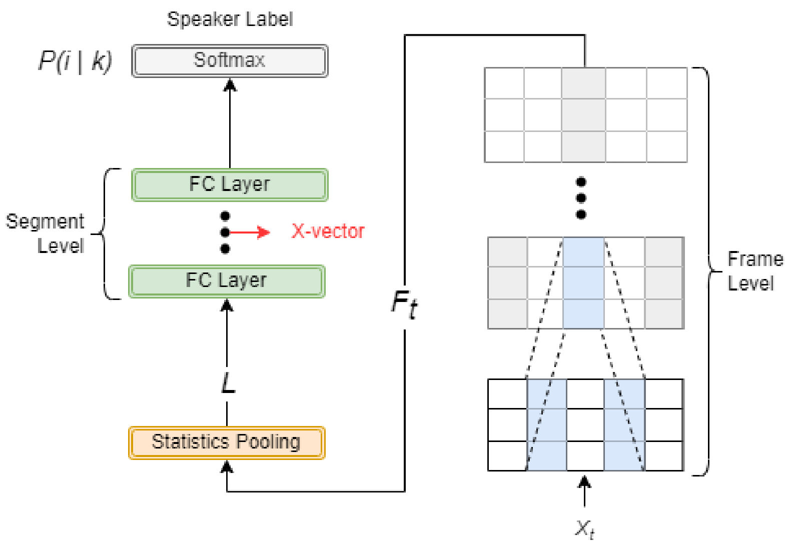


Figure 3.3: The x-vector architecture combines frame-level processing, statistics pooling, and segment-level sub-networks to handle specific tasks at different levels of speech processing.

3.1.2 ECAPA-TDNN

ECAPA-TDNN architecture, developed from SpeechBrain [6], is an improved version of the x-vector architecture depicted in 3.3.

The ECAPA-TDNN architecture incorporates several enhancements based on advancements in face verification and computer vision, resulting in improved performance, the netwrok architecture is depicted in 3.4.

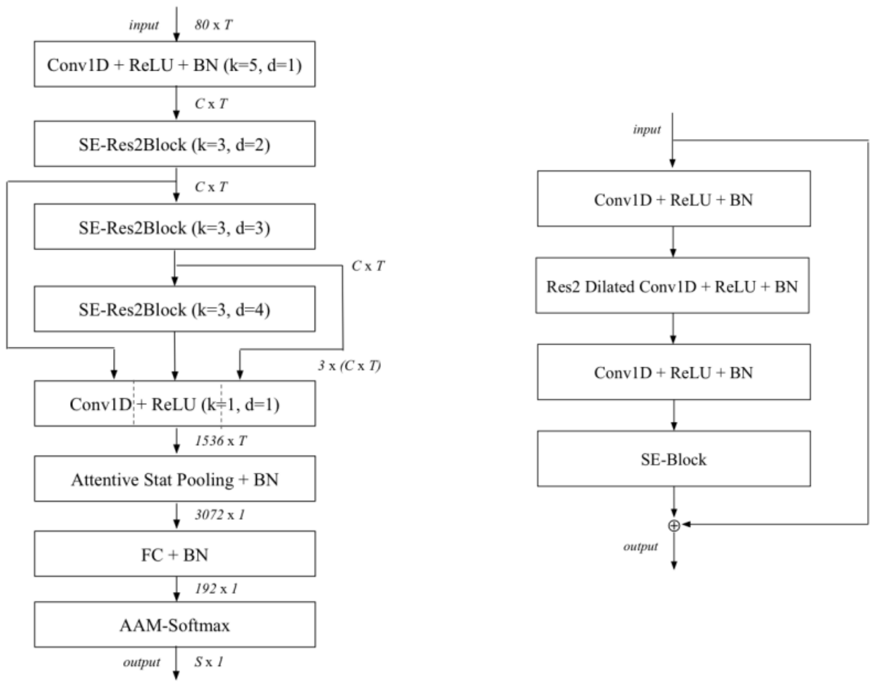


Figure 3.4: The network topology of the ECAPA-TDNN architecture is shown on the left. The Conv1D layers and SE-Res2Blocks are characterized by the kernel size (k) and dilation spacing (d). The channel (C) and temporal (T) dimensions represent the intermediate feature-maps, while S denotes the number of training speakers. On the right, the SE-Res2Block of the ECAPA-TDNN architecture is depicted. The standard Conv1D layers have a kernel size of 1. The central Res2Net Conv1D with a scale dimension (s = 8) enhances the temporal context through the kernel size (k) and dilation spacing (d). Figure adapted from [6].

Channel- and context-dependent statistics pooling

This enhancement allows the network to focus on speaker characteristics specific to different channels, enabling a more fine-grained analysis of speaker-related properties, such as variations in vowels and consonants.

The channel-dependent self-attention mechanism is implemented by calculating self-attention scores for each frame and channel. These scores represent the importance of each frame within the specific channel. Weighted statistics, including the weighted mean vector μ' and weighted standard deviation vector σ', are computed based on the channel-dependent self-attention scores. The final output of the pooling layer incorporates the channel-specific information by concatenating the vectors of the weighted mean and weighted standard deviation. This enhancement improves the x-vector architecture's ability to capture and utilize channel-specific speaker characteristics, enhancing its performance in speaker-related tasks.

1-Dimensional Squeeze-Excitation Res2Blocks

The original x-vector system has a limited temporal context of 15 frames for its frame layers. To address this limitation, 1-dimensional Squeeze-Excitation (SE) blocks can be used, these are a successful computer vision approach that models global channel interdependencies.

An SE block consists of two components: the squeeze operation and the excitation operation. In the squeeze operation, a descriptor for each channel is obtained by calculating the mean vector of the frame-level features across the time domain. This descriptor captures global properties of the recording. The excitation operation then calculates weights for each channel based on the descriptors obtained from the squeeze operation.

To integrate the 1-dimensional SE-block into the x-vector architecture, residual connections are leveraged. The SE-Res2Block incorporates dilated convolutions with preceding and succeeding dense layers, each with a context of 1 frame. The first dense layer reduces the feature dimension, while the second dense layer restores it to the original dimension. The SE-block scales each channel, and the entire unit is connected through a skip connection.

By incorporating traditional ResBlocks, the x-vector architecture benefits from advancements in popular computer vision architectures. For example, the integration of the Res2Net module allows the central convolutional layer to process multi-scale features using hierarchical residual-like connections. This integration enhances performance while minimizing the number of model parameters.

Multi-layer feature aggregation and summation

In the original x-vector system, only the feature map of the last frame-layer is used for calculating pooled statistics. However, evidence suggests that incorporating shallower feature maps can contribute to more robust speaker embeddings. To address this, the proposed system incorporates multi-layer feature aggregation.

In this approach, for each frame, the output feature maps of all the SE-Res2Blocks are concatenated. This concatenated information is then processed by a dense layer for Multi-layer Feature Aggregation (MFA), allowing the utilization of information from both shallow and deep feature maps. This enhances the robustness of the speaker embeddings by capturing comprehensive information across multiple layers.

Additionally, an alternative approach leverages multi-layer information by summing the outputs of all preceding SE-Res2Blocks and the initial convolutional layer. By incorporating the information from multiple layers into each frame layer block, the model benefits from the hierarchical nature of the TDNN architecture. This summation approach effectively restrains the model parameter count compared to concatenation.

The integration of multi-layer feature aggregation and summation enables the system to leverage both complex and shallow feature maps, leading to more comprehensive and robust speaker embeddings.

The experimental results on benchmark datasets highlights the effectiveness of the proposed enhancements in capturing speaker-specific information and achieving state-of-the-art results in speaker recognition.

Overall, the ECAPA-TDNN architecture represents a significant advancement in speaker recognition technology. Its incorporation of Res2Net modules, SE blocks, hierarchical feature learning, and channel-dependent frame attention extends the capabilities of the x-vector architecture and contributes to the development of more robust and accurate voice cloning systems.

3.2 Text to Speech Synthesis (TTS)

3.2.1 Tacotron 1

Among the notable architectures in the field of text-to-speech synthesis, Tacotron stands out as a pioneering and widely recognized model developed by Google. The Tacotron series represents one of the early attempts to create an end-to-end system for generating speech.

Tacotron 1 [30] is a seq2seq model with attention, specifically designed for text-to-speech synthesis. Its architecture comprises three essential components: an encoder, an attention-based decoder, and a post-processing network.

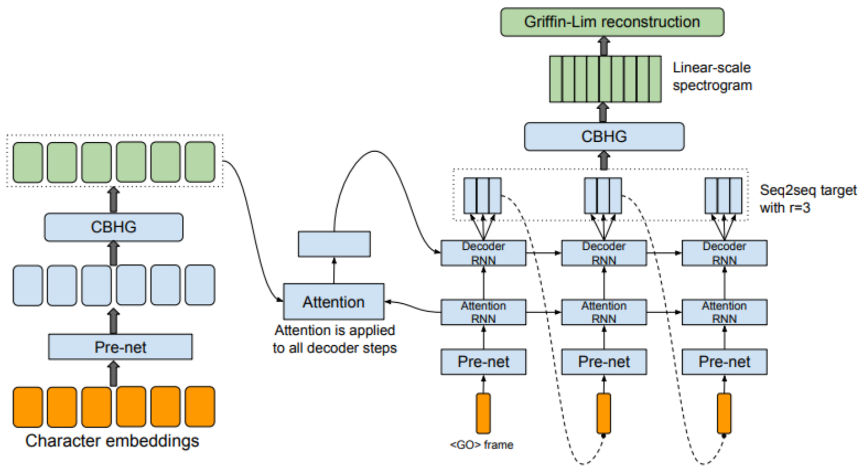


Figure 3.5: The architecture of the model. It receives characters as input and generates the corresponding raw spectrogram. This spectrogram is then used as input for the Griffin-Lim reconstruction algorithm, which synthesizes the speech. Source from [30]

Encoder

The encoder's role is to extract sequential representations of text. It takes a character sequence as input, where each character is represented as a one-hot vector. The input embeddings go through a pre-net, which includes a bottleneck layer with dropout. The pre-net outputs are then processed by a CBHG (Convolutional Banks + Highway Networks + Bidirectional GRU) module. The CBHG module applies 1-D convolutional filters to the input sequence, modeling local and contextual information. The convolution outputs are stacked and max-pooled along time to increase local invariances. The processed sequence is further passed through fixed-width 1-D convolutions, with the outputs added back to the original input sequence via residual connections. A multi-layer highway network extracts high-level features, and a bidirectional GRU RNN is stacked on top to capture sequential features from both forward and backward contexts.

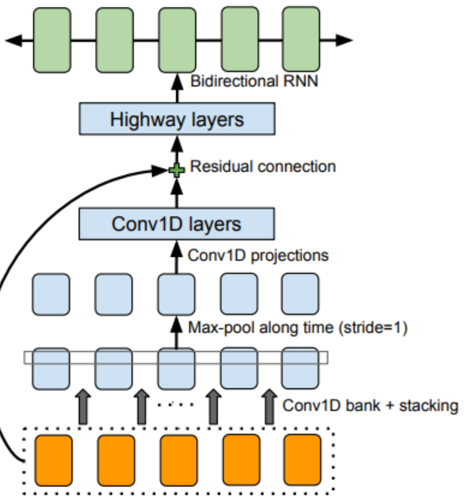


Figure 3.6: The CBHG module, which combines a 1-D convolution bank, a highway network, and a bidirectional GRU. Source from [30]

Attention-based decoder

The attention-based decoder generates the spectrogram frames based on the encoder representations. It uses a content-based tanh attention mechanism, where a stateful recurrent layer produces the attention query at each decoder time step. The context vector and attention RNN cell output are concatenated and fed as input to the decoder RNNs. The decoder consists of a stack of GRUs with vertical residual connections, which speed up convergence. In Tacotron 1, an 80-band mel-scale spectrogram is used as the decoder target, allowing for efficient learning of alignment between the speech signal and text. The decoder predicts multiple non-overlapping output frames at each step to reduce the number of decoder steps, improve convergence speed, and capture correlations between neighboring speech frames.

Post-processing network

The post-processing network converts the predicted spectrogram to a target that can be synthesized into waveforms. It learns to predict spectral magnitude sampled on a linear-frequency scale. A CBHG module is used for the post-processing network, but simpler architectures can also be effective. The post-processing network has the advantage of being able to see the full decoded sequence, incorporating both forward and backward information to correct prediction errors for each frame. The Griffin-Lim algorithm is used for waveform synthesis, converting the predicted spectrogram to waveforms.

The design choices in Tacotron 1 were made to address the challenges of generating high-quality waveforms from input text. The use of an attention mechanism allows the model to focus on different parts of the input text at each decoder step, capturing the alignment between text and speech. The CBHG module in the encoder and post-processing network helps in capturing local and contextual information and refining the predicted spectrogram. These design choices contribute to Tacotron 1's strengths in generating natural-sounding speech but also have limitations in terms of computational complexity and the need for additional post-processing for waveform synthesis.

3.2.2 Tacotron 2

Tacotron 2 [11] builds upon the success of Tacotron 1 and introduces improvements to generate even higher-quality audio. The model consists of two main components: a recurrent characters encoder network and a recurrent decoder for melody spectrogram prediction integrated with a Location Sensitive Attention. Moreover a modified version of WaveNet was used as vocoder. The use of mel-frequency spectrograms is a key element in Tacotron 2's design. These spectrograms are derived from the short-time Fourier transform (STFT) magnitude through a non-linear transform. By applying the mel scale, which emphasizes important lower frequencies for speech intelligibility while reducing the emphasis on less critical high-frequency details, Tacotron 2 achieves a more effective representation for speech synthesis.

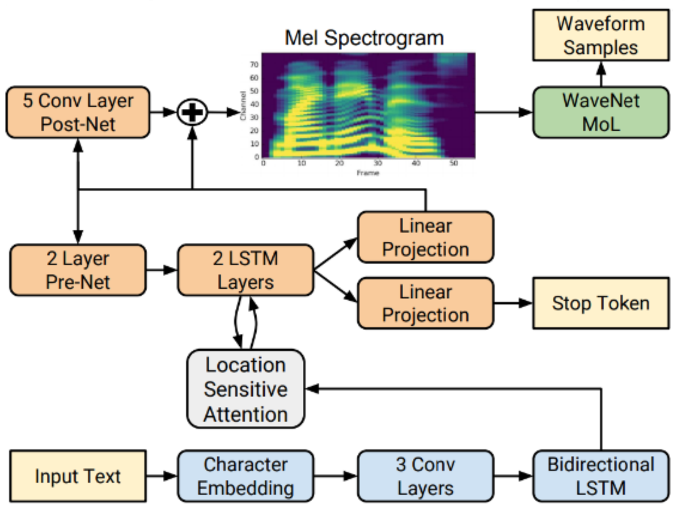


Figure 3.7: Architecture of Tacotron 2: in blue the characters encoder network. In gray the Location Sensitive Attention. In orange the decoder for mel spectrogram prediction, in green the WaveNet model used as vocoder.

Character encoder

The encoder takes the input character sequence and converts it into a hidden feature representation. This representation is then consumed by the decoder to predict the spectrogram. The encoder includes convolutional layers to capture longer-term context in the character sequence, while the decoder utilizes an attention network to summarize the encoded sequence. The attention mechanism ensures consistent progression through the input by incorporating cumulative attention weights from previous decoder time steps.

Location Sensitive Attention

The Location Sensitive Attention is an enhancement to the traditional additive attention mechanism. It introduces the concept of cumulative attention weights from previous decoder time steps as an additional feature. In the additive attention mechanism, the decoder attends to different parts of the input sequence to gather relevant information at each time step. However, in certain cases, the decoder may encounter difficulties in progressing smoothly through the input, resulting in repeated or skipped subsequences. This can lead to unnatural and distorted speech synthesis. To address this issue, Tacotron 2 incorporates the Location Sensitive Attention mechanism. By incorporating cumulative attention weights from previous decoder time steps, the model is encouraged to consistently move forward through the input. This means that the attention mechanism takes into account not only the current attention distribution but also the historical attention weights. This additional feature helps mitigate potential failure modes where subsequences are repeated or ignored by the decoder.

Mel spectrogram decoder

The decoder, functioning as an autoregressive recurrent neural network, generates the mel spectrogram frame by frame. It begins by passing the prediction from the previous time step through a pre-net, which consists of fully connected layers. This pre-net acts as an information bottleneck and plays a crucial role in learning attention. The pre-net output and the attention context vector are concatenated and processed through a stack of uni-directional LSTM layers. The resulting LSTM output, along with the attention context vector, undergoes linear transformation to predict the target spectrogram frame. A post-net, consisting of convolutional layers, predicts a residual that enhances the overall reconstruction.

Tacotron 2 use simpler building blocks compared to Tacotron 1, the motivation was to reduce computational complexity and improve training efficiency. Tacotron 2 uses vanilla LSTM and convolutional layers in the encoder and decoder, respectively, instead of the "CBHG" stacks and GRU recurrent layers used in Tacotron 1. These simplifications make the model more accessible and easier to train while still achieving high-quality audio generation.

Overall, the Tacotron series represents significant advancements in end-to-end text-to-speech synthesis. Tacotron 1's architecture combines seq2seq modeling with attention mechanisms and post-processing techniques to generate high-quality waveforms from input text. Tacotron 2 builds upon these ideas and introduces improvements in architecture and training efficiency, resulting in even higher-quality audio synthesis.

3.2.3 FastSpeech 1

This section, introduce the architecture design of FastSpeech [31].
FastSpeech is a model that aims to generate a target mel-spectrogram sequence in parallel, addressing the challenges of slow inference speed and limited controllability in autoregressive TTS systems.

The overall model architecture of FastSpeech is depicted in Figure 3.8. It consists of several components, with three principal blocks: the Feed-Forward Transformer, and two new blocks introduced in FastSpeech: the Length Regulator block, and the Duration predictor.

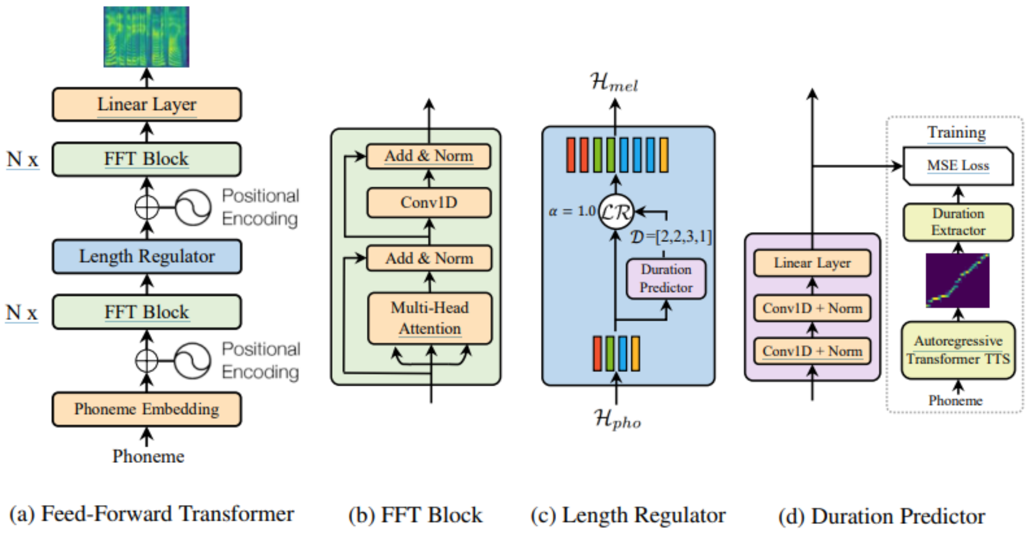


Figure 3.8: Architecture of FastSpeech 1: (a) Overview of the Feed-forawrd Transformer network, consisting of different blocks. (b) Details of the FFT block. (c) The Length Regulator block aims to predict duration of each frame. (d) Details of the Duration Prediction block. Source [31]

Feed-Forward Transformer

FastSpeech utilizes a feed-forward structure called the Feed-Forward Transformer (FFT). This structure combines self-attention from Transformer and 1D convolution to facilitate phoneme-to-mel-spectrogram transformation. Multiple FFT blocks are stacked, with N blocks on the phoneme side and N blocks on the mel-spectrogram side. A length regulator is incorporated between these blocks to bridge the length gap between the phoneme and mel-spectrogram sequences.

Length Regulator

To address the problem of length mismatch between the phoneme and mel-spectrogram sequences and enable control over voice speed and prosody, FastSpeech incorporates a length regulator. The length regulator expands the hidden states of the phoneme sequence to match the length of the mel-spectrogram sequence. By adjusting the expansion factor, controlled by a hyperparameter , the voice speed can be modified. Additionally, breaks between words can be controlled by adjusting the duration of space characters, allowing for influence over the prosody of the synthesized speech.

Duration Predictor

Phoneme duration prediction is crucial for the functioning of the length regulator. FastSpeech includes a duration predictor, which consists of a 2-layer 1D convolutional network with ReLU activation, followed by layer normalization, dropout, and an additional linear layer. The duration predictor is stacked on top of the FFT blocks on the phoneme side and is jointly trained with the FastSpeech model. It predicts the length of mel-spectrograms for each phoneme using mean square error (MSE) loss. The predicted durations are computed in the logarithmic domain for improved training performance.

The duration predictor is trained by extracting ground-truth phoneme durations from an autoregressive teacher TTS model. The attention alignments from the teacher model are used to obtain the phoneme durations, ensuring accurate alignment between phonemes and mel-spectrograms.

These components work together to enable FastSpeech to generate mel-spectrograms in parallel, overcoming the challenges of slow inference speed and limited controllability in traditional TTS systems.

3.2.4 FastSpeech 2

This section continue with the architecture of FastSpeech 2 [12] and describe the key differences from FastSpeech 1. The overall model architecture of FastSpeech 2 is shown in Figure 3.9. It enhance the FastSpeech 1 architecture and consists of an encoder, a variance adaptor, and a mel-spectrogram decoder. Moreover, in this work, FastSpeech 2s was not utilized as the vocoder. Instead, the HifiGAN vocoder, which is a state-of-the-art model, was employed for generating the speech waveforms.

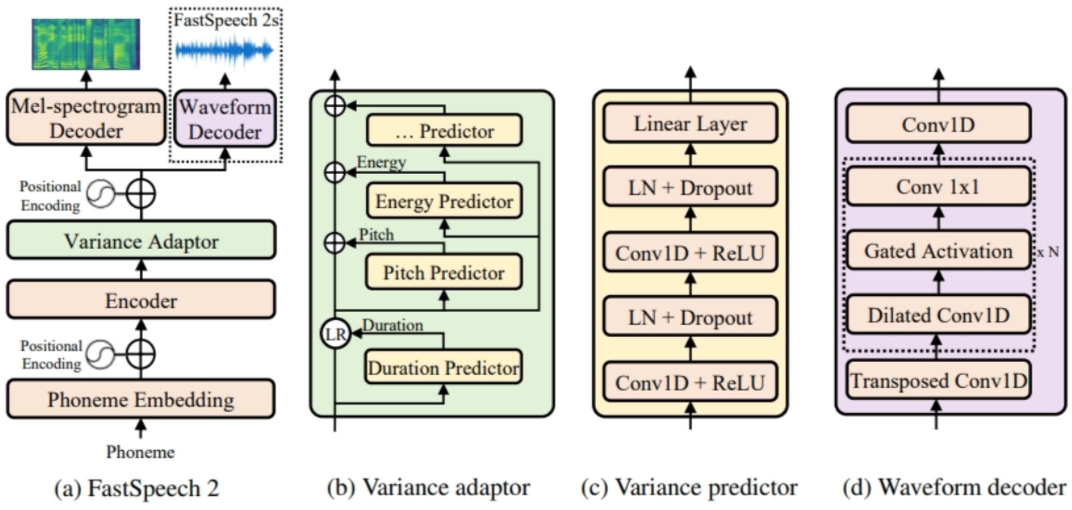


Figure 3.9: The overall architecture for FastSpeech 2 and 2s. (a) The architecture includes the encoder, variance adaptor, mel-spectrogram decoder and the waveform decoder. (b) Detail of the variance adaptor block. (c) Detail of a variance predictor, the structure is the same for energy, pitch and duration predictors. (d) Detail of the waveform decoder used in FastSpeech 2s. Source [12]

The encoder takes the phoneme embedding sequence and converts it into the phoneme hidden sequence. The variance adaptor adds various types of variance information, including duration, pitch, and energy, to the hidden sequence. Finally, the mel-spectrogram decoder converts the adapted hidden sequence into a mel-spectrogram sequence in parallel.

FastSpeech 2 utilizes the feed-forward Transformer block as the basic structure for both the encoder and mel-spectrogram decoder, similar to FastSpeech 1. However, FastSpeech 2 introduces several improvements:

Removal of Teacher-Student Distillation Pipeline

FastSpeech 2 removes the teacher-student distillation pipeline used in FastSpeech 1 for duration predictor. Instead, it uses the phoneme duration obtained by the Montreal Forced Alignment algorithm [32]. This extraction method improves alignment accuracy, avoids information loss during the distillation process and reduces the information gap between input and output.

Enhanced Variance Adaptor

The variance adaptor in FastSpeech 2 includes not only a duration predictor but also pitch and energy predictors.

- The **duration predictor** predicts the duration of each phoneme, representing the number of mel frames corresponding to that phoneme.
- The **pitch predictor** uses continuous wavelet transform (CWT) to decompose the pitch contour into a pitch spectrogram, which is then used as the training target. Inference involves predicting the pitch spectrogram and converting it back into a pitch contour using inverse continuous wavelet transform (iCWT).
- The **energy predictor** computes the L2-norm of the amplitude of each short-time Fourier transform (STFT) frame to obtain the energy. The energy predictor predicts the original energy values and is optimized using MSE loss.

FastSpeech 2s

FastSpeech 2 also introduces a simplified training pipeline called FastSpeech 2s. It directly generates waveforms from text without relying on cascaded mel-spectrogram generation (acoustic model) and waveform generation (vocoder). This move towards a fully end-to-end system simplifies the training process.

FastSpeech2 leverages these enhancements to generate mel-spectrograms in parallel, resulting in improvements in training speed and voice quality. Moreover, it introduces more control over speech synthesis by allowing modulation of the outputs of the variance predictors, such as phoneme duration, pitch, and energy, before feeding them into the mel-spectrogram decoder.

3.2.5 Vocoder: HiFiGAN

Speech synthesis tasks have greatly benefited from the utilization of **generative adversarial networks (GANs)**, a prominent class of deep generative models. GANs excel at capturing complex distributions and generating realistic samples by training a generator and discriminator in an adversarial framework. During training, the discriminator is initially provided with a set of ground truth data to establish a baseline for discrimination. This enables the discriminator to learn and improve its ability to differentiate between real data samples and the fake/generated samples produced by the generator. As training progresses, the generator receives feedback from the discriminator on how to improve its generated samples, leading to a refined and more accurate synthesis of speech audio. However, accurately modeling the periodic patterns inherent in speech audio remains a unique challenge due to the complex nature of sinusoidal signals with varying periods.

In this thesis, we focus on the integration and evaluation of the **HiFiGAN vocoder** [25] in voice cloning systems. HiFiGAN is a state-of-the-art vocoder that addresses the issue of modeling periodic patterns in speech audio. It introduces a discriminator composed of small sub-discriminators, each responsible for capturing specific periodic parts of raw waveforms. By extracting different parts of the audio signal, HiFiGAN effectively captures the intricate periodicity present in speech. Furthermore, HiFiGAN employs a module that incorporates multiple residual blocks, allowing the generator to observe patterns of various lengths in parallel. This design choice enhances the generator's ability to generate high-quality speech audio with accurate temporal structure.

The architecture of the model consists of a generator and two discriminators: a multi-scale discriminator and a multi-period discriminator. They work together through adversarial training to improve stability and model performance.

Generator

The generator, depicted in Figure 3.10 is a fully convolutional neural network. It takes a mel-spectrogram as input and uses transposed convolutions to upsample the input until the output sequence matches the temporal resolution of raw waveforms. Each transposed convolution is followed by a multi-receptive field fusion (MRF) module, which allows the generator to observe patterns of different lengths simultaneously. The MRF module combines the outputs of multiple residual blocks, each with distinct kernel sizes and dilation rates, to capture diverse receptive field patterns. The generator has adjustable parameters such as the hidden dimension, kernel sizes of transposed convolutions, and kernel sizes and dilation rates of MRF modules. These parameters can be customized to balance synthesis efficiency and sample quality.

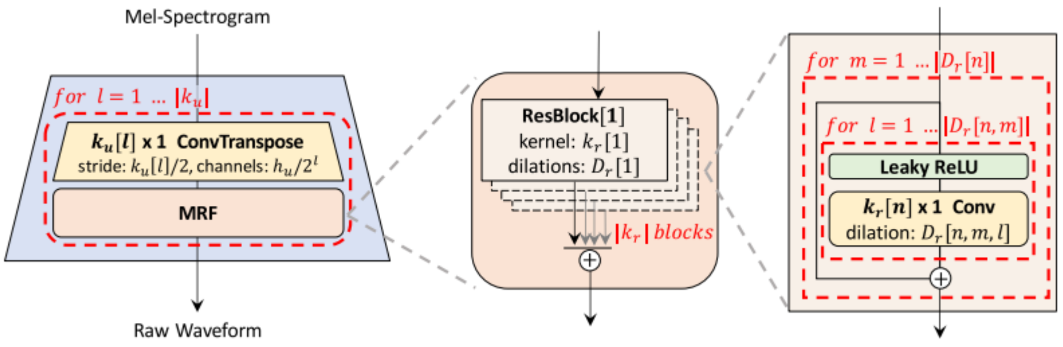


Figure 3.10: On the left, the full architecture of the generator, consisting of transposed convolution followed by the Multi Receptive Field (MRF) block that upsample the input up to $|K_u|$. In the center, a detailed view of the MRF block, showcasing different residual blocks with $|K_r|$ kernel size combined together. On the right, the structure of the $n-th$ residual block within the MRF module, characterized by a kernel size of $kr[n]$ and dilation rates of $Dr[n]$. Source [25].

Discriminator

The discriminator in HiFi-GAN, as a fundamental component, performs a core function in capturing long-term dependencies and identifying diverse periodic patterns in speech audio. It consists of two distinct components, as illustrated in Figure 3.11, with each component comprised of multiple sub-discriminators.

- **Multi-Period Discriminator (MPD):** the Multi-Period Discriminator (MPD) consists of multiple sub-discriminators, each specifically designed to handle a distinct portion of periodic signals present in the input audio. By examining different segments of the audio, these sub-discriminators effectively capture diverse implicit structures.

  Each sub-discriminator is composed of a stack of strided convolutional layers with leaky rectified linear unit (ReLU) activation. Furthermore, weight normalization is applied to the MPD, enhancing its stability and training dynamics. By transforming the input audio into 2D data instead of sampling periodic signals individually, gradients from the MPD can be propagated across all time steps of the input audio, enabling more effective learning and discrimination.

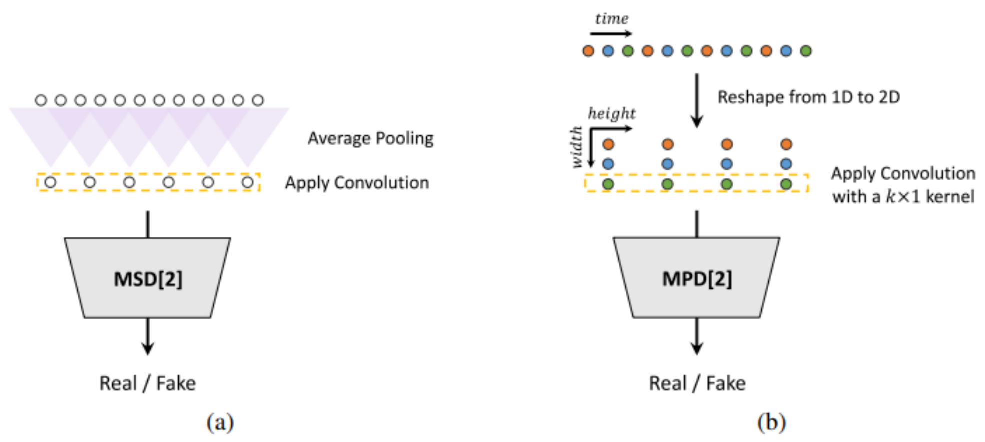


Figure 3.11: (a) View of the second sub-discriminator of MSD. (b) View of the second sub-discriminator of MPD. Source [25].

- **Multi-Scale Discriminator (MSD):** The Multi-Scale Discriminator (MSD) is a crucial component of HiFi-GAN, designed to capture consecutive patterns and long-term dependencies in the audio sequence. It evaluates the audio at multiple scales, allowing for a comprehensive analysis of the input.

  The MSD consists of three sub-discriminators, each operating on a different input scale: raw audio, ×2 average-pooled audio, and ×4 average-pooled audio. The raw audio sub-discriminator processes the original, uncompressed audio, while the other sub-discriminators handle downsampled versions of the audio at different scales.

  Each sub-discriminator is constructed as a stack of strided and grouped convolutional layers, featuring leaky ReLU activation. The discriminator size is increased by incorporating additional layers and reducing the stride. This architectural choice enables the MSD to capture fine-grained details and facilitate the detection of consecutive patterns and long-term dependencies.

  Weight normalization is applied to all sub-discriminators except the first one, which operates on the raw audio scale. This normalization technique enhances the stability and training dynamics of the discriminator, contributing to improved discrimination performance.

By combining the MPD and MSD, HiFi-GAN effectively captures both long-term dependencies and periodic patterns in speech audio. Overall, HiFi-GAN's vocoder architecture, with its generator and discriminators, enables high-quality speech synthesis by generating realistic waveforms based on mel-spectrograms.

3.3 Zero Shot Voice Cloning System

The Zero Shot Voice Cloning system is a highly promising implementation that utilizes three independently trained neural networks: the Speaker Embedder, the Synthesizer, and the Vocoder. In the following subsections, we will delve into the abstract architecture of the Voice Cloning System and subsequently explore the implemented architecture in this work. These architectures form the foundation of the system, enabling it to achieve remarkable capabilities in generating high-quality synthesized speech.

3.3.1 Abstract architecture

Figure 3.12 illustrates the abstract architecture of the Voice Cloning System, consisting of three independently trained components:

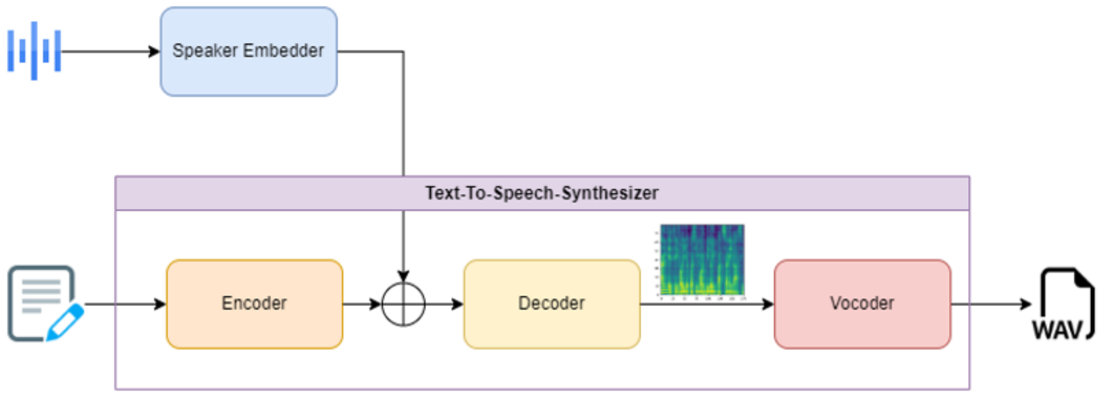


Figure 3.12: Abstract architecture of the Voice Cloning System comprising three independently trained models: Speaker Embedder, Synthesizer, and Vocoder.

Speaker Embedder

The speaker embedder plays a crucial role in the Voice Cloning System as it generates fixed-dimensional vectors from speech waveforms or speaker references. These vectors capture the unique characteristics of individual speakers and are essential for voice cloning, even for unseen voices.

It is important to note that Multi-Speaker TTS models also utilize speaker embeddings. However, these embeddings are computed internally using a simple embedding layer that maps unique IDs of known speakers to an internal hidden space. This means that embeddings can only be extracted for speakers that are already known and represented in the model. It is not possible to extract new speaker embeddings for unseen speakers using this approach.

By employing a Speaker Verification model to extract speaker embeddings, we can leverage the generalization power of the model and generate speaker embeddings for speakers who have not been seen before. This can be achieved by providing an input audio of the speaker's voice, allowing the Speaker Embedder to learn and capture the unique characteristics of that speaker, regardless of whether they were previously known or unseen.

Synthesizer

The synthesizer is a crucial component of the Voice Cloning System as it is responsible for predicting mel spectrograms based on input graphemes or phonemes. It plays a vital role in generating high-quality synthesized speech.

One important aspect to note is that the Speaker Embedding and the Synthesizer are trained on different datasets. The embeddings used by the Synthesizer are never seen during the training of the Speaker Embedder. This means that the Synthesizer can generate speech in various voices, including those that were not present in the training data for the Speaker Embedder. It demonstrates the versatility and generalization capabilities of the Voice Cloning System.

Additionally, it is worth mentioning that while the Speaker Embedding is language independent, allowing it to capture speaker characteristics across different languages, the Synthesizer is not inherently language independent. The training of the Synthesizer is typically specific to a particular language or a dataset in that language.

Vocoder

The vocoder is an essential component of the Voice Cloning System, responsible for converting synthesized mel spectrograms into time-domain waveforms. It plays a critical role in the text-to-speech pipeline by ensuring the generation of high-quality and natural-sounding speech.

To further enhance the performance of the Voice Cloning System, it is common practice to fine-tune the vocoder on the generated mel-spectrograms. Fine-tuning allows the vocoder to adapt its parameters specifically to the characteristics of the synthesized speech, optimizing the waveform generation process for the specific voice being cloned.

By leveraging these three networks, the Zero Shot Voice Cloning system achieves the capability to generate high-quality speech in various voices while conditioning the synthesis process on the speaker characteristics extracted by the speaker encoder.

3.3.2 Implemented architecture

To integrate the three components of the Voice Cloning System, various choices need to be made regarding the selection of models and the methods of integration. In this work, as depicted in 3.13, specific models were chosen based on their distinct advantages and capabilities.

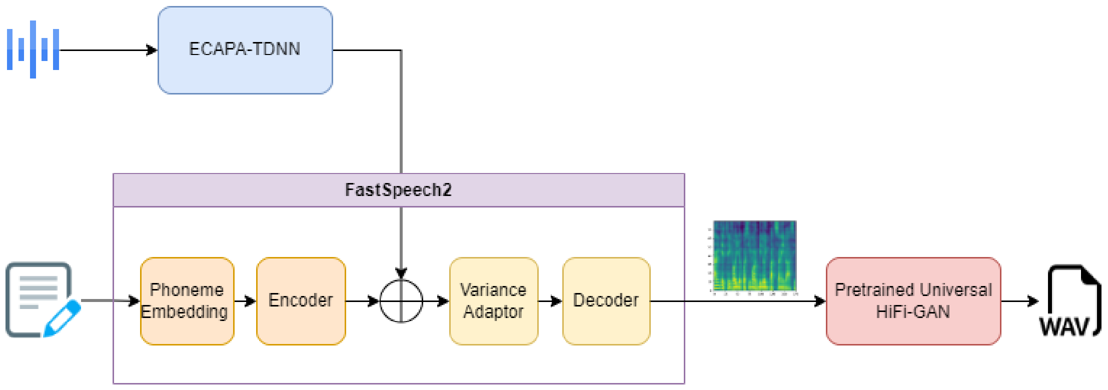


Figure 3.13: Implemented architecture of the Voice Cloning System comprising: ECAPA-TDNN as Speaker Embedder, FastSpeech2 as Synthesizer, and HiFiGAN as Vocoder.

- The **FastSpeech2** model was selected as the **Synthesizer** component because it offers several advantages over Tacotron 2. FastSpeech2 is known for its faster inference speed and allows for precise control over pitch, energy, and duration in the synthesized speech. These characteristics make it an ideal choice for generating high-quality and customizable speech.

- For the **Speaker Embedder** component, the **ECAPA-TDNN** model was chosen due to its outstanding performance in speaker verification tasks. Through experimental evaluation 4, it has been shown to achieve the best performance compared to WAVLM.

- As for the **Vocoder** component, **HiFiGAN** was selected at the time of writing as it represented the state-of-the-art vocoder model.

Furthermore, in the integration of the Zero Shot Voice Cloning system, different choices can be made. This work specifically focuses on two aspects:

- **Speaker Embedding averaging**:

  One approach is to average all the embeddings extracted from a speaker and use the averaged embedding for all utterances. However, in this work, a different strategy was chosen to better simulate the behavior at inference time. Instead of averaging, a unique embedding is computed for each utterance, even if the speaker remains the same. This decision enhances the system's ability to capture subtle variations in the speaker's voice across different utterances.

- **Integration of embeddings into TTS**:

  In this work, the integration of speaker embeddings into the Text-to-Speech (TTS) process was achieved using a linear layer. This layer projects the speaker embedding into the dimension of the hidden vector and then combines them by summation. An alternative approach would be to expand the dimension of the decoder input and concatenate the speaker embedding to the hidden vector.

By adopting these strategies, the Voice Cloning System effectively incorporates speaker characteristics into the synthesis process. This enables the generation of speech in various voices with control over variance, pitch, and energy, leveraging the capabilities of the FastSpeech2 model.

Chapter 4

Experiments

"In questions of science, the authority of a thousand is not worth the humble reasoning of a single individual."
Galileo Galilei

4.1 Speaker Verification

In this work, for the speaker verification task, the primary objective is to compare the two available models on HuggingFace, namely ECAPA-TDNN and WAVLM. The reported metrics are assessed, and the models are fine-tuned using the GE2E loss to better model the behavior at inference time and generalize the speaker embedding space effectively.

The subsequent section introduces the VoxCeleb 1 dataset, which is a vast collection of audio speech. The methods section reports the techniques, algorithms, tips, and notes employed in the experiment.

Additionally, results subsection presents qualitative outputs trough UMAP and explains the metric used to evaluate the models, the Equal Error Rate, demonstrating the effectiveness of utilizing the GE2E loss.

4.1.1 Data: VoxCeleb

The VoxCeleb dataset [9] is a valuable resource for speaker recognition and verification tasks. It offers a vast collection of audio speech from diverse speakers, including recordings from interviews, movies, and YouTube videos. This dataset addresses the need for large-scale, diverse, and challenging data tailored for speaker recognition.

The VoxCeleb dataset has become a benchmark in recent years, extensively used for evaluating speaker recognition algorithms. Researchers rely on its standardized evaluation metrics and protocols to compare their models' performance against state-of-the-art results, fostering advancements in the field. VoxCeleb provides different trial pairs for the verification task, consisting of pairs of utterances with corresponding ground truth labels. These labels indicate 0 if the utterances belong to different speakers and 1 otherwise.

The VoxCeleb dataset is available in two versions. VoxCeleb1 contains over 145,000 utterances from 1,211 celebrities, while VoxCeleb2 consists of more than 1 million utterances from 6,112 celebrities, all extracted from YouTube videos.

It is important to note that VoxCeleb2 offers a larger dataset compared to VoxCeleb1, providing more extensive coverage of different speakers. However, despite attempts to utilize VoxCeleb2 for the work, it is no longer accessible due to privacy policy restrictions. This limitation unfortunately prevents further use and exploration of the VoxCeleb2 dataset.

Split	**# of speakers**	**# of videos**	**# of utterances**
Verification (dev)	1,211	21,819	148,642
Verification (test)	40	677	4,874

Table 4.1: VoxCeleb1 Dataset Statistics

The dataset not directly present the audio of utterances instead it comprises a collection of YouTube url videos with timestamps for utterances. To prepare the data for training, a pipeline was employed to download the YouTube videos, convert them to audio, and extract the speech segments.

For the VoxCeleb1 dataset, there are different splits available for verification and identification tasks. The verification split includes a development set and a test set, with a varying number of speakers, videos, and utterances in each. Similarly, the identification split also has separate sets for development and testing.

There are different trial pairs list for evaluation: VoxCeleb1-O contains original version, VoxCeleb1-E contains only english speakers, VoxCeleb1-C is the cleaned version, VoxCeleb1-H is the hardest list.

It is important to note that the VoxCeleb1-H and VoxCeleb1-E lists were originally designed to be used with the VoxCeleb1 training set. These lists had significance when VoxCeleb2 was available, as the data from VoxCeleb1 were used as the test set in that scenario. However, with the unavailability of VoxCeleb2, the context has changed. Since VoxCeleb2 is no longer accessible, using the VoxCeleb1-H and VoxCeleb1-E lists for evaluation purposes may not serve the same purpose as before. Therefore, it is essential to consider the current limitations and adjust the evaluation approach accordingly, given the absence of VoxCeleb2.

4.1.2 Methods

In the training pipeline different choices can be made to further enhance the learning process and asses correct validation and testing. In particular the training pipeline for this work focus on two aspect: data augmentation and the batch construction.

Data augmentation

Data augmentation techniques were applied using established recipes found in the literature. First, reverberation effects were applied to audio signals by convolving them with room impulse responses from the well-known RIR and Noise Database [32]. Additionally, background audios from the MUSAN dataset were randomly added to input samples [33].

Batch construction

To create training batches a random cycler algorithm was used. It selects M random speakers at each step. For each selected speaker, N random utterances were drawn, randomly cropped to a fixed size of 3 seconds, and included in the batch. The random cycler algorithm ensures that the same samples are not picked until a training epoch is reached.

Algorithm 1 Random Cycler

```
1: procedure RandomCycler(items, sample_length)
2:     Initialize empty list seen_items
3:     cut ← length of items  mod sample_length
4:     while length of seen_items < (length of items − cut) do
5:         Select a random item from (items − seen_items) and assign it to item
6:         Add item to seen_items
7:         return item
8:     end while
9: end procedure
```

In this pseudo code, the RandomCycler class is represented as a procedure. The algorithm initializes an empty list `seen_items` to keep track of the items that have been seen. The variable `cut` is calculated as the remainder of the length of items divided by `sample_length`. The algorithm then enters a loop that continues until the length of `seen_items` is equal to the difference between the length of items and `cut`. Inside the loop, a random item is selected from the set difference of items and `seen_items`, assigned to the variable item, and added to `seen_items`. Finally, the selected item is returned and added to the batch.

4.1.3 Results

To evaluate the training results, this section provides both qualitative visualization through UMAP and quantitative evaluation using the Equal Error Rate (EER).

Qualitative Results: UMAP

For qualitative visualization of the finetuning process in speaker verification, UMAP (Uniform Manifold Approximation and Projection) was employed to represent the speaker embeddings extracted from a batch in a 2D scatterplot [34].

UMAP is a dimension reduction technique that offers similar visualization capabilities to t-SNE while providing general non-linear dimension reduction capabilities. It is based on a theoretical framework rooted in Riemannian geometry and algebraic topology. This practical and scalable algorithm is applicable to real-world data and competes with t-SNE in terms of visualization quality, while preserving more of the global structure and offering superior runtime performance. Furthermore, UMAP does not have computational restrictions on the embedding dimension, making it a versatile dimension reduction technique for machine learning applications.

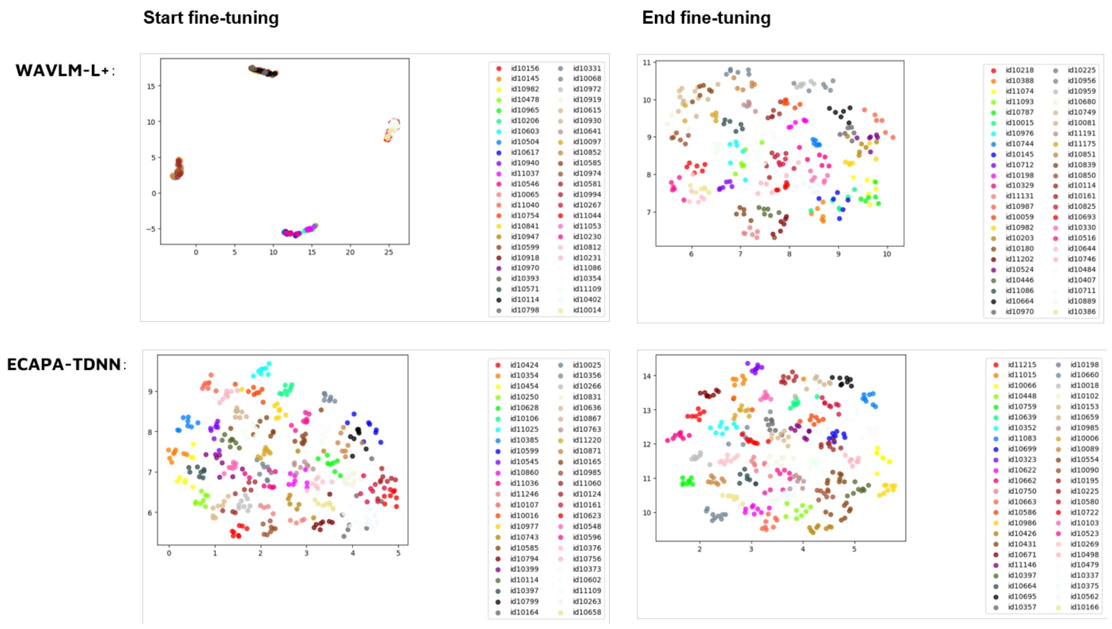


Figure 4.1: Visualization of the start and end of the finetuning process using UMAP dimensionality reduction for the two analyzed models: WAVLM-L+ and ECAPA-TDNN

Figure 4.1 illustrates the finetuning process using the GE2E loss on two principal model architectures: WAVLM-L+ and ECAPA-TDNN. The ECAPA-TDNN architecture is pretrained on both VoxCeleb1 and VoxCeleb2 (with the HuggingFace model also trained on VoxCeleb2), enabling it to already exhibit the ability to group different speakers into clusters at the beginning of the finetuning process. As the finetuning progresses, the clusters become more refined, indicating improved performance.

In contrast, the WAVLM-L+ architecture has a backbone pretrained on different speech processing benchmarks, excluding VoxCeleb, and the x-vector head pretrained on VoxCeleb1. During finetuning, only VoxCeleb1 data is used to re-finetune the head of the model with the GE2E loss. The UMAP visualization demonstrates that the model initially lacks clear speaker clustering. However, as the finetuning continues, the model gradually learns to cluster different speakers, showcasing its ability to discern and separate speakers based on their embeddings.

Quantitative Metrics: Equal Error Rate (EER)

The results were quantified using the Equal Error Rate (EER), which is a verification algorithm that determines threshold values for false acceptance rate and false rejection rate. The equal error rate occurs when these rates are equal, and a lower EER indicates higher model accuracy. Table 4.2 presents the EERs computed on two lists of utterance pairs from the VoxCeleb1 test set, namely V0 and V0-cleaned. The results of this study were compared to existing pretrained models on the HuggingFace Hub. Notably, both models developed in this study outperformed the available models, suggesting that the use of the GE2E loss leads to improved performance. Specifically, despite ECAPA-TDNN having significantly fewer parameters (22.3 million) compared to WAVLM-L+ (304 million), it achieved superior results. This performance difference can be attributed to ECAPA-TDNN's pretraining, which made it more specialized in VoxCeleb data compared to WAVLM.

Model	From	EER VO-C	Treshold VO-C	EER VO	Treshold VO
ECAPA-TDNN	SpeechBrain (HF)	0.90%	28.02%	1.04%	27.28%
ECAPA-TDNN	**This work**	**0.86%**	**32.04%**	**0.97%**	**30.55%**
WAVLM+	Microsoft (HF)	4.72%	87.46%	6.41%	84.37%
WAVLM+	This work	3.64%	76.03%	4.11%	68.55%
WAVLM-L+	This work	1.55%	45.09%	2.03%	40.22%

Table 4.2: Equal Error Rate results for Speaker verification models on VoxCeleb 1 data. Models from this work are trained using GE2E Loss. WAVLM-L+ is the larger model tested with 316.62M parameters and requires 4 GPUs (NVIDIA Tesla V100 SXM2 - 32 GB - 5120 CUDA cores) for a batch size of 288.

4.2 Text To Speech & Zero Shot Voice Cloning

In this section, we explore the methods and techniques used in Text To Speech (TTS) and Zero Shot Voice Cloning. We focus on two datasets: the LibriTTS corpus and the 61PangramITA dataset. The LibriTTS corpus provides a comprehensive English dataset with unique features such as higher sampling rate and sentence-level segmentation. It offers valuable resources for training and evaluating TTS models. The 61PangramITA dataset is a privately collected dataset specifically designed for voice cloning. It consists of pan-gram utterances from Italian speakers, providing high-quality input examples with high variability.

We discuss the modifications made to Tacotron 2, a popular TTS model, to handle Italian accented characters, ensuring accurate representation of the Italian language. Additionally, we explore the integration of speaker embeddings from ECAPA-TDNN into the FastSpeech2 text encoder output, forming the basis of Zero Shot Voice Cloning.

To evaluate the voice cloning system's performance, we utilize specific metrics. The Voice Clone Error Rate (VC-ER) measures the similarity between cloned and real speaker voices, while the Word Error Rate (WER) assesses the intelligibility of the synthesized speech.

In the subsequent sections, we present the results obtained from the voice cloning experiments, analyzing the Voice Clone Error Rate and Word Error Rate. These results provide insights into the performance and intelligibility of the synthesized speech. We also discuss the implications of these findings for future research and development in the field of Text To Speech and Voice Cloning.

4.2.1 Data: LibriTTS, Pangram61ITA

LibriTTS

The LibriTTS corpus [35], a product of collaborative efforts between Heiga Zen, the Google Speech team, and the Google Brain team, is a multi-speaker English dataset comprising approximately 585 hours of meticulously recorded read English speech, it is divided in different set.

Split	Number of Examples	Memory Size	Comment
dev_clean	5,736	1.2G	Development set, clean speech
dev_other	4,613	924M	Development set, more challenging speech
test_clean	4,837	1.2G	Test set, "clean" speech
test_other	5,120	964M	Test set, "other" speech
train_clean100	33,236	7.7G	Training set derived from the original materials of the train-clean-100 subset of LibriSpeech
train_clean360	116,500	27G	Training set derived from the original materials of the train-clean-360 subset of LibriSpeech
train_other500	205,044	44G	Training set derived from the original materials of the train-other-500 subset of LibriSpeech

Table 4.3: LibriTTS Dataset Statistics

Created as an extension of the renowned LibriSpeech corpus, LibriTTS inherits its foundation from the mp3 audio files sourced from LibriVox and the text files derived from Project Gutenberg. However, several significant modifications have been made to enhance its usability in TTS research. For the Text To Speech (TTS) models, the LibriSpeechTTS dataset was used. This dataset is a multi-speaker English corpus that consists of 585 hours of read English speech from 2,456 speakers, along with their corresponding texts.

One notable improvement in LibriTTS is the higher 24kHz sampling rate of the audio files, which improves the overall audio fidelity and ensures a more accurate representation of speech characteristics. Furthermore, the speech in LibriTTS has been segmented at sentence breaks, allowing for fine-grained analysis and modeling of individual utterances.

Another valuable aspect of the LibriTTS corpus is the inclusion of both the original and normalized texts. This duality provides researchers with the flexibility to experiment with different text representations and processing techniques, thereby facilitating the development of more robust and adaptable TTS systems.

61PangramITA

The 61PangramITA dataset is a privately collected dataset that includes pangram utterances from 7 Italian speakers with different accents. These utterances consist of sentences containing all the letters of the alphabet, known as pangrams. Pangrams provide high-quality input examples that are rich in information. The motivation behind choosing pangrams is their high variability in terms of different characters present in each utterance. By using pangrams at each training step, all possible characters are encountered, reducing the number of samples required to clone a voice. This approach demonstrates that high-quality data can be more effective than a high amount of data. Although an improved version could consider all possible phonemes, it would be more complex.

The objective of the study was to train the model to imitate a specific voice (the author's voice) using only these 61 sentences, amounting to approximately 5 minutes of speech data.

In Table 4.4, some pangram utterances from the 61PangramITA dataset are reported.

4.2.2 Methods

Case Study: Cloning Italian Voices with Few Samples

To explore the utilization of pangram utterances and enhance the model's understanding in the domain of voice cloning, a series of supplementary experiments were conducted. Specifically, for Tacotron 2, a case study was designed using the

Sentence	**Number of Character**
Qualche vago ione tipo zolfo, iodio, sodio	27
Pochi sforzan quel gambo di vite	28
O templi, quarzi, vigne, fidi boschi!	28
Qui gli ampi stronzi, bove, defechi?	28
Joe Magic beve quel whisky d'Oxford a Potenza	37
Che tempi brevi, zio, quando solfeggi	30
Mai posto quiz vaghi o indecifrabili	31
Voglio quei fiaschi di bronzo in tempo	32

Table 4.4: Pangram Utterances from 61PangramITA Dataset

61PangramITA dataset to showcase the efficacy of employing pangram utterances for reducing the data requirements in voice cloning tasks.

Modifications were made to Tacotron 2 to handle Italian accented characters. It was pretrained on the LJspeech-ITA dataset, which consisted of approximately 20,000 Italian utterances from the same speaker. The decoder was then fine-tuned using the 61 pangram utterances from the author's voice, after warming up the mel predictor layers.

The experimental findings indicated that incorporating pangram utterances significantly facilitated the training process and improved the model's ability to generalize to unseen input texts.

Table 4.5 shows the validation loss of two finetuning approaches with Tacotron 2. The first approach used the 61 pangram utterances, while the second approach was finetuned using 200 randomly selected utterances from the LJspeech-ITA dataset, recorded with the same voice as the pangram utterances. Both models were trained for 20 epochs.

Finetuning Approach	**Validation Loss**
61 Pangram Utterances	0.33
200 Random Utterances	0.42

Table 4.5: Validation loss comparison of Tacotron 2 finetuning with and without pangram utterances

However, training Tacotron 2 for this task proved to be time-consuming due to alignment problems. These alignment problems arise from discrepancies between the generated output frames and the corresponding input characters. To overcome this challenge, alignment plots were utilized to ensure accurate associations between the input text and the generated speech.

Alignment plots provide a visual representation of the alignment process during training. They consist of two axes: the encoder (y-axis) and the decoder (x-axis). The encoder takes an input character and its current state at each step and outputs a real vector representing the status of the model at that moment along the y-axis.

The decoder, on the other hand, takes all the vectors from the y-axis and uses them to generate audio frames or mel-spectrograms. The decoder works step-by-step, with each step determining which vectors from the y-axis are important for generating the audio frames at that particular moment. Bright colors in the alignment plot indicate areas where the decoder should focus more attention along the y-axis, while darker colors indicate areas where less attention is needed.

During the training process, it was necessary to continue training until the alignment plot exhibited a correct diagonal alignment, even if low loss values were achieved. Figure 4.2 displays examples of alignment plots generated during the fine-tuning process.

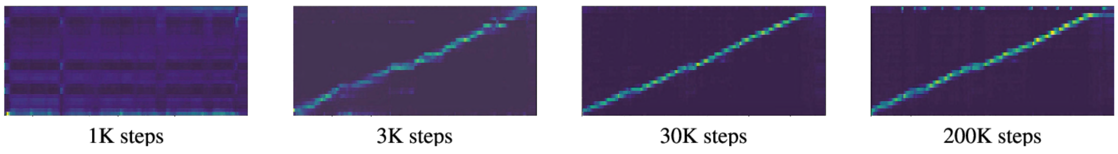


Figure 4.2: Alignment plot during different finetuning steps. The x-axis represents the decoder, which generates audio frames (mel-spectrograms), and the y-axis represents the encoder, which takes input characters and their current state to output a real vector representing the model's status. Bright colors indicate areas where the decoder focuses more attention along the y-axis, while darker colors indicate areas with less emphasis.

Zero Shot Voice Cloner

The FastSpeech2 approach tackled the alignment problem by utilizing multiple sets of ground truth, including mel spectrogram, phoneme duration, pitch, and energy. While extracting mel spectrogram, pitch, and energy is relatively straightforward, a more sophisticated technique known as Montreal forced alignment (MFA) was employed to extract phoneme duration. MFA leverages an orthographic transcription and a pronunciation dictionary to create a time-aligned version of the audio.

Multiple smaller training sessions were conducted to select parameters and determine the optimal models for the final voice cloning system. The selected models for development were ECAPA-TDNN and FastSpeech2. ECAPA-TDNN achieved the best Equal Error Rate (EER) score, while FastSpeech2 was faster, more robust, and allowed for postmodulation of pitch, energy, and speed of the output.

The Voice Cloner integrated the speaker embedding from ECAPA-TDNN into the FastSpeech2 text encoder output. Different strategies can be used to combine the embeddings, such as averaging all the embeddings from a speaker or computing a unique embedding for each utterance. The speaker embedding can also be combined with the hidden representation of the encoder using a linear layer or by expanding the decoder input dimension and concatenating the speaker embedding to the hidden vector.

In this work, a speaker embedding was used for each utterance during training. It was projected into the dimension of the encoder output and summed to it. The resulting Voice Cloning System Architecture is shown in Figure 1.

4.2.3 Results

To evaluate the performance of the voice cloning system, conventional human Mean Opinion Score (MOS) evaluations were not conducted. Instead, two alternative metrics were used Voice Clone Error Rate to measure similarity with original voice and Word Error Rate to measure intellegibility of the synthesized speech.

Voice Clone Error Rate (VC-ER

The Voice Clone Error Rate (VC-ER) was used to assess the similarity between cloned and real speaker voices. One hundred speakers from the LibriTTS dataset were selected, with 25 utterances per speaker. Two sets of 2,500 pairs were created, one comprising real and synthesized utterances with identical text, and the other with different text in the synthesized utterances. The best speaker verification (SV) model (ECAPA-TDNN with GE2E) and its Equal Error Rate (EER) threshold (0.86%) were employed. A lower VC-ER score indicates a better voice cloning system, while a higher score signifies a greater ability of the SV model to detect cloned speech. Results of the Voice Cloning System from this work are shown in table 4.6

Voice Clone Error Rate	
Same utterances text	1120/2500 (44.83%)
Different utterances text	1665/2500 (66.62%)

Table 4.6: Voice Cloning System metrics: Voice Clone Error Rate measures voice similarity between cloned and real utterances.

The results indicate the effectiveness of utilizing speaker embeddings to condition TTS systems. When comparing original and synthesized speech with identical text, the Speaker Verification system successfully detects speech cloning in 55.87% of the cases. In contrast, when different text is used, it confirms the cloning in 33.38% of the cases. These metrics reflect the quality of the cloning process, where the synthesized voice closely resembles the original voice but is not identically. However, there is room for improvement in achieving even better results.

Word Error Rate (WER)

Word Error Rate (WER) served as a measure of intelligibility. The Whisper OpenAI model was utilized to extract text from the 2,500 pairs of real and synthesized utterances that contained the same text. The extracted text from Whisper was then compared with the real text for both the real utterances (to establish the ground truth) and the synthesized utterances. Results of the Voice Cloning System from this work are shown in table 4.7

Word Error Rate	
Ground truth (real utterances)	13.39%
Voice cloner (synth utterances)	20.54%

Table 4.7: Voice Cloning System metrics: Word Error Rate measures intelligibility of the synthesized utterances.

The results indicate a difference of 7.15% between the automatic transcription of the original audio input and the automatic transcription of the synthesized utterances. This suggests that the synthesized utterances are slightly less comprehensible (as assessed by the OpenAI Whisper model) compared to the originals. However, the degradation in intelligibility is considered minimal.

Chapter 5

Conclusion

"In God we trust; all others must bring data."
William Edwards Deming

5.1 Final comments

In the chapters discussed, a framework was first provided on two tasks belonging to the field of speech processing: Speaker Recognition and Text-to-Speech Synthesis, deepening the methods used in literature with particular attention to recent Deep Learning models with which several experiments were carried out.

For the Speaker Recognition task, recent works on model architectures and loss were followed and two well-known models, were fine-tuned using newer loss, enhancing the available models from HuggingFace.

Then, an extension of the Text-to-Speech task was introduced to allow synthesis with a particular voice never seen by the model, starting from about 5 seconds of audio. To reach this extension, it was discussed how the two tasks, SR and TTS, could be exploited.

TTS models can be fine-tuned on a particular voice, but this requires a large amount of data in order to converge to the desired voice. It was shown how to reduce the amount of data needed, by exploiting pangrams, and a private dataset was collected in order to verify the goodness of this approach. In any case, the need for a fine-tuning and therefore a model for each speaker is a considerable limitation, and it was discussed how zero-shot learning methods can provide a solution in this direction, integrating the TTS and SR tasks.

The following sections summarize the main results divided for the two tasks analyzed and their integration. Subsequently, in future directions, some possibilities for the continuation were discussed with particular attention to the reduction of risks derived from a voice cloning technology.

Speaker Recognition

For the speaker recognition the Generalized end-to-end loss was exploited, the fine-tuning of the two models with it led to improvements in performance, evaluated on the VoxCeleb1 benchmark, compared to the starting models made available on HuggingFace. The generalized approach of this loss allows the integration of random components in the training and the generation of batches, generating many more different examples than the initial data at each epoch.

For WAVLM in its large version, it was expected to reach the best performance, however, despite the EER obtained being considerably low, it does not surpass ECAPA-TDNN, which with many fewer parameters obtains better performance. Comparing also the results presented by the authors of WAVLM [7], it is assumed that the model made available on HuggingFace does not correspond to the one described in the paper. ECAPA-TDNN available with pretrained weights on VoxCeleb 1 and 2 is more specialized on the VoxCeleb data than WAVLM, which does not know the VoxCeleb 2 data. Unfortunately, VoxCeleb 2 is no longer available for copyright reasons, so a fairer comparison could retrain ECAPA-TDNN from scratch using only VoxCeleb in order To enhance the performance of speaker verification models, additional information such as accent can be leveraged. The generalized end-to-end (GE2E) loss can be extended in a hierarchical manner, where each speaker is assigned to a class representing their accent. By incorporating the GE2E approach, we can compute not only speaker distances but also accent distances to better model the speaker space. This enables a more comprehensive representation of the latent speaker space by considering individual characteristics, accents, and any other pertinent information.

Text To Speech Synthesis

In terms of TTS, the series of Tacotron and FastSpeech models were analyzed, in particular Tacotron2 was used for the case study on the 61PangramITA dataset containing Italian pangrams from 7 different speakers. This demonstrates the potential to greatly reduce the required sample size by employing phrases with high variability in character composition. Through fine-tuning Tacotron2 using a mere 61 phrases, equivalent to approximately 5 minutes of audio, it becomes feasible to successfully clone the voice of the author of this thesis. The results obtained are not considered of excellent quality but they are good enough to demonstrate the efficiency in the use of pangrams.
The pangram experiment highlighted how Tacotron2 suffers in terms of training speed due to the alignment problem discussed, this combined with the greater controllability offered, led to the choice of FastSpeech2 as the model to be integrated into the Zero Shot Voice Cloning system.

Zero-Shot Voice Cloning System

Finally, the experiments on SR and TTS led to the choice of ECAPA-TDNN and FastSpeech2 as models to be integrated in order to implement a Zero-Shot Voice Cloning System. The experiments led to two integration choices: during the training of FastSpeech2 an embedding for each phrase is used to condition the system instead of using the mean of the embeddings for the speaker, furthermore the speaker embedding is integrated in the decoder of the synthesis model through a projection and then added to the input of this.
The final solution yielded satisfactory results, showcasing the effectiveness of the approach. The evaluation of the model's performance involved two aspects: similarity to the cloned voices and intelligibility. It is important to note that the reported metrics are discrete and there exists significant room for improvement. For instance, finetuning the HiFiGAN vocoder on the generated melspectrograms can ensure a higher quality waveform output.

5.2 Future Directions: Containing Malicious Applications

While various technical improvements can be pursued to enhance the quality, controllability, and speed of voice cloning systems, it is crucial to address the potential risks they pose to society. The ability to steal someone's identity through their voice opens the door to numerous malicious applications. As responsible researchers, it is imperative to acknowledge and mitigate the risks associated with the development of such high-risk technologies.

In this regard, future work should focus on effectively limiting the potential for malicious applications. One approach is to leverage Speaker Verification models as a form of discrimination, determining whether a vocal signal is synthetic or authentic. This methodology has been utilized to calculate the Voice Clone Error Rate metric. However, future research could explore training Speaker Verification models using both synthetic and non-synthetic data to further mitigate the risks arising from inappropriate use of these technologies.

It is important to note that while a Speaker Verification system can serve as a preliminary discriminator, it is not a definitive solution. A determined voice cloning system can potentially be refined to outperform the discriminator. Nevertheless, incorporating a Speaker Verification model into the development process represents an initial step towards controlling the risks. Therefore, researchers developing voice cloning systems should consider releasing a system that includes mechanisms to mitigate the potential harm, such as incorporating a speaker verification model capable of accurately discriminating cloned voices.

Bibliography

[1] D. A. Reynolds. «Automatic speaker recognition using gaussian mixture speaker models». In: The Lincoln laboratory journal volume B. number 2 (1995), p. 173 (cit. on pp. 2, 5).

[2] S. Furui. «Speaker recognition». In: *Scholarpedia* 3.4 (2008). revision #64889, p. 3715. DOI: 10.4249/scholarpedia.3715 (cit. on pp. 2, 4).

[3] Noor Salwani Ibrahim and Dzati Athiar Ramli. «I-vector Extraction for Speaker Recognition Based on Dimensionality Reduction». In: *Procedia Computer Science* 126 (2018). Knowledge-Based and Intelligent Information Engineering Systems: Proceedings of the 22nd International Conference, KES-2018, Belgrade, Serbia, pp. 1534–1540. ISSN: 1877-0509. DOI: https://doi.org/10.1016/j.procs.2018.08.126. URL: https://www.sciencedirect.com/science/article/pii/S1877050918314042 (cit. on pp. 2, 7).

[4] Ehsan Variani, Xin Lei, Erik McDermott, Ignacio Lopez Moreno, and Javier Gonzalez-Dominguez. «Deep Neural Networks for Small Footprint Text-dependent Speaker Verification». In: *Proc. ICASSP*. 2014 (cit. on pp. 2, 7).

[5] David Snyder, Daniel Garcia-Romero, Gregory Sell, Daniel Povey, and Sanjeev Khudanpur. «X-Vectors: Robust DNN Embeddings for Speaker Recognition». In: *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2018, pp. 5329–5333. DOI: 10.1109/ICASSP.2018.8461375 (cit. on pp. 2, 7).

[6] Brecht Desplanques, Jenthe Thienpondt, and Kris Demuynck. «ECAPA-TDNN: Emphasized Channel Attention, Propagation and Aggregation in TDNN Based Speaker Verification». In: *Interspeech 2020*. ISCA, Oct. 2020. DOI: 10.21437/interspeech.2020-2650. URL: https://doi.org/10.21437%2Finterspeech.2020-2650 (cit. on pp. 2, 27).

[7] Sanyuan Chen et al. «WavLM: Large-Scale Self-Supervised Pre-Training for Full Stack Speech Processing». In: *IEEE Journal of Selected Topics in Signal Processing* 16.6 (Oct. 2022), pp. 1505–1518. DOI: 10.1109/jstsp.2022.

3188113. URL: https://doi.org/10.1109%5C%2Fjstsp.2022.3188113 (cit. on pp. 2, 23, 24, 58).

[8] Li Wan, Quan Wang, Alan Papir, and Ignacio Lopez Moreno. *Generalized End-to-End Loss for Speaker Verification*. 2017. DOI: 10.48550/ARXIV.1710.10467. URL: https://arxiv.org/abs/1710.10467 (cit. on pp. 2, 10).

[9] Arsha Nagrani, Joon Son Chung, and Andrew Zisserman. «VoxCeleb: A Large-Scale Speaker Identification Dataset». In: *Interspeech 2017*. ISCA, Aug. 2017. DOI: 10.21437/interspeech.2017-950. URL: https://doi.org/10.21437%5C%2Finterspeech.2017-950 (cit. on pp. 2, 46).

[10] Anna Favaro, Licia Sbattella, Roberto Tedesco, and Vincenzo Scotti. «ITAcotron 2: Transfering English Speech Synthesis Architectures and Speech Features to Italian». In: *Proceedings of the 4th International Conference on Natural Language and Speech Processing (ICNLSP 2021)*. Trento, Italy: Association for Computational Linguistics, Dec. 2021, pp. 83–88. URL: https://aclanthology.org/2021.icnlsp-1.10 (cit. on p. 2).

[11] Jonathan Shen et al. *Natural TTS Synthesis by Conditioning WaveNet on Mel Spectrogram Predictions*. 2018. arXiv: 1712.05884 [cs.CL] (cit. on pp. 2, 17, 33).

[12] Yi Ren, Chenxu Hu, Xu Tan, Tao Qin, Sheng Zhao, Zhou Zhao, and Tie-Yan Liu. *FastSpeech 2: Fast and High-Quality End-to-End Text to Speech*. 2022. arXiv: 2006.04558 [eess.AS] (cit. on pp. 2, 17, 18, 37).

[13] Wei Ping, Kainan Peng, Andrew Gibiansky, Sercan O. Arik, Ajay Kannan, Sharan Narang, Jonathan Raiman, and John Miller. *Deep Voice 3: Scaling Text-to-Speech with Convolutional Sequence Learning*. 2018. arXiv: 1710.07654 [cs.SD] (cit. on p. 3).

[14] Sercan O. Arik, Jitong Chen, Kainan Peng, Wei Ping, and Yanqi Zhou. *Neural Voice Cloning with a Few Samples*. 2018. arXiv: 1802.06006 [cs.CL] (cit. on p. 3).

[15] Giuseppe Ruggiero, Enrico Zovato, Luigi Di Caro, and Vincent Pollet. *Voice Cloning: a Multi-Speaker Text-to-Speech Synthesis Approach based on Transfer Learning*. 2021. arXiv: 2102.05630 [cs.SD] (cit. on p. 3).

[16] Matt Edwards. *Mix it up Monday: Begin with the source*. 2017 (cit. on p. 5).

[17] Georg Heigold, Ignacio Moreno, Samy Bengio, and Noam Shazeer. *End-to-End Text-Dependent Speaker Verification*. 2015. DOI: 10.48550/ARXIV.1509.08062. URL: https://arxiv.org/abs/1509.08062 (cit. on pp. 8, 9).

[18] Ciwan Ceylan and Michael U. Gutmann. *Conditional Noise-Contrastive Estimation of Unnormalised Models*. 2018. DOI: 10.48550/ARXIV.1806.03664. URL: https://arxiv.org/abs/1806.03664 (cit. on p. 9).

[19] Elad Hoffer and Nir Ailon. *Deep metric learning using Triplet network*. 2014. DOI: `10.48550/ARXIV.1412.6622`. URL: `https://arxiv.org/abs/1412.6622` (cit. on p. 9).

[20] P. Taylor. *Text-to-Speech Synthesis 1st Edition*. NY, USA, 2017 (cit. on p. 13).

[21] Georg Heigold, Ignacio Moreno, Samy Bengio, and Noam Shazeer. *Introduction to speech processing*. URL: `https://speechprocessingbook.aalto.fi` (cit. on pp. 14, 15).

[22] *Scikit learn Gaussian mixture*. URL: `https://scikit-learn.org/stable/modules/mixture.html` (cit. on p. 16).

[23] *Statistical Parametric Speech Synthesis Incorporating Generative Adversarial Networks*. 2017 (cit. on p. 17).

[24] D. W. Griffin and J. S. Lim. *Signal estimation from modified short-time Fourier transform*. 1984 (cit. on p. 17).

[25] Jungil Kong, Jaehyeon Kim, and Jaekyoung Bae. *HiFi-GAN: Generative Adversarial Networks for Efficient and High Fidelity Speech Synthesis*. 2020. arXiv: `2010.05646 [cs.SD]` (cit. on pp. 17, 39–41).

[26] Sercan Arik, Gregory Diamos, Andrew Gibiansky, John Miller, Kainan Peng, Wei Ping, Jonathan Raiman, and Yanqi Zhou. *Deep Voice 2: Multi-Speaker Neural Text-to-Speech*. 2017. arXiv: `1705.08947 [cs.CL]` (cit. on p. 18).

[27] Sercan O. Arik, Jitong Chen, Kainan Peng, Wei Ping, and Yanqi Zhou. *Neural Voice Cloning with a Few Samples*. 2018. arXiv: `1802.06006 [cs.CL]` (cit. on pp. 18, 19).

[28] Yongqin Xian, Christoph H. Lampert, Bernt Schiele, and Zeynep Akata. *Zero-Shot Learning – A Comprehensive Evaluation of the Good, the Bad and the Ugly*. 2020. arXiv: `1707.00600 [cs.CV]` (cit. on p. 19).

[29] Abgeiba Isunza Navarro. *Zero-Shot Learning in NLP*. URL: `https://modulai.io/blog/zero-shot-learning-in-nlp/` (cit. on p. 20).

[30] Yuxuan Wang et al. *Tacotron: Towards End-to-End Speech Synthesis*. 2017. arXiv: `1703.10135 [cs.CL]` (cit. on pp. 30, 31).

[31] Yi Ren, Yangjun Ruan, Xu Tan, Tao Qin, Sheng Zhao, Zhou Zhao, and Tie-Yan Liu. *FastSpeech: Fast, Robust and Controllable Text to Speech*. 2019. arXiv: `1905.09263 [cs.CL]` (cit. on p. 35).

[32] *RIR dataset*. URL: `https://www.openslr.org/28/` (cit. on p. 48).

[33] David Snyder, Guoguo Chen, and Daniel Povey. *MUSAN: A Music, Speech, and Noise Corpus*. arXiv:1510.08484v1. 2015. eprint: `1510.08484` (cit. on p. 48).

[34] Leland McInnes, John Healy, and James Melville. *UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction*. 2020. arXiv: `1802.03426 [stat.ML]`. URL: `https://www.openslr.org/17/` (cit. on p. 49).

[35] Heiga Zen, Viet Dang, Rob Clark, Yu Zhang, Ron J. Weiss, Ye Jia, Zhifeng Chen, and Yonghui Wu. *LibriTTS: A Corpus Derived from LibriSpeech for Text-to-Speech*. 2019. arXiv: `1904.02882 [cs.SD]` (cit. on p. 51).